Multiple Description Vector Quantizer Design Based on Redundant Representation of Central Code

Akinori Ito
Graduate School of Engineering
Tohoku University
Sendai, Miyagi 980-8579 Japan
Email: aito@spcom.ecei.tohoku.ac.jp

***Abstract*—A design method of a multiple description vector quantizer (VQ) is proposed. VQ is widely used for data compression, transmission and other processing. Here, we assume transmission channels with data erasure such as a packet-based network. Multiple description coding is a coding method used to achieve “graceful degradation” when transmitting signals through lossy channels. The proposed method is inspired by the vector quantizer design of Poggi et al., which combines VQ design based on the self-organizing map (SOM) and the multiple description scalar quantizer (MDSQ). The method also uses the SOM-based VQ; the difference is that the proposed method combines a bit-error-tolerant VQ designed by SOM and a novel scheme for cell arrangement of SOM based on Redundant Representation of Central Code (RRCC). The method is not only easy to design for any bit rate but is also more robust against data erasure compared with the conventional VQ.**

I. INTRODUCTION

For digital communication, packet-based networks such as the Internet are often used. In a network based on the Internet Protocol (IP) [1], sent data packets are eventually lost because of bit errors introduced in the transmission channel or congestion. To recover the lost packet, methods of retransmitting the packets are often used such as transmission control protocol (TCP) [2]. However, sometimes packet retransmission is not appropriate or is even impossible for real-time or multicast application. In this case, various error correction coding methods are commonly used such as Reed-Solomon code [3], Turbo code [4], LDPC code [5], etc. These error correction codes completely recover the original data when the loss rate is under a certain threshold. However, if more losses than the threshold are introduced, almost all of thje data in the packets are lost.

When the data to be protected are multimedia signals such as image, video or audio signals, complete bit-by-bit recovery is not always necessary. Packet loss concealment [6], [7] is based on the assumption that the quality of the recovered signal can be degraded from the original signal if the degradation is sufficiently small. Multiple description coding (MDC) is a channel coding scheme used in such a situation, where slight degradation is permitted for a digital communication channel with erasure.

MDC is a channel coding method that splits the original signal into two or more bitstreams. The MD encoder encodes the input signal into two (or more) descriptions, and the descriptions are sent to the destination independently. At the receiver side, if the two descriptions are received, they are decoded at the central decoder and a high-quality signal is obtained. When only a part of the descriptions is received, the received description is decoded by the side decoder, and the decoded signal is degraded, but it is better than nothing.

Many MDC methods have been proposed so far. Time series data such as speech signal can be divided using pairs of two contiguous samples [8]–[11]. As a more general method, the MD scalar quantizer (MDSQ) [12] splits one scalar value into two descriptions. As the MDSQ does not need any assumption about the distribution of the value to be split, it can be used for any data.

When we want to encode a vector, a vector quantizer (VQ) is used because of its optimality compared with the scalar quantizer. Several methods to combine VQ and MDC have been proposed such as lattice MDVQ [13] or SOM-based MDVQ [14]. The proposed method is an improvement of the SOM-based MDVQ, which has the problem that it is difficult to design the index assignment matrix (IAM). The design of IAM greatly affects the performance of the quantizer, but there is no fixed methodology that enables the quantizer to be designed automatically.

This paper proposes a new design method of MDVQ. The key point of the method is to assign redundant bit sequences to the code vectors of the quantizer, which enables the erased code to be estimated when one of the descriptions is lost on the channel.

II. CONVENTIONAL MDSQ AND MDVQ

A. MDSQ

MDSQ is a method to split a scalar value into two descriptions, while the total rate of the descriptions is less than twice the rate of the original value. Figure 1 shows an example of an index assignment matrix (IAM) when splitting a three-bit value into two two-bit codes. For example, if we want to send the original value of 3, we send “10” and “01” to channel A and channel B, respectively. When both of the descriptions are received, the original code can be recovered at the receiver side using the same IAM. When the description of channel A is lost, we receive only “01” from channel B. In this case, we know that the original value should be either 2 or 3. If the prior probability distribution of the original value is known, the most probable value among the possibilities (2 or 3) is chosen.

