BlinQS: Blind quality scalable image compression algorithm without using PCRD optimization

Naveen Cheggoju[1] · Vishal Satpute[2]

Abstract
Quality Scalability is one of the important features of interactive imaging to obtain better perceptual quality at a specified target bit rate. In JPEG 2000, it is achieved using quality layers obtained by Rate-Distortion (R-D) optimization techniques in Tier-II coding. Some important concerns here are: (i) inefficient conventional Post-Compression Rate-Distortion (PCRD) optimization algorithms, (ii) lack of quality scalability for less or single quality layer string. This paper takes the above mentioned concerns into account and proposes a Blind Quality Scalable (BlinQS) algorithm that provides scalability with the least computational complexity. The novel part of this method is to eliminate the Tier-II coding and add a blind string selection i.e., transcoding algorithm through a normal distribution function for efficient rate control. The results obtained suggest that the proposed method achieves better results than JPEG-2000 at single quality layer and achieves results close to JPEG-2000 without using PCRD optimization algorithms.

Keywords Blind quality scalability (BlinQS) · Image compression ·
Rate-distortion optimization · JPEG-2000 standard

1 Introduction

Scalability is one of the main features of any interactive device. Scalability may refer to adaptation in size, shape, quality, rate, etc. In the field of image compression, scalability refers to adaptation in resolution, rate, quality and component. Among these, rate scalability and quality scalability need to achieve a good trade-off for maintaining the image

✉ Naveen Cheggoju
naveen.c@iiitu.ac.in

Vishal Satpute
vrsatpute@ece.vnit.ac.in

[1] Department of Electronics and Communication Engineering, School of Electroics, Indian Institute of Information Technology, Una, Haroli, Una, 177209, Himachal Pradesh, India

[2] Electronics and Communication Engineering, Visvesvaraya National Institute of Technology, Nagpur, South Ambazari, Nagpur, 440010, Maharashtra, India
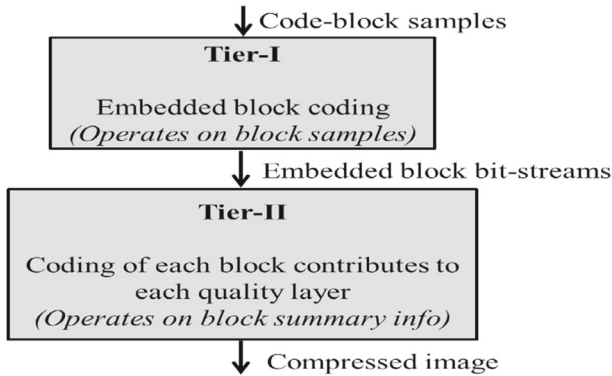
Fig. 1 Coding blocks of JPEG-2000

quality at a specified or required target rate. This trade-off has been addressed in the new image compression standard JPEG-2000 using the concept of *quality layers* [38, 40]. These layers are generated in iterative manner using Post-Compression Rate-Distortion (PCRD) optimization algorithms for all the individual code-blocks. To implement R-D optimization algorithms, JPEG-2000 creates a Tier-II coding system module as shown in Fig. 1 [38]. This module takes the block summary of each code-block as the inputs and arranges them in increasing order of quality for the specified target rates. Some important concerns of this coding are: (i) inefficient conventional Post-Compression Rate-Distortion (PCRD) optimization algorithms (iterative in nature), (ii) lack of quality scalability for less or single quality layer string. Research communities around the globe have been carrying out research to address these key issues in scalable compression. This has been the prime obstacle of JPEG-2000 widespread adoption in entertainment and broadcast sectors [41].

For interactive imaging in entertainment and broadcast sectors, scalability refers to two targets: (i) low computational complexity for fast processing, and, (ii) highly independent data rearrangement to achieve optimal quality at any required rate. JPEG-2000 has attempted to achieve this and earned more popularity in the field of interactive imaging, especially in the cloud-based content distribution applications. Having earned all the praises for its flexibility, JPEG-2000 has its own limitations pertaining to the computational complexity of the block coding (PCRD) algorithms. This has been the prime obstacle of JPEG-2000 widespread adoption in entertainment and broadcast sectors [41]. To break through the obstacles and make JPEG-2000 more flexible researchers throughout the world are continuing their research by keeping the following as the prime targets:

1. low computational complexity for fast processing, and,
2. highly independent data rearrangement to achieve optimal quality at any required rate.

To achieve scalability at low computational complexity, choosing effective optimization algorithm is necessary. This challange has been taken up by [14, 17, 23, 48] by coding the data and obtain the rate-disortion simultaneously . However, these algorithms fail to decrease the computational load on the encoder. To further reduce the computational complexity, wavelet data based and step size based rate-distortion algorithms have been presented in [30] and [20] respectively. Later, other approaches including Lagrange multiplier have been proposed in [1, 49]. In [47], authors have proposed three rate control methods (successive bit-plane rate allocation (SBRA), priority scanning rate allocation (PSRA)

and priority scanning with optimal truncation (PSOT)) over PCRD algorithms for reducing computational complexity and memory usage. These rate control methods have provided different trade-off among quality, complexity, memory utilization and coding delay. In [19], a low complexity R-D optimization method based on a reverse order for resolution levels and coding passes has been proposed. This method has attained a comparable quality performance with the conventional method, maintaining low complexity. These algorithms have proven to be efficient, but, high throughput has not been achieved. In [41], D. Taubman et. al, have coined the term FBCOT (Fast Block Coding with Optimized Truncation), to widespread the adoption of JPEG-2000 in entertainment and broadcast sectors. This JPEG-2000 compatible algorithm has sole target of increasing the throughput by reducing the computational complexity. This algorithm offers 10X or higher speed compared to the previous algorithms with a slight sacrifice in the coding efficiency. In order to optimize JPEG-2000 for image transmission through wireless networks, Joint Source-Channel Coding (JSCC) has been proposed in [10] and congestion control for interactive applications over SDN networks has been proposed in [28]. Further detailed study of rate-distortion optimization for JPEG-2000 is found in [9, 29]. These R-D optimization algorithms have been significant in reducing the computational complexity but the problem of scalability for any required rate remains unsolved.

To solve this problem of scalability for any bit rate, there is a need to develop algorithms which do not rely on the rate-distortion algorithms at the encoder, rather calculate the quality layers at the transcoder without actually having the knowledge of the code-block information. One such method has been proposed in [6], in which characterisation of code-blocks does not depend on the distortion measures related to the original image. The method proposed in [6], has been inspired by the algorithms presented in [5, 8], which also speak about achieving the better quality of reconstruction when there is a compressed string with single quality layer or less number of quality layers. As this method is computationally expensive another method called Block-Wise Layer Truncation (BWLT) has been proposed in [7]. The main insight behind BWLT is to dismantle and reassemble the to-be-fragmented layer by selecting the most relevant codestream segments of codeblocks within that layer. All these methods are targetted to achieve optimum scalability for single layered or less number of quality layered strings. In [3] and [4], authors focussed on proposing new estimators to approximate the distortion produced by the successive coding of transform coefficients in bitplane image coders. Recently CNN based lossy image compression with multiple bit-rate has been proposed in [13]. This paper focusses on learning multiple bit-rates from a single CNN using Tuceker Decomposition Network (TDNet). The pace of research on scalable image compression has picked up exponentially in the recent years. Some of the latest literature on scalable image compression using machine learning include [15, 21, 24, 25, 37, 42, 46]. Along with these approaches, a new modification to JPEG-2000 has been introduced to achieve high throughput and to reduce the latency. This new approach for JPEG-2000 is termed as High-throughput JPEG-2000 (HTJ2K) [18, 26, 27, 31, 45]. By taking all these inputs from the worldwide researchers into account, the authors have proposed a new blind quality scalable algorithm in this paper.

Investigation has been done in modification of bit plane strategies using several theoretical-practical mechanisms conceived from rate-distortion theory. The research work presented in this paper, is mainly focussed on 2 entities:

- Reducing the encoding complexity and increasing the throughput.
- Achieving scalability for even a single-layered string without using PCRD optimization algorithms.

To achieve this, a strong decision making is necessary at the transcoder, to optimally choose the code-blocks and the truncation points.

Further the paper is organized in the following order: Section 2 discusses R-D optimizaion and quality scalable algorithm used in JPEG-2000, Section 3 introduces the proposed method for achieving Blind Quality Scalable Image compression, Section 4 presents comparative analysis of BlinQS with the JPEG-2000 standard and finally, Section 5 draws the conclusion of the work followed by the references.

2 Quality scaling in JPEG-2000

In JPEG-2000, quality scalability is achieved by arranging the obtained bit streams in the form of layers as shown in Fig. 2 [11]. To get the clear interpretation of quality layer, the basic terms code-block and sub-band are indicated in Fig. 3. Each quality layer contains the truncation point for each code-block, thus having an interpretation of the overall image quality. As per the experimentation done for JPEG-2000, it is found that the number of quality layers should be approximately twice the number of sub-bit-planes to achieve optimal quality scalability. Increased number of layers may create the same quality reconstructed at different rates which are approximately same, causing an increase in the overhead [11]. Hence, the practice of more number of quality layers is not followed in JPEG-2000.

