ELSEVIER

Signal Processing: *Image Communication* 17 (2002) 73–84

SIGNAL PROCESSING: IMAGE COMMUNICATION

www.elsevier.com/locate/image

An overview of quantization in JPEG 2000

Michael W. Marcellin[a,*], Margaret A. Lepley[b], Ali Bilgin[a], Thomas J. Flohr[c], Troy T. Chinen[d], James H. Kasner[e]

[a] *Department of Electrical and Computer Engineering, University of Arizona, Tucson, AZ, USA*
[b] *The MITRE Corporation, Bedford, MA, USA*
[c] *Science Applications International Corporation, Tucson, AZ, USA*
[d] *Fuji Film Software Inc, Sunnyvale, CA, USA*
[e] *Aerospace Corporation, Chantilly, VA, USA*

Abstract

Quantization is instrumental in enabling the rich feature set of JPEG 2000. Several quantization options are provided within JPEG 2000. Part I of the standard includes only uniform scalar dead-zone quantization, while Part II allows both generalized uniform scalar dead-zone quantization and trellis coded quantization (TCQ). In this paper, an overview of these quantization methods is provided. Issues that arise when each of these methods are employed are discussed as well. © 2002 Elsevier Science B.V. All rights reserved.

Keywords: JPEG 2000; Uniform quantization; Dead-zone quantization; Trellis coded quantization (TCQ)

1. Introduction

JPEG 2000 is the latest ISO/IEC image compression standard. It is distinct from previous standards, in that it creates a framework where the image compression system can act like an image processing system rather than just a simple input–output storage filter. The decision on several key compression parameters such as quality or resolution can be delayed until after the creation of the compressed codestream. JPEG 2000 enables decompression of many image products from a single compressed file and offers several opportunities for compressed domain processing.

Quantization is one of the critical ingredients of the JPEG 2000 image compression system. Many of the desirable elements of the JPEG 2000 feature set are enabled due to careful selection of quantization methods. In this paper, we provide an overview of these methods.

This paper is organized as follows. In Section 2, an introduction to quantization methods used in JPEG 2000 is presented. Section 3 describes how these quantization methods are used within the framework of JPEG 2000, and Section 4 provides a brief summary.

*Corresponding author.

E-mail addresses: marcellin@ece.arizona.edu (M.W. Marcellin), mlepley@mitre.org (M.A. Lepley), bilgin@ieee.org (A. Bilgin), flohrt@aries.tucson.saic.com (T.J. Flohr), TChinen@FujifilmSoft.com (T.T. Chinen), James.H.Kasner@aero.org (J.H. Kasner).

2. Quantization

Quantization is the element of lossy compression systems responsible for reducing the precision of data in order to make them more compressible. JPEG 2000 offers several different quantization options. Only uniform scalar (fixed-size) deadzone quantization is included in Part I of the standard. Part II of the standard generalizes this quantization method to allow more flexible deadzone selection. Furthermore, trellis coded quantization (TCQ) is offered in Part II as a value-added technology.

2.1. Scalar quantization

The simplest form of quantization is scalar quantization. JPEG 2000 employs a dead-zone uniform scalar quantizer to coefficients resulting from the wavelet transform of image samples. Fig. 1 illustrates such a quantizer with stepsize Δ. A scalar quantizer (SQ) can be described as a function Q that maps each element in a subset of the real line to a particular value. For a given wavelet coefficient x, the quantizer produces a signed integer q given by

$$q = Q(x). \tag{1}$$

The quantization index q indicates the interval in which x lies. In Fig. 1, the endpoints of the quantization intervals are indicated by the vertical lines.

Given q, the decoder produces an estimate of x as

$$\hat{x} = \overline{Q^{-1}}(q). \tag{2}$$

In Fig. 1, the heavy "dots" represent these estimates (or reconstruction values). For a given step size Δ, q is computed as

$$q = Q(x) = \operatorname{sign}(x)\left\lfloor \frac{|x|}{\Delta} \right\rfloor. \tag{3}$$

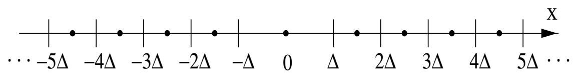

Fig. 1. Dead-zone uniform scalar quantizer with stepsize Δ.

Notice that the wavelet coefficients inside the interval $(-\Delta, \Delta)$ are quantized to zero for the quantizer in Fig. 1. Thus, the interval $(-\Delta, \Delta)$ is called the "deadzone". The width of this interval is 2Δ, while all other intervals are of width Δ.

The inverse quantizer is given by

$$\hat{x} = \overline{Q^{-1}}(q) = \begin{cases} 0, & q = 0, \\ \operatorname{sign}(q)(|q| + \delta)\Delta, & q \neq 0, \end{cases} \tag{4}$$

where δ is a user selectable parameter within the range $0 \leqslant \delta < 1$ (typically $\delta = 1/2$). δ can be chosen to achieve the best objective or subjective quality at reconstruction.

The quantizer illustrated in Fig. 1 is the quantizer used in JPEG 2000 Part I. However, in Part II of the standard, it is possible to generalize this quantizer to allow a deadzone of variable width, while maintaining the fixed width Δ for all other intervals. For a deadzone of width $2(1 - nz)\Delta$, the quantization index is given by

$$q = \begin{cases} 0, & |x| < -nz\Delta, \\ \operatorname{sign}(n)\left\lfloor \frac{|x| + nz\Delta}{\Delta} \right\rfloor, & |x| \geqslant -nz\Delta. \end{cases} \tag{5}$$

Here, nz is a parameter that adjusts the dead-zone size, and is transmitted as side information to the decoder. Notice that $nz = 0$ corresponds to the default quantizer given in Fig. 1, and values of nz in the interval (0, 1) result in a smaller deadzone, while values in $(-1, 0)$ result in a larger deadzone. The generalized dead-zone uniform scalar quantizer is illustrated in Fig. 2. In this case, the reconstructed wavelet coefficients are given as

$$\hat{x} = \begin{cases} 0, & q = 0, \\ \operatorname{sign}(q)(|q| - nz + \delta)\Delta, & q \neq 0. \end{cases} \tag{6}$$

In JPEG 2000, there is some indication that using $nz \approx 0.25$ (i.e., dead-zone size about 1.5Δ) can provide a very slight decrease in MSE and generate more visually pleasing low-level texture reconstruction.

