EURASIP Journal on Applied Signal Processing 2004:17, 2601–2613

Vector Quantization of Harmonic Magnitudes in Speech Coding Applications—A Survey and New Technique

Wai C. Chu

Media Laboratory, DoCoMo Communications Laboratories USA, 181 Metro Drive, Suite 300, San Jose, CA 95110, USA
Email: wai@docomolabs-usa.com

Received 29 October 2003; Revised 2 June 2004; Recommended for Publication by Bastiaan Kleijn

A harmonic coder extracts the harmonic components of a signal and represents them efficiently using a few parameters. The principles of harmonic coding have become quite successful and several standardized speech and audio coders are based on it. One of the key issues in harmonic coder design is in the quantization of harmonic magnitudes, where many propositions have appeared in the literature. The objective of this paper is to provide a survey of the various techniques that have appeared in the literature for vector quantization of harmonic magnitudes, with emphasis on those adopted by the major speech coding standards; these include constant magnitude approximation, partial quantization, dimension conversion, and variable-dimension vector quantization (VDVQ). In addition, a refined VDVQ technique is proposed where experimental data are provided to demonstrate its effectiveness.

Keywords and phrases: harmonic magnitude, vector quantization, speech coding, variable-dimension vector quantization, spectral distortion.

1. INTRODUCTION

A signal is said to be *harmonic* if it is generated by a series of sine waves or harmonic components where the frequency of each component is an integer multiple of some fundamental frequency. Many signals in nature—including certain classes of speech and music—obey the harmonic model and can be specified by three sets of parameters: fundamental frequency, magnitude of each harmonic component, and phase of each harmonic component. In practice, a noise model is often used in addition to the harmonic model to yield a high-quality representation of the signal. One of the fundamental issues in the incorporation of harmonic modeling in coding applications lies in the quantization of the magnitudes of the harmonic components, or *harmonic magnitudes*; many techniques have been developed for this purpose and are the subjects of this paper.

The term *harmonic coding* was probably first introduced by Almeida and Tribolet [1], where a speech coder operating at a bit rate of 4.8 kbps is described. For the purpose of this paper we define a *harmonic coder* as any coding scheme that explicitly transmits the fundamental frequency and harmonic magnitudes as part of the encoded bit stream. We use the term *harmonic analysis* to signify the procedure in which the fundamental frequency and harmonic magnitudes are extracted from a given signal.

As explained previously, in addition to the harmonic magnitudes, two additional sets of parameters are needed to complete the model. Encoding of fundamental frequency is straightforward and in speech coding it is often the period that is being transmitted with uniform quantization. Uniform quantization for the period is equivalent to nonuniform quantization of the frequency, with higher resolution for low frequency values; this approach is advantageous from a perceptual perspective, since sensitivity toward frequency deviation is higher in the low-frequency region. Phase information is often discarded in low bit rate speech coding, since sensitivity of the human auditory system toward phase distortions is relatively low. Note that in practical deployment of coding systems, a gain quantity is transmitted as part of the bit stream, and is used to scale the quantized harmonic magnitudes to the target power levels.

The harmonic model is an attractive solution to many signal coding applications, with the objective being an economical representation of the underlying signal. Figure 1 shows two popular configurations for harmonic coding, with the difference being the signal subjected to harmonic analysis, which can be the input signal or the excitation signal obtained by inverse filtering through a linear prediction (LP) analysis filter [2, page 21]; the latter configuration has the advantage that the variance of the harmonic magnitudes is greatly reduced after inverse filtering, leading to more efficient quantization. Once the fundamental frequency and the harmonic magnitudes are found, they are quantized and grouped together to form the encoded bit stream. In the present work we consider exclusively the configuration of

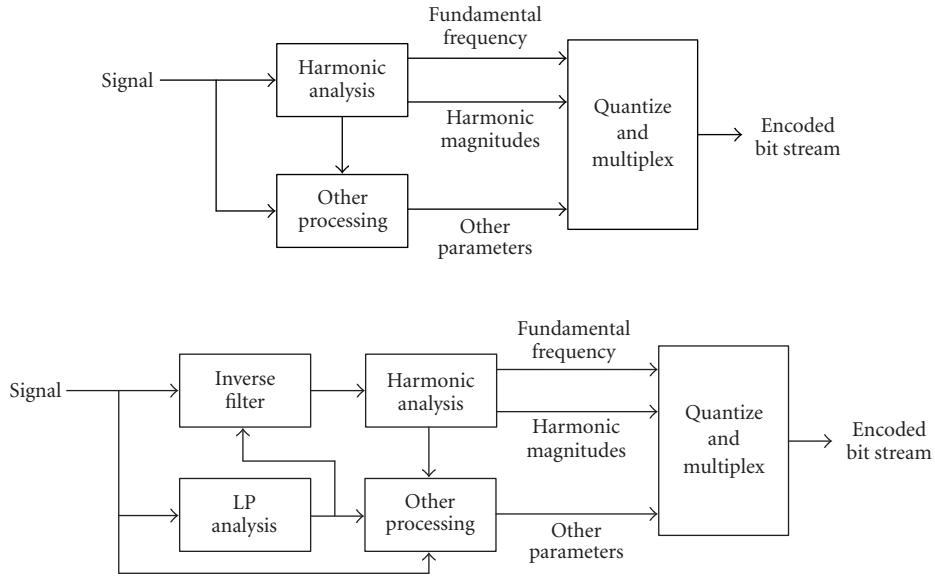

Figure 1: Block diagrams of harmonic encoders, with harmonic analysis of the signal (top), and with harmonic analysis of the excitation (bottom).

harmonic analysis of the excitation, since it has achieved remarkable success and is adopted by various speech coding standards.

In a typical harmonic coding scheme, LP analysis is performed on a frame-by-frame basis; the prediction-error (excitation) signal is computed, which is windowed and converted to the frequency domain via fast Fourier transform (FFT). An estimate of the fundamental period T (measured in number of samples) is found either from the input signal or the prediction error and is used to locate the magnitude peaks in the frequency domain; this leads to the sequence

$$x_j, \quad j = 1, 2, \ldots, N(T), \tag{1}$$

containing the magnitude of each harmonic; we further assume that the magnitudes are expressed in dB; $N(T)$ is the number of harmonics given by

$$N(T) = \left\lfloor \alpha \cdot \frac{T}{2} \right\rfloor \tag{2}$$

with $\lfloor \cdot \rfloor$ denoting the floor function (returning the largest integer that is lower than the operand) and α a constant that is sometimes selected to be slightly lower than one so that the harmonic component at $\omega = \pi$ is excluded. The corresponding frequency values for each harmonic component are

$$\omega_j = \frac{2\pi j}{T}, \quad j = 1, 2, \ldots, N(T). \tag{3}$$

As we can see from (2), $N(T)$ depends on the fundamental period; a typical range of T for the coding of narrowband speech (8 kHz sampling) is 20 to 147 samples (or 2.5 to 18.4 milliseconds) encodable with 7 bits, leading to $N(T) \in [9, 69]$ when $\alpha = 0.95$.

Various approaches can be deployed for the quantization of the harmonic magnitude sequence (1). Scalar quantization, for instance, quantizes each element individually; however, vector quantization (VQ) is the preferred approach for modern low bit rate speech coding algorithms due to improved performance. Traditional VQ designs are targeted at fixed-dimension vectors [3]. More recently researchers have looked into variable dimension designs, where many interesting variations exist.

Harmonic modeling has exerted a great deal of influence in the development of speech/audio coding algorithms. The federal standard linear prediction coding (LPC or LPC10) algorithm [4], for instance, is a crude harmonic coder where all harmonic magnitudes are equal for a certain frame; the federal standard mixed excitation linear prediction (MELP) algorithm [5, 6], on the other hand, uses VQ where only the first ten harmonic magnitudes are quantized; the MPEG4 harmonic vector excitation coding (HVXC) algorithm uses an interpolation-based dimension conversion method where the variable-dimension vectors are converted to fixed dimension before quantization [7, 8]. The previously described algorithms are all standardized coders for speech. An example for audio coding is the MPEG4 harmonic and individual lines plus noise (HILN) algorithm, where the transfer function of a variable-order all-pole system is used to capture the spectral envelope defining the harmonic magnitudes [9].