978-0-9928-6265-7/16/$31.00 ©2016 IEEE

Side channel B \ Side channel A	00	01	10	11
00	0	1		
01		2	3	
10		4	5	
11			6	7

Fig. 1. An example of IAM of the MDSQ

If the distribution is uniform, the value is chosen randomly. Although the IAM determines the bit rate, robustness against erasure and distortion, the design methodology of IAM is not completely established [12].

B. Lattice-based MDVQ by Vaishampayan

The multiple description vector quantizer (MDVQ) is an MDC for VQ that encodes one vector into two or more vectors. The lattice-based MDVQ proposed by Vaishampayan [13] quantizes the input vector into a point of the regularly arranged lattice. Among the lattice points, coarse lattice points (sub-lattice) are defined. Using the lattice-MDVQ, any point of the lattice can be associated with two sub-lattice points uniquely, and the sub-lattice points are located relatively near to the original lattice point. Thus we send two numbers of the sub-lattice points instead of sending the number of the lattice point. If the two sub-lattice points are received, we can recover the original lattice point. When only one sub-lattice point is received, that sub-lattice point is used as the received data.

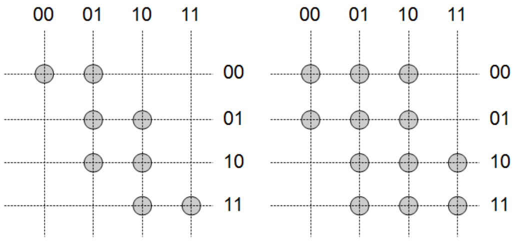
This method can be used for any multidimensional vectors, but the lattice cannot be designed from the distribution of the data to be coded. Another difficulty with this method is that, while theoretically possible, it is not easy to design a sub-lattice for any lattice.

C. SOM-based MDVQ by Poggi et al.

Poggi et al. proposed an MDVQ using the self-organizing map (SOM) [14]. Poggi's method consists of two components; one is to design the cell arrangement using a similar strategy as MDSQ, and the other is to train the code vectors using SOM.

SOM [15] is a kind of neural network that simulates the topographic map in the central nervous system. From an engineering point of view, SOM is regarded as a clustering algorithm where some kind of order is introduced among the code vectors.

In SOM, we use "cells" instead of "codes" of the ordinary VQ. The k-th cell has a coordinate in the cell space $\mathbf{r}_k$ and a weight vector (which is the same as a code vector) $\mathbf{m}_k$ in the same space as the input vector. When training data $\mathbf{x}_1, \ldots, \mathbf{x}_N$ are given, SOM determines the arrangement of the weight vectors so that the quantization error becomes smaller and the correlation between $||\mathbf{r}_j - \mathbf{r}_k||$ and $||\mathbf{m}_j - \mathbf{m}_k||$ becomes larger. The arrangement of $\mathbf{r}_k$ is usually given at the beginning of training and does not change during training. Using SOM, we can design VQ so that code vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ are near when $\mathbf{r}_i$ and $\mathbf{r}_j$ are near.



(a) n=8 (b) n=12

Fig. 2. Arrangement of SOM cells $\mathbf{r}_k$ according to the IAM

Poggi's method uses the property of SOM, and arranges $\mathbf{r}_k$ according to the IAM used for an MDSQ. Figure 2 shows examples of the arrangement of cells. As shown, the cells are arranged in the same way as the values in the IAM shown in Fig. 1. When the code vectors are trained using this cell arrangement, we can expect that the cells that are near in the cell space also have code vectors that are near in the vector space. At the same time, we can assign two codes to a cell in exactly the same way as that in MDSQ. The decoding method is also the same as those of MDSQ; the only difference is that the MDVQ outputs the code vector assigned to the cell instead of outputting the cell number.

The problems of this method are almost the same as those of MDSQ. It is difficult to design the cell arrangement (which corresponds to an IAM), especially when the bit length of the code is large. Another problem is that there is no theoretical background on how to choose the cell among the possible cells when one description is lost.

