6th International Conference on Spoken Language Processing (ICSLP 2000)
Beijing, China
October 16-20, 2000

A Novel Search Algorithm for LSF VQ

Jinyu Li   Xin Luo   Ren-Hua Wang
Department of Electronic Engineering and Information Science
University of Science and Technology of China Hefei
rhw@ustc.edu.cn

ABSTRACT

Because classical fast vector quantization(VQ) algorithms can't be used in the LSF vector quantizers that use varying weighted Euclidean distance, a novel fast VQ search algorithm —CRVQ-CS (Constrained Range Vector Quantization based on Component Searching) is presented in this paper. The CRVQ-CS algorithm works well with the varying weighted Euclidean distance and yields the same result as full search VQ with reduced computational complexity does. Although the CRVQ-CS algorithm is proposed for VQ using varying weighted Euclidean distance measure, it is also suitable for VQ using simple Euclidean distance measure.

1. INTRODUCTION

In many speech coding systems, LPC coefficients are transformed to the line spectrum frequency (LSF) parameters which are very effective representation for the quantization of the LPC information. F.K.Soong and B.H.Juang [1] proved that LSF parameters have very good performance in terms of quantization and interpolation. So, LSF parameters are widely used in speech coding systems.

K.K.Paliwal and B.S.Atal [2] proposed a split-vector approach, where the LSF vector is split into two parts and each part is separately vector quantized with different quantizers.

But the heavy complexity constrains the usage of vector quantization. So, there is great need to use fast search algorithm to decrease the search complexity while maintaining the quality. Many fast search algorithms, such as multi-stage VQ and tree-search VQ, have been used to vector quantize the LSF parameters. But they increase the distortion while reducing the search complexity. It is desired to have a fast search algorithm that reduces the search complexity while getting the same result as the original full search algorithm. There are many fast search algorithms of vector quantization that can achieve this goal. R.L.Joshi and P.G.Poonacha [3] proposed a near-neighbor search algorithm to reduce the search complexity in terms of near neighbors. C.M.Huang, Q.Bi, G.S.Stiles and R.W.Harris [4] proposed three fast search algorithms when the distance measure could satisfy the triangle inequality and be previously sorted. But all these algorithms can't be used in the LSF vector quantizers that use varying weighted Euclidean distance.

2. LSF VECTOR QUANTIZATION

K.K.Paliwal and B.S.Atal [2] proposed a varying weighted Euclidean distance measure in the LSF domain which tries to assign weights to individual LSFs according to their spectral sensitivity. The varying weighted Euclidean distance measure $d\left(f,\hat{f}\right)$ between the input LSF vector f and the reference vector $\hat{f}$ is given by $d\left(f,\hat{f}\right)=\sum_{i=1}^{K}\left[v_i\left(f_i-\hat{f}_i\right)\right]^2$, where f_i and $\hat{f}_i$ are the i-th LSF components in the input and reference vector, and v_i is the weight assigned to the i-th LSF component.

The classical fast search algorithms in [3][4] use the character that every vector has its near neighbor and the distance between each pair of reference vectors can be previously computed and arranged in order. According to the near neighbors previously computed, fast search algorithms can be performed. The near neighbors are computed only once before searching.

But these algorithms can't be used with the varying weighted

Euclidean distance. Because the weight v_i is not fixed and changes from frame-to-frame, the predetermined near neighbors also change from frame-to-frame. Because the computational cost of near neighbors is heavy, they can only be computed once before searching. In the course of searching, it is impossible to compute the near neighbors in every frame, so the classical fast search algorithms based on predetermined near neighbors are invalid with the varying weighted Euclidean distance.

3. CONSTRAINED RANGE VECTOR QUANTIZATION BASED ON COMPONENT SEARCHING

We propose here a new fast search algorithm for LSF vector quantization—CRVQ-CS (Constrained Range Vector Quantization based on Component Searching). CRVQ-CS can work well with the varying weighted Euclidean distance and yield the same results as that of full search with reduced computational complexity.