In this work a review of the existent techniques for harmonic magnitude VQ is given, with emphasis on those adopted by standardized coders. Weaknesses of past methodologies are analyzed and the variable-dimension vector quantization (VDVQ) scheme developed by Das et al. [10] is described. A novel VDVQ configuration that is advantageous

for low codebook dimension is introduced and is based on interpolating the elements of the quantization codebook. The rest of the paper is organized as follows: Section 2 reviews existent spectral magnitudes quantization techniques; Section 3 describes the foundation of VDVQ; Section 4 proposes a new technique called interpolated VDVQ or IVDVQ, with the experimental results given in Section 5, some conclusion is included in Section 6.

2. APPROACHES TO HARMONIC MAGNITUDE QUANTIZATION

This section contains a review of the major approaches for harmonic magnitude quantization found in the literature. Throughout this work we rely on the widely used spectral distortion (SD) measure [2, page 443] as a performance indicator for the quantizers, modified to the case of harmonics and given by

$$\mathrm{SD} = \sqrt{\frac{1}{N(T)} \sum_{j=1}^{N(T)} (x_j - y_j)^2} \tag{4}$$

with x and y the magnitude sequences to be compared, where the values of the sequences are in dB.

2.1. Constant magnitude approximation

The federal standard version of LPC coder [4] relies on a periodic uniform-amplitude impulse train excitation to the synthesis filter for voiced speech modeling. The Fourier transform of such an excitation is an impulse train with constant magnitude, hence the quantized magnitude sequence, denoted by y, consists of one value:

$$y_j = a, \quad j = 1, 2, \ldots, N(T). \tag{5}$$

It can be shown that

$$a = \frac{1}{N(T)} \sum_{j=1}^{N(T)} x_j \tag{6}$$

minimizes the SD measure (4); that is, the optimal solution is given by the arithmetic mean of the target sequence in log domain. The approach is very simple and in practice provides reasonable quality with low requirement in bit rate and complexity.

A similar approach was used by Almeida and Tribolet [1] in their 4.8 kbps harmonic coder. Even though the coder produced acceptable quality, they found that some degree of magnitude encoding, at the expense of higher bit rate (up to 9.6 kbps), elevated the quality. This is expected since the harmonic magnitudes for the prediction-error signal are in general not constant. Techniques that take into account the nonconstant nature of the magnitude spectrum are described next.

2.2. Quantization to a partial group of harmonic magnitudes

The federal standard version of MELP coder [5, 6] incorporates a ten-dimensional VQ at 8-bit resolution for harmonic magnitudes, where only the first ten harmonic components are transmitted. In the decoder side, voiced excitation is generated based on the quantized harmonic magnitudes and the procedure is capable of reproducing with accuracy the first ten harmonic magnitudes, with the rest having the same constant value. In an ideal situation the quantized sequence is

$$y_j = x_j, \quad j = 1, 2, \ldots, 10, \tag{7}$$

$$y_j = a, \quad j = 11, \ldots, N(T), \tag{8}$$

with the assumption that $N(T) > 10$; according to the model, SD is minimized when

$$a = \frac{1}{N(T) - 10} \sum_{j=11}^{N(T)} x_j. \tag{9}$$

In practice (7) cannot be satisfied due to finite resolution quantization. This approach works best for speech with low pitch period (female and children), since lower distortion is introduced when the number of harmonics is small. For male speech the distortion is proportionately larger due to the higher number of harmonic components. Justification of the approach is that the perceptually important components are often located in the low-frequency region; by transmitting the first ten harmonic magnitudes, the resultant quality should be better than the constant magnitude approximation method used by the LPC coder.

2.3. Variable-to-fixed-dimension conversion via interpolation or linear transform

Converting the variable-dimension vectors to fixed-dimension ones using some form of transformation that preserves the general shape is a simple idea that can be implemented efficiently in practice. The HVXC coder [7, 8, 11, 12] relies on a double-interpolation process, where the input harmonic vector is upsampled by a factor of eight and interpolated to the fixed dimension of the VQ equal to 44. A multistage VQ (MSVQ) having two stages is deployed with 4 bits per stage. Together with a 5-bit gain quantizer, a total of 13 bits are assigned for harmonic magnitudes. After quantization, a similar double-interpolation procedure is applied to the 44-dimensional vector so as to convert it back to the original dimension. The described scheme is used for operation at 2 kbps; enhancements are added to the quantized vector when the bit rate is 4 kbps. A similar interpolation technique is reported in [13], where weighting is introduced during distance computation so as to put more emphasis on the formant regions of the LP synthesis filter.

The general idea of dimension conversion by transforming the input vector to fixed dimension before quantization can be formulated with

$$\mathbf{y} = \mathbf{B}_N \mathbf{x}, \tag{10}$$

where the input vector $\mathbf{x}$ of dimension N is converted to the vector $\mathbf{y}$ of dimension M through the matrix $\mathbf{B}_N$ (of dimension $M \times N$). In general $M \neq N$ and hence the matrix is nonsquare. The transformed vector is quantized with an M-dimensional VQ, resulting in the quantized vector $\hat{y}$, which is mapped back to the original dimension leading to the final quantized vector

$$\hat{\mathbf{x}} = \mathbf{A}_N \hat{\mathbf{y}} \tag{11}$$

with $\mathbf{A}_N$ an $N \times M$ matrix. The described approach is known as nonsquare transform [14, 15], and optimality criterion can be found for the matrices $\mathbf{A}_N$ and $\mathbf{B}_N$. Similar to the case of partial quantization, the method is more suitable for low dimension, since the transformation process introduces losses that are not reversible. However, by elevating M, the losses can be reduced at the expense of higher computational cost. In [16], a harmonic coder operating at 4 kbps is described, where the principle of nonsquare transform is used to design a 14-bit MSVQ; a similar design appears in [17].

In [18] a quantizer is described where the variable-dimension vectors are transformed into a fixed dimension of 48 using discrete cosine transform (DCT), and is quantized using two codebooks: the transform or matrix codebook and the residual codebook, with the quantized vector given by the product of a matrix found from the first codebook and a vector found from the second codebook. A variation of the described scheme is given in [19], which is part of a sinusoidal coder operating at 4 kbps. Another DCT-based quantizer is described in [20]. To take advantage of the correlation between adjacent frames, some quantizers combine dimension conversion within a predictive framework; for instance, see [21].

2.4. Multi-codebook

One possible way to deal with the vectors of varying dimension is to provide separate codebooks for different dimensions; for instance, one codebook per dimension is an effective solution at the expense of elevated storage cost. In [22] an MELP coder operating near 4 kbps is described where the harmonic magnitudes are quantized using a switched predictive MSVQ having four codebooks. Two codebooks are used for magnitude vectors of dimension less than 55, and the other two for magnitude vectors of dimension greater than 45; the ranges overlap so that some spectra are included in both groups. For the two codebooks in each dimension group, one is used for strongly predictive case (current frame is strongly correlated with the past) and the other for weakly predictive case (current frame is weakly correlated with the past). During encoding, a longer vector is truncated to the size of the shorter one; a shorter vector is extended to the required dimension with constant entrants. A total of 22 bits are allocated to the harmonic magnitudes for each 20 millisecond frame.

In [23], vectors of dimension smaller than or equal to 48 are expanded via zero padding according to five different dimensions: 16, 24, 32, 40, and 48; for vectors of dimension larger than 48, it is reduced to 48. One 14-bit two-stage VQ is designed for each of the designated dimensions.