2.1.1. Embedded scalar quantization

A very desirable feature of compression systems is the ability to successively refine the reconstructed data as the bit-stream is decoded. In this situation, the decoder reconstructs an approximation of the reconstructed data after decoding

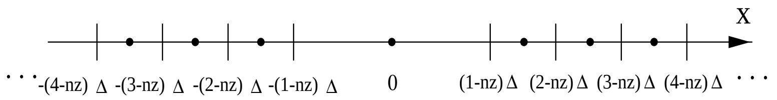
X
· · · -(4-nz) Δ -(3-nz) Δ -(2-nz) Δ -(1-nz) Δ
0
(1-nz)Δ (2-nz)Δ (3-nz)Δ (4-nz)Δ · · ·

Fig. 2. Generalized dead-zone uniform scalar quantizer with stepsize Δ and dead-zone size $2(1-nz)\Delta$.

a portion of the compressed bit-stream. As more of the compressed bit-stream is decoded, the reconstruction quality can be improved, until the full quality reconstruction is achieved upon decoding the entire bit-stream.

Since this property is one of the key focuses of JPEG 2000, the quantizers used in the standard have been designed to enable embedded quantization. Let us consider the uniform dead-zone quantizer defined by Eqs. (3) and (4). This quantizer has embedded within it, all uniform dead-zone quantizers with step sizes $2^p\Delta$ for integer $p \geqslant 0$.

To see this, let $K = \max_q \lceil \log_2(|q|) \rceil$. That is, K is the number of bits required to represent all quantization indices. Then, we can represent q in sign magnitude form as

$$q = s,\ q_0 q_1 q_2 \cdots q_{K-1}, \tag{7}$$

where s is the sign, q_0 is the most significant bit (MSB), and q_{K-1} is the least significant bit (LSB) of q. Now, let

$$q^{(p)} = s,\ q_0 q_1 q_2 \cdots q_{K-1-p} \tag{8}$$

denote the index obtained by dropping the p least significant bits of q. It is then easy to see that

$$q^{(p)} = Q_p(x), \tag{9}$$

where Q_p is the uniform dead-zone quantizer with step size $2^p\Delta$. Fig. 3 illustrates this embedding for $p = 1$ and 2.

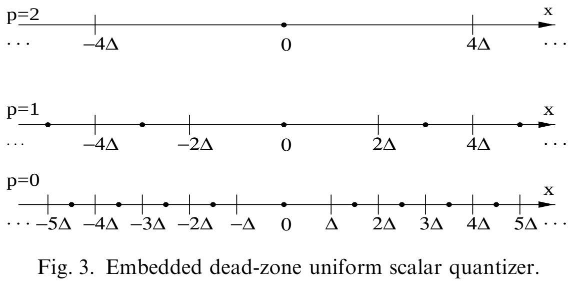
p=2
x
· · · −4Δ 0 4Δ · · ·
p=1
x
· · · −4Δ −2Δ 0 2Δ 4Δ · · ·
p=0
x
· · · −5Δ −4Δ −3Δ −2Δ −Δ 0 Δ 2Δ 3Δ 4Δ 5Δ · · ·

Fig. 3. Embedded dead-zone uniform scalar quantizer.

Eq. (9) suggests that if the p LSBs of $|q|$ are not available at the decoder, it is still possible to obtain an approximation to x, but at a lower level of quality. In this case, the inverse quantization is performed using the dequantizer $\overline{Q_p^{-1}}$ such that

$$\hat{x} = \overline{Q_p^{-1}}(q^{(p)}) = \begin{cases} 0, & q^{(p)} = 0, \\ \operatorname{sign}(q^{(p)})(|q^{(p)}| + \delta)2^p\Delta, & q^{(p)} \neq 0. \end{cases} \tag{10}$$

Note that $p = 0$ yields the full quality dequantization given by Eq. (4).

Similar embedding properties exist for the generalized dead-zone uniform scalar quantizer defined by Eqs. (5) and (6). In this case, when p LSBs of the index $|q|$ are not available at the decoder, the reconstructed value is computed as

$$\hat{x} = \begin{cases} 0, & q^{(p)} = 0, \\ \operatorname{sign}(q^{(p)})(|q^{(p)}| - nz2^{-p} + \delta)2^p\Delta, & q^{(p)} \neq 0. \end{cases} \tag{11}$$

Note that uniform dead-zone SQ has one important embedding property that is not shared by the generalized dead-zone scalar quantizer when $nz \neq 0$. For $Q^p(x)$, as illustrated in Fig. 3, any truncation of p bits is exactly equivalent to choosing a larger step size $2^p\Delta$. As a consequence, the dead-zone width is always exactly twice the effective step size of $2^p\Delta$. So full reconstruction of a quantization with a large step size will produce identical results to a partial reconstruction of a quantization with a smaller step size. This allows flexibility in setting the step size magnitude.

When $nz \neq 0$ the deadzone after truncating p LSBs is $2(1 - nz2^{-p})$ times the effective step size of $2^p\Delta$. Thus, the dead-zone size does not remain constant throughout the embedding. Benefits gained by setting $nz \neq 0$ will have fullest impact when coefficients are fully decoded, so step size choice becomes quite important.