III. PROPOSED METHOD

A. Overview

The proposed MDVQ is basically similar to the MDVQ by Poggi et al., where SOM is used to design code vectors and cells are arranged according to the descriptions sent to two channels. The difference is that a bit pattern is assigned in addition to the cell number to a cell, which determines the cell arrangement and multiple descriptions simultaneously. This bit pattern is called "Redundant Representation of Central Code (RRCC)." An overview of the proposed method is as follows. Here, we only consider a two-channel case, but it is not difficult to extend the proposed method to the case of three or more channels.

1) Each cell has an RRCC in addition to the cell number (k of $\mathbf{r}_k$). When the length of a description sent to one of two channels is B bits, the length of the RRCC becomes $2B$ bits.
2) Arrange the cells $\mathbf{r}_k$ at the vertices of a $2B$-dimensional hypercube so that the distance between two cells is related to the Hamming distance between two RRCCs of the cells.
3) Train code vectors using SOM.
4) In the encoding phase, when a vector is given, choose the nearest code vector, and determine the two descriptions based on the corresponding RRCC.

	000	001	011	010	110	111	101	100
000	000000	000001		000010				000100
001	001000	001001	001011				001101	
011		011001	011011	011010		011111		
010	010000		010011	010010	010110			
110				110010	110110	110111		110100
111			111011		111110	111111	111101	
101		101001				101111	101101	101100
100	100000				100110		100101	100100

Fig. 3. An example of RRCCs for three-bit side decoders

5) In the decoding phase, when the two descriptions are received at the receiver side, the RRCC is recovered by concatenating the two descriptions and the corresponding code vector is output. When one description is lost, the RRCC is estimated by repeating the received description twice, and the corresponding code vector is output.

The details of the method are explained below.

B. Designing RRCCs

First, the method of designing RRCCs is described. Let the bit length of a description sent to one channel be B. Here, the description is an integer $0 \leq c < 2^B$. Let $b(x, B)$ be a B-bit representation of integer x, and $b_1.b_2$ be the concatenation of two bit sequences. Then the set of RRCC R is determined as follows.

1) $b(c, B).b(c, B) \in R$ for $0 \leq c < 2^B$
2) $b(c, B).b(c', b) \in R$ when $D_H(b(c, B), b(c', B)) = 1$ for $0 \leq c, c' < 2^B$

Here, $D_H(b, b')$ is the Hamming distance between two equal-length bit patterns. The property of RRCC assures that the Hamming distance between the first half and the second half of an RRCC is at most one. Figure 3 shows an example of RRCCs for $B = 3$, where the row and column are the codes sent to the two channels. The bit patterns in the table represent RRCCs. Here, we can transmit one of 64 codes (5 bits) by sending $3 + 3 = 6$ bits using both channels.

C. Training code vectors using SOM

After designing RRCCs, code vectors are trained using SOM. The key point is to arrange a cell at the coordinate determined from the corresponding RRCC. When an RRCC is a $2B$-bit pattern $b_1 \dots b_{2B}$, the coordinate of the corresponding cell is $\mathbf{r}_k = (b_1, \dots, b_{2B})$. We can arrange the code vectors so that two cells with similar RRCCs also have similar code vectors. This kind of VQ, which has tolerance against bit errors introduced in the transmission channel, is called "channel-optimized VQ" [16], [17]. A channel-optimized VQ using SOM was proposed by de Bodt et al. [18], and our method of training the code vectors is a variant of de Bodt's method.

D. Decoding

When we receive two descriptions at the receiver side, we can recover the original RRCC by concatenating the two descriptions. When one of the two descriptions is lost, RRCC is estimated by repeating the received description. For example, assume that $B = 2$ and the original RRCC is 0100. Here, the first half 01 and the second half 00 are sent to the receiver using different channels. When we only receive the first description 01, we estimate the RRCC to be 0101.