3. VARIABLE-DIMENSION VECTOR QUANTIZATION

VDVQ [10, 24] represents an alternative for harmonic magnitude quantization and has many advantages compared to other techniques. In this section the basic concepts are presented with an exposition of the nomenclatures involved; Section 4 describes a variant of the basic VDVQ scheme. Besides harmonic magnitude quantization, VDVQ has also been applied to other applications; see [25] for quantization of autoregressive sources having a varying number of samples and [26] where techniques for image coding are developed.

3.1. Codebook structure

The codebook of the quantizer contains N_c codevectors:

$$\mathbf{y}_i, \quad i = 0, \ldots, N_c - 1, \tag{12}$$

with

$$\mathbf{y}_i^T = \begin{bmatrix} y_{i,0} & y_{i,1} & \cdots & y_{i,N_v-1} \end{bmatrix}, \tag{13}$$

where N_v is the dimension of the codebook (or codevector). Consider the harmonic magnitude vector $\mathbf{x}$ of dimension $N(T)$ with T being the pitch period; assuming full search, the following distances are computed:

$$d(\mathbf{x}, \mathbf{u}_i), \quad i = 0, \ldots, N_c - 1, \tag{14}$$

where

$$\mathbf{u}_i^T = \begin{bmatrix} u_{i,1} & u_{i,2} & \cdots & u_{i,N(T)} \end{bmatrix}, \tag{15}$$

$$u_{i,j} = y_{i,\mathrm{index}(T,j)}, \quad j = 1, \ldots, N(T), \tag{16}$$

with

$$\begin{aligned} \mathrm{index}(T, j) &= \mathrm{round}\left(\frac{(N_v - 1)\omega_j}{\pi}\right) \\ &= \mathrm{round}\left(\frac{2(N_v - 1)j}{T}\right), \quad j = 1, \ldots, N(T), \end{aligned} \tag{17}$$

where $\mathrm{round}(x)$ converts x to the nearest integer. Figure 2 contains the plot of $\mathrm{index}(T, j)$ as a function of T, with the position of the index marked by a black dot. As we can see, vertical separation between dots shrinks as the period increases.

The scheme works as follows: a vector $\mathbf{u}_i$ having the same dimension as $\mathbf{x}$ is extracted from the codevector $\mathbf{y}_i$ by calculating a set of indices using the associated pitch period. These indices point to the positions of the codevector where elements are extracted. The idea is illustrated in Figure 3 and can be summarized by

$$\mathbf{u}_i = \mathbf{C}(T)\mathbf{y}_i \tag{18}$$

with $\mathbf{C}(T)$ the selection matrix associated with the pitch period T and having the dimension $N(T) \times N_v$. The selection

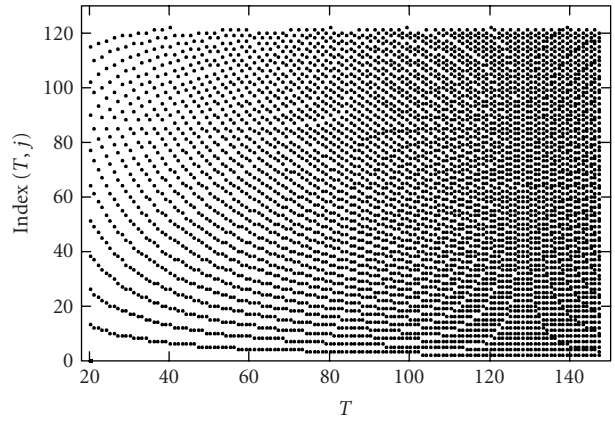

Figure 2: Indices to the codevectors' elements as a function of the pitch period $T \in [20, 147]$ when $N_v = 129$.

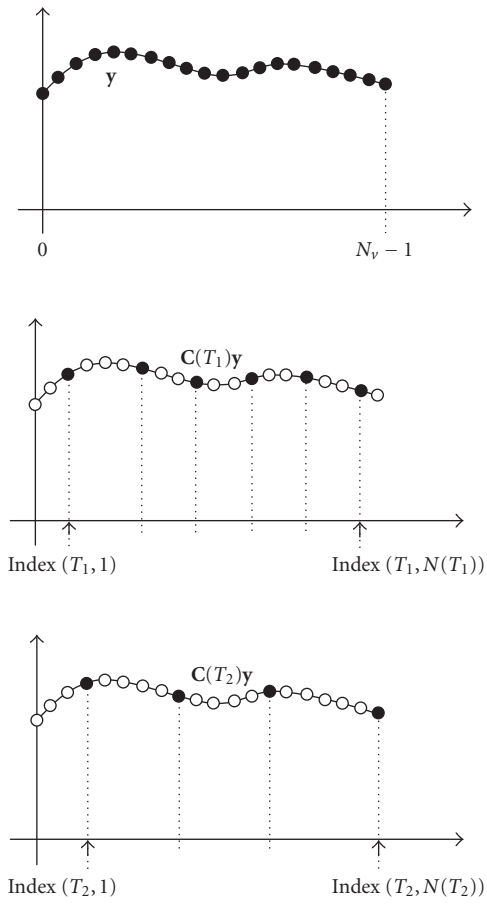

Figure 3: Illustration of codevector sampling: the original codevector with N_v elements (top) and two sampling results where $T_1 > T_2$.

matrix is specified with

$$\mathbf{C}(T) = c_{j,m}^{(T)}\big|_{j=1,\ldots,N(T);\, m=0,\ldots,N_v-1},$$
$$c_{j,m}^{(T)} = \begin{cases} 1 & \text{if index}(T, j) = m, \\ 0 & \text{otherwise.} \end{cases} \tag{19}$$

3.2. Codebook generation

We assume that the set of training data

$$\{\mathbf{x}_k, T_k\}, \quad k = 0, \ldots, N_t - 1, \tag{20}$$

is available, with N_t the size of the training set. Each vector $\mathbf{x}_k$ within the set has a pitch period T_k associated with it, which determines the dimension of the vector. The N_c codevectors divide the whole space into N_c cells. The vector $\mathbf{x}_k$ is said to pertain to the ith cell if

$$d(\mathbf{C}(T_k)\mathbf{y}_i, \mathbf{x}_k) \le d(\mathbf{C}(T_k)\mathbf{y}_j, \mathbf{x}_k) \tag{21}$$

for all $j \neq i$. Thus, given a codebook, we can find the sets

$$\{\mathbf{x}_k, T_k, i_k\}, \quad k = 0, \ldots, N_t - 1, \tag{22}$$

with $i_k \in [0, N_c - 1]$ the index of the cell that $\mathbf{x}_k$ pertains to. The task of obtaining (22) is referred to as *nearest-neighbor search* [3, page 363]. The objective of codebook generation is to minimize the sum of distortion at each cell

$$D_i = \sum_{k|i_k=i} d(\mathbf{x}_k, \mathbf{C}(T_k)\mathbf{y}_i), \quad i = 0, \ldots, N_c - 1, \tag{23}$$

by optimizing the codevector $\mathbf{y}_i$; the process is referred to as *centroid computation*. Nearest-neighbor search together with centroid computation are the key steps of the generalized Lloyd algorithm (GLA [3, page 363]) and can be used to generate the codebook. Depending on the selected distance measure, centroid computation is performed differently. Consider the distance definition

$$d(\mathbf{x}_k, \mathbf{C}(T_k)\mathbf{y}_i) = \left|\left|\mathbf{x}_k - \mathbf{C}(T_k)\mathbf{y}_i + g_{k,i}\mathbf{1}\right|\right|^2. \tag{24}$$

