韓國音響學會誌 5 卷 3 號 (1986)

유한상태 벡터양자화를 이용한 격리단어인식

Isolated Word Recognition Based on Finite-State Vector Quantization

* 윤 원 식 (Youn, W. S.)
** 은 종 관 (Un, C. K.)

요 약

본 논문은 유한상태 벡터양자화(**finite-state vector quantization**) 방법을 이용한 격리단어인식에 관하여 기술하고 있다. 이 인식시스템은 **codebook**과 **next-state function**으로 구성된 일종의 **finite-state machine**으로 볼 수 있다. 유한상태 벡터양자화방법을 이용한 격리단어인식시스템은 일반적인 벡터양자화방법을 이용한 인식시스템에 비하여 인식소요시간이 감소하며 입력음성을 분할할 필요도 없는 한편 두 시스템의 인식율은 비슷한 것으로 나타났다. **Next-state function**을 구하는 방법에는 **conditional histogram**방법과 **omniscient design**방법이 있으며, 이 방법들의 성능비교를 위해 영부터 구까지의 한국어 숫자음성에 대한 인식실험을 수행하였다.

ABSTRACT

In this paper, we propose an isolated word recognition system based on the finite-state vector quantization (FSVQ) method. The recognition system can be viewed as a finite state machine composed of a codebook and next-state functions. As compared to an isolated word recognition system that uses the conventional memoryless vector quantization, the proposed system requires far less search time, and needs no segmentation of input speech, yet yields comparable recognition accuracies. For the design of next-state functions, two techniques, that is, the conditional histogram and omniscient design methods, are used, and their performances are compared in recognition of the ten Korean digits.

I. INTRODUCTION

Various forms of the memoryless vector quantization technique have been applied with considerable success to the problem of isolated word recognition for a small size of vocabulary [1],[2]. In this method, either single-section or multisection codebook is used to represent the reference pattern for each word in the recognition vocabulary. In the case of using a multisection codebook, depending on how the input speech is segmented, the computational complexity, the recognition accuracy, and the memory size of the recognition system can vary significantly. In the case of a single-section codebook, segmentation of input speech that is required for the system using a multisection codebook need not be done, and the memory size for codebook storage can be reduced. However, the search time for classification takes much longer than the system with a multisection codebook.

To alleviate these problems, we propose an improved isolated word recognition system using the finite-state vector quantization (FSVQ) method. The reference pattern for each word consists of a codebook and next-state functions. By using a next-state function, the proposed system takes far less recognition time than a system based on the ordinary memoryless vector quantization method. In this paper we will consider two techniques (that is, the conditional histogram and omniscient design methods) of finding a next-state function, given a codebook.

In what follows, we first review the finite-state vector quantization (FSVQ) method proposed by Gray et al. [3],[4]. We then present the isolated word recognition algorithms based on the FSVQ method. Finally, we present and discuss simulation results.

II. FINITE-STATE VECTOR QUANTIZATION

A memoryless vector quantizer (VQ) is a source coder that maps consecutive vectors independently into one of the elements of a reproduction set or codebook. Unlike the conventional VQ which is memoryless, an FSVQ requires memory [3],[4]. Because it utilizes the correlational property between successive source vectors, it yields better performance than that of the memoryless VQ.

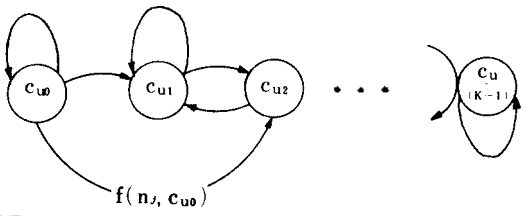
An FSVQ is a finite state machine composed of an encoder and a decoder. Let R^k, S and N denote a k-dimensional Euclidean space, a finite state space and an alphabet of finite channel symbols, respectively. Then, the encoder α maps in such a way that $\alpha: R^k \times S \rightarrow N$. That is, given a state $s \in S$, an input vector $x \in R^k$ is encoded into a channel symbol. And the encoder uses a next-state function f, a mapping $f: N \times S \rightarrow S$ that determines a next state from a current state and a channel symbol. The decoder β maps according to $\beta: N \times S \rightarrow R^k$, and has the same next-state function as that of the encoder. A finite-state vector quantizer encodes a sequence of input vectors $\{x_n, n=0,1,\cdots\}$ as follows [4]: Given an initial state $s_0 \in S$, the channel symbol sequence $\{u_n, n=0,1,\cdots\}$, the state sequence $\{s_n, n=0,1,\cdots\}$, and the reproduction sequence $\{\hat{x}_n, n=0,1,\cdots\}$ are determined recursively for n=0,1, ... as