2.1 Rate-distortion (R-D) optimization algorithm in JPEG-2000

Rate (R) and Distortion (D) in JPEG-2000 should satisfy the equations (1) and (2) [11],

$$D = \Sigma_i D_i^{n_i} \tag{1}$$

$$R = \Sigma_i R_i^{n_i} \tag{2}$$

where, i represents the current code-block number, n_i is the truncation point for the code-block B_i. Here, the main target is to find the values of n_i, which minimizes D corresponding to constrained target rate $R_{max} \geq R$. This optimization problem can be solved by minimizing
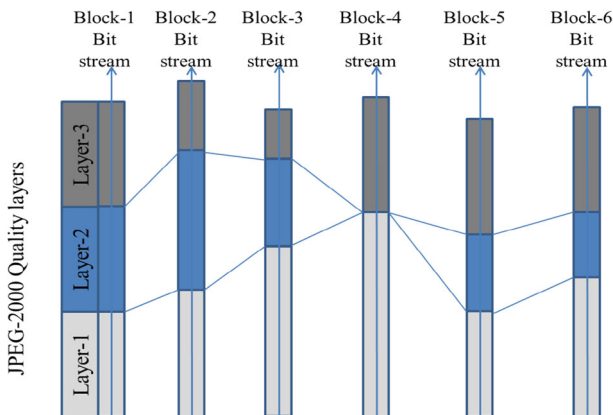

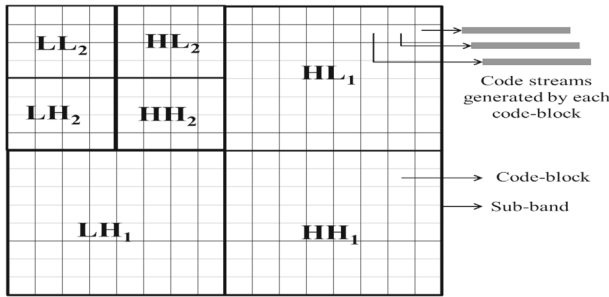**Fig. 2** Illustration of quality layers in JPEG-2000

Fig. 3 Illustration of code-blocks and sub-bands in JPEG-2000

(3) which is obtained by using the well-known method of Lagrange multipliers,

$$\Sigma(R_i^{n_i} - \lambda D_i^{n_i}) \tag{3}$$

where, λ is the Lagrange multiplier which should be varied until the target rate is achieved with minimum distortion. To obtain the values of λ, Algorithm-1 need to be followed [11]. This algorithm is an iterative method to obtain the best possible values of λ which achieves optimum distortion for a required target rate. To form progressive quality layers, n_i obtained for the calculated values of λ are taken as the truncation points and the bits are arranged accordingly.

Require: λ

1:   $n_i = 0$
2:   **for** $k = 1, 2, 3 \ldots$ **do**
3:       $\Delta R_i^k = R_i^k - R_i^{n_i}$
4:       $\Delta D_i^k = D_i^k - D_i^{n_i}$
5:       **if** $\frac{\Delta D_i^k}{\Delta R_i^k} > \lambda^{-1}$ **then**
6:          set $n_i = k$
7:       **end if**
8:   **end for**

Algorithm 1 Procedure get the value of λ.

3 BlinQS: Blind quality scalability algorithm

This section introduces the proposed BlinQS image compression algorithm to achieve optimum quality at target rate without using PCRD algorithms. As discussed in Section 2, R-D optimization algorithms used for generating quality layers are iterative in nature, thus they increase the computational complexity. BlinQS aims to bypass the R-D optimization algorithm and achieve near optimal quality, thus reducing the computation load on the encoder. This has been achieved by using blind selection of the code-blocks obtained through gaussian normal distribution. It has shown good approximation in chosing the code-blocks optimally for required target rate. This is because, the code-blocks are arranged in the order

of descending information content and selected depending on the variance boundary in which the code-block falls. As this operation is computationally effective, it does not create any prominent load on the transcoder, where BlinQS algorithm has been placed. Thus the aim of reducing the computational complexity and memoryless scalability has been achieved using BlinQS algorithm. Complete algorithm is explained in three sections: (i) Encoding, (ii) Inclusion map, and, (iii) Decoding. BlinQS is a part of the transcoder which forms the quality layers blindly from the encoded string using the inclusion map. Main tasks to be performed by the BlinQS trancoder on the received string are:

1. Getting the value of δ_b for each sub-band.
2. Calculating the truncation point of the code-blocks.

3.1 BlinQS: Encoding

Encoding module consists of three sub-modules, (i) Image Transformation, (ii) Image Compression using Set Partition In Hierarchical Trees (SPIHT), and, (iii) String arrangement along with the header. The encoding procedure is briefly explained in Fig. 4a.

3.1.1 Image tranformation

As shown in Fig. 4a, the first step of BlinQS encoding algorithm is to transform the image using Discrete Wavelet Transform (DWT). To decompose the image, '*Biorthogonal 4.4 (bior4.4)*' wavelet family has been used [12]. Before decomposing, an intensity level shift of -127 is performed on the image and then it is decomposed using DWT into sub-bands.

After decomposition, the sub-bands are divided into code-blocks as illustrated in Fig. 3, which can be considered as the building blocks of the coding. They are encoded using the SPIHT algorithm. To improve the efficiency of SPIHT, DWT is applied for each code-block in High-pass region using the same *'bior4.4'* wavelet family before encoding. This has shown substantial improvement in terms of compression ratio.

3.1.2 Image compression using SPIHT

SPIHT is an improvement to Embedded Zerotress of Wavelet (EZW) having main characteristics of SPIHT that include: efficiency, self-adaptiveness, precise rate-control, simple and fast, and fully embedded output [32–34]. SPIHT directly provides the binary output, hence, there is no need of using another algorithm for converting bits to symbols [22].

3.1.3 Quantization factor (δ_b)

SPIHT algorithm is applied on each code-block (CB), after quantizing them by a factor of predefined quantization parameter (δ_b) (obtained from Algorithm-2), to obtain the compressed string of the corresponding code-block. Here, quantization parameter (δ_b) is the function of encoding sub-band $\delta_b = f(SB)$, i.e., the quantization factor depends on, in which sub-band the code-block is present as shown in Fig. 5. Let δ_b denote the quantization parameter for sub-band SB_b, where b denotes DWT level of the sub-band. Each subband has different δ_b value, which is calculated as per Algorithm-2. Higher the value of δ_b, lesser the string length (L_i) generated for the code-block B_i of the sub-band SB_b and higher the
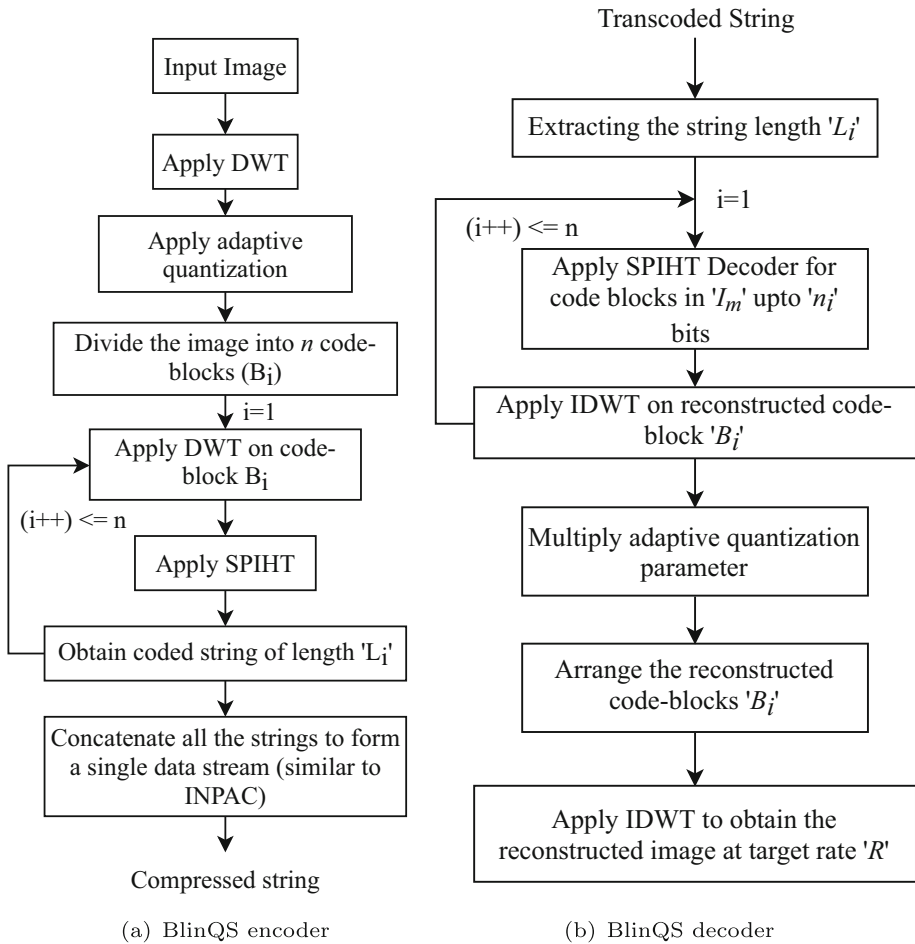
Fig. 4 Flow chart of proposed BlinQS image compression algorithm

quantization error (Q_e) i.e, $\delta_b \propto \frac{1}{L_i} \propto Q_e$. Therefore, to maintain the reconstruction quality of image, LL components are not quantized before coding (i.e., $\delta_b = 1$) and rest of the image is quantized with $\delta_b > 1$ upto a maximum point δ_{max}. Hence, δ_b is adapted with the sub-band in which the process of quantization is going on, which is termed as adaptive delta (δ_{adap}).