2.1.2. Dequantization with reversible transform

The inverse quantizer formulas given in Eqs. (4) and (10) have a problem when followed by a reversible transform. There is no guarantee that the estimate $\hat{x}$ is an integer. To avoid this problem, JPEG 2000 modifies the inverse quantization definition in this special case to

$$\begin{aligned}\hat{x} &= \overline{Q_p^{-1}}(q^{(p)}) \\ &= \begin{cases} 0, & q^{(p)} = 0, \\ \operatorname{sign}(q^{(p)})\lfloor(|q^{(p)}| + \delta)2^p \Delta\rfloor, & q^{(p)} \neq 0. \end{cases}\end{aligned} \qquad (12)$$

2.2. Trellis coded quantization

Trellis coded quantization (TCQ) is based on the ideas of an expanded signal set and set partioning from coded modulation, and has been shown to be an efficient method with modest complexity for encoding of memoryless sources [5]. As mentioned previously, TCQ is included in JPEG 2000 Part II.

A trellis is nothing more than a state transition diagram (that takes time into account) for a finite state machine. Trellises are used to study sequences of state transitions, or equivalently, sequences of states. A typical trellis with 8 states is shown in Fig. 4. In the figure, each column of heavy dots represent the eight possible states at any given point in time. These states are labeled from 0 to 7, from top to bottom. Each branch in the trellis represents a transition from one state to another, at the next point in time. For example, in the figure, two possible transitions from state 1 are to either state 0 or state 4. Specifying a path through the trellis is equivalent to specifying a sequence of states. Given an initial state at $t = 0$, this path can be specified by a binary sequence, since there are only two possible transitions from one state to another in the figure.

For the variant of TCQ included in JPEG 2000, a uniform scalar quantizer is partitioned into four subsets called D_0, D_1, D_2 and D_3. This is illustrated in Fig. 5. The subsets D_i are used to label the branches of a trellis. Fig. 6 shows a single stage of the 8-state trellis used in Fig. 4 with branch labeling. The union of the quantizers associated with each state is called a *union quantizer*. Notice that the two union quantizers used for the trellis in Fig. 6 are $A_0 = D_0 \bigcup D_2$ and $A_1 = D_1 \bigcup D_3$. These quantizers are illustrated in Fig. 7. In Fig. 7, the reconstruction values $\hat{x}$ corresponding to each union quantizer and the corresponding union quantizer indices $q_{(A_i)}$ are shown as well.

The structure of the trellis allows encoding to be done using the Viterbi Algorithm [2]. For encoding a data sequence $\boldsymbol{x} \in \mathbb{R}^m$, an N-state trellis of m stages is employed. Note that such a trellis has $m+1$ columns of states. Let $\mathcal{S}_{i,l}$, $i = 0, 1, \ldots, m$, $l = 0, 1, \ldots, N-1$ denote state l of column i.

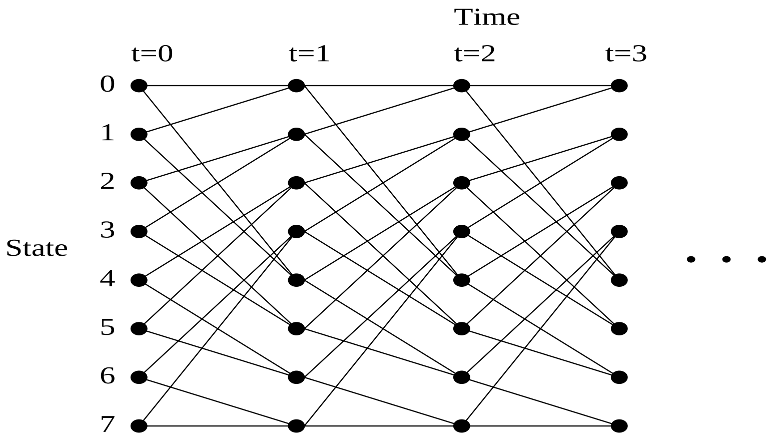

Fig. 4. A typical trellis diagram.

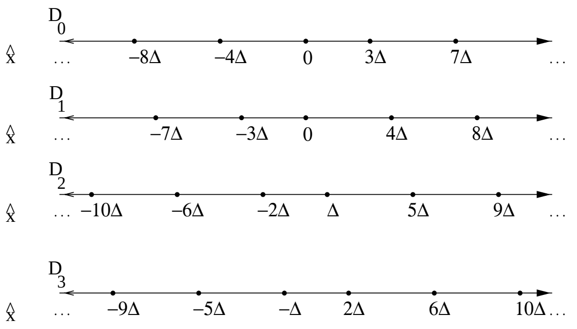

Fig. 5. Scalar quantizers used for TCQ.

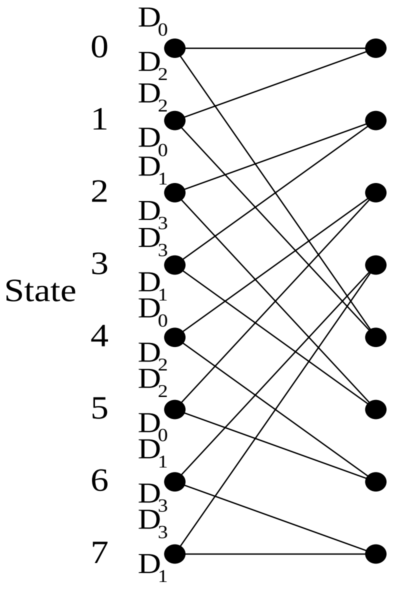

Fig. 6. Branch labeling for TCQ.

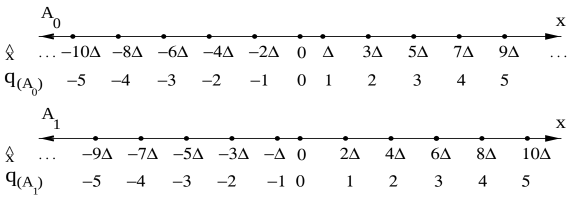
A0
x
x̂ … −10Δ −8Δ −6Δ −4Δ −2Δ 0 Δ 3Δ 5Δ 7Δ 9Δ …
q(A0) −5 −4 −3 −2 −1 0 1 2 3 4 5
A1
x
x̂ … −9Δ −7Δ −5Δ −3Δ −Δ 0 2Δ 4Δ 6Δ 8Δ 10Δ
q(A1) −5 −4 −3 −2 −1 0 1 2 3 4 5

Fig. 7. Union quantizers for TCQ.

