Decoding CREW (Compression with Reversible Embedded Wavelets) codestreams

Martin Boliek, Michael J. Gormish, Edward L. Schwartz, Alexander Keith
Ricoh Silicon Valley, Inc.
2882 Sand Hill Road, Suite 115, Menlo Park, CA 94025
email: crew@crc.ricoh.com, www.crc.ricoh.com/CREW/

Abstract: *As the applications of digital imagery expand in resolution and pixel fidelity there is a greater need for more efficient compression and extraction of images and sub-images. No longer is it sufficient to compress and decompress an image for a specific target device. The ability to handle many types of image data, extract images at different resolutions and quality, lossless and lossy, zoom and pan, and extract regions-of-interest are the new measures of image compression system performance.*

CREW is a high quality image compression system that is progressive from high compression to lossless, and pyramidal in resolution. CREW supports regions-of-interest, and multiple image types, such as bi-level and continuous-tone. This paper describes the CREW system and format, shows how the correct data can be quickly extracted from a CREW file to support a variety of target devices, describes the mechanisms needed for panning, zooming, and fixed-size compression, and explains the superior performance on bi-level and graphic images.

2 Introduction

Wavelet transforms have been studied for image compression systems for over ten years [1]. The results are generally considered superior to Discrete Cosine Transform systems, such as the JPEG Still Image Compression standard [2][3]. Because wavelets have compact support and are overlapped, blocking the image to reduce computation is unnecessary. (Region-of-interest can be obtained by tile blocks, however.)

Wavelet image compression systems vary widely in complexity and character. However, except for a few notable systems, quantization is performed while encoding, just like JPEG, allowing only one possible decoding of the image.

Zerotree-based systems [4][5][6] are embedded (progressive by pixel fidelity), therefore, the quantization is a function of how much of the compressed file is decoded. Although progressive with respect to pixel fidelity, zerotree-based systems do not allow efficient extraction of the image at different resolutions. The wavelet transform's natural "pyramid" of image resolutions are not exposed, or available, in the same codestream in the zerotree-based methods. An earlier video coding system which offers bit-plane coded wavelet coefficients and scaling similar to CREW is in [7].

The FlashPix file format [8] has a pyramidal structure that allows the extraction of different resolution images. This structure allows zooming and panning through the image. However, it is achieved through the use of redundant copies of the image. Further, the only compression currently allowed is the lower quality lossy JPEG with only one quality available per redundant image.

Compression with Reversible Embedded Wavelets (CREW) [9][10][11][12][1] is a wavelet compression system that delivers excellent compression quality and enables great flexibility in quantization of the image while decoding. CREW technology enables compression with several features including:

- lossless and lossy encoding or decoding (lossless decoding requires that the entire file has been encoded),
- progression by both resolution and pixel fidelity selectable at decode time,
- fixed-rate and fixed-size encoding and decoding,
- region-of-interest (tiled) decoding,
- idempotency,
- color, gray-level, graphic, text, bi-level images,
- "parsable" codestream without decoding,
- and different coding styles for multiple components.

CREW's features enable many modern imaging systems to utilize image data in compressed form. For example, an image browser can pan and zoom through an image, without redundant decoding or encoding, using the progressive, pyramidal, and region-of-interest features.

In another example, using the fixed-size and idempotency or the lossless features, an image editor can decode the appropriate portions of an image, perform editing or filtering, and recompress those portions into the codestream. These operations can be performed at full resolution or postponed until a particular resolution is needed.

Finally, a World Wide Web (WWW) example is perhaps most illustrative of the need for this type of compression system. Using the pyramidal, progressive, and parsing features, a number of thumbnail images can be displayed on a web page. The client can select the appropriate image resolution and fidelity or size and the server can parse the compressed file. When the client wants to print, the remaining data needed for a print resolution image can be parsed by the server and sent to the client.

3 CREW image compression technology and standard

CREW is a complete compression system and syntax rather than just an algorithm. There is an architecture and file format associated with CREW. This section describes the various technology components that make up the CREW system. Fig. 1 shows the flow of the CREW system.