Let 'L_i' denote the length of the compressed string obtained from the code-block 'B_i', and 'L_{ip}' denote the length of compressed string obtained from bit plane 'p' of code-block 'B_i'. For a given code-block 'B_i', the total compressed string length 'L_i' is the summation of the individual 'L_{ip}' from each bit plane as mentioned in equation (4) and complete string length 'L' is given by (5). Before transmitting or storing this string, header is formed for ease of access and flexibility in operation, details of which are given in Section 3.1.4.

$$L_i = \sum_{p=1}^{No. \ of \ bit \ planes} L_{ip} \qquad (4)$$

Require: δ_b
1: δ_{max} (Maximum value of δ_b)
2: $\delta_b = 1$, M_{SB} = Maximum sub-band level
3: **for** $b = M_{SB}, M_{SB} - 1, M_{SB} - 2 \dots$ **do**
4:      $\delta_b = \delta_b + 1$
5:      **if** $\delta_b \geq \delta_{max}$ **then**
6:          break;
7:      **end if**
8: **end for**

Algorithm 2 Finding δ_b.

$$L = \sum_{i=1}^{No.\ of\ codeblocks} L_i \qquad (5)$$

where, N_p represents number of planes in the transformed image and N_c represents the number of code-blocks.
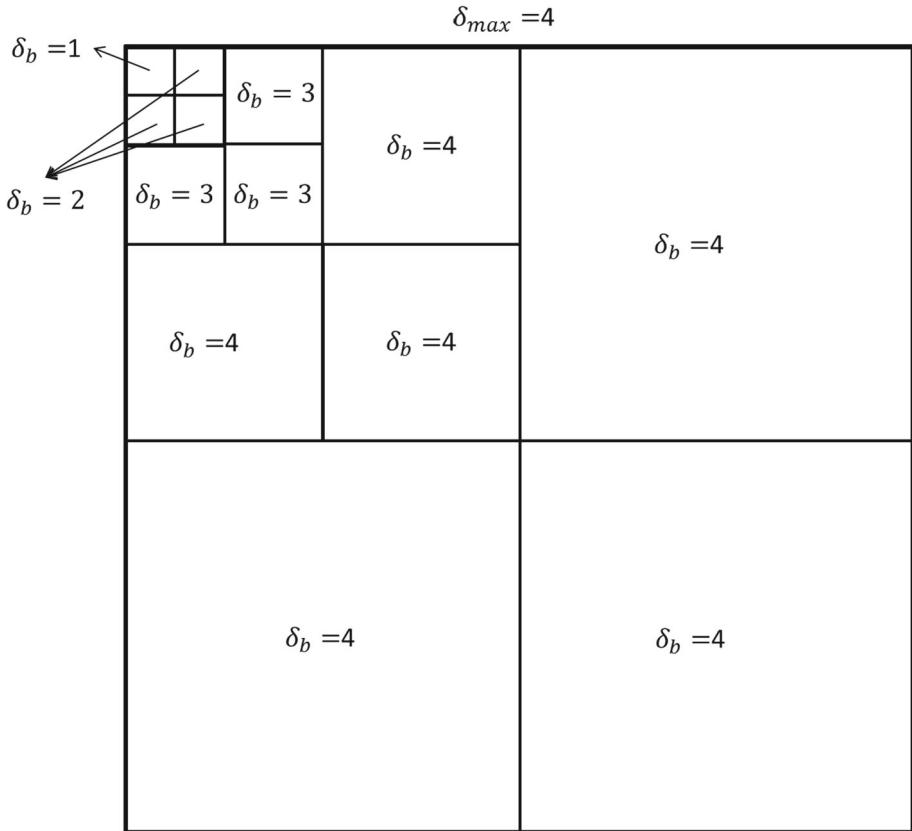
Fig. 5 Representation of δ_{adap} selection based on sub-band

Compressed string with header

Get lengths 'L_i' from the header

Compute % contribution 'PL_i'

Find norm 'Xi' for each value of 'PL_i'

Check for the required bit rate 'R_s in bpp'

Standard Rate

No

Yes

Get 'j' and 'k' from Algorithm 3

Calculate x_{new}

Plot 'X_i' vs 'PL_i' and find 'I_m'

Get 'n_i' from 'I_m'

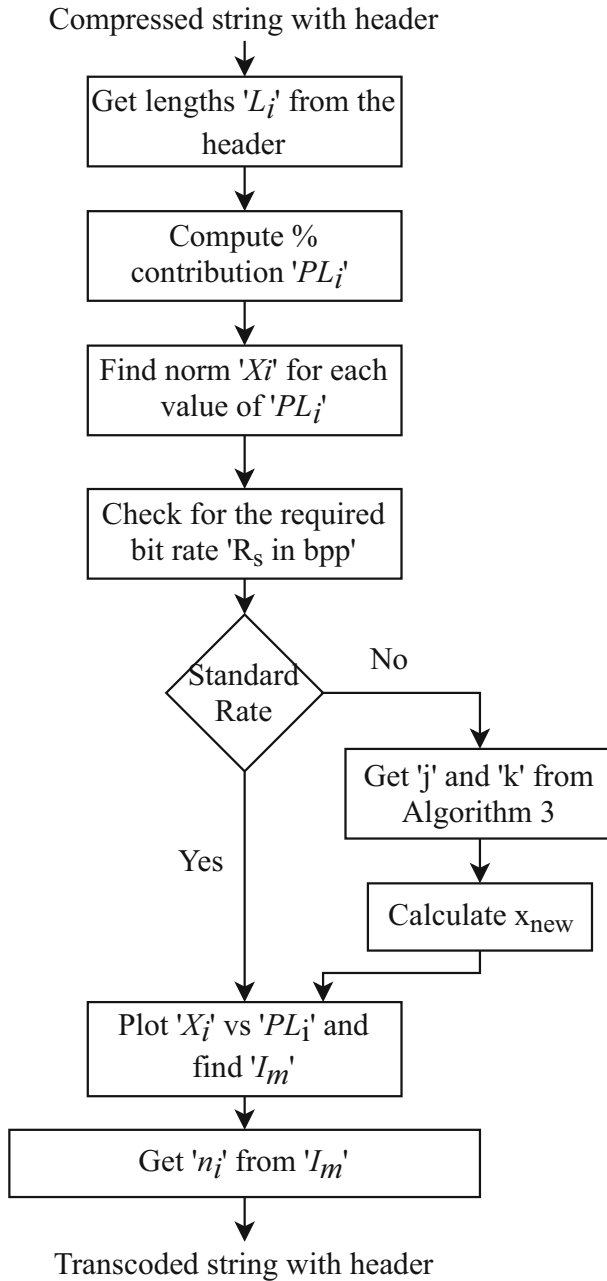Transcoded string with header

Fig. 6 BlinQS inclusion map selection

3.1.4 String arrangement along with header

The lengths L_{ip}, obtained from the SPIHT encoder are arranged in the order of their sub-bands i.e., LL_2, HL_2, LH_2 and so on. While arranging the string in header, strings obtained from each bit plane 'p' of code-block 'B_i' is considered as a separate entity and is placed in the header along with marker bytes which stores the length of the string i.e., L_{ip}. Therefore, effective length of the string stored is 'Bits occupied by string + marker bytes'. Along with this information, basic information such as size of image, level of DWT applied on image, code-block size etc, are appended to the header, which forms the basic information header. Using the basic information header, marker byte lengths are extracted first at the transcoder to form the quality layers before decoding, which is discussed in brief in Section 3.2.

3.2 BlinQS: Inclusion map

Blind Quality Scalability refers to *"obtaining the inclusion map and truncation points for the optimum reconstruction of an image at a specified bit rate."* An overview of this algorithm is given in Fig. 6. Inclusion map (I_m) is a matrix which consists of the information regarding code-blocks needed for decoding the image for the specified rate. The procedure for obtaining the inclusion map generation is briefly discussed in the following steps.

3.2.1 Calculation of string contribution: Step-1

Calculate the precentage contribution (PL_i) of each compressed code-block (CB) using (6), where i=1,2,..., N_c (Number of code-blocks).

$$PL_i = \left(\frac{L_i}{L}\right) * 100 \tag{6}$$

3.2.2 Calculation of Normal Distribution Coefficients: Step-2

Find the normal distribution coefficients (X) of array 'PL' using the mean (μ) and variance (σ^2) of the elements in the array using the (7), (8) and (9).

$$\mu = \frac{\sum PL_i}{N_c} \tag{7}$$

$$\sigma^2 = \frac{\sum_i (PL_i - \mu)^2}{N_c} \tag{8}$$

$$X_i = f(PL_i/\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(PL_i-\mu)^2}{2\sigma^2}} \tag{9}$$

