Markov system for image vector quantization coding

Shen-Chuan Tai
Yung-Gi Wu
Ling-Shiou Huang
National Cheng Kung University
Institute of Electrical Engineering
University Road 1
Tainan 70101, Taiwan
E-mail: bug@eembox.ee.ncku.edu.tw
wyg@rose.ee.ncku.edu.tw

Abstract. Vector quantization (VQ) has been accepted as one of the most effective image compression methods with provable rate-distortion optimality. The outputs of VQ are a collection of indices, which correspond to the addresses of the codevectors in the codebook. The indices are, however, not mutually independent. They are in fact very highly correlated and are thus appropriately described by a Markov system. In this paper, a Markov system for VQ indices is introduced. Statistics are gathered for various scans, such as the zig-zag, Peano, row-major and column-major scans. The proposed method, like address VQ, achieves the same image quality as conventional VQ. Simulation results show that the proposed method achieves a better bit-rate reduction than Address-VQ. Besides, both the computational complexity and memory needed for the proposed method are lower. Nevertheless, the only extra operation needed by the proposed method is a simple table retrieval operation on both the encoder side and the decoder side. We believe that it is a method worth further exploration.

Subject terms: Vector quantization; address VQ; Markov system.

1 Introduction

Vector quantization (VQ) has been shown to be the best block-based compression technique for a given block size.\(^1\) After decades of intensive research, however, the theoretical promise of VQ has not been fully realized. The basic concept of VQ is that instead of encoding pixels separately, it encodes blocks of pixels as vectors. In a typical VQ system, an image is partitioned into nonoverlapping subblocks of size 4×4, each subblock being considered as a vector. Every vector is compared with codevectors of the codebook, which may be considered as representatives selected in some way, and replaced by the nearest neighboring codevector in a least-mean-squared-error sense. The address of that codevector in the codebook is then transmitted to the channel or storage. To reconstruct the image, the receiver just retrieves the corresponding vector from the same codebook for each received address. That is, all that needs to be done by the decoder in VQ is a table lookup operation.

To reduce the bit rate, a designer may use either a larger vector dimension or a smaller codebook size. The vector dimension itself, however, must be small, considering the computational complexity and the size of storage. An intuitive and appealing way to overcome this problem is to exploit the correlation among the indices for the codevectors of neighboring vectors.

Among all previous proposals for processing VQ indices, address VQ\(^5\)\(^–\)\(^6\) has achieved the best rate-distortion performance. However, the application of address VQ to lossy techniques, such as scalar or transform coding, has proven rather unsatisfactory because it does not allow one to accurately control the trade-off between bit rate and distortion.

Many proposals for VQ try to obtain the best possible fidelity at a given rate, or the lowest possible rate for a given fidelity. In this paper, an alternative way of approaching rate-distortion optimality is proposed. The Markov property is adopted to reduce the bit rate even at the same fidelity as for conventional VQ. The Markov property has been employed in many applications in the field of signal processing, including word recognition,\(^7\) speech recognition,\(^8\) frequency estimation,\(^9\) texture classification,\(^10\) document decoding,\(^11\) and the Markov prefetcher, which acts as an interface between on-chip and off-chip caches.\(^12\)

In this paper, an application of Markov processes to VQ-based still-image compression is proposed. It achieves better bit-rate reduction, has lower computational complexity, and requires less memory space than address VQ.

2 Address VQ

To make this paper self-contained, the concept of address VQ\(^5\)\(^–\)\(^6\) is briefly described in this section. Address VQ is a method of lossless coding on VQ indices designed for image compression. Let the VQ system encode 4×4 pixel blocks with a codebook of size \(k=128\). The basic idea of address VQ is to explore the representatives of \(4=2\times2\) neighboring VQ indices. Each of those representatives is then coded as a single symbol.