$$u_n = \alpha(x_n, s_n),\ s_{n+1} = f(u_n, s_n),$$
$$\hat{x}_n = \beta(u_n, s_n). \quad (1)$$

As seen in the above FSVQ scheme, the encoder mapping is done by using different codebooks for each input vector where the codebooks are chosen by previous channel symbol and encoder state. A state codebook $C_s = \{\beta(u, s), u \in N\}$ is the codebook of possible reproduction vectors obtainable by an encoder and a decoder in each state $s \in S$. In other words, the finite-state vector quantizer achieves encoder mapping using a state codebook or a part of the entire (or super) codebook. In what follows, we consider application of the FSVQ method to isolated word recognition.

III. ISOLATED WORD RECOGNITION BASED ON FSVQ

In the proposed word recognition system based on the FSVQ method, a reference pattern for each word in the vocabulary consists of a single-section codebook and a next-state function. Each word may be represented as a finite state model related with a Moore finite state machine. Fig. 1 shows an example of a finite state machine with K states and N transitions. Each code represents a state, and each branch means a next-state function. A codeword corresponds to a state, and a codebook may be visualized as a finite state space. The path from one state to another is determined from the next-state function. Unlike a hidden Markov model [5], this finite state model does not require transition probabilities.



STATE:

c_{u0}	c_{u1}	c_{u2}	$\cdots$ $c_{u(K-1)}$

NEXT-STATE: FUNCTION

$f(n_0, c_{u0})$	$f(n_0, c_{u1})$	$f(n_0, c_{u2})$	$f(n_0, c_{u(K-1)})$
$f(n_1, c_{u0})$	$f(n_1, c_{u1})$	$f(n_1, c_{u2})$	$f(n_1, c_{u(K-1)})$
$\vdots$ $\vdots$	$\vdots$	$\vdots$	$\vdots$
$f(n_{N-1}, c_{u0})$	$f(n_{N-1}, c_1)$	$f(n_{N-1}, c_{u2})$	$f(n_{N-1}, c_{u(K-1)})$

Fig 1. A finite state model.

In the recognition system using an ordinary memoryless VQ, computation time for classification depends on the codebook size. But, the recognition speed in the proposed system is affected by the number of transitions. When the number of transitions is equal to the number of codewords (states), this model becomes an ordinary memoryless VQ.

For the proposed recognition system we design the reference pattern using the design algorithms for finite-state vector quantizers which combine ad hoc algorithms with an algorithm for the memoryless vector quantizer. The procedure includes design of a codebook and next-state functions. Let V be the number of words in the recognition vocabulary. Then, in the first step a single-section codebook $C_u = \{c_{u0}, c_{u1}, \cdots, c_{us}, \cdots, c_{u(K-1)}\}, u = 1, 2, \cdots, V$, having K codewords for word, is designed using a splitting method on the training sequence $L_u = \{x_i, i = 0, 1, \cdots\}$ [6]. Here we use the gain-normalized Itakura-Saito distortion measure in designing a codebook [7].