For each state $\mathcal{S}_{i+1,l}$ let $\mathcal{S}_{i,l'}$ and $\mathcal{S}_{i,l''}$ be the two states having branches ending in $\mathcal{S}_{i+1,l}$. Also, let $D^{l',l}$ and $D^{l'',l}$ be the subsets associated with those branches, respectively. Let $c_{l',l}$ and $c_{l'',l}$ be the codewords in $D^{l',l}$ and $D^{l'',l}$ that minimize $\rho(x_i, c) = (x_i - c)^2$, and let $d_{l',l} = (x_i - c_{l',l})^2$ and $d_{l'',l} = (x_i - c_{l'',l})^2$. Finally, let $s_{i+1,l}$ be the "survivor distortion" associated with the survivor path at state $\mathcal{S}_{i+1,l}$.

The ith step ($i = 0, \ldots, m-1$) in the Viterbi algorithm then consists of setting $s_{i+1,l} = \min\{s_{i,l'} + d_{l',l}, s_{i,l''} + d_{l'',l}\}$, preserving the branch that achieves this minimum, while deleting the other branch from the trellis. If two values compared for minimum survivor distortion are equal, the "tie" can be resolved arbitrarily with no impact on MSE.

When the end of the data is reached ($i = m-1$), the trellis is traced back from the final state having the lowest survivor distortion, and the corresponding set of TCQ indices are produced. For long data sequences ($m \gg \log_2 N$) the choice of initial state has negligible impact on MSE. Thus, we arbitrarily fix the initial state at 0.

A simple modification can be applied to the TCQ quantization indices obtained as described above, to allow more efficient entropy coding. If we assume that the input sequence x_i has a symmetric probability distribution, it can be seen in Fig. 7 that the quantization indices 1 from A_0 and -1 from A_1 have the same probability. Similarly, the probability of the quantization index -1 from A_0 is equal to that of quantization index 1 from A_1. Thus, we negate the ± 1 indices in A_1 to bring the probabilities of ± 1 indices in both union quantizers into agreement. This allows for more efficient entropy coding of quantization indices. Note that, it is also possible to negate *all* the quantization indices in A_1 [3]. This will bring the probabilities of all the quantization indices in A_0 and A_1 into agreement, and will improve the performance of entropy coding. However, this is avoided since the negation of the indices of A_1 (except ± 1) compromises the embedding property of TCQ, as discussed in the next section.

The dequantization of TCQ indices at the decoder is straightforward. The sequence of indices specifying which codeword was chosen from the appropriate union quantizer at each stage is sufficient to allow the decoder to reproduce the reconstructed values, given the initial state. The dequantization utilizes the same trellis employed at the encoder, and initializes the state index $l = 0$, and the stage index $i = 0$. If the union quantizer used for the current state $\mathcal{S}_{i,l}$ is A_1 and the quantization index is ± 1, the quantization index is negated. The reconstructed value is computed using the union quantizer of the current state. The next state is chosen using the branch labeling of the trellis as follows: if the quantization index of the union quantizer is from subset D_j, the next state is selected by following the branch labeled D_j.

2.2.1. *Embedded trellis coded quantization*

As in the case of SQ, the sign magnitude representation of TCQ indices can be employed to achieve an embedding for TCQ. To see this, first notice that codewords from the two subsets (within a union quantizer) differ in the least significant bit of their index $q_{(A_i)}$. In other words, the least significant bit of the quantization index determines the subset to which the codeword belongs.

For example, in A_0 all the codeword indices for the codewords in D_2 have '1's in their least significant bit, while the indices for codewords in D_0 have '0's. Since the decoder determines the next state by identifying the subset each codeword belongs to, the least significant bit thus determines the path through the trellis. Since the decoder needs to be able to determine the path, it is not possible to invert the TCQ indices precisely until all of the least significant bits are available. However, it is possible to form an approximate reconstruction value in the absence of the p least significant bits of the TCQ quantization index. Similar to the case of SQ, this value can be computed as

$$\hat{x} = \begin{cases} 0, & q^{(p)} = 0, \\ \operatorname{sign}(q^{(p)})(|q^{(p)}| + \delta)2^{p+1}\Delta, & q^{(p)} \neq 0. \end{cases} \tag{13}$$

Note that this operation is equivalent to performing inverse scalar quantization with twice the TCQ step size.

As mentioned in the previous section, negation of the indices of A_1 will improve the performance of entropy coding. However, since the path through the trellis cannot be determined until the least significant bitplane becomes available at the decoder, the decoder cannot identify the codewords belonging to A_1. Thus, this negation will jeopardize the embedding property, and is avoided. Note, however, that the negation of ± 1's in A_1 will not jeopardize embedding. The ± 1's are transmitted in the least significant bitplane and at this point the decoder can determine the path through the trellis. This allows identification of ± 1's that belong to A_1. The reader is referred to [1] for further discussion on embedded coding of TCQ indices.

3. Quantization in JPEG 2000

In JPEG 2000, each resolution of wavelet coefficients is partitioned into *precincts*. Precincts are one of the ingredients to low memory implementations. They also provide a method of spatial random access. Compressed data from a precinct are grouped together to form a *packet*.

Another one of the geometric structures used in JPEG 2000 are *codeblocks*. Codeblocks are formed by partitioning subbands. Since the precinct size (resolution dependent) and codeblock size (resolution independent) are both powers of 2, the two partitions are forced to "line up". Thus, it is reasonable to view the codeblocks as partitions of the precincts (rather than of the subbands).