Considering $d\left(f,\hat{f}\right)=\sum_{i=1}^{K}\left[v_i\left(f_i-\hat{f}_i\right)\right]^2$, it is a varying weighted Euclidean distance in K-dimensional space. We define $C=\left\{\hat{f}^{\,j},1\le j\le N\right\}$ as a codebook of size N where $\hat{f}^{\,j}=\left(\hat{f}_1{}^j,\hat{f}_2{}^j,\mathrm{K}\ ,\hat{f}_k{}^j\right)$ is a K-dimensional vector. The steps of CRVQ-CS are:

1) Sort the K-dimension codebook in every dimension in an ascending order, get the order information and store it in the index matrix $Index=\left\{ind_i^j,1\le j\le N,1\le i\le K\right\}$. Because the $Index$ stores the order information of every dimension, we can get the following relationship
$$\hat{f}_i^{ind_i^1}\le\hat{f}_i^{ind_i^2}\le\Lambda\ \Lambda\ \le\hat{f}_i^{ind_i^j}\le\hat{f}_i^{ind_i^{j+1}}$$
$$\le\Lambda\ \Lambda\ \le\hat{f}_i^{ind_i^N},1\le i\le K,1\le ind_i^j\le N$$
.

2) Set $COUNT=\left\{count^j,1\le j\le N\right\}$ to indicate the count of the j-th reference vector in the following steps. Clear $count^j,1\le j\le N$ to be zero.

3) If the K-dimensional input vector is $f=\left(f_1,f_{2,\mathrm{K}}\ ,f_K\right)$. with the method of half finding, we can easily find the index lp_i and up_i, $1\le i\le K$. In every dimension the following relationship is satisfied:
$\hat{f}_i^{ind_i^{lp_i}}\le f_i\le\hat{f}_i^{ind_i^{up_i}},1\le i\le K,up_i=lp_i+1$. That is to say $\hat{f}_i^{ind_i^{lp_i}}$ and $\hat{f}_i^{ind_i^{up_i}}$ are the closest components (lower bound and upper bound) to f_i in the i-th dimension.

4) In every dimension $1\le i\le K$, increase $count^{ind_i^{lp_i}}$ and $count^{ind_i^{up_i}}$ by 1. If $count^{ind_i^{lp_i}}$ or $count^{ind_i^{up_i}}$ is equal to K, go to step 6).

5) Decrease lp_i by 1 and increasing up_i by 1, go to step 4).

6) Assuming $count^l=K$, we can get an initial reference vector $\hat{f}^{\,l}$. The reference vector is the first reference vector whose total index in all dimensions is closest to the input vector.

7) After getting the initial reference vector $\hat{f}^{\,l}$, we can determine the constrained range with this initial reference vector $\hat{f}^{\,l}$ and the input vector f. First, Clear $count^j,1\le j\le N$ to be zero.

Then in every dimension $1\le i\le K$, if $\hat{f}_i{}^l\le f_i$, set the count of the reference vectors whose i-th component is in the range $\left[\hat{f}_i{}^l,2f_i-\hat{f}_i{}^l\right]$ to be 1; if $\hat{f}_i{}^l>f_i$, set the count of the reference vectors whose i-th component is in the range $\left[2f_i-\hat{f}_i{}^l,\hat{f}_i{}^l\right]$ to be 1. The reference vectors in these ranges are all possible to be the best vector that is closest to the input vector. With the index matrix $Index$ we can easily determine the ranges.

8) According to Step 4), 5), 6) and 7), the constrained range is determined. The count of the reference vectors in the

constrained range is equal to 1. Compute the weighted Euclidean distance between the input vector and the reference vectors in the constrained range. The reference vector $\hat{f}^n$ that has the minimum weighted Euclidean distance to the input vector is selected as the quantization result.