In the second step we design the next-state function $f(n_J, c_{us})$ or next possible N states for each state in a single-section codebook C_u. For the design of the next-state functions one can use the conditional histogram technique or the omniscient design method. These are now described.

in which empty cells do not occur [8]. When a state includes a self state as a next state, the number of codewords in the state codebook is (N-1). Consequently, we have

$$f(n_j, c_{us}) = \min_i{}^{-1} d(c_{usj}, c_{ui}) \tag{3}$$

where $\min^{-1}$ means that $f(n_j, c_{us})$ has the index i for which the codeword c_{ui} yields the minimum distortion over all codewords in the cdcebook C_u. After designing next-state functions, all state codebooks are discarded. And, like the conditional histogram method, the initial states for each word are saved in the reference pattern.

The proposed recognition system performs classifications using the codebooks and next-state functions. Let A_l be the feature vector for the i_{th} frame in the input utterance. Denoting D_{ul} the minimum distortion between A_l and codewords in the codebook C_u, we have

$$D_{u1} = \min_i d(A_1, c_{ui}) \tag{4}$$

where "i" indicates one of the initial states. Also, selecting c_{uj} as the initial state, we have for the second frame A_2

$$D_{u2} = \min_i d(A_2, c_{ui}),\ \ i = f(n_0, c_{uj}),\\ f(n_1, c_{uj}), \cdots, f(n_{N-1}, c_{uj}). \tag{5}$$

In general, if a codeword c_{uj} is selected for A_m we have

$$D_{u(m+1)} = \min_i d(A_{m+1}, c_{ui}),\\ i = f(n_0, c_{uj}), f(n_1, c_{uj}), \cdots, f(n_{N-1}, c_{uj}). \tag{6}$$

Hence, when an input utterance is encoded in a codebook C_u, the average distortion D_u is determined as

$$D_u = \frac{1}{L} \sum_{l=1}^{L} D_{ul} \tag{7}$$

where L is the number of frames in the input utterance. Consequently, if we have

$$D_r = \min_u D_u, \tag{8}$$

the utterance is then classified as the r_{th} word in the recognition vocabulary.

Since there exists relatively large confusion in recognition of Korean digits, we introduce a second classification stage in the proposed recognition system. Those digits except for 0(yuŋ) and 6(yook) which cause significant confusion are divided into three classes as the following: C_1 = {1(il), 2(i:), 7(chil)}, C_2 = {3(sä:m), 4(sä:), 8(pä:l)}, C_3 = {5(ō:), 9(gu:)}. In this we will call the classification procedure by the FSVQ method the first classification stage, and the decision process within one class the second classification stage.

We have the second classification stage when the following two conditions are met. The first condition is that the r_{th} word having minimum average distortion D_r is included in one of the three classes. And the second condition is

$$D_i / D_r < \epsilon_1,\ \ i \neq r \tag{9}$$

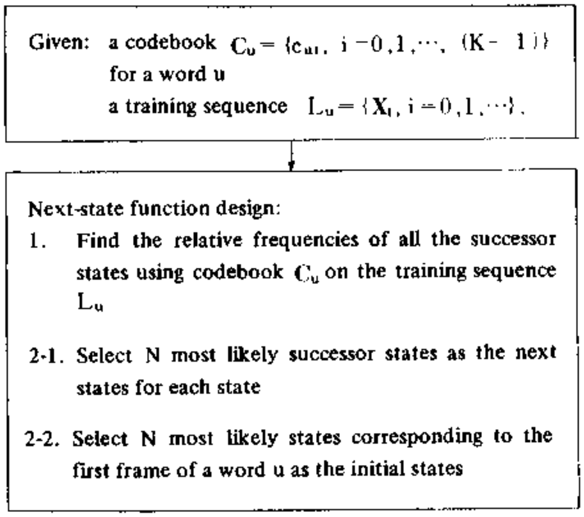
where ϵ_1 is a constant, and D_i is the average distortion of a word in a class that includes the r_{th} word. If the two conditions are not met, the recognized word is the r_{th} word.

A. Conditional Histogram Method

A simple method for finding a next-state function is to form a conditional codeword histogram using a codebook C_u on the training sequence L_u. In other words, this method is to find the occurrence frequencies of all the successor states for each state. Then, the N most likely next states are selected as the next states for each state.