In JPEG 2000, wavelet coefficients of each codeblock are scanned in a particular order for the purpose of quantization and entropy coding. This order was chosen to facilitate efficient pipelining and/or parallel processing. Fig. 8 illustrates the scan pattern used for a given codeblock. Each codeblock is quantized and entropy coded independently.

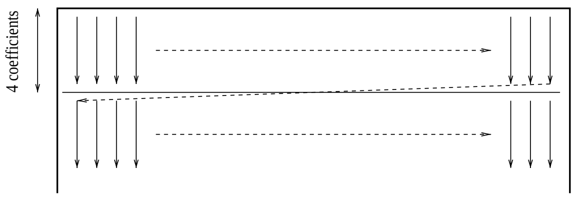

Fig. 8. Scan order for quantization sequence of a codeblock.

Entropy coding is performed using context-dependent, binary, arithmetic coding of bitplanes (For details of entropy coding methods used in JPEG 2000, refer to [6]).

Consider a quantized codeblock to be an array of integers in sign-magnitude representation, then consider a sequence of binary arrays with one bit from each coefficient. The first such array contains the most significant bit (MSB) of all the magnitudes. The second array contains the next MSB of all the magnitudes, continuing in this fashion until the final array which consists of the LSBs of all the magnitudes. These binary arrays are referred to as bitplanes.

The number of all-zero "leading" bitplanes for the codeblock is signaled as side information and bitplanes are encoded starting from the first bitplane having at least a single 1. Note that this way of coding complements embedded quantization and facilitates successive refinement of wavelet coefficients. If $K-p$ of the most significant bitplanes for a codeblock are available at the decoder, the quantization index $q^{(p)}$ can be formed for each wavelet coefficient of the codeblock. The approximate reconstruction values are then given by Eq. (10).

The encoder may take advantage of this embedding facility and choose not to encode p of the LSBs for a particular codeblock. When this occurs, the *effective* quantization step size of this codeblock, $2^p\Delta$, is due to a combination of the step size Δ and encoder truncation. How a desired effective quantization Δ' is generated—either all in the quantization step size ($\Delta' = \Delta$) or some portion occurring via bitstream truncation ($\Delta' = 2^p\Delta$)—is typically an encoder choice.

The ability to transfer much of the quantization into the bitstream truncation process is a powerful tool within JPEG 2000. Since JPEG 2000 codestream construction allows truncation to be controlled on a sub-bitplane level at each code block, coefficients in different areas of the image can have different effective quantization levels.

3.1. Determining amount of embedding

All the embedded inverse quantization formulas rely on a knowledge of p, the number of LSBs unavailable at the decoder. In JPEG 2000 each subband b has a maximum number of magnitude bitplanes that can be decoded, M_b. By counting the number of magnitude bits actually decoded for each coefficient, $N_b(u, v)$, the decoder determines

$$p = M_b - N_b(u, v), \tag{14}$$

where M_b itself is composed of the predicted number of quantized bits, ε_b, and some number of guard bits, G, to accomodate unpredicted overflow, such that

$$M_b = G + \varepsilon_b - 1. \tag{15}$$

Both G and ε_b are chosen by the encoder and signaled in the codestream. A typical value for G is 2. The values ε_b set by the encoder are a function of both the quantization step size and the predicted bitdepth of the wavelet coefficients,

$$R_b = R_I + \log(\text{gain}_b), \tag{16}$$

where R_I is the original image bitdepth and gain_b is the gain expected from the wavelet transform in subband b. The exact relationship between ε_b, the quantization step size, and R_b will become apparent in the next section.

3.2. Selection and signaling of quantization step sizes

Image quality and compression rate are controlled by the amount of quantization applied to each coefficient. JPEG 2000 is a decoder standard so the quantization step size selection is quite flexible. However, there are a few restrictions imposed by the standard.

3.2.1. Reversible wavelets

When reversible wavelets are utilized in JPEG 2000, uniform dead-zone scalar quantization with a step size of $\Delta = 1$ must be used. Reversible wavelets produce integer wavelet coefficients. Thus, a step size of $\Delta = 1$ results in no quantization. If all the bitplanes are encoded, lossless reconstruction is possible once all bitplanes are received by the decoder.

The constant step size $\Delta = 1$ is not signaled to the decoder. Instead, the value $\varepsilon_b = R_b$ is signaled in a 5-bit unsigned integer field. If $R_b > 31$, $\varepsilon_b = 31$ and lossless reconstruction is impossible.

Restricting the step size to 1 does not prevent the encoder from embedding the bitstream to allow intermediate (or even final) lossy results. The effective quantization step size caused by this embedding can be any power of two.

3.2.2. Irreversible wavelets

When irreversible wavelets are utilized in JPEG 2000, the step size selection is restricted only by the signaling syntax itself.

Every subband b has a single quantization step size, Δ_b, that is represented as a two-part quantity, (ε_b, μ_b) such that

$$\Delta_b = \left(1 + \frac{\mu_b}{2^{11}}\right) 2^{R_b - \varepsilon_b}, \tag{17}$$

where μ_b is an 11-bit unsigned integer and ε_b is a 5-bit unsigned integer.

This syntax has several consequences.

- Only one quantization step size per subband. Therefore, Δ_b must be smaller than (or equal to) the effective quantization desired in different areas in the subband. If in doubt, choose a somewhat small quantization step size and then truncate later as required. This also means a single scale factor $(1 + \mu_b/2^{11})$ must be used across the entire subband.
- The accuracy of the step size is restricted to 12 significant binary bits. This can impact convergence of integrative step size refinement techniques and transcoding of step sizes defined outside JPEG 2000.
- Upper bound: $\Delta_b < 2^{R_b+1}$. A step size that is over twice the predicted subband magnitude will quantize almost all subband coefficients to zero. The same effect can be achieved using a reduced step size (set $\varepsilon_b = 0$) and applying encoder truncation.
- Lower bound: $2^{R_b-31} \leqslant \Delta_b$. This bound is more restrictive than the upper bound, since it limits high accuracy coding. For 8-bit image data 21 fractional bits of the HH coefficients can be encoded. This is certainly enough for many compression needs. However, when the image bitdepth becomes large, this bound has more impact. For example, the HH subband of 30-bit image data must be truncated above the original decimal point. When computations are performed in 32-bit registers this does not impact overall performance, but the lower bound must be applied prior to using the step size for any intermediate computations.