1. See http://www.crc.ricoh.com/CREW/.

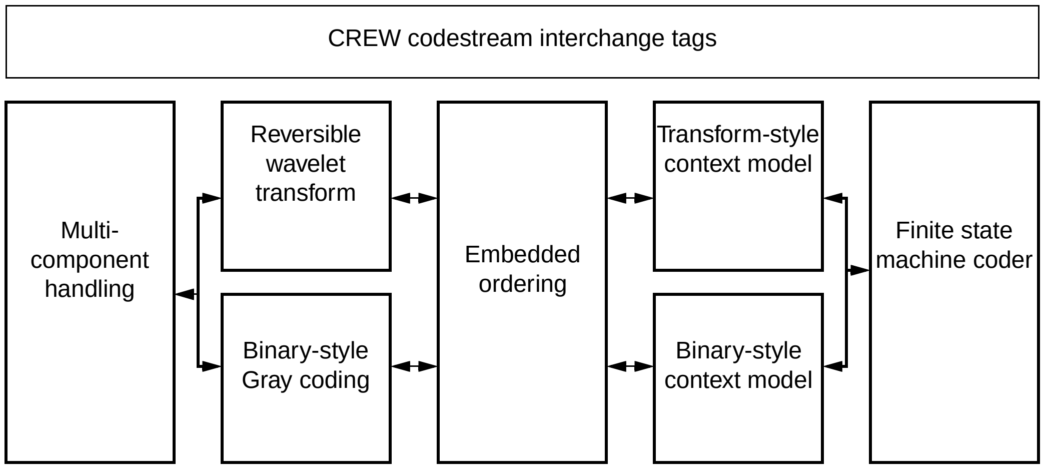

CREW codestream interchange tags
Multi-component handling
Reversible wavelet transform
Binary-style Gray coding
Embedded ordering
Transform-style context model
Binary-style context model
Finite state machine coder

Fig. 1 CREW standard block diagram (encode left to right, decode right to left)

3.1 *Image model and tag structure of a CREW codestream*

CREW can handle multiple component images. If the components are classic RGB or CYM color planes, CREW offers a reversible (lossless or lossy) color space conversion that is similar to YCrCb transform found in CCIR 601 [13].

For random access to regions, the image may be tiled using a regular rectangular grid. Thus, the image is divided into “tile-components.” The tile size is user chosen and arbitrary, however there is an impact on compression performance if the tile is too small. The tiles are rectangular regions of the image laid out in a regular way. For multiple component images each tile is assumed to have at least one element from each component. The components within a tile are coded independently and can use different coding styles.

CREW allows for a binary-style in addition to the wavelet transform-style of coding. For tile-components that have graphic or text regions (such as overlays, alpha planes, and transparency planes), binary-style coding can achieve higher quality than transform-style coding. Essentially, the transform is replaced with a simple Gray code for the designated tile-components.

The CREW tag format is much like the JPEG marker and tag format [2][3]. There is a main header that signals information about the image and tile size, components and component depth, and the coding style or styles. This header can also contain information about the size of the coded tiles and pointers to the entry points. (Entry points are locations in the file where the entropy coder is initialized, so the compressed data can be randomly accessed. This is described in greater detail below).

Each coded tile has its own header, called a tile header. This header can override the coding styles from the main header as appropriate for the data in that tile. The tile length and entry point information can also be signalled in the tile header if it is not signalled in the main header. This allows progressive encoding of an image on a tile by tile basis without the need to rewrite the main header.

3.2 *Reversible color space transform*

One of the most important features of CREW is the ability to encode and decode either losslessly or lossy. To have state-of-the-art compression for multiple component color images (red, green, blue or RGB) it is necessary to have a color space conversion, or transform, into an opponent color space such as YCrCb. This allows quantization in a Human Visual System weighted space. The CREW reversible color space transformation has no systemic error for when there is no quantization.

The CREW reversible color space transform is based on the following matrix:

$$\begin{bmatrix} Y \\ v \\ u \end{bmatrix} = \begin{bmatrix} \frac{1}{4} & \frac{2}{4} & \frac{1}{4} \\ 1 & -1 & 0 \\ 0 & -1 & 1 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{1}$$

The reversible approximation is defined by:

$$\begin{cases} Yr = \left\lfloor \dfrac{R + 2G + B}{4} \right\rfloor \\ Ur = R - G + 2^{depth} - 1 \\ Vr = B - G + 2^{depth} - 1 \end{cases} \tag{2}$$

where depth is the number of bit-planes required to store the original R, G, and B components (for example 8 bits for 0-255 unsigned components or -128 to +127 signed components). The symbol $\lfloor \circ \rfloor$ is the floor function meaning “greatest integer less than or equal to x,” e.g., floor (-0.3) = -1, floor (0.3) = 0, floor (-1) = -1.

