Visual Progressive Coding

Jin Li*

Sharp Laboratories of America
5750 NW Pacific Rim Blvd., Camas, WA 98607, USA.

ABSTRACT

The embedded coder has an attractive feature that the coding bitstream can be truncated later at any point and still decoded a perceptible image. It is common in conventional coding to improve the subjective quality of the coded image through adjusting the quantization step size inversely proportional to a set of visual weights. However, such scheme will not be effective in the embedded coder as different viewing condition may be called for at different stages of embedding. In this paper, we propose the visual progressive coding (VIP), which uses the visual weights to determine the order of embedding, rather than to requantize the transform coefficients. VIP can change the weights halfway through redefining the order of embedding according to the active weights. With such "reordering by weight" strategy, it is feasible to adjust the visual weights flexibly during the embedding process, and improve the subjective appearance of the image over the entire bit rate.

KEYWORDS: Image coding, embedded coding, scalability, visual frequency weighting, visual progression

1. INTRODUCTION

Visual weighting has proven itself as an effective tool to improve the subjective quality of an encoded image. By allocating more bits to the coefficients in the visual sensitive frequency band and less bits to those in the visual insensitive band, visual weighting emphasizes the features more perceivable by the human eyes, and improves the subjective quality of the image. Traditionally, the visual weights are the same throughout the coding process, and the weighting is implemented in one of two ways: (1) by multiplying the transform coefficients with the weights:

$$\hat{f}_{i,j} = f_{i,j} \times w_{i,j}, \tag{1}$$

and then quantize and entropy encode the weighted coefficient $\hat{f}_{i,j}$, or (2) by adjusting the quantization step size inversely proportional to the weight:

$$x_{i,j} = Q\left[\frac{f_{i,j}}{q_i}\right], \quad \text{with } q_i \propto \frac{1}{w_i}. \tag{2}$$

We denote such scheme as the fixed visual weighting. In (1) and (2), $f_{i,j}$ and $\hat{f}_{i,j}$ are the transform coefficient without and with weighting, $x_{i,j}$ is the quantized coefficient, i indexes the frequency band, and j is a position within the band i. q_i and w_i are the quantization step size and the visual weight associated with the band i. The weight w_i may be derived from a contrast sensitivity function (CSF) model of the visual system and the distance the image is to be viewed. For more details of the derivation of the weights, we refer to [2]. Implementation (2) is more simply and is therefore preferred for schemes explicitly involve a quantization operation, such as JPEG. In many embedding schemes, there is no quantization operation, in such a case implementation (1) can be used.

One of the recent achievements in image coding is the embedded coding. Pioneered by Shapiro in the embedded zero tree wavelet coding (EZW)[3], embedded coder had an attractive feature that the coding bitstream could be truncated later at any point and still decoded a perceptible image. Important applications of embedding can be found in internet image browsing, image database, digital camera, etc. EZW encoded the image from the most significant bitplane to the least significant bitplane, and within each bitplane, it used a zerotree structure to group

* Correspondence: Email: lijin@sharplabs.com, Tel. (360) 817-8486, Fax. (360) 817-8436.

the insignificant bits of coefficients and encoded it with one symbol. Said and Pearlman improved EZW and developed the scheme of set partitioning in hierarchical trees (SPIHT)[4]. SPIHT redefined the coding symbols into 3 sets -- the list of insignificant pixels (LIP), the list of significant pixels (LSP), and the list of insignificant sets (LIS) which consisted of a type A or B tree group of insignificant coefficients. SPIHT performed superior to EZW. Moreover, with a little sacrifice of performance, one mode of SPIHT greatly reduced the computational complexity by eliminating the arithmetic entropy coder. Zandi *et al* proposed the compression with reversible embedded wavelet (CREW)[6] which might encode the image all the way to lossless. Taubman and Zakhor proposed the layered zero coding (LZC) in [5]. In stead of grouping the insignificant coefficients into zerotrees, LZC used a context adaptive arithmetic coder to efficiently encode the location of the insignificant coefficients. LZC not only outperformed SPIHT, but also generated a flexible coding bitstream which might be organized into progression by quality or progression by resolution. Wang and Kuo proposed a multi-threshold wavelet coder (MTWC)[7] which further improved the performance of LZC by first encoded the subband bitplane with the largest quantization step size. Li and Lei indicated that a rate-distortion optimal embedding could be achieved by first coding the bits with the largest rate-distortion slope, i.e., the largest distortion decrease per bit spent. A rate-distortion optimized embedding coder (RDE) was demonstrated in [1] which provided a smooth rate-distortion curve and a superior performance compared with both SPIHT and LZC.