The exponent/mantissa pairs (ε_b, μ_b) are either signaled in the codestream for every subband (expounded quantization), or else signaled only for the lowpass subband $(\varepsilon_{LL}, \mu_{LL})$ and derived for all other subbands (derived quantization).

Derived quantization can only be used when the step size exponent/mantissa pairs obey

$$(\varepsilon_b, \mu_b) = (\varepsilon_{LL} - n_{LL} + n_b, \mu_{LL}), \tag{18}$$

where n_b is the number of levels of decomposition required to reach subband b, and the subscript LL indicates the lowpass subband. With the Mallat (or dyadic) decomposition used in JPEG 2000 Part I, this means that the step size decreases by a factor of 2 at each decomposition level. This gives a rough approximation of the step sizes which would be chosen based on high rate theory using the L_2 synthesis gains of the wavelet subbands.

Average performance often improves if the actual L_2 synthesis gains are used [7] to set the relative step sizes. Specifically, if W_b is the L_2 synthesis gain (or "energy weight") of band b, then

$$\Delta_b = \sqrt{\frac{W_0}{W_b}}\Delta_0. \tag{19}$$

When this methodology of step size selection is employed, expounded quantization is usually required.

3.3. Lagrangian rate allocation

As discussed above, when expounded quantization is employed, quantization step sizes can be selected separately for each subband. Since the selection of the quantization step sizes is an encoder issue, a particular selection method is not specified within the standard. One method that can be used to determine these step sizes is described in [3]. In this procedure, every subband of wavelet coefficients is assigned a generalized-Gaussian density (GGD) model. The GGDs are a family of symmetric, unimodal probability density functions (pdf). The zero-mean pdf is given by

$$p(x) = \frac{\alpha}{2\sigma\Gamma(1/\alpha)}\sqrt{\frac{\Gamma(3/\alpha)}{\Gamma(1/\alpha)}} \times \exp\left\{-\left(\sqrt{\frac{\Gamma(3/\alpha)}{\Gamma(1/\alpha)}}\left(\frac{|x|}{\sigma}\right)\right)^{\alpha}\right\}. \qquad (20)$$

The α parameter determines the shape of the function, where $\alpha = 1.0$ corresponds to the Laplacian density, and $\alpha = 2.0$ corresponds to the Gaussian density. In the procedure, five allowable choices for α are 0.5, 0.75, 1.0, 1.5, 2.0. These values have been chosen to cover the extent of the observed pdfs of wavelet coefficients in typical imagery. The parameter α for a given subband is estimated from its sample kurtosis. Given α_i and $d_i(\mathcal{R}_i)$, $i = 1, \ldots, B$, where B is the number of subbands and $d_i(\mathcal{R}_i)$ is the MSE obtained by quantizing at rate $\mathcal{R}_i$ a unit variance GGD source having parameter α_i the rate allocator attempts to minimize the overall MSE constrained by an average rate constraint. For details of this Lagrangian rate allocation strategy, the interested reader is referred to [3].

3.4. *Considerations for TCQ*

For TCQ, the Δ_b signaled in the codestream is actually 2Δ, i.e. twice the step size used in the TCQ description. This allows decoders that have no knowledge of TCQ to decode embedded data correctly using the standard inverse scalar quantizer.

It is important to reiterate that the LSBs of the TCQ indices determine the path through the trellis and are required for precise dequantization. If only a portion of the LSBs are available at the decoder (as would be the case if only one or two of the three LSB coding passes have been received by the decoder), full TCQ dequantization cannot be utilized. For this reason, the following policy is followed in the current verification model (VM):[1] If any of the three coding passes for the LSB of a codeblock is included within a packet, the encoder is forced to include all three passes. This allows full TCQ dequantization to be utilized at the decoder after this particular packet is received. It should be noted, however, that this is an encoder issue, and thus, is not specified by the standard.

TCQ provides visually superior results compared to SQ, when images have fine grain texture. Such an example is illustrated in Figs. 9–11. In Fig. 9, the original image is displayed. Figs. 10 and 11 displays the same image encoded at 0.3 bits/pixel using SQ and TCQ, respectively. By comparing Figs. 10 and 11, it can be seen that texture is better preserved using TCQ.

3.5. *Multi-component and tiled images*

In JPEG 2000, an image is a collection of two-dimensional rectangular arrays of samples. Each of these arrays is called an image component. Tiles are non-overlapping rectangular regions of the image and the array of samples from one component that fall within a tile is called a tile-component. Within JPEG 2000 the (ε_b, μ_b) pairs may be separately chosen for each tile-component, but cannot change within a tile-component. This allows step size choices to vary in different image components (for example intensity versus chromaticity in a color image) and in different tiles.

A signaling priority is used for efficient storage of this quantization information. A single default set of image quantization pairs (ε_{bI}, μ_{bI}) is signaled in the main header and used whenever no other quantization signaling overrides it. Default component quantization pairs may be signaled to override the image default on a particular component. A tile-component using non-default quantization must signal the new quantization as part of its header. The new quantization only effects this particular tile-component.