When a state has less than N successor states, the next states for the state consist of states numbering less than N. Here the number N is the maximum number of next states from one state to another including a self state. The state may be considered as an approximation of the current input vector and prediction of the next input vector.

In getting the next-state function, initial states are obtained to find the state corresponding to the first frame of a word. Note that this initial state of the first frame is found among initial states in the reference pattern at the time of classification.



Given: a codebook $C_u = \{c_{ui},\ i = 0, 1, \cdots, (K-1)\}$
for a word u
a training sequence $L_u = \{X_i,\ i = 0, 1, \cdots\}$.

Next-state function design:

1. Find the relative frequencies of all the successor states using codebook C_u on the training sequence L_u

2-1. Select N most likely successor states as the next states for each state

2-2. Select N most likely states corresponding to the first frame of a word u as the initial states

Fig. 2 Next-state function design algorithm by conditional histogram method.

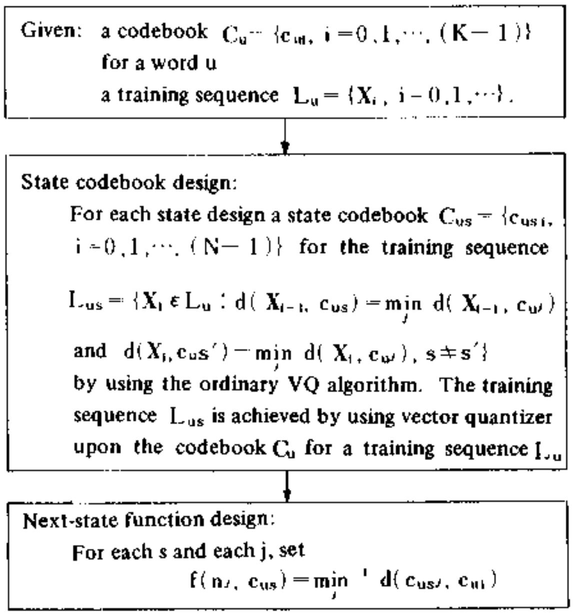
B. Omniscient Design Method

The omniscient design method is more complex than the conditional histogram method. With this method, one can design the next-state function $f(n_j, c_{us})$ after the construction of a state codebook having N codewords.

Given the cocdebook $C_u = \{c_{u0}, c_{u1}, \cdots, c_{us}, \cdots, c_{u(K-1)}\}$ for word u, a state codebook $C_{us} = \{c_{us0}, c_{us1}, \cdots, c_{us(N-1)}\}$ related with a state c_{us} is designed on the training sequence L_{us} given by

$$L_{us} = \{x_i \in L_u : d(x_{i-1}, c_{us}) = \min_j d(x_{i-1}, c_{uj}) \quad \text{and } d(x_i, c_{us'}) = \min_j d(x_i, c_{uj}),\ s \neq s'\}. \tag{2}$$

Here the gain-normalized Itakura-Saito distortion measure is again used as the distortion measure d(x,y) between x and y, and the k-means algorithm is used in the state codebook design



Given: a codebook $C_u = \{c_{ui},\ i = 0, 1, \cdots, (K-1)\}$
for a word u
a training sequence $L_u = \{X_i,\ i = 0, 1, \cdots\}$.

State codebook design:
For each state design a state codebook $C_{us} = \{c_{usi},\ i = 0, 1, \cdots, (N-1)\}$ for the training sequence

$$L_{us} = \{X_i \in L_u : d(X_{i-1}, c_{us}) = \min_j d(X_{i-1}, c_{uj})$$

$$\text{and } d(X_i, c_{us'}) = \min_j d(X_i, c_{uj}),\ s \neq s'\}$$

by using the ordinary VQ algorithm. The training sequence L_{us} is achieved by using vector quantizer upon the codebook C_u for a training sequence L_u

Next-state function design:
For each s and each j, set

$$f(n_j, c_{us}) = \min_j{}' \ d(c_{usj}, c_{ui})$$

Fig. 3 Next-state function design algorithm by omniscient design method.