3.2.3 Obtaining the code-blocks: Step-3

Plot the values of 'X_i', against the percentage lengths (PL_i) obtained from (6), i.e. X_i vs PL_i and divide the plot by taking the step-size of $x_s\sigma$. This plot for '$Lena$' image is shown in Fig. 7. Here, the bit rates mentioned in set $R_{std} = \{0.0625, 0.125, 0.25, 0.5, 1.0, 2.0\}$ are considered as standard bit rates for which $x_s = 1$, where s ϵ [1,5], and for bit rates $R_s < R_{new} < R_{s+1}$, a new term x_{new} is introduced, which can be found out using
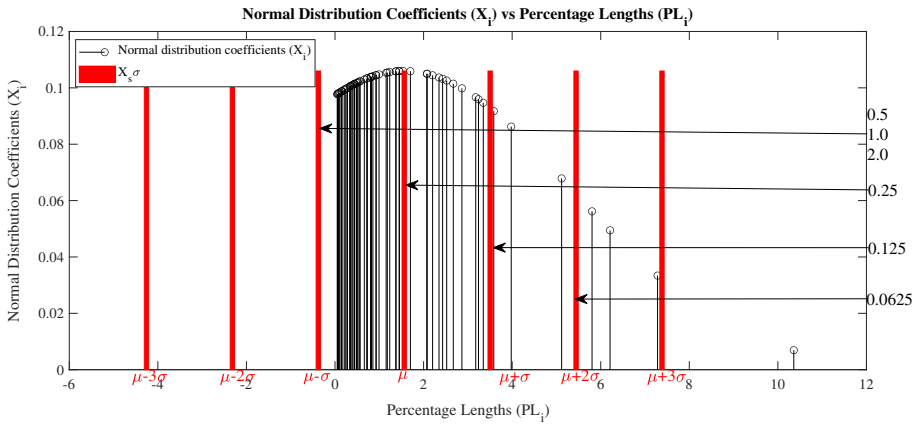
Fig. 7 Normal distribution of percentage lengths. Code-blocks to the right of the arrow head are considered for reconstruction for the rate (bpp) given at the arrow tail

Algorithm-3. $x_{new} \epsilon (0, 1)$ acts as the local step-size generator between R_s and R_{s+1} to precisely choose the code-blocks B_i for the new rate R_{new}. Standard rates are addressed here as $R_1, R_2, \ldots, R_6$ and the partition number for each rate is denoted by loc_{R_s}. From normal distribution plot in Fig. 7, inclusion map for standard rates can be obtained as indicated by the arrows i.e., for 0.0625: code-blocks upto $\mu + 2\sigma$ (i.e. $loc_{R_1} = 2$), for 0.125: code-blocks upto $\mu + 1\sigma$ (i.e. $loc_{R_2} = 3$) and so on. x_{new} plays a major role in obtaining the inclusion map for R_{new}. If $x_{new} \epsilon (0,1)$, total number of partitions increase by a factor of 'k'. Therefore, value of $loc_{R_{new}}$ can be obtained from (10), where, $s \ and \ j$ are indicated in Algorithm-3.

$$loc_{R_{new}} = loc_{R_s} + j \quad if \ \Delta_L < \Delta_H \tag{10a}$$

$$loc_{R_{new}} = loc_{R_s} + (k - j) \quad if \ \Delta_L > \Delta_H \tag{10b}$$

3.2.4 Calculation of inclusion map: Step-4

After obtaining $loc_{R_{new}}$ and x_{new} values, all the code-block strings present ahead of $loc_{R_{new}}$ are included in the inclusion map (I_m). The included code-blocks can be determined from (11) as indicated below.

$$I_m = CB_i \ \forall f^{-1}(X_i/\mu, \sigma^2) \geq (x_s + x_{new})\sigma \quad if \ \Delta_L < \Delta_H \tag{11a}$$

$$I_m = CB_i \ \forall f^{-1}(X_i/\mu, \sigma^2) \geq (x_{s+1} - x_{new})\sigma \quad if \ \Delta_L > \Delta_H \tag{11b}$$

3.2.5 Calculation of truncation points: Step-5

The truncation points are obtained by solving (12), which considers the code-blocks in the inclusion map (I_m).

After obtaining the inclusion map, truncation points 'n_i' for each code-block 'B_i' in the inclusion map have to be identified for a specified target rate 'R_{max}'. The truncation points obtained here must follow the conditions specified in (2) and (12). On solving (12),

Require: $x_{new} = \frac{j}{k+1}$, where, $j, k \in Z$
1: known values, $R_{std}, R_{new}, k_{max}$
2: find the location of R_{new}
3: $R_s < R_{new} < R_{s+1}, s \in [1,5]$
4: get $\Delta_L = \mid R_s - R_{new} \mid$ and $\Delta_H = \mid R_{s+1} - R_{new} \mid$
5: assign $k = 1$
6: **while** $k \le k_{max}$ **do**
7:      assign $j = 1$
8:      **while** $j \le k$ **do**
9:          **if** $\beta_j > T$ **then**
10:             **if** $\Delta_L < \Delta_H$ **then**
11:                 $R_{new_k} = R_s + (R_{s+1} - R_s) * \frac{j}{k+1}$
12:             **end if**
13:             **if** $\Delta_L > \Delta_H$ **then**
14:                 $R_{new_k} = R_{s+1} - (R_{s+1} - R_s) * \frac{j}{k+1}$
15:             **end if**
16:             $\beta_j = \mid R_{new} - R_{new_k} \mid$
17:             $j++$
18:         **else**
19:             break;
20:         **end if**
21:     **end while**
22:     $k++$
23: **end while**
24: obtain j and k
25: **if** $\Delta_L == \Delta_H$ **then**
26:     assign $k = 1$ and $j = 1$
27: **end if**

Algorithm 3 Finding x_{new}.

$n_i \approx R/\Sigma_i L_i$, where, i follows the values obtained from the inclusion map. On obtaining the values of n_i for each code-block in the inclusion map, a new header with the obtained lengths and string is formed and sent to decoder for image decoding.

3.3 BlinQS: Decoding

Firstly, in decoding module, bits received are separated and required information is extracted from the header formed in Section 3.2. Functionalities of the decoder are summarized in the flowchart given in Fig. 6. Value of δ_b for each sub-band is calculated as per the algorithm given in Algorithm-2. After obtaining the adaptive delta, truncation points 'n_i' for each code-block 'B_i' have to be identified for a specified target rate 'R_{max}'. The truncation points obtained here must follow the conditions specified in (12).

$$\sum_i L_i * n_i = R \le R_{max} \tag{12}$$

On solving (12), $n_i \approx R/\Sigma_i L_i$. On obtaining the values of n_i for each code-block, SPIHT decoding is applied to the respective blocks up to the truncation points 'n_i'. Inverse
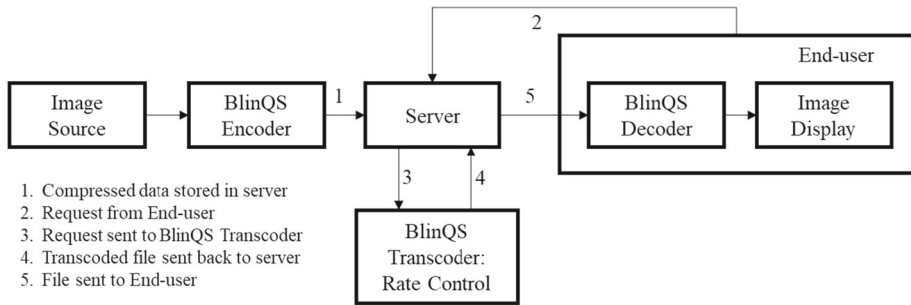
Fig. 8 Overall block diagram of the proposed work

SPIHT is applied on the blocks in the inclusion map upto the obtained truncation points. δ_b obtained from Algorithm-2 is multiplied with the corresponding block values and then Inverse Discrete Wavelet Transform (IDWT) is applied using *'bior4.4'* wavelet family and arranged in image format. The blocks which are not available in the inclusion map are filled with zeros and image is transformed into spatial domain by applying IDWT. And finally, the transformed image is level shifted by "+127" to get the reconstructed image. The general block diagram of the proposed in presented in Fig. 8.

4 Results and discussions

This section presents the performance comparison of BlinQS with JPEG-2000 standard. Results have been presented at standard and non-standard rates for three (3) standard images: Lena (512 × 512), Barbara (512 × 512) and Elaine (512 × 512). This section is divided into three subsections: (i) Inclusion map and truncation points, (ii) Quantitative and qualitative comparison of BlinQS, and (iii) Computational complexity and trade-off.