4. DISCUSSION AND EXPERIMENT OF CRVQ-CS

The idea of CRVQ-CS is to find a constrained range of reference vectors in the codebook. Only the reference vectors in the constrained range are searched. The constrained range is determined according to step 4), 5), 6) and 7). Only the reference vectors in the constrained range can be the candidates of the best vector that is closest to the input vector.

The operation of step 1) is previously done. The matrix $Index$ is determined according to the codebook, and will not change with the varying weight v_i.

Step 2) initializes the counter of every reference vector, step 3) finds the upper and lower boundary of the input vector in every dimension using the method of half finding. Both step 2) and 3) have very small computational cost.

Step 4) 5) 6) and 7) determine the constrained range. After these steps, assuming the counter of the reference vector $\hat{f}^m$ is zero, it is obvious that the relationship between the initial reference vector $\hat{f}^l$ and $\hat{f}^m$ is: $\left|f_i - \hat{f}_i^l\right| \le \left|f_i - \hat{f}_i^m\right|, 1 \le i \le K$. Then the relationship of the weighted Euclidean distance is:

$$d\left(f, \hat{f}^l\right) = \sum_{i=1}^{i=K}\left[v_i\left(f_i - \hat{f}_i^l\right)\right]^2$$

$$\le \sum_{i=1}^{i=K}\left[v_i\left(f_i - \hat{f}_i^m\right)\right]^2 = d\left(f, \hat{f}^m\right)$$. So, if the count of a reference vector is zero, it must not be the best vector that is closest to the input vector in the case of weighted Euclidean distance. Through these steps, the constrained range is determined. In the constrained range, the count of reference vectors is 1.

Step 8) computes and compares weighted Euclidean distance between the input and each reference vector in the constrained range. The computational cost of searching the constrained range in step 8) is the majority of the full computational cost of CRVQ-CS. Because $\hat{f}^l$ is the first reference vector that terminates the loop of step 4) and 5), it is obvious that the constrained range is much smaller than the whole codebook. In addition, it has been proved that any reference vector outside the constrained range won't be the best vector that is closest to the input vector. So, CRVQ-CS can yield the same result as that of the full search VQ with reduced computational complexity.

Although the weight v_i changes from frame-to-frame, the order of the reference vectors' i-th component doesn't change. That is to say, the matrix $Index$ doesn't change with the varying weight v_i. The algorithm of determining the constrained range is based on the component order matrix $Index$ and the codebook, so the determination is not affected by the varying weight v_i. According to this, CRVQ-CS can work well with the varying weighted Euclidean distance measure.

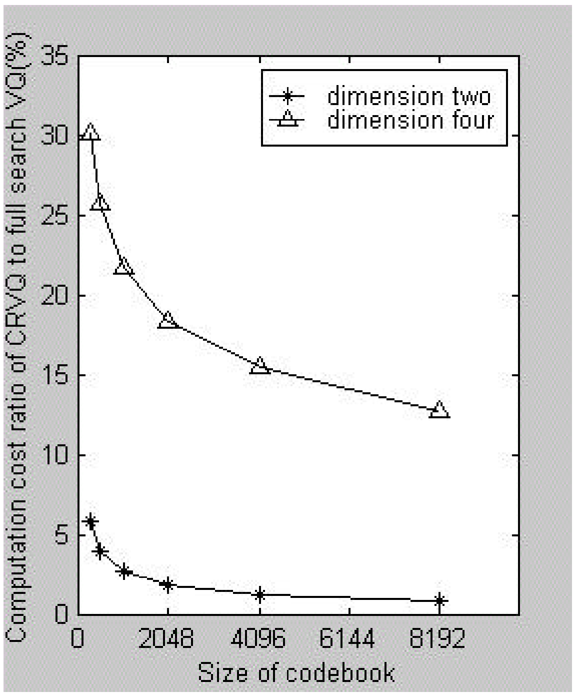

figure 1. CRVQ-CS performance with codebook size changing

Experiments have been done on CRVQ-CS performance with changing codebook size and changing vector dimension. We figure out the computational cost ratio of CRVQ-CS to full search

VQ. In figure 1, dimension K is fixed to be 2 and 4, and the computational cost ratio changes along with the change of codebook size. In figure 2, codebook size N is fixed to be 4096 and 8192, and the computational cost ratio changes along with the change of vector dimension. From the figures, we know that the performance of CRVQ-CS is perfect when the codebook size is large or vector dimension is small. But when the codebook size becomes smaller or vector dimension becomes larger, the performance of CRVQ-CS decreases.