Let \(r_i, i=1, 2, 3, 4\), be four random variables that denote the four neighboring indices, i.e.,
Consider a stochastic process \( \{X(t)\} \) taking values from a discrete state space \( S = \{0,1,2,\ldots\} \). For any two points in time, say \( t \) and \( t' \), and for any two states, say \( m \) and \( n \), define the transition probability as

\[
P_{m,n}^{t,t'} = P[X(t') = n | X(t) = m],
\]

which is the conditional probability that the process is in state \( n \) at time \( t' \), given that it is in state \( m \) at time \( t \). In general, given a sequence of time points \( t_1 < t_2 < \cdots < t_{k-1} < t_k \), the conditional probability that the process is in state \( m_k \) at time \( t_k \), given that it has been in states \( m_1 \) through \( m_{k-1} \) at times \( t_1 \) through \( t_{k-1} \), respectively, depends on the values of the process at times \( t_1, t_2, \ldots, t_{k-1} \). The Markov property can be defined as

\[
P[X(t_k)]
= m_k[X(t_{k-1}) = m_{k-1}, \ldots, X(t_2) = m_2, X(t_1) = m_1]
= P[X(t_k) = m_k | X(t_{k-1}) = m_{k-1}] = P_{m_{k-1},m_k}^{t_{k-1},t_k}.
\]

However, in some cases the conditional probability depends strongly on the most recent values prior to \( t_{k-1} \). In such a case, the analysis of the process is greatly simplified. Refer to Fig. 1, which shows a simplified version of a Markov process with four states, depicted as a directed graph. State \( s_i \), i.e., node \( s_i \) of the directed graph, represents index \( i \). A directed edge \( e_{ij} \) from state \( s_i \) to \( s_j \) means a transition from state \( s_i \) to state \( s_j \). Corresponding to each \( e_{ij} \) in the Markov process, there is a probability \( P_{i,j} \), which is the conditional probability \( P(s_j | s_i) \). If an entropy coding technique is used to encode the directed edges ejected from \( s_i \), then the code length \( L_{i,j} \) for edge \( e_{ij} \) is given by

\[
L_{i,j} = \log_2 \frac{1}{P_{i,j}}.
\]

This can achieve a smaller average code length.

In the proposed system, every VQ index is regarded as a state. The Markov system just exploits the relationship between the present state and the next one. The basic idea is straightforward: Since in natural images spatially close blocks are strongly correlated, the indices associated with them also exhibit statistical dependence, as can be seen from Fig. 2, which shows the VQ indices expressed as 128 \( \times \) 128 images where the codebook size is 256 and the codevectors, which are of dimension 4\( \times \)4, are sorted according to their mean values. Therefore, it is quite natural and will be effective to encode the VQ indices by extensively exploring this kind of dependence. The reader may refer to Ref. 13 for a detailed description of the Markov property.
4 Proposed Markov System for VQ Coding

Consider a finite sequence of VQ indices \( y_1, y_2, \ldots, y_n \) obtained by some kind of scan of the image subblock, and let \( k \) denote the size of the codebook that is generated by the LBG algorithm. So there are \( k \) states in the Markov system. According to information theory, the minimum code length of the VQ index sequence in bits is given by 
\[
- \log_2 \Pi_{i=1}^k P(y_i|y_{i-1})
\]
This is the minimum possible rate. The key issue in Markov systems is the estimation of \( P(y_i|y_{i-1}) \). Note that depending on the scan pattern, the previous state \( y_{i-1} \) may differ from \( y_i \). Typical scans that are considered in this paper include the zig-zag, Peano, row-major, and column-major scans as shown in Fig. 3. That figure shows the four scans applied to \( 4 \times 4 \) indices, to simplify the presentation. However, four scans are applied to \( 128 \times 128 \) \((512/4) \times (512/4)\) indices in the case where the image size is \( 512 \times 512 \) and the codeword is \( 4 \times 4 \).

To estimate \( P(y_i|y_{i-1}) \), \( i, j, y_{i-1} \in \{1, 2, \ldots, k\} \), a large training set is used. The estimated conditional probabilities \( P(y_i|y_{i-1}) \), \( i, j, y_{i-1} \in \{1, 2, \ldots, k\} \), are then stored and used to drive an entropy coder for compressing VQ index sequences. In order to achieve the most efficient and effective compression of VQ indices, all the scan patterns in Fig. 3 are explored.