4.1 Inclusion map and the truncation points

The inclusion map and the truncation points of the corresponding code-blocks for Lena image are presented in Fig. 9. In this figure, the white bar indicates the complete length of the string (L_i) obtained for code-blocks B_i and the black bar indicates the truncation point for B_i i.e., amount of string used for reconstruction for the target rate. Obtained Peak Signal to Noise Ratio (PSNR) values for the standard rates mentioned in R_{std} are shown in Fig. 9. These values clearly show the effect of inclusion map and the truncation points obtained using BlinQS. For required rates, 0.5 and 1.0, the inclusion map obtained is same as shown in Fig. 7, but the truncation points for these rates are different. Therefore, it is clear that the truncation points have played a major role in providing good quality at that rates. To optimally maintain the quality, BlinQS does not pick the blocks in the order, instead it picks up the blocks in the order of their contribution to the quality which can be derived using X_i. This can be clearly seen for the target rates 0.125 and 0.25 in Fig. 9, where some of the code-blocks are skipped by the algorithm to achieve optimum quality. For obtaining the optimum quality for the non-standard rates R_{new}, the local step size x_{new} plays a vital role in obtaining the inclusion map I_m, which is obtained from Algorithm-3.
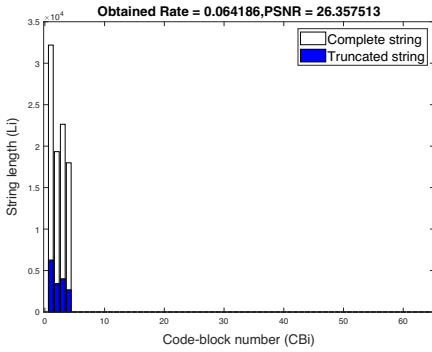
(a) Req rate= 0.0625bpp, PSNR= 26.35
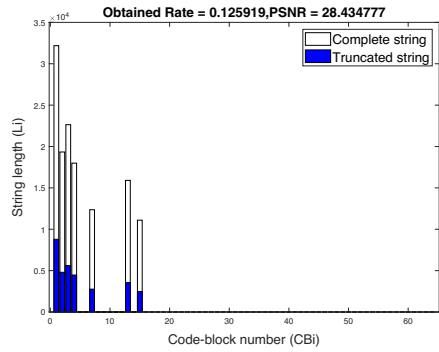
(b) Req rate= 0.125bpp, PSNR= 28.42
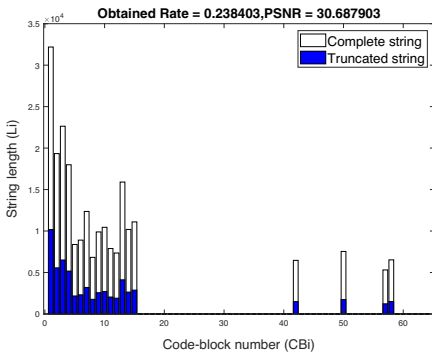
(c) Req rate= 0.25bpp, PSNR= 30.69
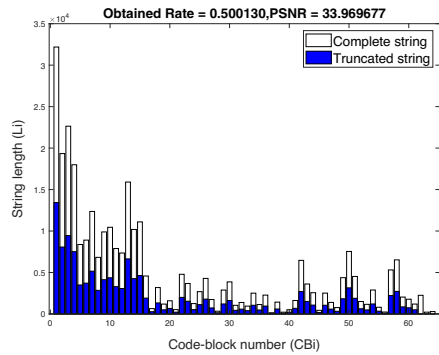
(d) Req rate= 0.5bpp, PSNR= 34.54
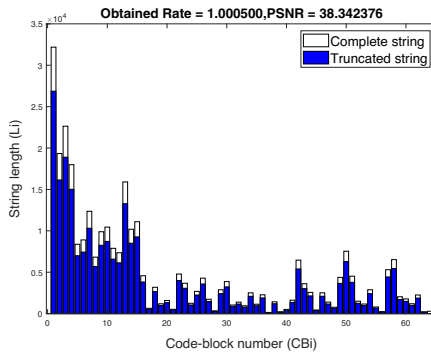
(e) Req rate= 1.0bpp, PSNR= 38.22

Fig. 9 Representation of inclusion map and truncation points for Lena image

4.2 Tabular and graphical results

To evaluate the performance of the BlinQS under same platform, results have been compared against JPEG-2000 at standard rates (bpp) with and without quality layer in Table 1. BlinQS has a clear domination over JPEG-2000 without quality layers and a near optimal

Table 1 PSNR (dB) comparison of BlinQS and JPEG-2000 at standard rates

Image	Rate (bpp)	J2K	J2K#	BlinQS
Lena	0.0625	26.76	22.31	26.35
	0.125	29.73	25.29	28.43
	0.25	32.82	27.3	30.69
	0.5	36.06	29.45	34.54
	1	38.78	33.13	38.22
Barbara	0.0625	22.51	20.08	22.64
	0.125	24.27	22.141	23.63
	0.25	27.05	22.894	25.18
	0.5	30.61	23.785	28.91
	1	35.56	24.929	34.17
Elaine	0.0625	28.17	23.15	27.61
	0.125	30.35	27.28	29.39
	0.25	31.79	28.615	30.51
	0.5	33.03	30.88	31.82
	1	35.12	32.66	34.37

J2K: JPEG-2000 from [39, 50], #-without quality layers

*Proposed BlinQS has comfortably ahead of J2K# at all the rates and performing equally well with J2K despite using the single layered string. This adds a new degree of freedom at the user end to chose any required rate independent from the encoder

value with quality layers. This shows that the estimation of BlinQS in optimizing the quality is very good. In JPEG-2000, the variation in PSNR is around 10dB for layered and non-layered string [6], but using BlinQS that has been reduced by a large extent and satisfactory results in terms of visual quality and PSNR are obtained.

To further investigate the proposed method, test images with various resolutions and textures have been selected for comparison and some of the sample test images are presented in Table 2. The detailed PSNR values of around 100 images are presented in Appendix A. PSNR and Structural Similarity (SSIM) index values obtained for these images, using BlinQS and JPEG-2000 are presented in Tables 3 and 4 respectively. From the PSNR values presented in Table 3, it can be observed that even at lower rates BlinQS is giving more

Table 2 Sample Database

S. No	Image Name	Resolution	Set
1	Baboon		
2	Plane		
3	Peppers	512×512	Set-I
4	Ship		
5	Boat		
6	Sand	3840×2160	Set-II (4K)

*Apart from the standard images, results for UHD and other standard images are taken for comparison. More images are taken for comparison in Appendix Table 5

Table 3 PSNR (dB) comparison of BlinQS and JPEG-2000 for sample database

Rate	0.0625 bpp			0.125 bpp			
S. No	J2K	J2K$^{\#}$	BlinQS	J2K	J2K$^{\#}$	BlinQS	
1	20.69	19.17	20.38	21.69	19.84	20.94	
2	26.17	20.48	24.84	29.31	22.62	26.41	
3	27.69	21.49	25.39	30.83	24.35	27.45	
4	25.21	21.01	24.19	27.41	22.79	25.51	
5	39.64	34.34	36.54	40.34	34.97	35.29	
6	39.64	28.7	36.54	40.34	29.10	35.29	
Average	29.84	24.20	27.98	31.65	25.61	28.48	
	0.25 bpp			0.5 bpp			1 bpp		
	J2K	J2K$^{\#}$	BlinQS	J2K	J2K$^{\#}$	BlinQS	J2K	J2K$^{\#}$	BlinQS
	23.15	19.99	21.16	25.48	20.90	22.87	28.97	21.79	26.11
	32.55	24.35	28.43	36.58	26.75	33.40	41.25	30.38	37.57
	33.44	24.96	29.80	35.73	28.70	32.76	38.20	31.66	36.23
	30.02	23.02	27.18	33.20	25.97	29.95	36.64	28.96	34.50
	41.13	36.55	37.08	42.53	37.36	39.77	44.88	39.66	41.05
	41.13	29.80	37.08	42.53	30.09	39.77	44.88	31.20	41.05
	33.57	26.45	30.12	36.01	28.30	33.09	39.14	30.61	36.08

*J2K: JPEG-2000, J2K$^{\#}$: JPEG-2000 without quality layers ([39])

quality than JPEG-2000 without quality layer (J2K$^{\#}$) and PSNR is ≈30dB (but <J2K value) which clearly tells the visual quality of the image is flawless. From the SSIM values presented in Table 4, it can be clearly seen that BlinQS is giving almost same results as that of J2K and giving very goos results when compared to J2K$^{\#}$. he comparison graph between "PSNR (dB) and Rate" of Lena (512×512) and Barbara (512×512) is depicted in Fig. 10. From this graph, it can be clearly seen that BlinQS is almost following JPEG-2000 interms of PSNR value. It is even closer at lower rates like 0.0625 bpp and 0.125 bpp when compared to other rates.

4.3 Visual quality representation

For qualitative analysis, reconstructed images of BlinQS at various standard rates have been presented in Figs. 11, 12 and 13. Where, sub-figure (a), represents the original image used for encoding and (b) to (g) represents the images reconstructed at standard rates as mentioned in the figure captions. The maximum possible rate that can be obtained through the compressed string is represented in subfigure (g). The visual quality is flawless when observed at rates R_{std} >0.5, and quite good even for lower rates. PSNR values for the respective rates are given in Table 1 and also mentioned in the figure along with the obtained rate. Hence, it can be clearly seen that in both qualitative and quantitative analysis BlinQS has provided nearly same results as that of JPEG-2000.

Table 4 SSIM comparison of BlinQS and JPEG-2000 for sample database

Rate	0.0625 bpp			0.125 bpp		
S. No	J2K	J2K#	BlinQS	J2K	J2K#	BlinQS
1	0.57	0.33	0.57	0.69	0.46	0.63
2	0.84	0.62	0.80	0.91	0.75	0.85
3	0.86	0.65	0.79	0.92	0.80	0.86
4	0.76	0.52	0.72	0.84	0.65	0.77
5	1.00	0.99	0.99	1.00	0.99	0.97
6	1.00	0.96	0.99	1.00	0.99	0.97
Average	0.84	0.68	0.81	0.89	0.77	0.84

0.25 bpp			0.5 bpp			1 bpp		
J2K	J2K#	BlinQS	J2K	J2K#	BlinQS	J2K	J2K#	BlinQS
0.79	0.50	0.62	0.88	0.69	0.74	0.95	0.81	0.88
0.95	0.85	0.88	0.97	0.93	0.95	0.99	0.98	0.99
0.95	0.83	0.90	0.97	0.94	0.93	0.98	0.97	0.98
0.92	0.67	0.82	0.96	0.87	0.90	0.98	0.94	0.97
1.00	0.99	0.98	1.00	0.99	1.00	1.00	0.99	1.00
.00	0.99	0.98	1.00	0.99	1.00	1.00	0.99	1.00
0.93	0.81	0.86	0.96	0.90	0.92	0.98	0.95	0.97

*J2K: JPEG-2000, J2K#: JPEG-2000 without quality layers ([39])