3.6. *Selection of δ*

The inverse scalar quantizer formulas incorporate a user selectable parameter δ which indicates where in the quantization interval the reconstruction value $\hat{x}$ is placed. A typical assignment of $\delta = 1/2$ will place the reconstruction at the center of the quantization interval. Other considerations such as transform coefficient distribution and

[1] The verification model for JPEG 2000 consists of software implementation of an encoding system, a decoding system, the bit-stream definition and syntax, input and output definitions of each system, operational procedures and documentations.

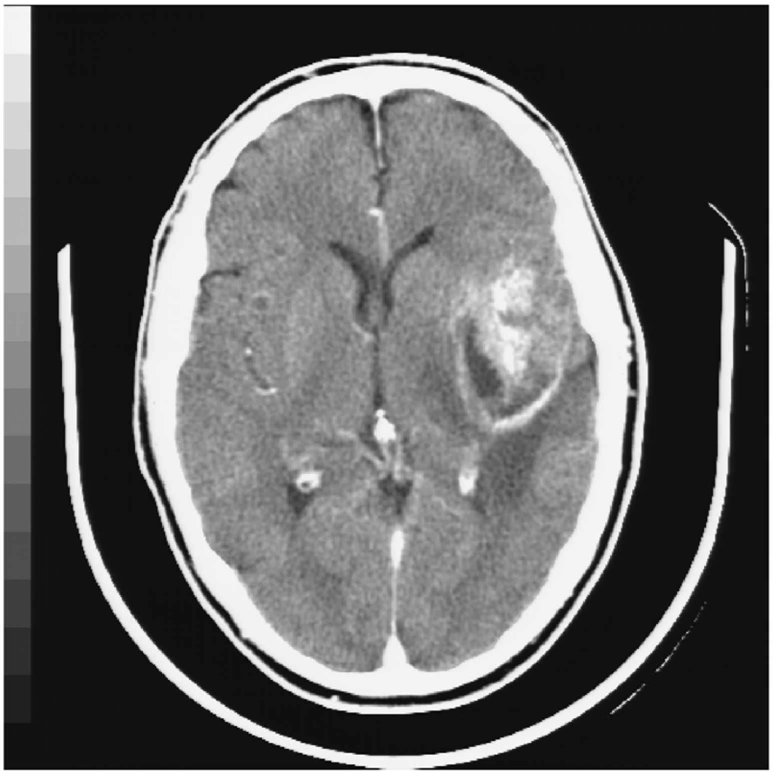

Fig. 9. Original image.

visual appearance may influence the location of the reconstruction value. Since there is no one optimal reconstruction positioning for all images, the JPEG 2000 standard allows the decoder free choice of δ.

Tuning of δ involves a large number of factors including: image type, subband coefficient distribution, number of missing LSBs p, quantization index, neighboring coefficient values, and encoder embedding rules. Over a range of image types, the default $\delta = 1/2$ assignment tends to provide the most reliable reconstruction performance. For further information the interested reader is referred to [4].

3.7. Unusual quantization effects

When there is no change in the image bitdepth between encoding and decoding, JPEG 2000 generates an image with data of the same magnitude as the original. However, there are situations requiring a change in the original image bitdepth. For example, maximum display depth restrictions or display of binary data with high contrast. Sometimes this is a decoder choice, but in other circumstances the encoder or fileserver may force the change within the compressed codestream. This section explores the effects of such changes.

Let R_{IS} be the bitdepth of the source image and R_{ID} be the image bitdepth signaled to the decoder. Reversibly and irreversibly transformed data react very differently when $R_{\mathrm{IS}} \neq R_{\mathrm{ID}}$.

3.7.1. Irreversible transform

When the transform is irreversible the decoder quantization step size is a function of R_{b}, which in turn is computed from the reported image bitdepth R_{ID}. If $R_{\mathrm{ID}} > R_{\mathrm{IS}}$ the decoder will see a larger step size than the encoder and if $R_{\mathrm{ID}} < R_{\mathrm{IS}}$ a smaller one. This difference translates into either a gain or loss in magnitude of the reconstructed image values. So altering the signaled image bitdepth causes the most significant bits of the input data to become the most

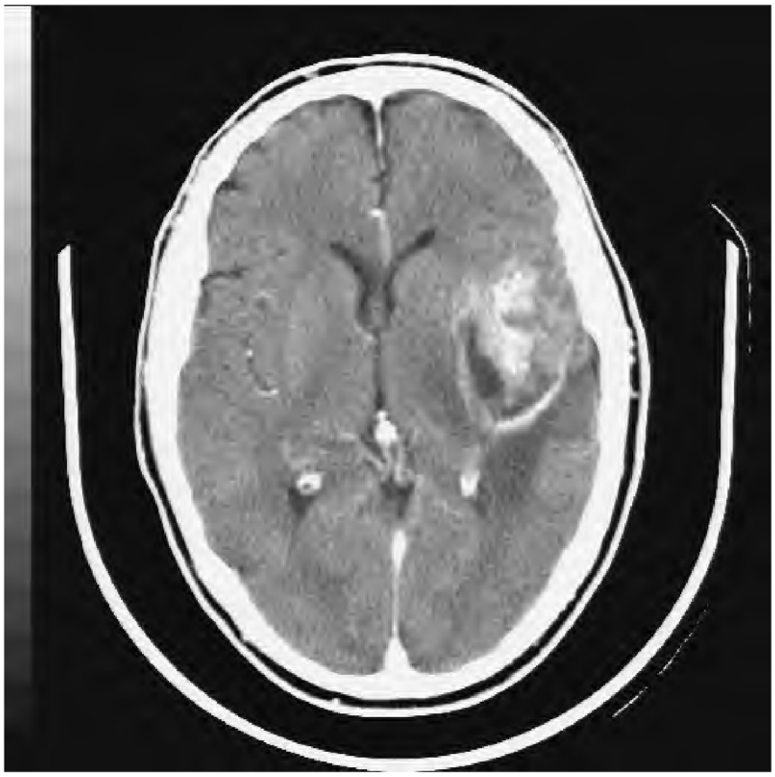
Fig. 10. SQ quantized image at 0.3 bits/pixel.

significant bits in the decoded output data without any extra manipulation.