If the embedded coded image is to be viewed at a specific distance, visual weighting can be easily incorporated into the coder by using (1), i.e., multiplying the transform coefficients with the weight w_i. However, in the embedded coding, different viewing conditions may be called for at different stages of embedding. Taking image database query as an example, with the embedding functionality, only one version of the compressed bitstream is stored in the central database. The user first requests only a small portion of the bitstream of each image to quickly browse through a large number of images at low resolution and fidelity, say $1/16^{th}$ screen per image. When an image of interest is found, he then previews the image at full screen resolution. He finds the image satisfactory, and requests the full losslessly compressed image to be analyzed and printed. During the query process, the viewing condition of the image changes. The image is enlarged or viewed closer and closer as more and more bits are received. At low bitrate, the image is usually viewed at a relatively far distance. The user is more interested in the global feature since the quality of the compressed image is poor and the detail image feature is not available anyway. The image quality improves as more bits are received, and the user becomes interested in not only the global features but also the details of the image. The image is examined at a closer distance, it may undergo image analysis operation, or even be blown up for examination, which equivalently decreases the viewing distance. Implementing changing weights with either (1) or (2) will be clumsy because the coefficients must be multiplied with a new weight, or be requantized every time the weights change. Furthermore, such implementation changes the binary representation of the coefficient sent to the entropy coder each time the weights change, the performance of the subsequent entropy coder may degrade due to the changing statistics.

In this paper, we propose a visual weighting strategy denoted as the visual progressive coding (VIP). VIP does not multiply the transform coefficients by the weights as in (1) or adjust the quantization step sizes inversely proportional to the weights as in (2), in stead, it uses the weights to change the order of embedding. A number of weights may take place during the VIP coding process, and each time a new set of weights is active, VIP just reorders the rest of the bitstream according to the new weights. Note that the new weights will not affect the order of the bitstream that has already been coded. VIP may be implemented upon any existing embedded coders and provide flexible visual adjustment for the entire embedded coding. We will discuss the implementation details of VIP in Section 2. The syntax facilitates VIP will be discussed in Section 3. Results and conclusions will be given in Section 4.

2. IMPLEMENTATION OF VISUAL PROGRESSIVE CODING

We first define some notations that will be used throughout the paper. In the visual progressive coding (VIP), the image is first transformed into a set of coefficients, where the transform may be DCT (discrete cosine transform), wavelet or even wavelet packet. Without loss of generality, we define a band in VIP as a group of trans-

form coefficients with the same visual characteristics. In the wavelet/wavelet packet transform, a band is just a wavelet/wavelet packet subband; while in DCT, a band consists of all coefficients of the same DCT basis. We index the transform coefficient as $f_{i,j}$, where i indexes the band, and j is a position within the band i. Let the binary representation of the transform coefficient $f_{i,j}$ be:

$$-b_1 b_2 b_3 \cdots b_u \cdots b_L \tag{3}$$

where b_1 is the most significant bit, and b_L is the least significant bit. Let $b_u(f_{i,j})$ be the u-th most significant bit or the u-th coding layer of coefficient $f_{i,j}$. A sample bit array produced by a transform can be shown in Figure 1, in which each row of the bit array represents a transform coefficient, and each column of the bit array represents a coding layer. We place the most significant bit at the left most column, and place the least significant bit at the right most column. It is apparent that a more significant bit $b_u(f_{i,j})$ should always be coded before a less significant bit $b_v(f_{i,j})$, for $u<v$. We denote a bit $b_u(f_{i,j})$ as a candidate bit if it is the most significant unencoded bit, i.e., if all of the more significant bits of the same coefficient $b_v(f_{i,j})$, $v=1,\cdots, u-1$ have already been encoded. At any given moment, the coder has to select from the set of candidate bits the next bit to encode. Let a coefficient be significant if any of its coded bits is non zero, otherwise let it be insignificant. The candidate bit of an insignificant coefficient is coded in the mode of significance identification, the candidate bit of a significant coefficient is coded in the mode of refinement.

