SECONDARY CODEBOOK STORAGE QUANTITATION

Thomas M. Chapman, Professor C. S. Xydeas
Speech and Image processing laboratory
Simon Engineering Building
The University of Manchester
Brunswick Street, Manchester, M13 9PL, UK

ABSTRACT

Efficient reduction of storage and complexity demands in VQ and MQ systems is a key issue when developing new, more powerful compression algorithms. Secondary Storage Quantisation (SSQ) is capable of drastically reducing VQ storage through an efficient representation of codebook elements. Rather than the conventional fixed or floating point representation, codebook elements are quantised using a set of “secondary codebooks” and represented as a set of quantisation indices. The number of bits required for these indices is relatively small and hence the amount of storage required for codebook representation is reduced. The potential of SSQ for codebook compression is demonstrated in a Split Matrix Quantisation (SMQ) application. A reduction of 65 - 75 % in the amount of memory required for SMQ codebooks is achieved.

1. INTRODUCTION

Vector Quantisation (VQ) is an important compression / classification technique and is an optimal method for encoding vector signals [2]. In two dimensions, VQ is generalised to Matrix Quantisation (MQ) [3], also a powerful compression tool.

However, applications of MQ and VQ are often limited by high computational and storage demands. Although the recent dramatic increase in the speed of DSP devices has in a way eased complexity constraints, large memory requirements continue to inhibit the implementation of VQ based techniques on low power devices.

In terms of complexity reduction, a frequently applied strategy is the use of sub-optimal schemes [2,4] which trade computational demands against performance. Recently, effort has been concentrated into Vornoi Projection techniques [5] which have been shown to significantly reduce VQ complexity without compromising performance. This technique does not, however overcome the large storage requirement, of 2KN elements, where K is the vector dimension and N is the codebook size.

Efficient quantisation of speech LPC spectral envelope information is an important issue in low bit rate speech coding. In this context, the application of Split Matrix Quantisation (SMQ) on LSP parameters [1] has been shown to be an effective compression technique. Single Track SMQ schemes (ST-SMQ) are capable of achieving “transparent” LPC quantisation at a rate of 912.5 bits / second, whilst “High Quality” quantisation is achieved at 825 bits / second. More efficient Double Track schemes (DT-SMQ) yield “High Quality” LPC quantisation at 700 bits / sec.

In common with many other MQ and VQ schemes, a drawback of certain SMQ configurations is the large amount of permanent storage required for codebooks. The 912.5 bits / sec ST-SMQ scheme, for example, needs 24000 codebook elements, which corresponds to 375kbits in 16 bit words. In the same way, DT - SMQ demands around 5.5 Mbits for “High Quality” LSP quantisation at 700 bits / sec.

In general VQ / MQ schemes represent codebook elements using a conventional fixed or floating point format. The finite numeric resolution means that the codebook elements are effectively quantised. In general little attention is paid to the ability of this “secondary” quantisation to yield an accurate codebook representation with a lowest possible storage demand.

This contribution examines a novel “Secondary codebook Storage Quantisation” (SSQ) technique in the context of reducing SMQ storage. In the proposed scheme, a “secondary” scheme quantises the elements of the “main” SMQ codebooks. In this way, SMQ codebooks are represented as sets of indices of “secondary codebooks”, which are large enough to ensure the accurate quantisation of SMQ elements and thus a minimal effect upon SMQ performance. Thus, the SSQ “index” representation of SMQ codebooks, together with a suitable packing arrangement designed to maximise the usage of memory words, drastically reduces the storage requirement of SMQ schemes,
despite the necessity of a small amount of “side information” storage for the secondary codebooks.

The organisation of the remainder of this paper is as follows: Section 2 details some essential features of SMQ. This is followed in section 3 by a specific description of the SSQ scheme. Section 4 then presents and discusses simulation results, while conclusions are given in section 5.