For the statistics of state transitions, a two-dimensional array \( \varphi(k \times k) \) that accumulates the frequencies of transitions between state pairs is used. Initially, we set every element of \( \varphi(k \times k) \) to 1. To avoid the zero-frequency problem, we have to make sure that every state can transit to all the other states, i.e., \( P_{ij} \neq 0 \), \( i, j \in \{1, 2, \ldots, k\} \). The algorithm is as follows:

- **Input**: A well-trained codebook generated by LBG. Training images that are used to generate the state-transition matrix.
- **Output**: A code table for every state transition.
- **Step 1**: Initially, for each \( i \) and \( j \), set \( \varphi[i][j] \) to 1.

\[
\text{Fig. 3 Four scan patterns.}
\]

\[
\text{Fig. 2 VQ indices show the same image (codebook size=256).}
\]

\[
\begin{align*}
\text{Step 2:} & \quad \text{Find out the best match codevectors } y_i \text{ from the codebook for the incoming unencoded vector } u_t \text{ at time } t. \\
\text{Step 3:} & \quad \text{Increase } \varphi[y_{i-1}][y_i] \text{ by 1.} \\
\text{Step 4:} & \quad \text{If all vectors in the training images have been processed, go to step 5, else go to step 2.} \\
\text{Step 5:} & \quad \text{Calculate the transition probability } P_{i,j} \text{ from state } i \text{ to } j \text{ as follows:} \\
& \quad \text{for } (i=0; i<k; i++) \\
& \quad \text{for } (j=0; j<k; j++) \\
& \quad \quad P_{i,j} = \frac{\varphi[i][j]}{\sum_{j'=0}^{k-1} \varphi[i][j']} \quad (4) \\
\text{Step 6:} & \quad \text{Calculate the minimum code-length table as follows:} \\
& \quad \text{for } (i=0; i<k; i++) \\
& \quad \text{for } (j=0; j<k; j++) \\
& \quad \quad L_{i,j} = \left[ \log_2 \left( \frac{1}{P_{i,j}} \right) \right] \quad (5)
\end{align*}
\]

If the model is properly selected, the \( P_{i,j} \)'s for each \( i \) will differ greatly from the uniform distribution, and thus, according to information theory, one can build a Huffman code tree for each state and the average code length for each state will be small. To make the proposed system simple and effective, the models used simply correspond to different scans. Figure 4 illustrates the count distribution of our Markov transition matrix \( \varphi_{i,j} \) that was attained by training 22 images with column-major scan. As can be seen from Fig. 4, for any present state \( s_i \), the distribution \( \{P_{i,j}\} \) does differ greatly from the uniform distribution; in fact, all values except \( P_{i,i} \) are small. Therefore, it is quite natural to adopt the Markov system to describe VQ indices. In that case, the codes to be transmitted or stored are not the VQ indices themselves. It is the state transitions that are transmitted or stored. For example, let the code sequence be...
Then the codes to be transmitted or stored for \(y_2, y_3, y_4\) will be Code(transition \((y_1 \rightarrow y_2)\)) + Code(transition \((y_2 \rightarrow y_3)\)) + Code(transition \((y_3 \rightarrow y_1)\)).

In conventional VQ, the coding order does not influence the results at all. In the proposed system, however, the orders, which correspond to the models, do affect the simulation results, as will be clear in the next section.

5 Simulation Results

To assess the performance of the Markov system, simulations have been carried out on a set of gray-scale images widely used in the research community. All images are of \(512 \times 512\) pixels with 8-bit pixel gray intensity. The code-vector dimension is \(4 \times 4\). A problem that must be considered in the proposed method that does not happen in conventional VQ is the error propagation problem. Once a single code in the sequence is corrupted, all the following codes will be wrong. To prevent the propagation of errors caused by channel noise, the codes are transmitted in segments. A special symbol is inserted into the sequences after every \(t\) transitions to indicate the start of a segment. Refer to Fig. 5. In this example, \(t\) is set to 4. In practical experiments, \(t\) is set to 128 so that the error propagation will be limited below 128. Extra bits for this special symbol are of course added to all the experimental results on bit rates in this paper.