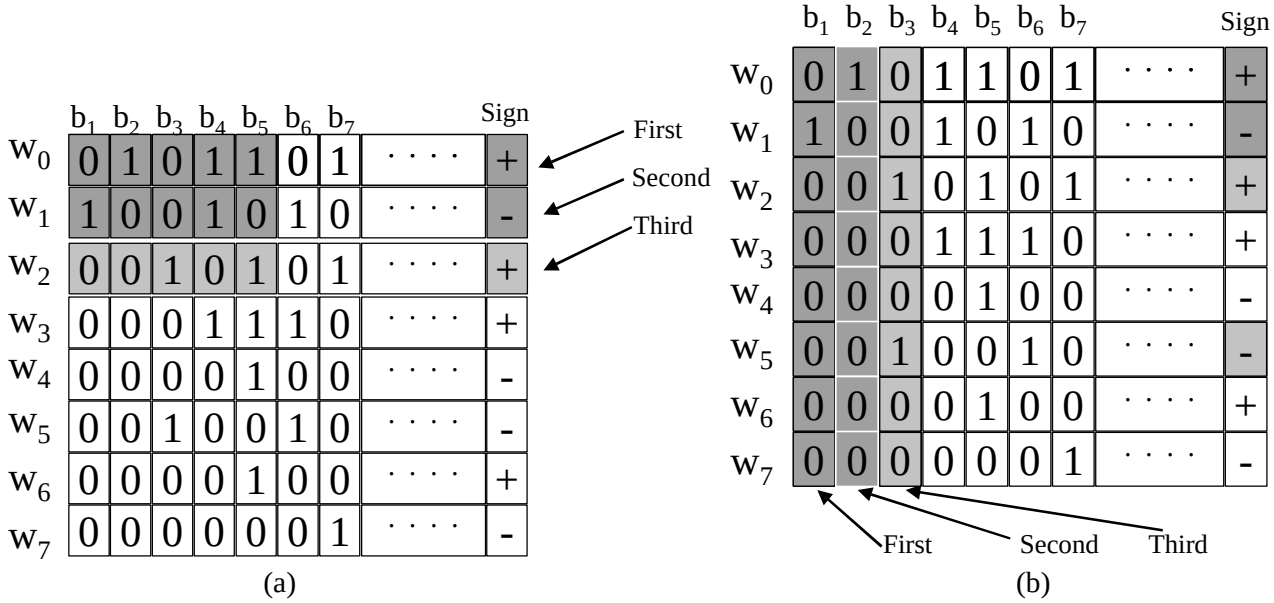

Figure 1 A bit array and the coding order of (a) the conventional coder, (b) the embedded coder.

The conventional and the embedded coder differ in the order of encoding the bit array. The conventional coder such as JPEG [8] or MPEG [9] first determines the quantization precision, or equivalently, the number of bits to encode for each coefficient, then sequentially encodes one coefficient after another. Using the bit array as an example, the conventional coding is ordered row by row as shown in Figure 1a. The embedded coding is distinctive as the image is coded bit-plane by bit-plane, or column by column as shown in Figure 1b. The embedding bitstream can be truncated and still maintain reasonable image quality since the most significant part of each coefficient is coded first. It is also suited for progressive image transmission because the quality of the decoded image gradually improves as more and more bits are received.