2. SPLIT MATRIX QUANTISATION

Split Matrix Quantisation (SMQ) is described fully in [1]. N input frames of p LSP parameters are grouped into an (N x p) matrix \( M_{j,i} \) (\( j = 1, 2, .. N \); \( i = 1, 2, .. p \)). In Single Track SMQ (ST - SMQ) each row of \( M_{j,i} \) is separately vector quantised yielding p indices \( y^m \) (\( m = 1, 2, .. p \)). In the double track scheme (DT - SMQ), \( M_{j,i} \) is split into \( p/2 \) submatrices by grouping rows in blocks of 2. Each submatrix is then Matrix Quantised, yielding \( p/2, y^m \) (\( m = 1, 2, .. p/2 \)) indices. Note that a separate bit allocation \( B_A^m \) is given to each \( M_{j,i} \) row in ST - SMQ, or each submatrix in DT - SMQ.

The N input speech frames may be either (i) all voiced, (ii) all unvoiced or (iii) a mixture of the two. In order to preserve perceptually significant voiced - unvoiced transitions and to improve coding efficiency, SMQ schemes use a separate set of codebooks for each case. Thus ST - SMQ schemes employ a total of \( 3*p \) codebooks \{SMQCB\} \( _m,t \) (\( m = 1, 2, ..., p; t = 1, 2, 3 \)) while DT - SMQ uses \( 3*p/2 \) codebooks (\( t \) is dependent upon the voicing status of the N frames). The entries of each \{SMQCB\} \( _m,t \) are vectors of length N elements in ST - SMQ and matrices of size \( 2*N \) elements in the dual track case. The number of codebook elements in each scheme are as follows:

\[
3*N*2^{BA^i} \quad \text{(ST - SMQ)} \\
3*2N*2^{BA^i} \quad \text{(DT - SMQ)}
\]

3. SECONDARY STORAGE QUANTISATION

3.1 ST - SMQ schemes

A set of p “secondary” codebooks, \{SECCB\} \( _m \) (\( m = 1, 2, .. p \)) of size \( 2^{s^m} \) are employed. The \( m \)th \{SECCB\} \( _m \) quantises the \( m \)th group of 3 \{SMQCB\} \( _{m,t} \) (\( t = 1, 2, 3 \)) codebooks which correspond to the \( m = i \)th row of \( M^i \).

The \{SECCB\} \( _m \) codebooks are simple scalar quantisers and are generated using the LBG algorithm [6]. Either (i) A database of LSP vectors obtained from natural input speech or (ii) the \{SMQCB\} \( _{m,t} \) codebook elements can be employed as training data. A total of 3 types of \{SECCB\} \( _m \) codebooks were produced, as follows:

(a) \( m = 1, 2, .. p \) uniform scalar quantisers, denoted by \{SECCB\} \( _{m,\text{scalar}} \). The range of the \( m \)th \{SECCB\} \( _{m,\text{scalar}} \) is determined from the minimum and maximum elements of the 3 \{SMQCB\} \( _{m,t} \) (\( t = 1, 2, 3 \)) codebooks.

(b) \( m = 1, 2, .. p \) nonuniform scalar quantisers, \{SECCB\} \( _{m,\text{nonuniform}} \). The \( m \)th \{SECCB\} \( _{m,\text{nonuniform}} \) is trained using the elements of the 3 \{SMQCB\} \( _{m,t} \) (\( t = 1, 2, 3 \)) codebooks.

(c) \( m = 1, 2, .. p \) nonuniform scalar quantisers, denoted by \{SECCB\} \( _{m,\text{speech}} \). The elements of the \{SMQCB\} \( _{m,t} \) codebooks are quantised using the appropriate \{SECCB\} \( _{m} \), yielding a set of \( S^m \) bit \{SECCB\} \( _{m} \) indices \( i_{SMQ}^m \) (\( m = 1, 2, .. p; \) \( t = 1, 2, 3 \)) which together with \{SECCB\} \( _{m} \) are stored in ROM. Note that the \{SECCB\} \( _{m} \) elements are stored using full machine accuracy. Some form of packing of the \( i_{SMQ}^m \) indices may be required if \( S^m \) differs from the word length. Based upon a 16 bit word, suitable packing arrangements have been devised for the configurations presented in this paper. The total storage requirement (in bits) of a SSQ - ST - SMQ scheme is:

\[
\sum_{m=1}^{p} (B_2^{s^m} + 3 \sum_{i=1}^{N} 2^{BA^i} \sum_{t=1}^{3} i_{SMQ}^m )
\]

where \( i_{SMQ}^m \) are the \( j = 1, 2, .. N \) \{SMQ\} indices relating to the \( u \)th SMQ codeword and \( B_2^{s^m} \) is the amount of storage used in representing \{SECCB\} \( _{m} \).