It is assumed in (24) that all elements of the vectors $\mathbf{x}_k$ and $\mathbf{y}_i$ are in dB values, hence (24) is proportional to SD^2 as given in (4). Note that in (24) $\mathbf{1}$ is a vector whose elements are all 1's with dimension $N(T)$. The variable $g_{k,i}$ is referred to as the gain and has to be found for each combination of input vector $\mathbf{x}_k$, pitch period T_k, and codevector $\mathbf{y}_i$. The optimal gain

value can be located by minimizing (24) and can be shown to be

$$\begin{aligned} g_{k,i} &= \frac{1}{N(T_k)}\left(\mathbf{y}_i^T \mathbf{C}(T_k)^T \mathbf{1} - \mathbf{1}^T \mathbf{x}_k\right) \\ &= \frac{1}{N(T_k)} \sum_{j=1}^{N(T_k)} (u_{i,j} - x_{k,j}), \end{aligned} \tag{25}$$

hence it is given by the difference of the means of the two vectors. In practice the gain must be quantized and transmitted so as to generate the final quantized vector. However, in the present study we will focus solely on the effect of shape quantization and will assume that the quantization effect of the gain is negligible, which is approximately true as long as the number of bits allocated for gain representation is sufficiently high. To compute the centroid, we minimize (24) leading to

$$\sum_{k|i_k=i} \Psi(T_k)\mathbf{y}_i = \sum_{k|i_k=i} \left(\mathbf{C}(T_k)^T \mathbf{x}_k + g_{k,i}\mathbf{C}(T_k)^T \mathbf{1}\right) \tag{26}$$

with

$$\Psi(T) = \mathbf{C}(T)^T \mathbf{C}(T) \tag{27}$$

an $N_v \times N_v$ diagonal matrix. Equation (26) can be written as

$$\Phi_i \mathbf{y}_i = \mathbf{v}_i, \tag{28}$$

where

$$\begin{aligned} \Phi_i &= \sum_{k|i_k=i} \Psi(T_k), \\ \mathbf{v}_i &= \sum_{k|i_k=i} \left(\mathbf{C}(T_k)^T \mathbf{x}_k + g_{k,i}\mathbf{C}(T_k)^T \mathbf{1}\right), \end{aligned} \tag{29}$$

hence the centroid is solved using

$$\mathbf{y}_i = \Phi_i^{-1} \mathbf{v}_i. \tag{30}$$

Since Φ_i is a diagonal matrix, its inverse is easy to find. Nevertheless, elements of the main diagonal of Φ_i might contain zeros and occur when some elements of the codevector $\mathbf{y}_i$ are not invoked during training. This could happen, for instance, when the pitch periods of the training vectors pertaining to the cell do not have enough variety, and hence some of the N_v elements of the codevector are not affected during training. In those cases one should avoid the direct use of (30) to find the centroid, but rather use alternative techniques to compute the elements. In our implementation, a reduced-dimension system is solved by eliminating the rows/columns of Φ_i corresponding to zeros in the main diagonal; also, $\mathbf{v}_i$ is trimmed accordingly. Elements of the centroid associated with zeros in the main diagonal of Φ_i are found by interpolating adjacent elements in the resultant vector.

3.3. Competitive training

Given a codebook, it is possible to fine-tune it through the use of competitive training. This is sometimes referred to as learning VQ [27, page 427]. In this technique, only the codevector that is closest to the current training vector is updated; the updating rule is to move the codevector slightly in the direction negative to the distance gradient. The distance gradient is found from (24) to be

$$\begin{aligned} &\frac{\partial}{\partial y_{i,m}} d(\mathbf{x}_k, \mathbf{C}(T_k)\mathbf{y}_i) \\ &\quad = \sum_{j=1}^{N(T_k)} 2(x_{k,j} - u_{i,j} + g_{k,i}) \left(\frac{\partial g_{k,i}}{\partial y_{i,m}} - \frac{\partial u_{i,j}}{\partial y_{i,m}}\right) \end{aligned} \tag{31}$$

for $m = 0, \ldots, N_v - 1$. From (25) we have

$$\frac{\partial g_{k,i}}{\partial y_{i,m}} = \frac{1}{N(T_k)} \sum_{j=1}^{N(T_k)} \frac{\partial u_{i,j}}{\partial y_{i,m}}, \tag{32}$$

and from (16),

$$\frac{\partial u_{i,j}}{\partial y_{i,m}} = \begin{cases} 1 & \text{if } m = \text{index}\,(T_k, j), \\ 0 & \text{otherwise.} \end{cases} \tag{33}$$

By knowing the distance gradient, the selected codevector is updated using

$$y_{i,m} \longleftarrow y_{i,m} - \gamma \frac{\partial}{\partial y_{i,m}} d(\mathbf{x}_k, \mathbf{C}(T_k)\mathbf{y}_i) \tag{34}$$

with γ an experimentally found constant known as the step size parameter which controls the update speed as well as stability. The idea of competitive training is to find a codevector for each training vector, with the resultant codevector updated in the direction negative to the distance gradient. The process is repeated for a number of epochs (one epoch refers to a complete presentation of the training data set). After sufficient training time, the codebook is expected to converge toward a local optimum.

4. INTERPOLATED VDVQ

A new configuration of VDVQ is proposed here, which is based on interpolating the elements of the codebook to obtain the actual codevectors. The VDVQ system described in Section 3 is based on finding the index of the codevectors' elements through (17). Consider an expression for the index where rounding is omitted:

$$\text{index}(T, j) = \frac{2(N_v - 1)\, j}{T}, \quad j = 1, \ldots, N(T). \tag{35}$$

The previous expression contains a fractional part and cannot be used to directly extract the elements of the codevectors; nevertheless, it is possible to use interpolation among the elements of the codevectors when the indices contain a

nonzero fractional part. We propose to use a first-order linear interpolation method where the vector $\mathbf{u}$ in (15) is found using

$$u_{i,j} = \begin{cases} y_{i,\text{index}(T,j)} \\ \text{if } \lceil \text{index}(T,j) \rceil = \lfloor \text{index}(T,j) \rfloor, \\ (\text{index}(T,j) - \lfloor \text{index}(T,j) \rfloor) y_{i,\lceil \text{index}(T,j) \rceil} \\ +(\lceil \text{index}(T,j) \rceil - \text{index}(T,j)) y_{i,\lfloor \text{index}(T,j) \rfloor} \\ \text{otherwise}, \end{cases} \tag{36}$$

that is, interpolation is performed between two elements of the codevector whenever the index contains a nonzero fractional part. The operation can also be captured in matrix form as in (18), with the elements of the matrix given by

$$c_{j,m}^{(T)} = \begin{cases} 1 \\ \text{if } \lceil \text{index}(T,j) \rceil = \lfloor \text{index}(T,j) \rfloor,\ m = \text{index}(T,j), \\ \text{index}(T,j) - \lfloor \text{index}(T,j) \rfloor \\ \text{if } \lceil \text{index}(T,j) \rceil \neq \lfloor \text{index}(T,j) \rfloor,\ \lceil \text{index}(T,j) \rceil = m, \\ \lceil \text{index}(T,j) \rceil - \text{index}(T,j) \\ \text{if } \lceil \text{index}(T,j) \rceil \neq \lfloor \text{index}(T,j) \rfloor,\ \lfloor \text{index}(T,j) \rfloor = m, \\ 0 \\ \text{otherwise}, \end{cases} \tag{37}$$

for $j = 1, \ldots, N(T)$ and $m = 0, \ldots, N_v - 1$. We name the resultant scheme interpolated VDVQ, or IVDVQ. For codebook generation we can rely on the same competitive training method explained previously. The distance gradient is calculated using a similar procedure, with the exception of

$$\frac{\partial u_{i,j}}{\partial y_{i,m}} = \begin{cases} 1 \\ \text{if } \lceil \text{index}(T,j) \rceil = \lfloor \text{index}(T,j) \rfloor,\ m = \text{index}(T,j), \\ \text{index}(T,j) - \lfloor \text{index}(T,j) \rfloor \\ \text{if } \lceil \text{index}(T,j) \rceil \neq \lfloor \text{index}(T,j) \rfloor,\ m = \lceil \text{index}(T,j) \rceil, \\ \lceil \text{index}(T,j) \rceil - \text{index}(T,j) \\ \text{if } \lceil \text{index}(T,j) \rceil \neq \lfloor \text{index}(T,j) \rfloor,\ m = \lfloor \text{index}(T,j) \rfloor, \\ 0 \\ \text{otherwise}, \end{cases} \tag{38}$$

which is derived from (36).