In VIP, there are many sets of visual weights:

$$\begin{aligned} \boldsymbol{w}^{(0)} &= \{w_0^{(0)}, w_1^{(0)}, \cdots, w_n^{(0)}\}, \\ \boldsymbol{w}^{(1)} &= \{w_0^{(1)}, w_1^{(1)}, \cdots, w_n^{(1)}\}, \\ &\cdots, \\ \boldsymbol{w}^{(m)} &= \{w_0^{(m)}, w_1^{(m)}, \cdots, w_n^{(m)}\}, \end{aligned} \tag{4}$$

There may be an optional global weighting set ***wg*** which is applied right after the transform operation and is in additional to the VIP weighting series:

$$\mathbf{wg}=\{wg_0, wg_1, \cdots, wg_n\}. \quad (5)$$

The global weighting set is implemented with the fixed visual weighting. At any given moment, one set of weights, which we denote as active weights ***w***, will be in effective:

$$\boldsymbol{w} =\{w_0, w_1, \cdots, w_n\}, \quad (6)$$

where w_i is the active weight for band i. The key concept of the VIP is in stead of weighting the transform coefficient as in implementation (1), or adjusting the quantization step inverse proportional to the weights as in implementation (2), VIP uses the weights to control the order of embedding. Let the smallest unit of reordering in VIP be a coding unit (CU), which is indexed by k. Depending on the specific embedding scheme based upon which the VIP is implemented, the CUs are different. Let a candidate CU be a coding unit which consists of only candidate bits. Since only candidate CUs can be coded, the operation of VIP is to order the candidate CUs according to the active weights. When a new weighting set is active, VIP just forms a new coding order for the remaining CUs. The coding order of CUs that have already been coded will not be affected by the new weights. It is the "reordering by weights" strategy that enables VIP to incorporate a number of weighting sets during the embedding process.

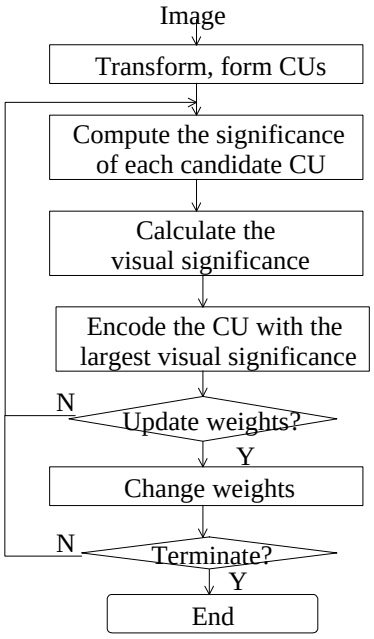

Figure 2 Operation flowchart of the visual progressive coding.

A general operation flowchart of VIP can be shown in Figure 2. After the transform, if there is a global weighting set ***wg***, it is applied with fixed visual weighting using either implementation (1) or (2). We initialize the active weights ***w***, and group the bits of transform coefficients to form CUs. VIP identifies the candidate CUs and calculates their significance s_k, which is a magnitude value related to the order of embedding without visual weighting. After that, the visual significance vs_k of the CU is caculated by multiplying the significance of the CU with its weight:

$$vs_k = s_k \times w_{i,} \quad (7)$$

where w_i is the active weight of the band where the CU resides. VIP encodes the CU with the largest visual significance. After a CU has been encoded, new candidate CUs may emerge. VIP then evaluates the significance and the visual significance of the newly emerged candidate CUs and selects to encode the one with the largest visual significance. Active weights may change any time, and when a set of new weights becomes active, it just affects the embedding order of the remaining CUs. The changing of weights has to be negotiated between the

encoder and the decoder, and there are several viable approaches. For a detail discussion, please refer to the syntax of VIP in Section 3. The coding process repeats until some termination criterion has been satisfied, such as all CUs have been encoded (the coding reaches losslessly), the final coding rate has been reached, or the coding distortion is smaller than a certain threshold.

We will elaborate with some sample VIP implementation on existing embedded coders.

2.1 Visual Progression on Individual Bits -- Visual Progressive Rate-Distortion Optimized Embedding (VIP RDE)

Pushing the performance of embedding to the limit, the rate-distortion optimized embedding (RDE) was developed in [1]. In RDE, the coding unit (CU) is a just single bit $b_u(f_{i,j})$ of a transform coefficient $f_{i,j}$. RDE encodes the candidate bits in the order of their expected rate-distortion (R-D) slope, i.e., the distortion reduction per coding bit:

$$slope_{i,j} = \frac{E[\Delta D_{i,j}]}{E[\Delta R_{i,j}]}. \tag{8}$$

In order to reduce the computational complexity, a lookup table is developed so that the calculation of the R-D slope for each candidate bit is just a single lookup table operation with its coding layer, significant status, and arithmetic coding context as index. For details of RDE, please refer to [1].