SSQ - ST - SMQ encoding consists of a codebook search for each row of \( M^i \). The \( i \)th search is carried out as follows:

(i) Construct a table \[D_{j,s} \] (\( j = 1, 2, .. N \); \( s = 1, 2, \ldots S^m \)), which stores the distortions formed between the (\( j \))th element of \( M^i \) and the \( S^m \) \{SECCB\} \( _{m} \) elements:

\[
D_{j,s} = w_{j,i}^2 (seccb^m_j - M_{j,i}^i)^2
\]

seccb^m_j are the elements of \{SECCB\} \( _m \) and \( w_{j,i}^2 \) is a weighting function as described in [1].
For each entry in \( \{\text{SMQCB}\}_m,t \), recover the \( j = 1, 2, \ldots, N \) \( \{\text{SMQ}\}_m,t \) indices. The total distortion for the \( u \)th entry is calculated by using \( [D^{u}] \) and the \( \{\text{SMQ}\}_m,t \) indices:

\[
\text{dist}^u = \sum_{j=1}^{N} D^{u,j}_{\text{SMQ}} \quad (3)
\]

Notice that this codebook search procedure requires an overhead calculation of \( [D^{u}] \), followed by two memory lookups and an addition for each element of each SMQ codebook entry. Furthermore, some additional complexity may result from an unpacking operation to recover \( \{\text{SMQ}\}_m,t \). This is compared to a total of 1 memory lookup and 1 MAC operation per SMQ element in a conventional search. \( [D^{u}] \) is temporary and may be stored in fast access memory. The complexity of the SSQ - SMQ search in comparison with conventional SMQ is implementation dependent, but of the same order of magnitude. At the decoder, the reconstructed SMQ is implementation dependent, but of the same performance as the set of data to be quantised (i.e. SMQ codebook elements). An explanation for this observation is that in ST - SMQ it is “speech” LSP vectors that are quantised. The SMQ codebook elements therefore cover the full vector space of natural speech. If the secondary codebooks are trained using “original input speech” derived LSPs, the high density of training points allows the codewords to reflect more accurately the distribution of natural speech. Thus even though the quantised SMQ elements, an LSSNR performance of 10.23 dB is achieved. Notice that using a 16 bit word representation for SMQ elements, an LSSNR performance of 10.23 dB is observed using \( \{\text{SECCB}\}_m \) codebooks.

\[
\sum_{m=1}^{p/2} (2B * 2^m)^3 \sum_{t=1}^{3} \sum_{j=1}^{N} \sum_{m=1}^{m} \sum_{t=1}^{3} [j]_{\text{SMQ}} \quad (4)
\]

2B * 2^m (\( m = 1, 2, \ldots, p/2 \)) is the amount of storage required for the \( \{\text{SECCB}\}_m \) codebooks.

### 4. RESULTS AND DISCUSSION

The application of SSQ on ST - SMQ performance was investigated using an 18.25 bits / 20msec, \( N = 4 \) ST - SMQ configuration. \( S^m \) was equal for all \( m = 1, 2, \ldots, p \) and increased from 1 to 8. The elements of each \( \{\text{SMQCB}\}_m \) were quantised using \( \{\text{SECCB}\}_m \) to allow the performance of the resulting ST - SMQ scheme to be assessed using the LogSegSNR (LSSNR) [1] metric. In addition, SMQ performance using “unquantised” SMQ codebook elements (represented as \( C \) type “float”) was measured.