5. EXPERIMENTAL RESULTS

This section summarizes the experimental results in regard to VDVQ as applied to harmonic magnitudes quantization. In order to design the vector quantizers, a set of training data must be obtained. We have selected 360 sentences from the TIMIT database [28] (downsampled to 8 kHz). The sentences are LP-analyzed (tenth order) at 160-sample frames with the prediction error found. An autocorrelation-based pitch period estimation algorithm is deployed. The prediction-error signal is mapped to the frequency domain via 256-point FFT after Hamming windowing. Harmonic magnitudes are extracted only for the voiced frames according to the estimated pitch period, which has the range of [20, 147] at steps of 0.25; thus, fractional values are allowed for the pitch periods. A simple thresholding technique is deployed for voiced/unvoiced classification, in which the normalized autocorrelation value of the prediction error for the frame is found over a range of lag and compared to a fixed threshold; if one of the normalized autocorrelation values is above the threshold, the frame is declared as voiced, otherwise it is declared as unvoiced.

There are approximately 30 000 harmonic magnitude vectors extracted, with the histogram of the pitch periods shown in Figure 4. In the same figure, the histogram is plotted for the testing data set with approximately 4000 vectors obtained from 40 files of the TIMIT database. As we can see, there is a lack of vectors with large pitch periods, which is undesirable for training because many elements in the codebook might not be appropriately tuned. To alleviate this problem, an artificial set is created in the following manner: a new pitch period is generated by scaling the original pitch period by a factor that is greater than one, and a new vector is formed by linearly interpolating the original vector. Figure 5 shows an example where the original pitch period is 31.5 while the new pitch period is 116. Figure 6 shows the histogram of the pitch periods for the final training data set, obtained by combining the extracted vectors with their interpolated versions, leading to a total of 60 000 training vectors.

5.1. VDVQ results

Using the training data set, we designed a total of 30 quantizers at a resolution of $r = 5$ to 10 bits, and codebook dimension $N_v = 41, 51, 76, 101$, and 129. The initial codebooks are populated by random numbers with GLA applied for a total of 100 epochs (one epoch consists of nearest-neighbor search followed by centroid computation). The process is repeated 10 times with different randomly initialized codebooks, and the one associated with the lowest distortion is kept. The codebooks obtained using GLA are further optimized via competitive training. A total of 1000 epochs are applied with the step size set at $\gamma = 0.2N_c/N_t$. This value is selected based on several reasons: first, the larger the training data set, the lower the γ should be, because on average the codevectors receive more update in one pass through the training data set, and the low γ allows the codevectors to converge steadily toward the intended centroids; on the other hand, the larger the codebook, the higher the γ should be, because on average each codevector receives less update in one pass through the training data set, and the higher γ compensates for the reduced update rate; in addition, experimentally, we found that the specified γ produces good balance between speed and quality of the final results. It is observed that by incorporating competitive training, a maximum of 3% reduction in average SD can be achieved.

The average SD results appear in Table 1 and Figure 7. The average SD in training reduces approximately 0.17 dB for one-bit increase in resolution. As we can see, training performance can normally be raised by increasing the codebook dimension; however, the testing performance curves

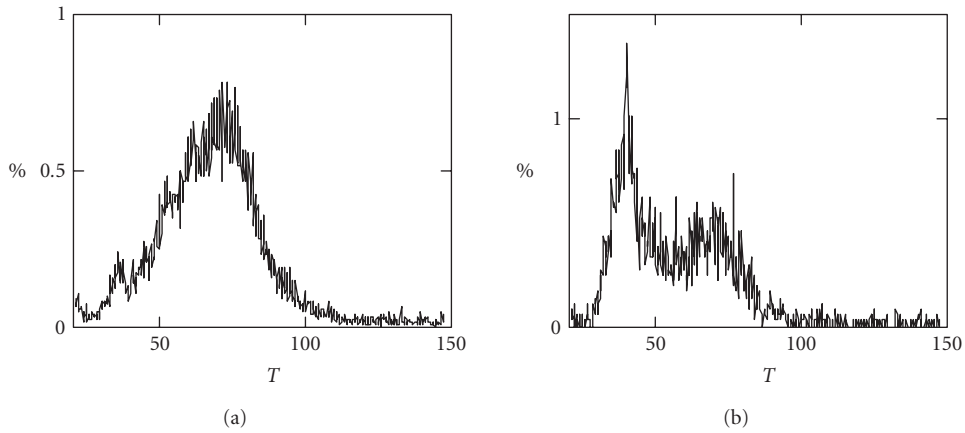

Figure 4: (a) Histogram of the pitch periods (T) for 30 000 harmonic magnitude vectors to be used as training vectors, and (b) the histogram of the pitch periods for 4000 harmonic magnitude vectors to be used as testing vectors.

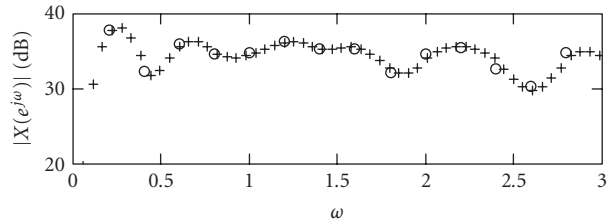

Figure 5: An example of harmonic magnitude interpolation. Original vector (∘) and interpolated version (+).

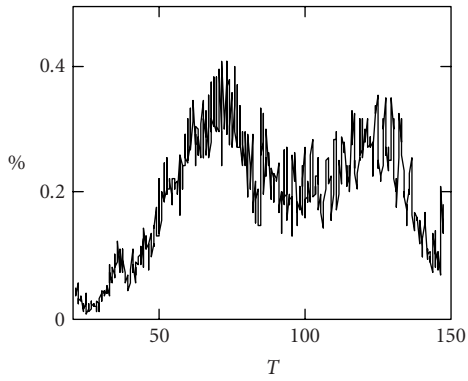

Figure 6: Histogram of the pitch periods (T) for the final training data set.

show a generalization problem when N_v increases. In most cases, increasing N_v beyond 76 leads to a degradation in performance. The phenomenon can be explained from the fact that overfitting happens for higher dimension; that is, the ratio between the number of training data and the number of codebook elements decreases as the codebook dimension increases, leading to overfitting conditions. In the present experiment, the lowest training ratio N_t/N_c is 60 000/1024 = 58.6, and in general can be considered as sufficiently high to achieve good generalization. The problem, however, lies in the structure of VDVQ, which in essence is a multi-codebook encoder, with the various codebooks (each dedicated to one particular pitch period) overlapped with each other. The amount of overlap becomes less as the dimensionality of the codebook (N_v) increases; hence a corresponding increase in the number of training vectors is necessary to achieve good generalization.

How many more training vectors are necessary to achieve good generalization at high codebook dimension? The question is not easy to answer but we can consider the extreme situation where there is one codebook per pitch period, which happens when the codebook dimension grows sufficiently large. Within the context of the current experiment, there are a total of 509 codebooks (a total of 509 pitch periods exist in the present experiment), hence the size of the training data set is approximately 509 times the current size (equal to 60 000). Handling such a vast amount of vectors is complicated and the training time can be excessively long. Thus, if resource is limited and low storage cost is desired, it is recommended to deploy a quantizer with low codebook dimensionality.

5.2. IVDVQ results

The same values of resolution and dimension as for the basic VDVQ are used to design the codebooks for IVDVQ.

Table 1: Average SD in dB for VDVQ as a function of the resolution (r) and the codebook dimension (N_v).