To implement VIP for RDE, we encode the CU, i.e., the individual bits of coefficients in a descend order of visual significance. We define the significance of the CU as the square root of its R-D slope:

$$s_{i,j} = \sqrt{slope_{i,j}} \tag{9}$$

We apply the square root because the R-D slope is an energy reduction measure, while the significance of a CU is a magnitude measure. Because the number of CUs is huge, we do not strictly search and encode the CU with the maximum visual significance, in stead, we adopt a threshold approach. A set of decreasing thresholds is defined as: $\gamma_0 > \gamma_1 > \cdots > \gamma_n > \cdots$. A typical threshold sequence reduces itself by a factor of α for each iteration:

$$\gamma_n = \gamma_0 \cdot \alpha^{-n}. \tag{10}$$

VIP RDE scans the transformed coefficients multiple times, and at scan n, all CUs with visual significance greater than γ_n is encoded. Since the active weight is the same within band i, in stead of calculating the visual significance of each coefficient and comparing it to the current threshold, we inversely weight the threshold for band i:

$$\gamma_i' = \gamma^2 / w_i^2 \tag{11}$$

and encode all candidate bits with visual significance greater than the adjusted threshold γ_i'. The procedure of VIP RDE can be described below as:

Step 1. Image transform.

Step 2. Fixed visual weighting: applying the global weights $\boldsymbol{wg}$ when available.

Step 3. Setting the initial threshold $\gamma = \gamma_0$ and active weights $\boldsymbol{w}$.

Step 4. Scanning and Coding.

The image is scanned from the lowest resolution band to the highest resolution band, and with raster line order within each band. For band i, the weighted threshold γ_i' is calculated according to (11). For each candidate bit, we calculate its R-D slope with a lookup table operation with its coding layer, significant status, and arithmetic coding context as index, as described in [1]. The R-D slope of the candidate bit is compared with the adjusted threshold γ_i', and only the bit with R-D slope greater than the adjusted threshold is encoded.

Step 5. Updating the active weights as necessary.

Step 6. Reducing threshold: after scanning the entire image, the threshold γ is reduced by a factor of α: $\gamma \leftarrow \gamma / \alpha$ and the coding goes back to Step 4. The coding continues until certain termination condition has been satisfied.

12.2 Visual Progressive Coding on a Subband or DCT index level.

For bitplane embedding schemes, such as the layered zero coding (LZC) proposed by Taubman and Zakhor [5], the compression with reversible embedded wavelets (CREW) proposed by Zandi *et al*[6], and the multi-threshold wavelet coder (MTWC) proposed by Wang and Kuo[7], a VIP coding unit(CU) can be just a band bitplane, which consists of all bits in the same coding layer and of the same band[1]. By enlarging the CU, we increase the granularity of reordering, however, the implementation is more simple as most part of the coder remains the same. A VIP implementation of this category is demonstrated with JPEG 2000 VM 2 as follows.

In JPEG 2000 VM2, the quantized coefficients are coded by a bitplane arithmetic coder. Within a band bitplane, the bits are further grouped into three sub- modes: (1) the predicted significance mode, where the current coefficient is insignificant but at least one of its neighbor coefficients is significant; (2) the refinement mode, where the current coefficient is significant; (3) predicted insignificance mode, where the current coefficients and all its neighbor coefficients are insignificant. Within a band, the coder always proceeds from the most significant bitplane to the least significant bitplane, and within a band bitplane, the coder always encodes first the predicted significance bits, then the refinement bits, and finally the predicted insignificance bits. To implement VIP, we define the CU to be one sub-mode of a band bitplane, and reorder the CU according to the active weights. The VIP enabled JPEG 2000 VM2 is implemented as follows:

Step 1. Image transform.

Step 2. Quantization (by scalar quantizer or TCQ) and fixed visual weighting with the global weights ***wg*** when available.