4.4 Computational complexity and trade-off

The basic idea of BlinQS is to reduce the computational load by removing the iterative R-D optimization algorithm and achieve blind scalbility. The computational complexity of R-D optimization algorithm is given by $\mathcal{O}(N_{crv} \times N_{pt})$, where N_{crv} represents the number of
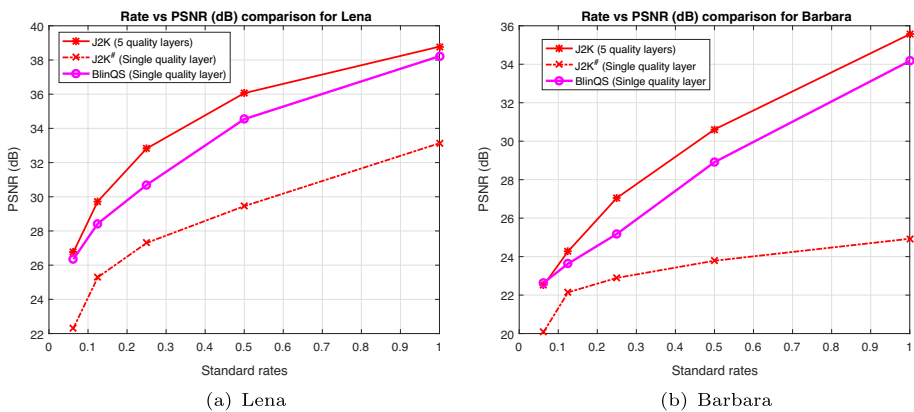

(a) Lena   (b) Barbara

Fig. 10 Rate vs PSNR (dB)

(a) Lena   (b) Rate= 0.0625 bpp   (c) Rate= 0.125 bpp   (d) Rate= 0.25 bpp

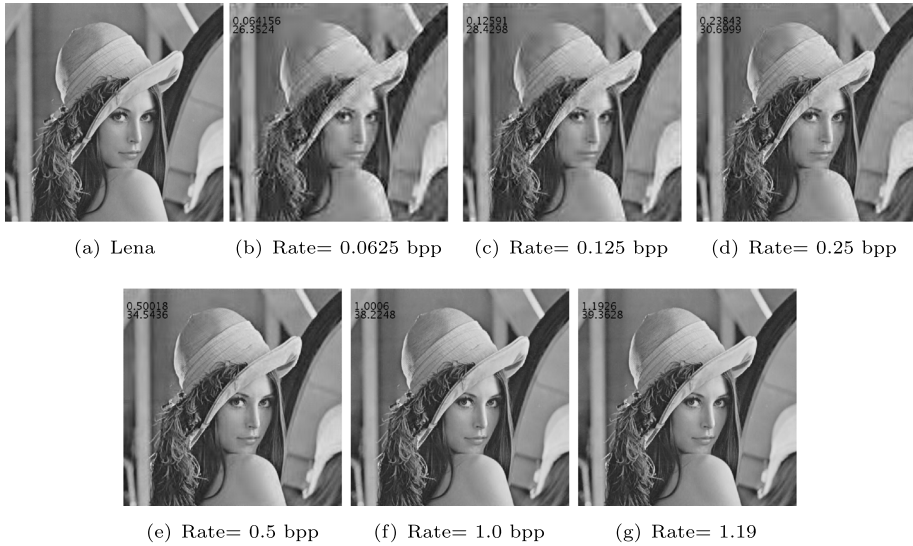(e) Rate= 0.5 bpp   (f) Rate= 1.0 bpp   (g) Rate= 1.19

Fig. 11 Lena reconstructed at standard rates: (a) Original Image, (b)-(g) Reconstructed at specified rates using the proposed BlinQS algorithm

code-blocks and N_{pt} represents number of average points in each code-block [2]. Hence, BlinQS has reduced the computational load by the order of $\mathcal{O}(N_{crv} \times N_{pt})$ at a sacrifice of $\approx 7\%$ of PSNR compared to JPEG-2000.

To achieve optimal quality, non-iterative and computationally less complex algorithm using gaussian normal distribution has been added to the decoder for standard rates. For non-standard rates, iterations are used to achieve optimal quality. It adds computational complexity of $\mathcal{O}(N_{depth})$, where N_{depth} represents precision of the target rate that depends

(a) Barbara   (b) Rate= 0.0625 bpp   (c) Rate= 0.125 bpp   (d) Rate= 0.25 bpp

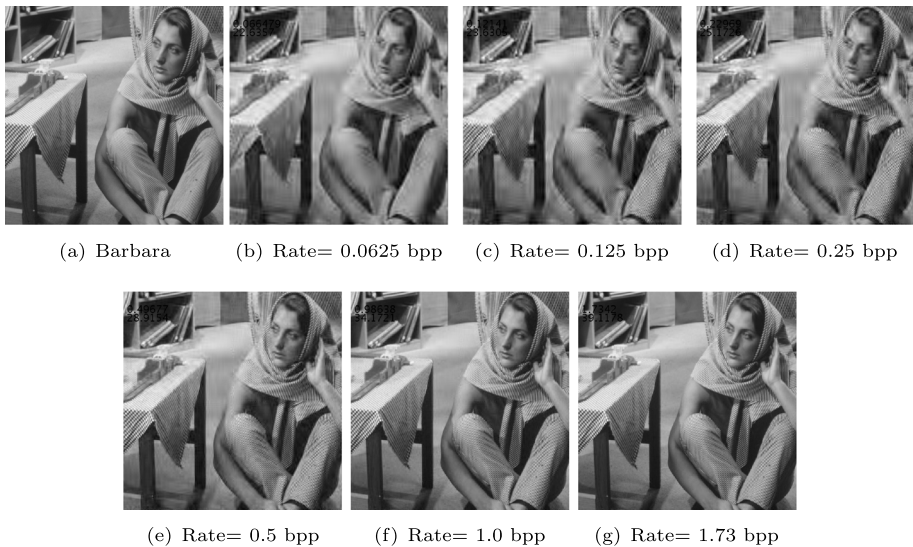(e) Rate= 0.5 bpp   (f) Rate= 1.0 bpp   (g) Rate= 1.73 bpp

Fig. 12 Barbara reconstructed at standard rates: (a) Original Image, (b)-(g) Reconstructed at specified rates using the proposed BlinQS algorithm

(a) boat   (b) Rate= 0.0625 bpp   (c) Rate= 0.125 bpp   (d) Rate= 0.25 bpp

(e) Rate= 0.5 bpp   (f) Rate= 1.0 bpp   (g) Rate= 1.38 bpp
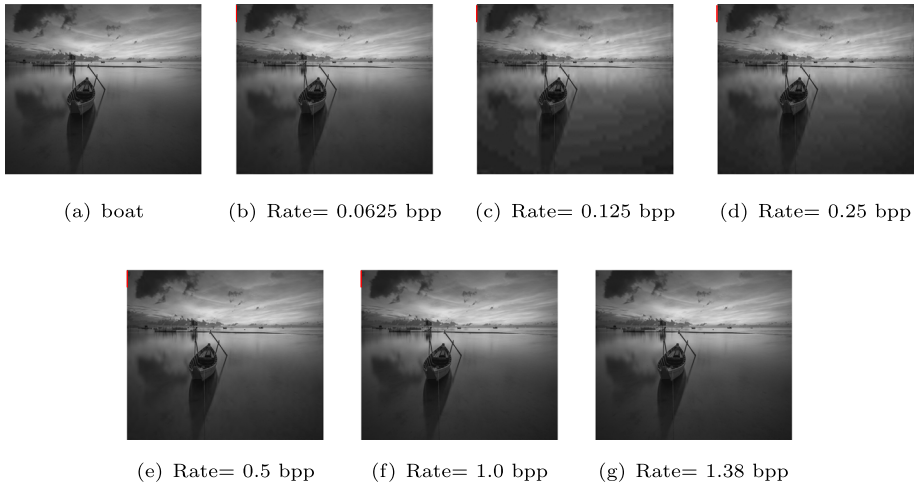
Fig. 13 Boat image reconstructed at standard rates: (a) Original Image, (b)-(g) Reconstructed at specified rates using the proposed BlinQS algorithm

on the threshold and target rate selected by the user. This has also added a *new degree of freedom* for choosing any required rate at the decoder rather than limiting to the rates calculated at the encoder. Hence, loss of $\approx 7\%$ of PSNR has resulted in the decoder independency for optimal reconstruction at target rate and reduced computational load.

5 Conclusion

This paper addresses the necessity of blind quality scalability for image compression and its implementation. The main concerns of quality scalability, iterative coding and lack of scalability for single layered string, are taken into consideration for developing BlinQS (Blind Quality Scalable) image compression. Normal distribution of percentage lengths has been used for getting the inclusion map for the target rate and this map is used for generating the truncation points (n_i) for the respective blocks. Algorithm to obtain the inclusion map for achieving optimum reconstruction quality without iterative process at decoder has been introduced. This has reduced the computational complexity by a factor of $\mathcal{O}(N_{crv} \times N_{pt})$. The PSNR values obtained by the proposed algorithm have been presented in the the the comparison table, which shows that BlinQS has obtained nearly same results using single string without using quality layers. Results shown for standard images clearly show the visual quality of the reconstructed image is flawless at higher rates and quite good even at lower rates. On an average, PSNR values obtained for BlinQS are 7% less than that of JPEG-2000 by reducing the computational load on encoder and making the single string scalable at any desired target rate.