N_v	41		51		76		101		129	
r (bit)	Train	Test	Train	Test	Train	Test	Train	Test	Train	Test
5	3.30	3.33	3.25	3.28	3.17	3.24	3.16	3.23	3.16	3.23
6	3.18	3.23	3.12	3.21	3.05	3.15	3.03	3.15	3.05	3.16
7	3.05	3.18	3.00	3.13	2.91	3.08	2.91	3.08	2.88	3.10
8	2.93	3.11	2.86	3.06	2.76	3.02	2.74	3.03	2.72	3.06
9	2.76	3.06	2.69	3.00	2.59	2.98	2.53	2.98	2.49	3.01
10	2.57	2.99	2.46	2.95	2.31	2.93	2.26	2.92	2.20	2.95

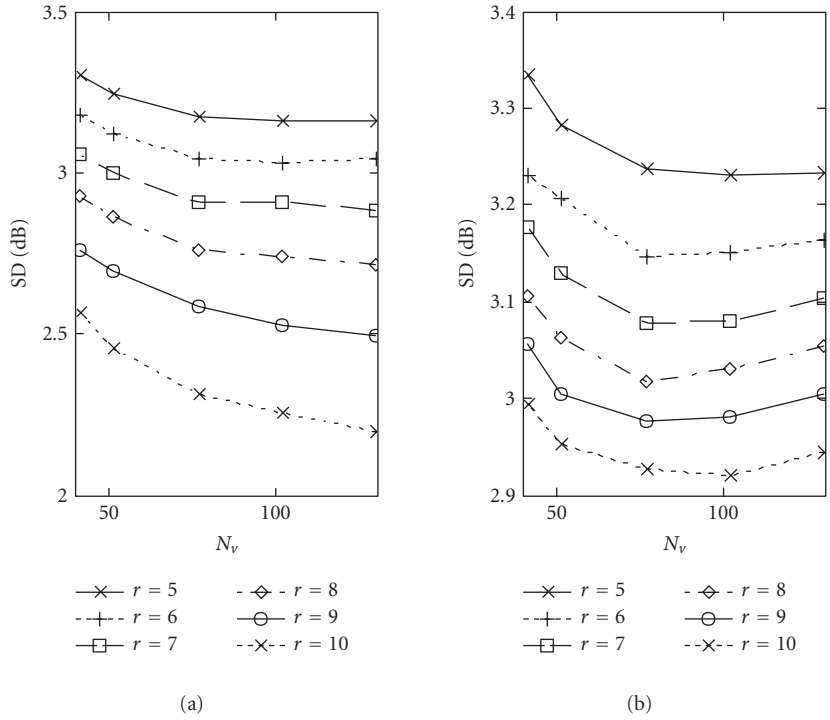

Figure 7: Plots of average spectral distortion (SD) as a function of the resolution (r) and codevector dimension (N_v) in VDVQ: (a) training performance and (b) testing performance.

We follow the competitive training method explained in Section 4, with the initial codebooks taken from the outcomes of the basic VDVQ designs. The average SD results appear in Table 2 and Figure 8. Similar to the case of VDVQ, we see that training performance tends to be superior for higher N_v, which is not true in testing, and is partly due to the lack of training data, as explained previously. Moreover, the problem with generalization tends to be more severe in the present case, and is likely due to the fact that the interpolation involved with IVDVQ allows the codebook to be better tuned toward the training data set. For VDVQ, the errors involved with index rounding make the training process less accurate as compared to IVDVQ; hence overtraining is of a lesser degree.

Figure 9 shows the difference between the SD results found by subtracting the present numbers to those of VDVQ (Table 1). As we can see, by introducing interpolation among the elements of the codevectors, there is always a reduction in average SD for the training data set, and the amount of reduction tends to be higher for low dimension and high resolution. Also for testing, the average SD values for IVDVQ are mostly lower.

The fact that IVDVQ shows the largest gain with respect to VDVQ for low codebook dimension is mainly due to the fact that the impact of index rounding (as in (17)) decreases as the codebook dimension increases. In other words, for larger codebook dimension, the error introduced by index rounding becomes less significant; hence the performance

Table 2: Average SD in dB for IVDVQ as a function of the resolution (r) and the codebook dimension (N_v).

N_v	41		51		76		101		129	
r (bit)	Train	Test	Train	Test	Train	Test	Train	Test	Train	Test
5	3.16	3.22	3.16	3.21	3.16	3.22	3.15	3.21	3.14	3.21
6	3.04	3.13	3.04	3.14	3.04	3.14	3.02	3.12	3.03	3.14
7	2.90	3.07	2.91	3.06	2.89	3.06	2.89	3.06	2.87	3.09
8	2.77	3.00	2.76	3.00	2.74	3.00	2.72	3.02	2.69	3.04
9	2.61	2.96	2.59	2.94	2.56	2.96	2.51	2.97	2.44	3.01
10	2.39	2.90	2.34	2.91	2.27	2.92	2.18	2.94	2.08	2.97

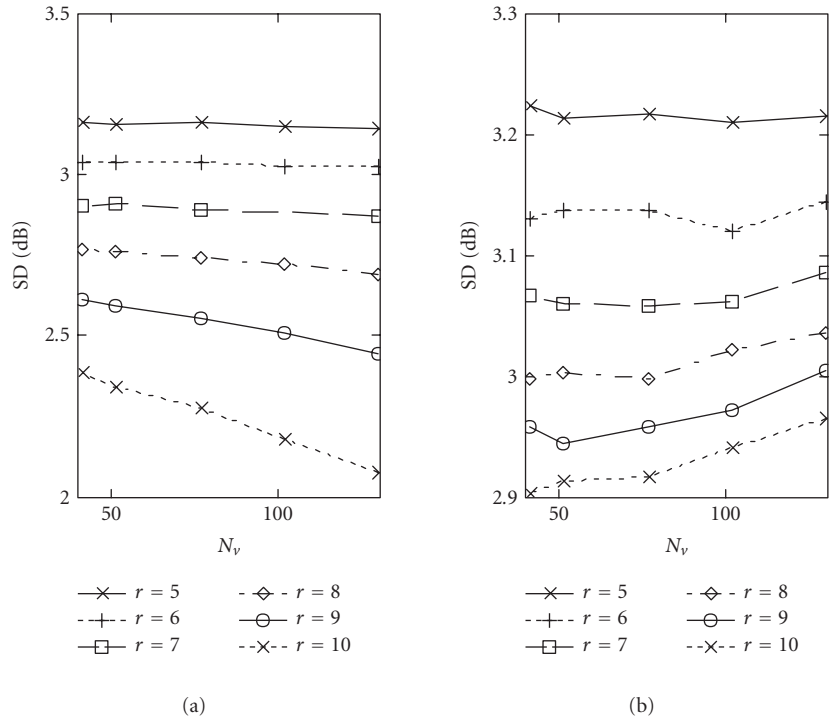

Figure 8: Plots of average spectral distortion (SD) as a function of the resolution (r) and codevector dimension (N_v) in IVDVQ: (a) training performance and (b) testing performance.

difference between VDVQ and IVDVQ tends to be closer. To quantify the rounding effect, we can rely on a signal-to-noise ratio defined by

$$\mathrm{SNR}\,(N_v) = 10\log\left(\frac{\sum_n \sum_j \left(\mathrm{index}\,(T_n, j)\right)^2}{\sum_n \sum_j \left(\mathrm{index}\,(T_n, j) - \mathrm{round}\left(\mathrm{index}\,(T_n, j)\right)\right)^2}\right) \tag{39}$$

with index$(\cdot)$ given in (35). The range of the pitch periods is $T_n = 20, 20.25, 20.50, \ldots, 147$ and has a total of 509 values; the index j ranges from 1 to $N(T_n)$. Table 3 summarizes the values of SNR, where we can see that SNR(129) is significantly higher than SNR(41); therefore at $N_v = 129$, the effect of index rounding is much lower than, for instance, at $N_v = 41$; hence the benefits of interpolating the elements of the codebook vanish as the dimension increases.