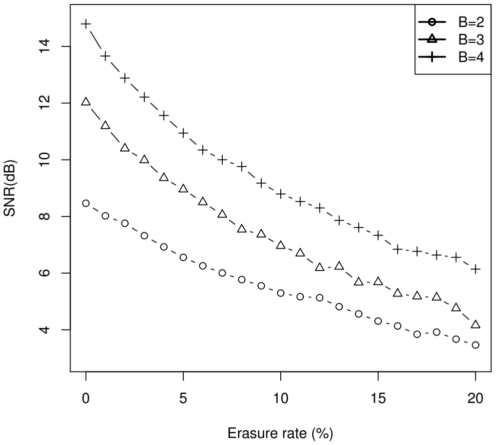


Fig. 4. SNR of the proposed vector quantizer

Consider an RRCC $B_R = B_R^1.B_R^2$ where $B_R^1 = b_1 \dots b_B$, $B_R^2 = b_{B+1} \dots b_{2B}$. From the definition of RRCC, it is assured that $D_H(B_R^1, B_R^2) \leq 1$. Thus, when we estimate RRCC as $B_R' = B_R^1.B_R^1$,

$$\begin{aligned} D_H(B_R, B_R') &= D_H(B_R^1.B_R^2, B_R^1.B_R^1) \\ &= D_H(B_R^2, B_R^1) \leq 1 \end{aligned} \quad (1)$$

and the same result holds for $B_R' = B_R^2.B_R^2$. We trained the code vectors so that code vectors associated with similar RRCCs are located at nearby positions, so we expect the code vector recovered from the estimated RRCC to be similar to the original one.

The advantage of the proposed method is that we can design the quantizer automatically; we do not need to design the IAM, and we can design the quantizer for any bit length of the code. A drawback of the method is that the number of code vectors is fixed when we determine the bit length of the description. The number of code vectors (and the number of RRCCs) is $(B+1)2^B$, which means that RRCCs become sparse when B is large.

IV. EXPERIMENT

Simulation experiments were conducted to compare the proposed method with conventional vector quantizers. The input vectors had two-dimensions, where each dimension obeys $N(0, 1)$ independently. 1000 training samples and 1000 test samples were used. The batch SOM algorithm was used for training the SOM, and the number of iteration was 30. When testing, discriptions were randomly lost according to the erasure rate. When one description was lost, we estimated the RRCC using the above-mentioned method. If two descriptions were lost, we just used a fixed code vector. We conducted the same experiment 10 times with different random seeds, and the results were averaged.

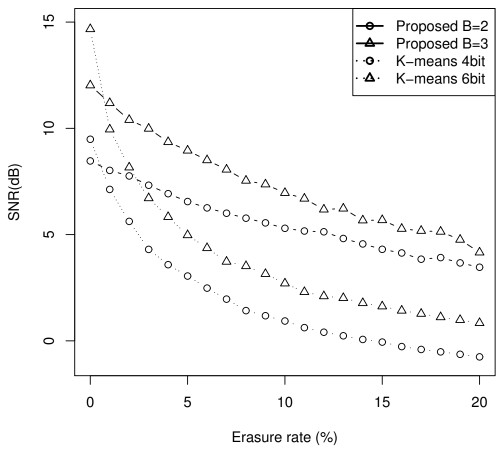


Fig. 5. SNR of the proposed vector quantizer and k-means vector quantizer

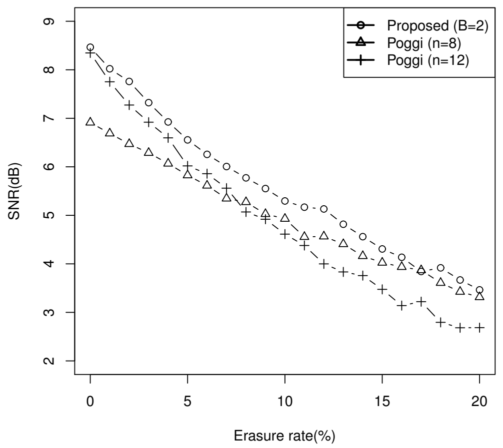


Fig. 6. SNR of the proposed vector quantizer and Poggi's vector quantizer

Figure 4 shows the result of the proposed method for different values of B. The x-axis is the erasure rate. In all cases, the degradation of SNR at an erasure rate of 10 % was around 5 dB.

Figure 5 compares the results of the proposed method and the k-means method, the most popular method for designing an ordinary VQ. In this experiment, the total rate was set to be the same. We can see that, when no errors were introduced, the k-means method outperformed the proposed method. However, for an erasure rate of 1 %, the proposed method showed higher SNR.