Appendix

This appendix presents the comparison results of BlinQS algorithm against JPEG-2000 with and without quality layers in Table 5. Images presented here comprise of standard images

and other images downloaded from standard datasets like SCIEN test images and videos
[43], Laboratory for Image & Video Engineering, Texas [35, 36, 44], Robert Chung colour
management database and Roger K. Fawcett Distinguished Professor [16]. In Table 5, J2K#
stands and J2K stands for JPEG-2000 without and with quality layers.

Table 5 Comparison results of BlinQS algorithm against JPEG-2000 with and without quality layers

Image Name	Method	PSNR for different rates				
		0.0625	0.125	0.25	0.5	1
1.pgm	J2K	23.37	25.42	28.30	32.08	37.08
	J2K#	20.08	22.14	22.89	23.78	24.92
	BlinQS	22.64	23.63	25.17	28.54	34.36
109.pgm	J2K	20.69	21.68	23.12	25.47	28.96
	J2K#	19.17	19.84	19.99	20.90	21.79
	BlinQS	20.37	20.94	21.16	22.72	26.06
111.pgm	J2K	26.11	29.29	32.53	36.54	41.26
	J2K#	20.48	22.62	24.35	26.75	30.38
	BlinQS	24.82	26.40	28.42	33.01	37.74
113.pgm	J2K	27.69	30.81	33.43	35.72	38.19
	J2K#	21.49	24.35	24.96	28.70	31.66
	BlinQS	25.39	27.43	29.78	31.96	36.12
198.pgm	J2K	25.21	27.41	30.02	33.20	36.64
	J2K#	21.01	22.79	23.02	25.97	28.96
	BlinQS	24.19	25.51	27.18	29.51	34.50
N1A.pgm	J2K	23.48	25.02	27.27	30.66	35.72
	J2K#	21.47	21.78	22.10	22.51	24.33
	BlinQS	21.97	22.58	23.12	25.31	30.37
N5A.pgm	J2K	21.41	23.71	26.78	30.64	35.51
	J2K#	16.69	17.70	18.53	20.07	22.47
	BlinQS	18.66	19.23	20.30	22.25	28.15
building2.bmp	J2K	17.79	19.14	20.92	23.42	27.30
	J2K#	15.46	16.06	16.78	18.45	20.14
	BlinQS	15.26	16.74	16.95	18.27	21.37
buildings.bmp	J2K	19.44	21.09	23.43	26.75	31.49
	J2K#	15.99	17.42	18.20	19.74	21.71
	BlinQS	18.67	19.25	20.71	22.31	27.14
coins.bmp	J2K	23.17	24.93	27.25	29.98	34.18
	J2K#	19.42	21.35	21.83	24.45	26.29
	BlinQS	22.28	21.09	22.24	23.89	29.23
elaine.pgm	J2K	29.34	31.14	32.33	33.52	36.07
	J2K#	23.15	27.28	28.62	30.88	32.66
	BlinQS	27.06	27.85	28.70	30.69	34.38
fhd2.pgm	J2K	23.63	25.20	27.19	30.05	34.51
	J2K#	20.83	22.04	22.84	24.36	25.75
	BlinQS	22.90	22.39	22.72	24.71	30.19

Table 5 (continued)

Image Name	Method	PSNR for different rates 0.0625	0.125	0.25	0.5	1
fhd3.pgm	J2K	43.26	47.37	50.52	52.65	54.37
	J2K#	32.67	37.62	42.27	48.19	54.40
	BlinQS	35.82	38.61	42.28	46.63	46.63
fhd4.pgm	J2K	28.73	31.45	34.53	37.94	42.21
	J2K#	22.94	25.88	28.19	31.72	35.54
	BlinQS	24.39	23.16	25.62	29.16	38.78
flowers.bmp	J2K	18.39	19.65	21.47	24.05	28.50
	J2K#	16.18	16.98	17.50	18.88	20.27
	BlinQS	15.55	16.41	17.82	19.80	24.29
img2.pgm	J2K	34.19	36.93	39.14	41.08	43.49
	J2K#	25.42	28.15	29.62	32.25	37.43
	BlinQS	30.82	33.49	36.25	38.92	40.01
k01.bmp	J2K	29.40	32.06	35.40	38.88	42.37
	J2K#	23.08	25.42	26.82	30.87	35.65
	BlinQS	25.91	28.02	30.99	34.26	35.26
k06.bmp	J2K	32.18	35.28	38.73	41.86	45.04
	J2K#	25.47	27.85	29.65	33.55	39.66
	BlinQS	29.14	32.29	35.67	40.55	40.77
k08.bmp	J2K	27.45	29.86	32.22	34.27	36.90
	J2K#	21.23	23.60	25.66	28.30	31.52
	BlinQS	25.23	26.54	29.15	31.81	35.52
k09.bmp	J2K	34.78	35.40	36.01	37.11	38.96
	J2K#	29.61	31.45	33.14	34.08	36.68
	BlinQS	33.58	34.65	35.19	36.15	36.73
k10.bmp	J2K	35.04	35.74	36.43	37.58	39.59
	J2K#	29.83	32.13	33.65	35.60	37.19
	BlinQS	33.88	34.96	35.59	36.71	37.18
k12.bmp	J2K	39.51	41.53	43.11	44.53	46.74
	J2K#	31.52	34.15	37.69	39.35	42.51
	BlinQS	36.22	38.16	40.18	42.29	42.29
k13.bmp	J2K	26.14	28.34	31.17	34.65	38.91
	J2K#	21.96	23.72	24.90	28.45	30.20
	BlinQS	23.86	24.50	26.71	31.30	37.15
k15.bmp	J2K	37.01	39.54	42.05	44.52	47.39
	J2K#	29.84	32.32	35.36	37.95	41.83
	BlinQS	33.43	35.64	38.37	42.02	42.02
k16.bmp	J2K	34.02	36.71	39.45	42.07	44.61
	J2K#	27.23	29.51	30.72	33.83	39.44
	BlinQS	31.93	34.72	37.33	40.61	40.61
k19.bmp	J2K	33.63	36.06	38.44	40.62	43.69
	J2K#	26.61	28.69	31.05	34.18	39.98

Table 5 (continued)

Image Name	Method	PSNR for different rates				
		0.0625	0.125	0.25	0.5	1
	BlinQS	30.95	33.15	36.10	39.41	39.70
k20.bmp	J2K	36.20	38.41	40.16	41.85	44.29
	J2K[#]	29.19	31.42	34.46	36.59	41.06
	BlinQS	32.64	34.82	36.57	37.85	37.85
k21.bmp	J2K	32.08	34.47	36.65	38.68	41.52
	J2K[#]	25.36	28.16	29.08	32.78	37.17
	BlinQS	29.37	31.52	33.92	37.41	38.56
k22.bmp	J2K	33.90	36.28	38.71	41.07	44.42
	J2K[#]	28.72	30.67	32.84	35.97	40.38
	BlinQS	31.18	33.59	36.30	39.90	40.03
k23.bmp	J2K	40.14	41.31	42.53	44.36	46.99
	J2K[#]	33.16	36.10	39.89	41.72	44.46
	BlinQS	35.76	36.71	37.52	37.89	37.89
lena512.pgm	J2K	28.15	31.01	33.89	37.08	40.27
	J2K[#]	22.31	25.29	27.30	29.45	33.13
	BlinQS	22.26	25.01	28.91	33.97	38.34
lighthouse.bmp	J2K	24.72	26.38	28.59	31.98	37.12
	J2K[#]	20.96	22.40	23.06	24.48	26.60
	BlinQS	23.94	24.38	25.35	27.97	33.55
plane.bmp	J2K	28.41	30.70	33.33	37.12	43.08
	J2K[#]	23.07	24.54	26.69	28.09	31.36
	BlinQS	25.86	26.99	28.17	31.60	35.43
sailing1.bmp	J2K	24.41	25.68	27.80	30.88	35.51
	J2K[#]	21.20	22.40	22.84	24.31	26.50
	BlinQS	23.75	24.26	25.30	27.23	31.40
uhd.pgm	J2K	30.49	32.58	35.37	39.46	45.30
	J2K[#]	26.31	27.87	29.38	30.94	34.25
	BlinQS	25.59	26.24	28.74	33.97	41.53
uhd1.pgm	J2K	43.85	45.64	47.79	50.30	53.29
	J2K[#]	38.49	41.12	43.88	46.03	50.98
	BlinQS	35.50	38.62	42.81	45.65	45.87
uhd10.pgm	J2K	30.56	33.44	37.29	41.88	46.84
	J2K[#]	24.50	25.75	26.75	28.65	34.74
	BlinQS	25.11	27.08	29.72	35.52	42.14
uhd7.pgm	J2K	28.70	32.07	36.18	41.30	47.01
	J2K[#]	22.46	24.31	26.24	28.84	35.27
	BlinQS	22.39	24.75	28.31	34.41	42.47
uhd8.pgm	J2K	39.61	40.31	41.11	42.50	44.85
	J2K[#]	34.34	34.97	36.55	37.36	39.66
	BlinQS	36.50	35.22	37.05	39.46	41.34

Table 5   (continued)

Image Name	Method	PSNR for different rates 0.0625	0.125	0.25	0.5	1
uhd9.pgm	J2K	30.42	31.16	32.57	34.78	38.93
	J2K$^{\#}$	28.73	29.10	29.82	30.09	31.20
	BlinQS	29.32	25.25	25.20	30.19	35.54
us092.pgm	J2K	20.02	21.26	23.37	26.48	31.40
	J2K$^{\#}$	17.44	18.46	19.13	20.05	21.15
	BlinQS	17.49	18.80	17.95	20.42	25.76

J2K: JPEG-2000 with quality layers from [39, 50], J2K$^{\#}$- JPEG-2000 without quality layers from [39, 50]

Data Availability Data used in this work is downloaded from repositories which have been already cited in the manuscript. Data available on request from the authors.