For instance, using this manipulation 12-bit imagery can be scaled down to 8 bits and reduced bitdepth imagery can be stretched to 8 bits.

3.7.2. *Reversible transform*

When the transform is reversible, the step size is always 1 and the signaled image bitdepth has no impact on the magnitude of the inverse transformed image data. Instead the magnitude of the inverse transformed image data is proportional to the signaled values ε_b. This is exactly the reverse of the situation with irreversible transforms.

If ε_b is set as anticipated ($\varepsilon_b = R_{IS}+\log_2(\text{gain}_b)$) the inverse transform will indeed act reversibly, but the reconstructed image bitdepth will be R_{IS}. In other words, the decoder will create the original image bitdepth regardless of the signaled image bitdepth. If the image bitdepth is forced upwards, say from 1 to 8, the result will be an 8-bit image with significant values only at the very lowest bitplanes (values of 0, 1 and possibly 2). If instead the image bitdepth is forced down (say from 12 to 8), the output of the inverse transform will still have the higher number of bits (12), and the application will need to take extra steps unspecified by the standard to decide which bits should be output.

If the encoder attempts to modify ε_b to force scaling of the decompressed image data, another problem will occur. Any time $\varepsilon_b \neq R_{IS}$ the reversible property of the transform is lost since rounding/truncation no longer occurs at the correct bitplane. For example, lowering all the values of ε_b by a fixed value k, will cause the output magnitude to be reduced by k bits, but will also cause errors in the inverse transform that would not be present if the decoder application independently reduced the bitdepth after the transform. Likewise, increasing ε_b by a fixed value k will stretch the data by 2^k, but will also introduce errors in the lower level bitplane values. For

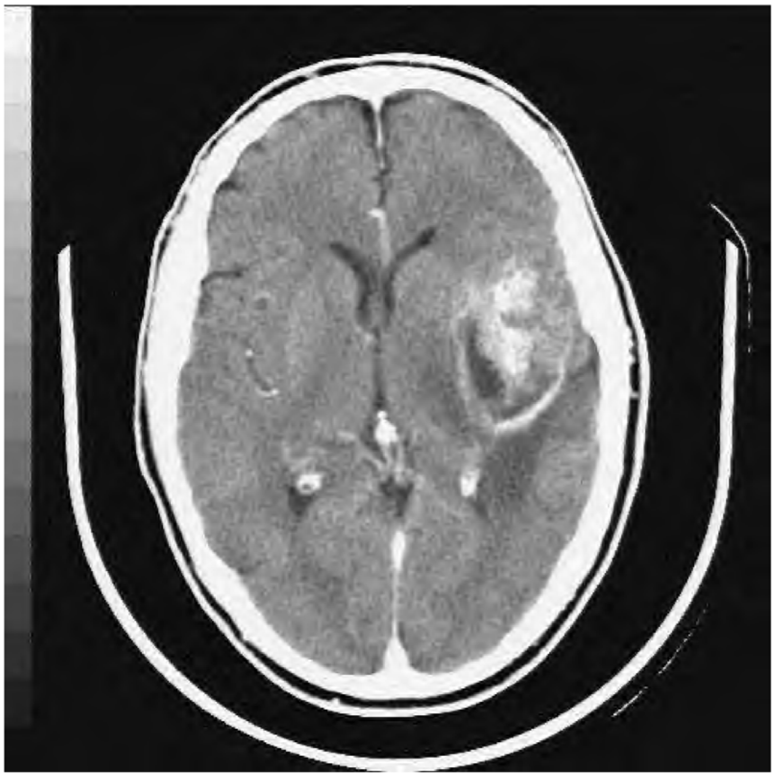
Fig. 11. TCQ quantized image at 0.3 bits/pixel.

example 1-bit data might be stretched to 8-bit via ε_b and R_{ID} signaling, but the output image will not be strictly binary even if all data are decoded.

4. Summary

Since JPEG 2000 is intended to serve a diverse set of applications, several quantization options are provided within the standard. Part I of the standard includes uniform scalar dead-zone quantization. Part II extends the quantization options to include generalized uniform scalar dead-zone quantization and trellis coded quantization. All of the quantization methods support embedded coding and are conducive to the rich feature set of JPEG 2000.

In Section 2 of this paper, the quantization methods used in the standard are reviewed. In Section 3, several issues pertaining to the use of these methods within the standard are discussed. These issues range from selection and signaling of the quantization parameters to dealing with the unusual quantization effects when the image bitdepth changes between encoding and decoding.

References

[1] A. Bilgin, P.J. Sementilli, M.W. Marcellin, Progressive image coding using trellis coded quantization, IEEE Trans. Image Process. 8 (11) (1999) 1638–1643.

[2] J.G.D. Forney, The Viterbi algorithm, Proc. IEEE (Invited Paper) 61 (1973) 268–278.

[3] J.H. Kasner, M.W. Marcellin, B.R. Hunt, Universal trellis coded quantization, IEEE Trans. Image Process. 8 (12) (1999) 1677–1687.

[4] M.A. Lepley, Tuning JPEG 2000 decompression performance via rounding: theory & practice, in: Proceedings of SPIE, Applications of Digital Image Processing XXIII, Vol. 4115, 2000.

[5] M.W. Marcellin, T.R. Fischer, Trellis coded quantization of memoryless and Gauss–Markov sources, IEEE Trans. Commun. 38 (1990) 82–93.

[6] D. Taubman, E. Ordentlich, M. Weinberger, G. Seroussi, Embedded block coding in JPEG 2000, Signal Processing: Image Communication 17 (1) (2002) 49–72.

[7] J.W. Woods, T. Naveen, A filter based bit allocation scheme for subband compression of HDTV, IEEE Trans. Image Process. 1 (1992) 436–440.