In the second classification stage we determine $P_{in}(u, i)$ and $P_{fi}(u, i)$ during the training period. These are defined as follows:

$P_{in}(u, i)$ = probability of codeword c_{ui} occurring in the first b frames of word u

$P_{fi}(u, i)$ = probability of codeword c_{ui} occurring in the last b frames of word u.

To test a word in each class, full search over a codebook is done for the first b frames, the last b frames, or both. When full search over C_u is done for the first b frames, A_i, $i = 1, \cdots, b$ the decision criterion $\overline{D}_u$ is obtained as follows:

$$\overline{D}_u \triangleq 1/\left(\frac{1}{b}\sum_{i=1}^{b} D_{ui}\right) + \varepsilon_2 \cdot \sum_{i=1}^{b} P_{in}(u, r_i) \qquad (10)$$

$$r_i \triangleq \min_j{}^{-1}\, d(A_i, c_{uj}), \text{ for } j = 0, 1, \cdots, (K-1) \qquad (11)$$

where ε_2 is a constant. The same method is used for the last b frames. In the case of having full search for the first and last b frames, $\overline{D}_u$'s for each part are added. Then, the input speech is finally recognized as the r_{th} word in the class, if we have

$$D_r = \max_u \overline{D}_u. \qquad (12)$$

IV. SIMULATION RESULTS AND DISCUSSION

Here we present simulation results of the recognition systems described in the preceding section.

The vocabulary used for recognition were ten Korean digits. These were repeated ten times by five male speakers in two different sessions. Table I shows the phonetic symbols of the Korean digits. All of the digits are monosyllable, and thus they are relatively difficult to recognize.

Table I. Phonetic symbols of the Korean digits

0/yoŋ/
1/il/
2/i:/
3/sa:m/
4/sa:/
5/o:/
6/yook/
7/chil/
8/pa:l/
9/gu:/

The input speech was low-pass filtered with 4 kHz, digitized at 10 kHz with 12-bit accuracy, and then preemphasized. Following automatic endpoint detection [9], 10-pole LPC analysis with Hamming window of 30 ms was executed in every 10 ms.

The results were obtained for different codebook sizes (K) and for different numbers (N) of transitions. For comparision, we used as a reference the full search classification which neglects the next-state functions.

Tables II and III show accuracies of the speaker-dependent recognition systems that use the FSVQ and the full search VQ methods, respectively. The reference patterns were generated by using 10 utterances for each word. The recognition accuracies for each speaker were based on the 10 utterances of each word. The recognition accuracies for each speaker were based on the 10 utterances of each word. One can note that the conditional histogram method yields better accuracy than the omniscient design method. In spite of the computational reduction, the speaker-dependent re-

Table II. Accuracy (%) of the speaker-dependent recognition system using the FSVQ method

SPEAKER	CH			OM
	K=8 N=3	K=8 N=4	K=16 N=3	K=16 N=3
A	92	95	100	100
B	98	98	99	100
C	99	100	100	100
D	96	96	98	99
E	95	95	96	91
AVERAGE	96.0	96.8	98.6	98.0

Note: CH – Conditional histogram method
OM – Omniscient design method

Table III. Accuracy (%) of the speaker-dependent recognition system using the full search VQ method

SPEAKER	K=8	K=16
A	95	99
B	95	96
C	100	100
D	88	100
E	100	100
AVERAGE	95.6	99.0

cognition system using the conditional histogram method yields almost the same performance as the full search VQ method. When K=8, the input speech was recognized based on the results of the first classification by the FSVQ or the full search VQ method. When K=16, we applied to the recognition system using the VQ method the second stage classification that is identical to that of the recognition system based on the FSVQ method.

Tables IV and V show accuracies of the speaker-independent recognition systems based on the FSVQ and VQ methods, respectively.

Table IV. Accuracies (%) of the speaker-independent recognition system using the conditional histogram method for different codebook size K.