Declarations

- This research is carried out as a part of Academic work. No External funding has been received to carry out this research.
- The research does not involve any Human Participation and/ or Animals.

Conflict of Interests The authors declare that they have no conflict of interest.

References

1. Aminlou A, Fatemi O (2005) A novel efficient rate control algorithm for hardware implementation in jpeg2000. In: Proceedings. (ICASSP'05), IEEE International conference on acoustics, speech, and signal processing, 2005., vol 5, pp v/21–v/24 Vol 5
2. Aminlou A, Fatemi O, Homayouni M, Hashemi MR (2006) A non-iterative rd optimization algorithm for rate-constraint problems. In: Image processing, 2006 IEEE international conference on, pp 2493–2496, IEEE
3. Auli-Llinas F, Marcellin MW (2009) Distortion estimators for bitplane image coding. IEEE Trans Image Process 18(8):1772–1781
4. Auli-Llinas F, Marcellin MW (2011) Scanning order strategies for bitplane image coding. IEEE Trans Image Process 21(4):1920–1933
5. Auli-Llinas F, Serra-Sagrista J (2007) Low complexity JPEG2000 rate control through reverse subband scanning order and coding passes concatenation. IEEE Signal Process Lett 14(4):251–254
6. Auli-Llinas F, Serra-Sagrista J (2008) Jpeg2000 quality scalability without quality layers. IEEE Trans Circuits Syst Video Technol 18(7):923–936
7. Auli-Llinas F, Serra-Sagrista J, Bartrina-Rapesta J (2010) Enhanced JPEG2000 quality scalability through block-wise layer truncation. EURASIP J Adv Signal Process 2010:56
8. Auli-Llinas F, Serra-Sagrista J, Monteagudo-Pereira JL, Bartrina-R J (2006) Efficient rate control for JPEG2000 coder and decoder. In: Data Compression Conference (DCC'06), pp 282–291 IEEE
9. Aulí-llinàs F, Serra Sagristà J (2007) Model-based JPEG2000 rate control methods Universitat Autònoma de Barcelona
10. Bi C, Liang J (2017) Joint source-channel coding of jpeg 2000 image transmission over two-way multi-relay networks. IEEE Trans Image Process 26(7):3594–3608
11. Boliek M (2002) Jpeg 2000 image coding system: Core coding system, ISO/IEC

12. Bruni V, Cotronei M, Pitolli F (2020) A family of level-dependent biorthogonal wavelet filters for image compression. J Comput Appl Math 367:112467

13. Cai J, Cao Z, Zhang L (2020) Learning a single tucker decomposition network for lossy image compression with multiple bits-per-pixel rates. IEEE Trans Image Process 29:3612–3625

14. Chebil F, Kurceren R (2006) Pre-compression rate allocation for jpeg2000 encoders in power constrained devices. In: Visual Communications and image processing 2006, vol 6077, p 607720 International society for optics and photonics

15. Choi H, Bajić IV (2022) Scalable image coding for humans and machines. IEEE Trans Image Process 31:2739–2754

16. Chung R (2019) Pictorial color reference images, http://rycppr.cias.rit.edu/cms.html

17. Du W, Sun J, Ni Q (2004) Fast and efficient rate control approach for jpeg2000. IEEE Trans Consum Electron 50:1218–1221

18. Edwards T, Smith MD (2021) High throughput jpeg 2000 for broadcast and ip-based applications. SMPTE Motion Imaging J 130(4):22–35

19. Etesaminia A, Mazinan A (2017) An efficient rate control algorithm for jpeg2000 based on reverse order. J Cent South Univ 24(6):1396–1405

20. Gaubatz MD, Hemami SS (2007) Robust rate-control for wavelet-based image coding via conditional probability models. IEEE Trans Image Process 16:649–663

21. Hu Y, Yang S, Yang W, Duan L-Y, Liu J (2020) Towards coding for human and machine vision: A scalable image coding approach. In: 2020 IEEE International Conference on Multimedia and Expo (ICME), pp 1–6 IEEE

22. Kanchi (2011) Spiht (set partitioning in hierarchical trees), https://in.mathworks.com/matlabcentral/fileexchange/4808-spiht?s_tid=prof_contriblnk. Accessed: 06 March 2020

23. Kim T, Kim HM, Tsai P-S, Acharya T (2005) Memory efficient progressive rate-distortion algorithm for jpeg 2000. IEEE Trans Circuits Syst Video Technol 15:181–187

24. Kim SH, Park JH, Ko JH (2021) Target-dependent scalable image compression using a reconfigurable recurrent neural network. IEEE Access 9:119418–119429

25. Mei Y, Li L, Li Z, Li F (2021) Learning-based scalable image compression with latent-feature reuse and prediction. IEEE Transactions on Multimedia

26. Naman AT, Taubman D (2019) Decoding high-throughput jpeg2000 (htj2k) on ag. In: 2019 IEEE International Conference on Image Processing (ICIP), pp 1084–1088, IEEE

27. Naman AT, Taubman D (2020) Encoding high-throughput jpeg2000 (htj2k) images on a gpu. In: 2020 IEEE International Conference on Image Processing (ICIP), pp 1171–1175, IEEE

28. Naman AT, Wang Y, Gharakheili HH, Sivaraman V, Taubman D (2018) Responsive high throughput congestion control for interactive applications over sdn-enabled networks. Comput Netw 134:152–166

29. Ortega A, Ramchandran K (1998) Rate-distortion methods for image and video compression. IEEE Signal Process Mag 15(6):23–50

30. Parisot C, Antonini M, Barlaud M (2002) High performance coding using a model-based bit allocation with ebcot. In: Signal Processing Conference, 2002 11th European, pp 1–4 IEEE

31. Rajput A, Li J, Akhtar F, Hussain Khand Z, Hung JC, Pei Y, Börner A (2022) A content awareness module for predictive lossless image compression to achieve high throughput data sharing over the network storage. Int J Distrib Sens Netw 18(3):15501329221083168

32. Said A, Pearlman WA (1996) A new, fast, and efficient image codec based on set partitioning in hierarchical trees. IEEE Trans Circuits Syst Video Technol 6:243–250

33. Shapiro JM (1993) Embedded image coding using zerotrees of wavelet coefficients. IEEE Trans Signal Process 41(12):3445–3462

34. Shapiro JM (1993) Smart compression using the embedded zerotree wavelet (ezw) algorithm. In: Signals, systems and computers 1993. 1993 Conference record of the 27th Asilomar Conference on, pp 486–490, IEEE

35. Sheikh H (2005) Live image quality assessment database release 2, http://live.ece.utexas.edu/research/quality

36. Sheikh HR, Sabir MF, Bovik AC (2006) A statistical evaluation of recent full reference image quality assessment algorithms. IEEE Trans image process 15(11):3440–3451

37. Su R, Cheng Z, Sun H, Katto J (2020) Scalable learned image compression with a recurrent neural networks-based hyperprior. In: 2020 IEEE International Conference on Image Processing (ICIP), pp 3369–3373, IEEE

38. Taubman D (2000) High performance scalable image compression with ebcot. IEEE Trans Image Process 9(7):1158–1170

39. Taubman D (2000) Kakadu software, http://www.kakadusoftware.com. Accessed 14 Jun 2000

40. Taubman DS, Marcellin MW (2002) Jpeg2000: Standard for interactive imaging. Proc IEEE 90(8):1336–1357

41. Taubman DS, Naman AT, Mathew R, Smith MD (2018) High throughput jpeg 2000 (htj2k): New algorithms and opportunities. SMPTE Motion Imaging Journal 127(9):1–7

42. Tu H, Li L, Zhou W, Li H (2021) Semantic scalable image compression with cross-layer priors. In: Proceedings of the 29th ACM International conference on multimedia, pp 4044–4052

43. University S (2015) Scien test images and videos, https://scien.stanford.edu/index.php/test-images-and-videos/

44. Wang Z, Bovik AC, Sheikh HR, Simoncelli EP (2004) Image quality assessment: from error visibility to structural similarity. IEEE Trans Image Process 13(4):600–612

45. Watanabe O, Taubman D (2019) A matlab implementation of the emerging htj2k standard. In: 2019 IEEE 8th Global Conference on Consumer Electronics (GCCE), pp 491–495, IEEE

46. Yan N, Gao C, Liu D, Li H, Li L, Wu F (2021) Sssic: Semantics-to-signal scalable image coding with learned structural representations. IEEE Trans Image Process 30:8939–8954

47. Yeung YM, Au OC (2005) Efficient rate control for jpeg2000 image coding. IEEE Trans Circuits Syst Video Technol 15(3):335–344

48. Yu W, Sun F, Fritts JE (2006) Efficient rate control for jpeg-2000. IEEE Trans Circuits Syst Video Technol 16:577–589

49. Zhang YZ, Xu C (2006) Analysis and effective parallel technique for rate-distortion optimization in jpeg2000. In: International conference on image processing, pp 2465–2468

50. openjpeg (2019) Official repository of the openjpeg project: openjpeg-v2.3.1, https://github.com/uclouvain/openjpeg. Accessed 15 Jun 2019