Step 3. Setting the initial active weights ***w***.

Step 4. Calculating the significance s_k for each candidate CU as

$$s_k=\begin{cases}\sqrt{3}\times 2^{-n_k} & \text{for predicted significance mode}\\ 1\times 2^{-n_k} & \text{for refinement mode}\\ \sqrt{0.96}\times 2^{-n_k} & \text{for predicted insignificance mode}\end{cases}, \qquad (12)$$

where n_k is the current coding layer. The constants $\sqrt{3},\ 1,\ and\ \sqrt{0.96}$ are designed through a coarse calculation of the R-D slope of different coding modes and to preserve the order of embedding when visual progression is not activated.

Step 5. Calculating the visual significance for each candidate CU according to (7).

Step 6. Encoding the candidate CU with the maximum visual significance.

Since there are relatively few CUs, instead of coding the changing weights, JPEG 2000 VM2 explicitly encodes the CU order. Before coding a CU, a tag will be encoded which identifies the CU. Since there is a unique coding order within the band, the tag only needs to specify the band where the CU occupies.

Step 7. Updating the active weights as necessary. The coding continues until certain termination condition has been satisfied.

12.3 Visual Progressive Coding for Embedding Schemes with Coding Unit across Multiple Bands.

In this section, we demonstrate the implementation of VIP on SPIHT [4], which consists of coding symbols with coefficients across multiple bands. Nevertheless, the implementation can be generalized to other similar embedding schemes, such as EZW[3]. There are three kinds of coding symbols in SPIHT -- the list of insignificant pixels (LIP), the list of significant pixels (LSP), and the list of insignificant sets (LIS). The member of LIP

[1] Such subband bitplane structure already exists in MTWC.

and LSP is just a single bit of a single coefficient. The member of LIS consists of a tree group of insignificant bits at the same coding layer across multiple bands. We define the CU, which is the smallest unit of VIP reordering, to be one member of LIP, LSP or LIS. Since the number of CUs is large, a threshold approach similar to VIP RDE is adopted. The coding procedure of VIP enabled SPIHT is described below:

Step 1. Image transform.

Step 2. Fixed visual weighting with the global weights ***wg*** when available.

Step 3. Setting the initial threshold $g = g_0$ and active weights ***w***.

Step 4. Traversing and Coding.

VIP traverses the LIS, LIP and LSP, evaluates the significance and the visual significance of each CU, and encodes the ones with visual significance greater than g. The significance of the CU is calculated by its quantization step size and coding mode:

$$s_k = \begin{cases} 1.9\times 2^{-n_k} & \text{a member of LIS} \\ \sqrt{3}\times 2^{-n_k} & \text{a member of LIP} \\ 1\times 2^{-n_k} & \text{a member of LSP} \end{cases}, \qquad (13)$$

where n_k is still the coding layer of the CU. The constants *1.9,* $\sqrt{3}$*, and 1* are again designed through a coarse calculation of the R-D slope of different coding modes and to preserve the order of embedding when visual progression is not activated. The visual significance of the CU is calculated by multiplying the significance of the CU with its weight. For the CU of a single bit (LIP or LSP), its weight is simply the active weight w_i of band i where the pixel resides. For the CU of a member of LIS, which consists of a tree of insignificant bits across multiple bands, its weight can be calculated either according to the most visual sensitive band:

$$w_{cur} = max\ \{ w_{i_0}, w_{i_1}, \cdots, w_{i_L} \} \qquad (14)$$

or as a weighted sum:

$$w_{cur} = \frac{p_0 \cdot w_{i_0}^{(k)} + p_1 \cdot w_{i_1}^{(k)} + \cdots + p_L \cdot w_{i_L}^{(k)}}{p_0 + p_1 + \cdots + p_L} \qquad (15)$$

where p_c denotes the number of pixels resides in band i_c, with $c=0,\cdots,L$. Method (14) is preferred since it guarantees the visual quality of the CU.

The calculated visual significance is compared with the current threshold g. Only those CUs with the visual significance greater than the threshold are encoded. The coding of CU follows precisely the rule described in [4].