5.3. Comparison with techniques from standardized coders

In order to compare the various techniques described in this paper, we implemented some of the schemes explained in Section 2 and measured their performance. For LPC we used (5) and (6) to measure the average SD and the results are 4.43 dB in training and 4.37 dB in testing. For MELP we used (7), (8), and (9); the average SD results are 3.32 dB in training and 3.31 dB in testing. Notice that these results are obtained assuming that no quantization is involved, that is, resolution is infinite. We conclude that MELP is indeed superior to the constant magnitude approximation method of the LPC coder.

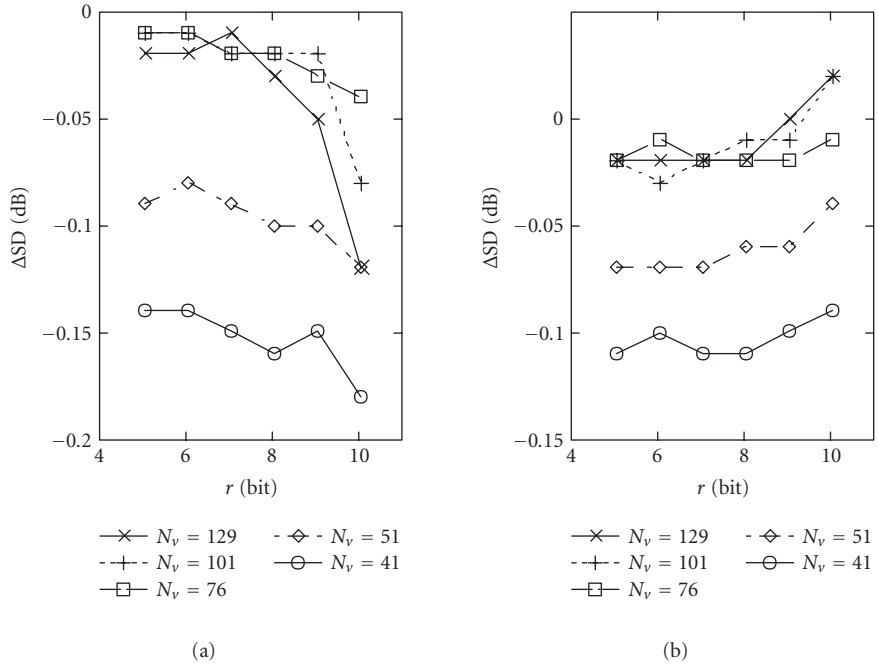

Figure 9: Difference in average spectral distortion (Δ SD) obtained by subtracting the SD results of IVDVQ from those of the basic VDVQ as a function of the resolution (r) and codevector dimension (N_v): (a) training data and (b) testing data.

Table 3: Signal-to-noise ratio related to index rounding as a function of the codebook dimension N_v.

N_v	41	51	76	101	129
SNR(N_v)	37.7	39.6	43.1	45.6	47.8

We also implemented the HVXC-dimension conversion method. Without quantization, average SD is 0.352 dB in training and 0.331 dB in testing. Therefore, HVXC is much better than both LPC and MELP. To measure its performance under quantization, we designed full search vector quantizers of dimension equal to 44 of 5-to-10 bit resolution. To accomplish this task, the variable-dimension training vectors are interpolated to 44 elements, which are used to train the quantizers where GLA is applied; ten random initializations are performed with each followed by 100 epochs of training; at the end only the best codebook is kept.

Plots of average SD for the discussed schemes appear in Figure 10, where we can see that at $N_v = 41$, the VDVQ schemes compare favorably with respect to other schemes. This value of dimension is slightly lower than the value of 44 for the HVXC coder. Notice that performance curves for LPC and MELP are plotted for reference only; they represent the best possible outcomes under the constraints set forth by each coding algorithm, reachable only at infinite resolution. It is also clear that IVDVQ is the best performing scheme, and from the testing performance curves we can see that deployment of IVDVQ leads to a saving of 1-bit resolution when compared to VDVQ. Nevertheless, advantage of IVDVQ is obtainable only for low codebook dimension (equal to 41 in the present case). For higher codebook dimension, the advantage of using IVDVQ vanishes, and both IVDVQ and VDVQ perform at a similar level.

6. CONCLUSION

A review of several techniques available for the quantization of harmonic magnitudes is given in this paper. The technique of VDVQ is studied, and an enhanced version is proposed where interpolation is performed among the elements of the codebook. It is shown through experimental data that they compare favorably to the schemes adopted by established standards. The present study has focused on the design of the shape codebook, and has assumed that the associated gain is quantized with infinite resolution; it is important to note that in practice, the gain quantization codebook must be jointly designed with the shape codebook, see [3, page 443] for codebook design algorithms of shape-gain VQs.

The advantage of VDVQ comes from the fact that no transformation is introduced to the input vector prior to distance computation during encoding. For the two configurations of VDVQ studied, the codebook of the quantizer is adjusted so as to reproduce as accurate as possible a given input vector. This is in sharp contrast to other propositions where the input vector is subjected to some sort of modification or transformation prior to encoding, such as partial quantization in the MELP standard and interpolation for dimension conversion for the HVXC coder; this step introduces a nonzero distortion that cannot be reduced even when the

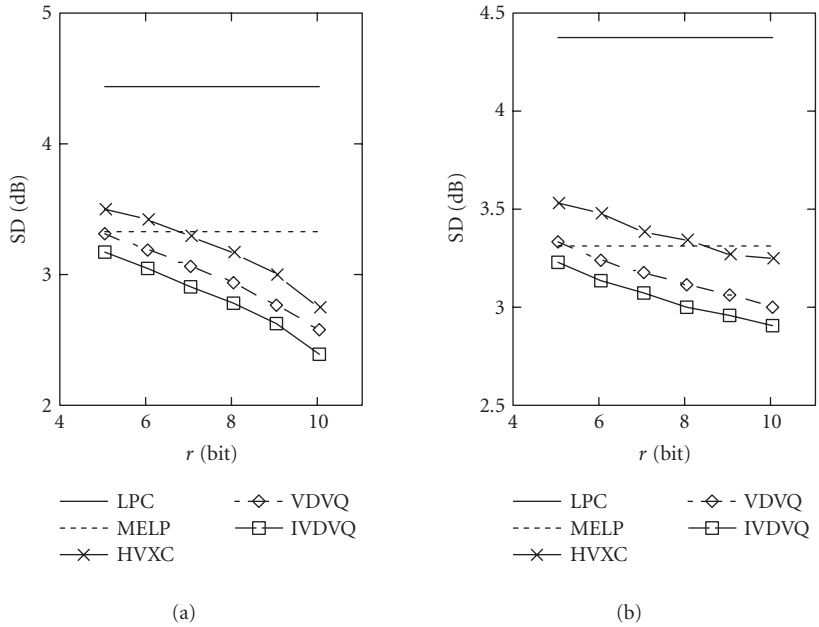

Figure 10: Comparison of (a) training performance and (b) testing performance for five schemes: LPC, MELP, HVXC, VDVQ, and IVDVQ; $N_v = 41$ for VDVQ and IVDVQ.

resolution of the quantizer is increased indefinitely. According to the present experimental outcomes, at $N_v = 41$ the average SD of VDVQ in testing is lower than that of HVXC by 0.22 dB; while the average SD of IVDVQ in testing is lower than that of HVXC by 0.33 dB.

Because of the simplicity of VDVQ, various kinds of structures can be incorporated so as to reduce computational cost. For instance, the popular structures for split VQ, multistage VQ, and predictive VQ can be incorporated to the VDVQ framework in a straightforward manner. Moreover, different types of weighting for distance computation can be added during codebook design.

By introducing interpolation among the elements of the codevectors to the basic VDVQ structure, the performance at low codebook dimension can be raised. However, as the codebook dimension increases, the performance advantage vanishes; this is due to the fact that the rounding error introduced in the sampling index becomes less significant. A simple first-order interpolation technique is considered in the present study, more complex schemes can be explored which might lead to better performance.