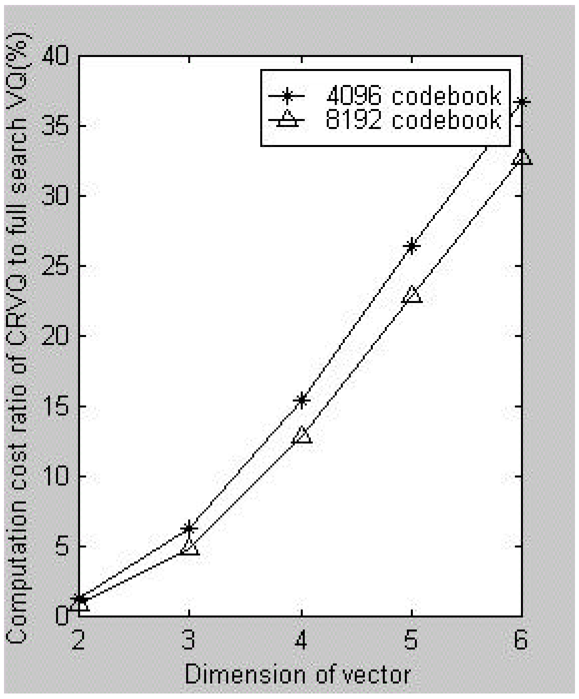

figure 2. CRVQ-CS performance with vector dimension changing

5. CONCLUSION

Because classical fast vector quantization(VQ) algorithms can't be used in the LSF vector quantizers that adopt varying weighted Euclidean distance, a novel fast VQ search algorithm —CRVQ-CS is presented in this paper. Based on the order of the components of the reference vectors in every dimension, CRVQ-CS determines the constrained range. It is proved that only the reference vectors in the constrained range can be the best vector that is closest to the input vector in the case of weighted Euclidean distance. Because the number of the reference vectors in the constrained range is much smaller than the codebook size, the computational cost of CRVQ-CS is much smaller than that of full search VQ. So, CRVQ-CS yields the same result as that of full search VQ with reduced computational complexity.

Because the determination of the constrained range is not affected by the change of weight, the CRVQ-CS algorithm works well with the varying weighted Euclidean distance. According to the experiments, the performance of CRVQ-CS is perfect when the codebook size is large or vector dimension is small. But when the codebook size becomes smaller or vector dimension becomes larger, the performance of CRVQ-CS decreases.

Although the CRVQ-CS algorithm is proposed for VQ using varying weighted Euclidean distance measure, it is also suitable for VQ using simple Euclidean distance measure.

REFERENCES:

[1] F.K.Soong and B.H.Juang, "Line spectral pair and speech data compression", in Proc. Int. Conf. Acoust., Speech, Signal Processing, 1984, pp.1.10.1-1.10.4

[2] K.K.Paliwal and B.S.Atal, "Efficient vector quantization of LPC parameters at 24 bits/frame", IEEE Transactions on Speech and Audio Processing, vol 1, no 1, Jan, 1993. pp3-14

[3] R.L.Joshi and P.G.Poonacha, "A new MMSE encoding algorithm for vector quantization", in Proc. Int. Conf. Acoust., Speech, Signal Processing, 1991, pp645-648

[4] C.M.Huang, Q.Bi, G.S.Stiles and R.W.Harris, "Fast full search equivalent encoding algorithms for image compression using vector quantization", IEEE Transactions on Image Processing, Vol. 1, No. 3, July 1992, pp412-416