Step 5. Updating weights as necessary.

Step 6. Reducing threshold: after scanning through the LIS, LIP and LSP, the threshold g is reduced by a factor of a: $g \leftarrow g/a$ and the coding goes back to Step 4. The coding continues until certain termination condition has been satisfied.

3. BIT STREAM SYNTAX OF VISUAL PROGRESSIVE CODING

In the visual progressive coding (VIP), the decoder has to be informed the change of the active weights. There are three ways of doing this. We may negotiate a default weight changing strategy between the encoder and the decoder. The default weight approach eliminates the overhead sent to the decoder, however, it also limits the flexibility of the visual progression, as the number of default weights is limited. A more common approach is to let the encoder control the change of weights (viewing condition) during the embedding process, and the decoder merely receives and updates the weights according to the instruction of the encoder. When the number of coding units (CUs) is small, we may directly encode a tag which specifies the order of CU embedding, as the implementation of JPEG 2000 VM2. When the number of CUs is large, a usual approach is to explicitly send a visual mark (VM) every regular interval to inform the decoder whether the weights have changed. The syntax can be shown

in Figure 3. The visual mark (VM) is lead by an one bit symbol *M* indicating whether the weights have changed. If *M* is '0', the previous weights remain active. In the case of *M* equals '1', VIP sends the updated weights for all bands. Such syntax minimizes the overhead where there is no weight change. The regular interval for weight update has to be negotiated between the encoder and the decoder in advance. It can be, for example, after coding a band bitplane, or after the scan of the entire image. The longer the weight update interval, the less the overhead for updating weights, however, the granularity of weight changing will also be more coarse.

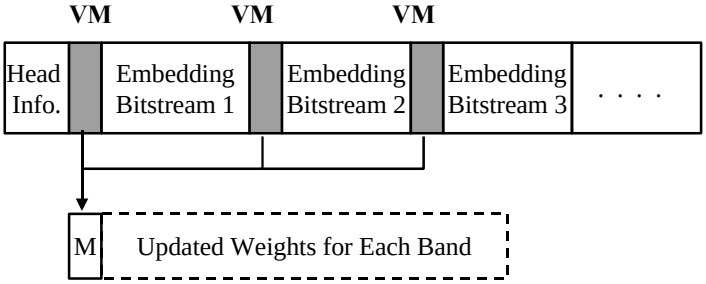

Figure 3 Syntax for visual progressive coding (VIP).

The proposed VIP syntax can support quality and spatial scalability as a special case as long as there are no CU with coefficients across multiple bands. For quality scalability, we just set the initial weights to be uniformly '1', and send a visual mark '0' every weight update interval indicating that the weights never change. To implement the spatial scalability, we first set the weights for the lowest resolution to be all '1' and those of the rest resolutions to be all '0'. Under such weights, the visual significance of coefficients reside outside the lowest resolution will be '0', so that VIP only encodes the coefficients in the lowest resolution. After all bit-planes of all coefficients in the lowest resolution have been encoded, VIP proceeds to the next lowest resolution. The weights of the new resolution are set to '1' and the weights of the rest resolutions are set to '0'. After all coefficients of that resolution have been encoded, VIP proceeds to an even higher resolution. The process repeats until all coefficients have been encoded.

4. EXPERIMENTAL RESULTS AND CONCLUSIONS

In this section, we demonstrate the subjective quality improvement offered by the visual progressive coding (VIP). The simulation software is the JPEG 2000 VM2, with non visual weighting mode (NW), fixed visual weighting mode (VW) and visual progression mode (VIP). The test image is the Bike with size 2048x2560. The image is compressed at 1.0 bit per pixel (bpp), and embedded decoded at 0.125 bpp and 1.0 bpp, respectively. For the fixed visual weighting, we assume that the image is to be viewed at a distance about 14 inches (35 cm), and the CSF visual weights are calculated according to the schemes proposed by Jones and Daly[2]. The same CSF weights are used in VIP before 0.125 bpp, and after that, the weights are set to be uniformly '1'. We show photo results in Figure 4, 5 and 6. Just for reference, we give the PSNR and RMSE values of the coded image in Table 1. However, keep in mind that the PSNR and RMSE is not a good measurement of the visual quality.