Next, we compared the proposed method with the SOM-based MDVQ proposed by Poggi et al. [14]. In this experiment, B was set to 2. We used the two cell arrangements shown in Fig. 2, where the numbers of code vectors were 8 and 12. The number of code vectors of the proposed method was 12.

Figure 6 shows the result, revealing that the proposed method has a higher SNR than the conventional method. When the number of code vectors was 8 in the conventional method, degradation upon increasing the erasure rate was small, but the total SNR was low; when the number was 12, SNR under no erasure was almost the same as that of the proposed method, but SNR fell rapidly when the erasure rate increased.

V. CONCLUSION

In this paper, a new MDVQ design methodology was proposed. The proposed method was inspired by the SOM-based MDVQ, but we do not need to design the IAM. A new concept, RRCC, was introduced in the training process. A channel-optimized VQ based on SOM was used to train the code vectors. Simulation experiments revealed that the proposed method was more robust than the conventional VQ and MDVQ.

REFERENCES

[1] E. J. Postel, "Internet protocol," RFC791, 1981.
[2] ——, "Transmission control protocol," RFC793, 1981.
[3] I. S. Reed and G. Solomon, "Polynomial codes over certain finite fields," *J. Soc. for Industrial and Applied Mathematics*, vol. 8, no. 2, pp. 300–304, 1960.
[4] C. Berrou and A. Glavieux, "Near optimum error correcting coding and decoding: turbo-codes," *IEEE Trans. Communication*, vol. 44, no. 10, pp. 1261–1271, 1996.
[5] M. C. Davey and D. J. C. MacKay, "Low density parity check codes over $gf(q)$," in *Proc. Information Theory Workshop*, 1998, pp. 70–71.
[6] C. Perkins, O. Hodson, and V. Hardman, "A survey of packet loss recovery techniques for streaming audio," *IEEE Network*, vol. 12, no. 5, pp. 40–48, 1998.
[7] B. Wah, X. Su, and D. Lin, "A survey of error-concealment schemes for real-time audio and video transmissions over the Internet," in *Proc. Int. Symp. on Multimedia Software Engineering*, 2000, pp. 17–24.
[8] W. Jiang and A. Ortega, "Multiple description speech coding for robust communication over lossy packet networks," in *IEEE Int. Conf. on Multimedia and Expo*, 2000, pp. 444–447.
[9] V. K. Goyal and J. Kovačević, "Generalized multiple description coding with correlating transform," *IEEE Trans. Inf. Theory*, vol. 47, no. 6, pp. 2199–2224, 2001.
[10] A. Ito and S. Makino, "Multiple description coding of an audio stream by optimum recovery transforms," *Journal of Digital Information Management*, vol. 6, pp. 189–195, 2008.
[11] ——, "Designing side information of multiple description coding," *Journal of Information Hiding and Multimedia Signal Processing*, vol. 1, no. 1, pp. 10–19, 2010.
[12] V. A. Vaishampayan, "Design of multiple description scalar quantizers," *IEEE Trans. Inform. Theory*, vol. 39, pp. 821–834, 1993.
[13] V. A. Vaishampayan and N. Sloane, "Multiple-description vector quantization with lattice codebooks: design and analysis," *IEEE Trans. Inf. Theory*, vol. 47, no. 5, pp. 1718–1734, 2001.
[14] G. Poggi, D. Cozzolino, and L. Verdoliva, "Self-organizing maps for the design of multiple description vector quantizers," *Neurocomputing*, vol. 122, pp. 298–309, 2013.
[15] T. Kohonen, "The self-organized map," *Proc. IEEE*, vol. 78, no. 9, pp. 1464–1480, 1990.
[16] N. Farvardin, "A study of vector quantization for noisy channels," *IEEE Trans. Information Theory*, vol. 36, no. 4, pp. 799–809, 1990.
[17] X. Yu, H. Wang, and E.-H. Yang, "Design and analysis of optimal noisy channel quantization with random index assignment," *IEEE Trans. Information Theory*, vol. 56, no. 11, pp. 5796–5804, 2010.
[18] E. de Bodt, M. Cottrell, P. Letremy, and M. Verleysen, "On the use of self-organizing maps to accelerate vector quantization," *Neurocomputing*, pp. 187–203, 2004.