The complexity involved in VDVQ is inferior to that of the method adopted by HVXC, because no dimension conversion is necessary for the former, and the actual codevector can be extracted directly from the codebook during encoding. For IVDVQ, there is a need to perform two products and one addition (assuming that the interpolation constants in (36) are precomputed and stored) for every sample of the actual codevector; this amount of computation represents the cost required for improved performance at low codebook dimension.

Although it is theoretically possible to increase the performance of a VDVQ by increasing the codebook dimension, the size of the training data set has to be increased as well so as to ensure a good generalization to data outside the training data set. This is because the codebook elements tend to be influenced by a lower amount of training data as the codebook dimension increases, hence more training data are necessary for codebook design.

In conclusion, we can say that VDVQ is a promising technique applicable to many signal coding applications. It has a simple structure with moderate complexity, with performance exceeding the schemes adopted by many established standards.

ACKNOWLEDGMENTS

The author is grateful to the anonymous reviewers for their valuable comments that improved the paper significantly. An abridged version of this paper was published in [29].

REFERENCES

[1] L. B. Almeida and J. M. Tribolet, "Nonstationary spectral modeling of voiced speech," *IEEE Trans. Acoustics, Speech, and Signal Processing*, vol. 31, no. 3, pp. 664–678, 1983.

[2] W. B. Kleijn and K. K. Paliwal, *Speech Coding and Synthesis*, Elsevier Science Publishers, Amsterdam, The Netherlands, 1995.

[3] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*, Kluwer Academic, Norwell, Mass, USA, 1992.

[4] T. Tremain, "The government standard linear predictive coding algorithm: LPC-10," *Speech Technology Magazine*, vol. 1, no. 2, pp. 40–49, 1982.

[5] L. Supplee, R. Cohn, J. Collura, and A. McCree, "MELP: the new federal standard at 2400 bps," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 2, pp. 1591–1594, Munich, Germany, April 1997.

[6] W. C. Chu, *Speech Coding Algorithms: Foundation and Evolution of Standardized Coders*, John Wiley & Sons, New York, NY, USA, 2003.

[7] M. Nishiguchi, A. Inoue, Y. Maeda, and J. Matsumoto, "Parametric speech coding-HVXC at 2.0–4.0 kbps," in *Proc. IEEE Speech Coding Workshop*, pp. 84–86, Porvoo, Finland, June 1999.

[8] ISO/IEC, "Information technology—Coding of audio-visual objects—Part 3: Audio," 14496-3, 1999.

[9] ISO/IEC, "Information technology—Coding of audio-visual objects—Part 3: Audio, amendment 1," 14496-3, 2000.

[10] A. Das, A. Rao, and A. Gersho, "Variable-dimension vector quantization," *IEEE Signal Processing Letters*, vol. 3, no. 7, pp. 200–202, 1996.

[11] M. Nishiguchi, "MPEG-4 speech coding," in *Proc. AES 17th International Conference on High-Quality Audio Coding*, Florence, Italy, September 1999.

[12] M. Nishiguchi and J. Matsumoto, "Harmonic and noise coding of LPC residuals with classified vector quantization," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 1, pp. 484–487, Detroit, Mich, USA, May 1995.

[13] S. Yeldener, J. C. de Martin, and V. Viswanathan, "A mixed sinusoidally excited linear prediction coder at 4 kb/s and below," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 2, pp. 589–592, Seattle, Wash, USA, May 1998.

[14] C. Li, P. Lupini, E. Shlomot, and V. Cuperman, "Coding of variable dimension speech spectral vectors using weighted nonsquare transform vector quantization," *IEEE Trans. Speech, and Audio Processing*, vol. 9, no. 6, pp. 622–631, 2001.

[15] P. Lupini and V. Cuperman, "Nonsquare transform vector quantization," *IEEE Signal Processing Letters*, vol. 3, no. 1, pp. 1–3, 1996.

[16] C. Li, A. Gersho, and V. Cuperman, "Analysis-by-synthesis low-rate multimode harmonic speech coding," in *Proceeedings of Eurospeech*, vol. 3, pp. 1451–1454, Budapest, Hungary, September 1999.

[17] C. Li and V. Cuperman, "Analysis-by-synthesis multimode harmonic speech coding at 4 kb/s," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 3, pp. 1367–1370, Istanbul, Turkey, June 2000.

[18] C. O. Etemoglu and V. Cuperman, "Coding of spectral magnitudes using optimized linear transformations," in *Proc. IEEE Speech Coding Workshop*, pp. 5–7, Delavan, Wis, USA, September 2000.

[19] C. O. Etemoglu and V. Cuperman, "Spectral magnitude quantization based on linear transforms for 4 kb/s speech coding," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 2, pp. 701–704, Salt Lake City, Utah, USA, May 2001.

[20] S. Yeldener, "A 4 kb/s toll quality harmonic excitation linear predictive speech coder," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 1, pp. 481–484, Phoenix, Ariz, USA, March 1999.

[21] Y. D. Cho, M. Y. Kim, and A. Kondoz, "Predictive and mel-scale binary vector quantization of variable dimension spectral magnitude," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 3, pp. 1459–1462, Istanbul, Turkey, June 2000.

[22] J. Stachurski, A. McCree, and V. Viswanathan, "High quality MELP coding at bit-rates around 4 kb/s," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 1, pp. 485–488, Phoenix, Ariz, USA, March 1999.

[23] E. Shlomot, V. Cuperman, and A. Gersho, "Hybrid coding: combined harmonic and waveform coding of speech at 4 kb/s," *IEEE Trans. Speech, and Audio Processing*, vol. 9, no. 6, pp. 632–646, 2001.

[24] A. Das and A. Gersho, "Variable dimension spectral coding of speech at 2400 bps and below with phonetic classification," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 1, pp. 492–495, Detroit, Mich, USA, May 1995.

[25] A. Makur and K. P. Subbalakshmi, "Variable dimension VQ encoding and codebook design," *IEEE Trans. Communications*, vol. 45, no. 8, pp. 897–899, 1997.

[26] S. Mohamed and M. Fahmy, "Image compression using block pattern VQ with variable codevector dimensions," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 3, pp. 357–360, San Francisco, Calif, USA, March 1992.

[27] S. Haykin, *Neural Networks: A Comprehensive Foundation*, Macmillan Publishers, New York, NY, USA, 1994.

[28] J. S. Garofolo, L. F. Lamel, W. M. Fisher, J. G. Fiscus, D. S. Pallett, and N. L. Dahlgren, "DARPA TIMIT, acoustic phonetic continuous speech corpus CD-ROM," Tech. Rep., National Institute of Standards and Technology, Gaithersburg, Md, USA, 1993.

[29] W. C. Chu, "A novel approach to variable dimension vector quantization of harmonic magnitudes," in *Proc. 3rd IEEE International Symposium on Image and Signal Processing and Analysis*, vol. 1, pp. 537–542, Rome, Italy, September 2003.

Wai C. Chu received the B.S. degree in electronics engineering from Universidad Simón Bolivar (Caracas, Venezuela) in 1990, the Master's of Engineering degree in electrical engineering from Stevens Institute of Technology (Hoboken, NJ, USA) in 1992, and the Ph.D. degree in electrical engineering from the Pennsylvania State University (University Park, Pa, USA) in 1998. From March 1993 to August 1994, he was with Texas Instruments Hong Kong as a field applications engineer, where he designed a digital telephone answering device. While enrolled as a graduate student at Penn State University, he spent two summers (1995 and 1996) working as an intern in Texas Instruments Inc (Dallas, Tex), and developed software for various speech coding and digital filtering applications. He joined DoCoMo USA Labs (San Jose, Calif) in 2001 and is currently involved with R&D activities in speech/audio coding, digital signal processing, and multimedia applications. Dr. Chu is a Member of the IEEE, the Audio Engineering Society (AES), and the Acoustical Society of America (ASA). He has served as a reviewer for numerous conferences and journals, and is the author of the textbook *Speech Coding Algorithms: Foundation and Evolution of Standardized Coders* (John Wiley & Sons, 2003).