The decoded image at 0.125bpp is shown in Figure 4. The NW, VIP and VW coded images are shown at the up, central and bottom row respectively. It is easily observed that the subjective quality of the VIP (Figure 4b) coded image is substantially better than that of NW coded image (Figure 4a), and is close to that of VW coded image (Figure 4c). By emphasizing the frequency component which is more perceptible to human eyes, the VIP coded images look much clear, with less ringing artifacts around the bicycle rings and bar chart. More background stripes are revealed too in VIP and VW coded images. The fully decoded image at 1.0 bpp is shown in Figure 5. We find that all images appear very close, whether they are coded by NW, VIP and VW (Figure 5a-c). However, at that high bit rate, the user may want to blow up the image for analysis. If the images are blown up 4 times, as shown in Figure 6, we observe that VW coded image (Figure 6c) is smoother, and with stronger ringing artifact around sharp edges, while the VIP and NW coded images (Figure 6a and 6b) have less such artifacts. At high bitrate, the "reordering by weight" strategy of VIP makes it feasible to gradually phase out the visual

weighting so that the image can be viewed at a closer distance, while the VW coded image does not have such flexibility.

As a conclusion, the VIP coded image enables more flexible adjustment of the visual weights during the embedding. It takes advantage of the visual weighting at low bitrate, assigns more bits to the low pass coefficients and improves the global appearance of the image. At high bitrate, it phases out the visual weighting to accommodate a more flexible viewing condition and to keep the high frequency image details. VIP improves the subjective quality of embedded coding.

We have proposed the visual progressive coding (VIP) in the JPEG 2000 standard. Due to its flexible visual adjustment functionality, VIP is adopted into JPEG 2000 VM2 at the Copenhagen, Denmark meeting in July, 1998.

Table 1. Bike image coded by JPEG 2000 VM2 with no visual weighting (NW), visual progressive coding (VIP), and fixed visual weighting (VW).

Schemes	Coding Rate: 0.125 bpp		1.0bpp	
	PSNR (dB)	RMSE	PSNR (dB)	RMSE
No visual weighting (NW)	25.82	13.0526	38.12	3.1677
Visual progressive coding (VIP)	23.47	17.0968	38.11	3.1692
Fixed visual weighting (VW)	23.66	16.7230	30.88	7.2874

REFERENCES

[1] J. Li and S. Lei, "An embedded still image coder with rate-distortion optimization", *SPIE: Visual Communication and Image Processing*, volume 3309, pp. 36-47, San Jose, CA, Jan. 1998.

[2] Jones, Daly, Gaborski and Rabbani, "Comparative study of wavelet and DCT decompositions with equivalent quantization and encoding strategies for medical images", *SPIE V. 2431, Proceedings of Conference Medical Imaging,* pp. 571-582, 1995.

[3] J. Shapiro, "Embedded image coding using zerotree of wavelet coefficients", *IEEE Trans. On Signal Processing,* vol. 41, pp.3445-3462, Dec. 1993.

[4] A. Said, and W. Pearlman, "A new, fast and efficient image codec based on set partitioning in hierarchical trees", *IEEE Trans. On Circuit and System for Video Technology,* Vol. 6, No. 3, Jun. 1996, pp. 243-250

[5] D. Taubman and A. Zakhor, "Multirate 3-D subband coding of video", *IEEE Trans. On Image Processing,* Vol. 3, No. 5, Sept. 1994, pp.572-588.

[6] A. Zandi, J. D. Allen, E. L. Schwartz, M. Boliek, "CREW: compression with reversible embedded wavelets", *IEEE Data Compression Conference*, Snowbird, UT, March 1995.

[7] H. Wang and C. J. Kuo, "A multi-threshold wavelet coder (MTWC) for high fidelity image", *IEEE International Conference on Image Processing* ' 1997.

[8] W. B. Pennebaker and J. L. Mitchell, "JPEG still image data compression standard", New York: Van Nostrand Reinhold, 1993.

[9] J. L. Mitchell, "MPEG video compression standard", New York: Chapman & Hall, 1997.