Figure 1 shows ST - SMQ LSSNR performance characteristics obtained using the 3 types of \( \{\text{SECCB}\}_m \) codebook. Notice that for storage levels greater than 100 kbits, no loss of SMQ performance is observed and the three types of codebook perform equivalently. As \( S^m \), and hence the storage requirement are reduced, using \( \{\text{SECCB}\}_m \) results in a somewhat better performance than \( \{\text{SECCB}\}_m \). This is easily explained since \( \{\text{SECCB}\}_m \) is better matched to the distribution of the SMQ codebook elements in the LSP vector space. In addition, \( \{\text{SECCB}\}_m \) yields better performance than \( \{\text{SECCB}\}_m \), even though the training data in the former (i.e. “natural” speech) is not the same as the set of data to be quantised (i.e. SMQ codebook elements). An explanation for this observation is that in ST - SMQ it is “speech” LSP vectors that are quantised. The SMQ codebook elements therefore cover the full vector space of natural speech. If the secondary codebooks are trained using “original input speech” derived LSPs, the high density of training vectors allows the codewords to reflect more accurately the distribution of natural speech. Thus even though the quantised SMQ codebooks elements are somewhat different to the original, they are still based upon “natural” speech centroids. When the secondary codebooks are trained using SMQ codebook elements, however there is only a low density of points to describe the distribution of speech in LSP vector space, and the resulting secondary codebook elements do not reflect properly the positions of “natural” speech centroids.

Notice that using a 16 bit word representation for SMQ elements, an LSSNR performance of 10.23 dB is observed using \( \{\text{SECCB}\}_m \) codebooks.
achieved while the total storage required is 375 kBits. The \( \text{SECCB}_{\text{speech,nonuniform}} \) SSQ using 6 bits achieves a storage reduction of 65 % while the reduction in LSSNR due to the secondary quantisation is around 0.075dB.

Figure 1 LSSNR performance of 18.25bpf, \( N = 4 \) SSQ, ST - SMQ using Uniform and Speech and SMQ trained nonuniform secondary quantisers.

Figure 2 shows the performance characteristics of a 14 bit per frame, \( N = 4 \) SSQ, DT - SMQ LSP quantisation scheme when secondary quantised using 2 dimensional SMQ and speech trained VQ \( \{ \text{CBSEC}\}_m \) codebooks. Again, using \( \{ \text{SECCB}\}_m^{\text{DT,Speech}} \) yields a superior performance to \( \{ \text{SECCB}\}_m^{\text{DT,SMQ,p/2}} \). When the secondary quantisation is fine enough to ensure that SMQ performance is not compromised, there is little difference between either of the \( \{ \text{CBSEC}\}_m \) codebook design schemes.

Notice that the conventional 14bpf DT - SMQ scheme requires around 5.44 Mbits storage and achieves an LSSNR performance of 9.70. On the other hand \( \{ \text{SECCB}\}_m^{\text{DT,Speech}} \) based SSQ requires around 1.39 Mbits when \( S^m = 8 \) (\( m = 1, 2, ... p/2 \)) and achieves an LSSNR performance of 9.59 dB. Thus the introduction of SSQ yields a 74 % reduction in SMQ storage. Notice also that the storage reduction achieved in the DT - SMQ system is higher than that obtained in ST - SMQ. This is due to the ability of the 2 dimensional secondary codebooks to exploit interparameter redundancy and thus achieve better codebook compression.

5. CONCLUSION

Secondary Storage Quantisation has been introduced and shown to be an effective means of reducing the storage requirement of both Single Track and Double Track SMQ schemes, achieving reductions in the order of 65 - 75 %. Further reductions may be obtained by means of exploiting the ordering and structure of the SMQ and SSQ indices. The proposed concept secondary “storage quantisation” is also applicable to other VQ / MQ schemes and is therefore a powerful and generic VQ design methodology.

6. REFERENCES

3. “Matrix Quantizer design for LPC speech using the Generalized Lloyd Algorithm” Tsao C. Gray R. M. IEEE ASSP Vol 33 No 3 June 1985 pp 537 - 545