Twenty-two images are used to generate the Markov state transition table. After that, the corresponding Huffman trees for every state are generated. Codebooks of size 128 and 256 are generated and tested. Due to the heavy computation required to determine the codebooks and Markov transition tables, the programs are run on a DEC 8400. However, since that is done offline in advance, it does not increase the burden on the coding procedures. Figures 6 and 7 illustrate the simulation results with codebooks of size 128 and 256, respectively. As can be seen from the two figures, adopting the column-major scanning pattern gives the best correlation reduction in most cases. Detailed numerical data are shown in Table 1.

In Table 1, the bit rates and PSNR obtained by the proposed system with the four different scanning orders are shown. The bit rates and PSNR of the proposed method are 0.233 bits/pixel and 29.72 dB for "Lena," versus 0.256 bits/pixel and 30.6 dB with address VQ.\(^6\) The PSNR depends on the way the codebooks are generated. Since the authors of Ref. 6 did not describe the way they generated the codebook, the PSNRs by the proposed method may be a little different from theirs. However, the extent of bit-rate reduction is obviously higher than with address VQ. In all tests, the test images are excluded from the training sets. Even so, their bit rates decrease significantly. As to the computational complexity and memory requirement of the proposed method, the computation burden for codebook searching is the same as in conventional VQ. Processing the Markov transition matrix is simply table retrieval. The memory requirement for the Markov transition matrix is given in Table 2. Four memory sizes required by the Markov transition matrices for the four scan patterns are shown. For the case of codebook size 128, the total memory needed is about 25 kbyte. It is about 95 kbyte for codebook size 256. Address VQ\(^6\) needs three codebook-searching operations, and the sizes of the three codebooks are 128, 16,384, and 1024 respectively. The total memory size needed is about 260 kbyte. It is clear that the proposed system needs less memory. Therefore, the proposed method outperforms that of Ref. 6 in computational complexity, memory requirement, and bit-rate reduction.

Results of the proposed method are also compared with those by JPEG. Three images—"Lenna," "Lena," and "Bird"—are compressed at the same bit rates and codebook size 128. The PSNR values of the three reconstructed images are 30.71, 29.80, and 35.27 dB, respectively. The proposed Markov system achieves a little worse results

---

1. Conventional VQ index sequences
\[
y_1, y_2, y_3, y_4
\]
2. Markov code sequences
\[
y_1, (y_1 \rightarrow y_2), (y_2 \rightarrow y_3), (y_3 \rightarrow y_4)
\]
3. Insert start symbol "\(\text{^{'}}\) to every segment
\[
@, y_1, (y_1 \rightarrow y_2), (y_2 \rightarrow y_3), (y_3 \rightarrow y_4), @, y_1, (y_1 \rightarrow y_2), (y_2 \rightarrow y_3), (y_3 \rightarrow y_4)
\]

Fig. 5 Example of sequences.
than JPEG in terms of PNSR, but it should be noted that the computation burden for inverse DCT is much heavier than for simple index retrieval.

References 14 and 15 also describe a VQ index compression technique. They achieved bit rates of 0.254 and 0.2185 bits/pixel, respectively. The proposed system does better.

6 Conclusion and Future Work

In this paper, a bit-rate reduction technique for still-image coding, referred to as the Markov system for VQ, is proposed. Like address VQ, it adopts statistical coding of the addresses obtained by VQ searching. However, it requires considerably less computation time and memory space than address VQ. On both the encoder and decoder sides, the extra operations needed are simply table retrievals, which add negligible complexity. Nevertheless, the bit rate is significantly reduced.

The proposed method provides the same reconstruction fidelity as conventional VQ does. However, the number of codes to be transmitted or stored decreases by 40% to 60%, which is a significant improvement. As compared with other bit-rate reduction technique such as address VQ, the performance of the proposed Markov system is better in reduction degree, computational complexity, and memory requirement. Furthermore, the simulation results will get better if better codebooks are used. In this paper, no special technique is used to generate optimal or near-optimal codebooks. The results will be better if a more sophisticated algorithm is used to generate the codebooks. Besides, the proposed Markov system is general in the sense that it can be used with other VQ systems, such as classified VQ and finite-state VQ, and with other optimal codebook generation strategies to raise the performance.