(a) K=16

N	4	5	6	UNCON-STRAINED
AN / SPEAKER	3.7	4.3	4.7	5.2
A	90	92	91	92
B	96	95	97	98
C	90	95	95	94
D	91	91	94	94
E	91	90	90	90
AVERAGE	91.6	92.6	93.4	93.6

(b) K=32

N	4	5	6	UNCON-STRAINED
AN / SPEAKER	3.7	4.1	4.5	5.0
A	88	81	87	86
B	97	96	95	96
C	86	88	91	91
D	97	98	96	96
E	99	99	97	97
AVERAGE	93.4	92.4	93.2	93.2

Note: AN – Average number of transitions
"Unconstrained" means all transitions for the next states occurring in the training process.

Table V. Accuracy (%) of the speaker-independent recognition system using the full search VQ method

SPEAKER	K=16	K=32
A	93	96
B	98	97
C	93	93
D	87	95
E	90	100
AVERAGE	92.2	96.2

One can note that, when K=16, the conditional histogram method yields better performance than the full search VQ method. In the case of having 16 codewords, the average search number was 5.2 when we had the best performance. But, the conditional histogram method has a shortcoming that the performance does not get improved as the number of codewords increases.

The average accuracy of the speaker-independent recognition system using the omniscient design method was 91.0% when the codebook rate was 4 and the maximum number of transitions was 5.

Each of the above recognition algorithms can be processed in real time if it is implemented in a signal processor chip such as TMS 32010 or 32020.

V. CONCLUSION

We have studied isolated word recognition based on the FSVQ method. This approach appears promising in that it reduces classification time up to about one fifth as compared to the system using the single-section VQ method, yet yields almost the same performance. It has been found that the performance of the recognition system is affected significantly by the accuracy of a next-state function. We believe that there is a room to improve the accuracy of the speaker-independent recognition system. This may be achieved by using the multisection FSVQ method and by improving the accuracy of a next-state function. We are presently looking into these aspects, and expect to report further results in the near future.

REFERENCES

1. D.K. Burton, J.E. Shore, and J.T. Buck, "Isolated-word speech recognition using multisection vector quantization codebooks," IEEE Trans. Acoust., Speech, Signal Processing, vol. ASSP-33, pp. 837-849, Aug. 1985.
2. M.A. Bush, G.E. Kopec, and N. Lauritzen, "Segmentation in isolated word recognition using vector quantization," in Proc. Int. Conf. on Acoust., Speech, Signal Processing, San Diego, March 1984, vol. 2, pp. 17.11.1-17.11.4.
3. J. Foster, R.M. Gray, and M.O. Dunham, "Finite-state vector quantization for waveform coding," IEEE Trans. Inform. Theory, vol. IT-31, pp. 348-359, May 1985.
4. R.M. Gray, "Vector quantization," IEEE ASSP Magazine, vol. 1, pp. 429, Apr. 1984.
5. L.R. Rabiner, S.E. Levinson, and M.M. Sondhi, "On the application of vector quantization and hidden Markov models to speaker-independent isolated word recognition," Bell Sys. Tech. J. vol. 62, No. 4, pp. 1075-1105, Apr. 1983.
6. Y. Linde, A. Buzo and R.M. Gray, "An algorithm for vector quantization," IEEE Trans. Commun., vol. COM-28, pp. 84-95, Jan. 1980.
7. R.M. Gray, A. Buzo, A.H. Gray, Jr. and Y. Matsuyama, "Distortion measures for speech procesing," IEEE Trans. Acoust., Speech, Signal Processing, vol. ASSP-28, pp. 367–376, Aug. 1980.
8. J.T. Tou and R.C. Gonzalez, *Pattern Recognition Principles*, Addison-Wesley, Reading, MA, 1974.
9. L.R. Rabiner and M.R. Sambur, "An algorithm for determing the endpoints of isolated utterance," Bell Syst. Tech. J., vol. 54, pp. 297-315, Feb. 1975.