We are now studying the possibility of utilizing the proposed coding scheme to transform images. In particular, we

<table>
<thead>
<tr>
<th>Image</th>
<th>PSNR (dB)</th>
<th>Row-major scan</th>
<th>Column-major scan</th>
<th>Peano scan</th>
<th>Zig-zag scan</th>
<th>Bit saving for best scan (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lenna</td>
<td>30.51</td>
<td>0.2469</td>
<td>0.2110</td>
<td>0.2373</td>
<td>0.2570</td>
<td>51.76</td>
</tr>
<tr>
<td>Lena</td>
<td>29.72</td>
<td>0.2692</td>
<td>0.2332</td>
<td>0.2604</td>
<td>0.2801</td>
<td>46.68</td>
</tr>
<tr>
<td>Fruit</td>
<td>29.53</td>
<td>0.2428</td>
<td>0.2363</td>
<td>0.2521</td>
<td>0.2829</td>
<td>45.97</td>
</tr>
<tr>
<td>F-16</td>
<td>29.38</td>
<td><strong>0.2104</strong></td>
<td>0.2274</td>
<td>0.2307</td>
<td>0.2539</td>
<td>51.89</td>
</tr>
<tr>
<td>F-18</td>
<td>24.13</td>
<td>0.2256</td>
<td><strong>0.2181</strong></td>
<td>0.2341</td>
<td>0.2618</td>
<td>48.41</td>
</tr>
<tr>
<td>Bird</td>
<td>35.74</td>
<td>0.1714</td>
<td><strong>0.1650</strong></td>
<td>0.1706</td>
<td>0.1949</td>
<td>62.27</td>
</tr>
<tr>
<td>Goldhill</td>
<td>29.96</td>
<td><strong>0.2391</strong></td>
<td>0.2555</td>
<td>0.2531</td>
<td>0.2831</td>
<td>45.34</td>
</tr>
<tr>
<td>Tiffany</td>
<td>28.67</td>
<td>0.2083</td>
<td><strong>0.1907</strong></td>
<td>0.2019</td>
<td>0.2192</td>
<td>56.40</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Codebook size 128; bit rate 0.4375 bits/pixel</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lenna</td>
</tr>
<tr>
<td>Lena</td>
</tr>
<tr>
<td>Fruit</td>
</tr>
<tr>
<td>F-16</td>
</tr>
<tr>
<td>F-18</td>
</tr>
<tr>
<td>Bird</td>
</tr>
<tr>
<td>Goldhill</td>
</tr>
<tr>
<td>Tiffany</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Codebook size 256; bit rate 0.5000 bits/pixel</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lenna</td>
</tr>
<tr>
<td>Lena</td>
</tr>
<tr>
<td>Fruit</td>
</tr>
<tr>
<td>F-16</td>
</tr>
<tr>
<td>F-18</td>
</tr>
<tr>
<td>Bird</td>
</tr>
<tr>
<td>Goldhill</td>
</tr>
<tr>
<td>Tiffany</td>
</tr>
</tbody>
</table>

Table 1 Simulation results. Boldface values correspond to test images, excluded from the training sets.

<table>
<thead>
<tr>
<th>Scan</th>
<th>Markov transition matrix (bytes)</th>
<th>Codebook (bytes)</th>
<th>Total (bytes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Row major</td>
<td>22,838</td>
<td>2048</td>
<td>24,886</td>
</tr>
<tr>
<td>Column major</td>
<td>26,815</td>
<td>2048</td>
<td>28,863</td>
</tr>
<tr>
<td>Peano</td>
<td>21,679</td>
<td>2048</td>
<td>23,727</td>
</tr>
<tr>
<td>Zig-zag</td>
<td>20,735</td>
<td>2048</td>
<td>22,783</td>
</tr>
</tbody>
</table>

Table 2 Memory requirement for Markov transition matrix and LBG codebook.
are interested in wavelet-transformed images where the horizontally, vertically, and diagonally oriented details are separated after wavelet decomposition. Applying different scanning orders to different band signals should raise the performance significantly.

Acknowledgment
This research was partially supported by the National Science Council of the Republic of China under contract no. NSC 89-2213-E-006-112. The authors appreciate the comments of the reviewers.

References