The Y component requires the same number of bit-planes (bit depth) as the original components, and Ur, Vr will require one more bit. The Y component is an approximation of the luminance of the image and can be used as a grey-scale version. Each of these components is then tiled and coded separately.

3.3 *Transform-style coding and the reversible wavelet transform*

For each tile-component one of two coding styles, transform-style or binary-style, is chosen. Fig. 2 shows the data path for

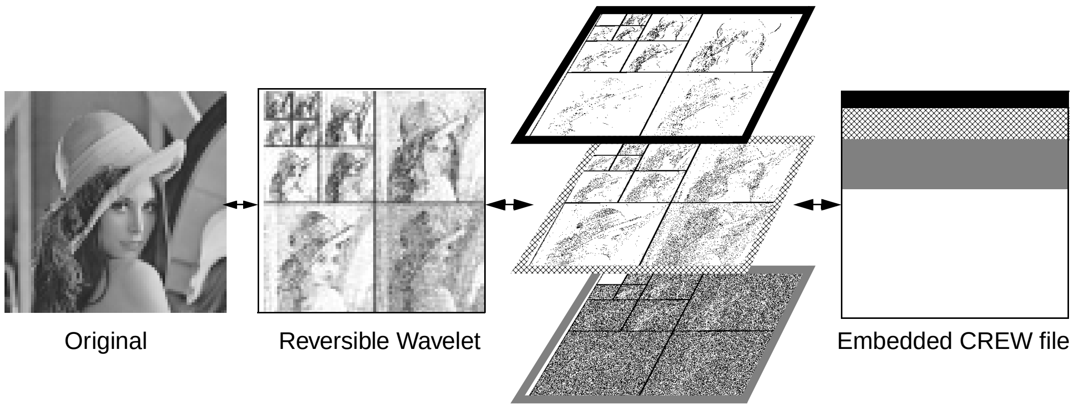


Fig. 2 CREW transform-style compression path

one tile-component coded in the transform-style. First, a reversible wavelet transform is performed. This is followed by shifting, or alignment, of the wavelet coefficients into a number of bit-planes called importance levels. These importance levels are then coded (either independently or in runs) by using a context model and binary entropy coder.

The image is transformed by the “Two-Ten” reversible wavelet transform (TT-transform) based on a decomposition of the Le Gall-Tabatabai polynomial [14]. This filter pair implementation has the special properties of reversibility (exact reconstruction with finite precision arithmetic) and no precision growth in the low pass coefficients. The one dimension filter is applied to the pixel components horizontally and vertically, decomposing the image in the classic wavelet form.

The two-tap low-pass, ten-tap high-pass reversible wavelet filters, called the TT-transform, are defined as follows. Let x(0), x(1), x(2),... be the input signal, and let s(0), s(1), and d(0), d(1), be the smooth and the detail outputs respectively. The smooth output and the detail output are the results of applying the low-pass and the high-pass filters, respectively. The TT-transform filters are the following:

$$\begin{cases} s(n) = \left\lfloor \dfrac{x(2n) + x(2n+1)}{2} \right\rfloor \\ d(n) = x(2n) - x(2n+1) + p(n) \end{cases} \tag{3}$$

where $p(n)$ is defined by

$$\left\lfloor \frac{3s(n-2) - 22s(n-1) + 22s(n+1) - 3s(n+2) + 32}{64} \right\rfloor \tag{4}$$

The inverse of the TT transform filters are given by,

$$\begin{cases} x(2n) = s(n) + \left\lfloor \dfrac{d(n) - p(n) + 1}{2} \right\rfloor \\ x(2n+1) = s(n) - \left\lfloor \dfrac{d(n) - p(n)}{2} \right\rfloor \end{cases} . \tag{5}$$

The TT-transform was chosen for several reasons. Filters of the Le Gall-Tabatabai form, including the TT and common seven-nine filter, have been shown to have excellent energy compaction and other desirable properties for compression [15]. The TT-transform is a good visual filter when the coefficients are quantized (lossy) and is reversible when not. Hence, the TT-transform lends itself quite well to both lossless and lossy compression. The TT-transform has no growth in the smooth output, i.e., if the input signal is b bits deep, so is the smooth output. And no systemic error is introduced due to rounding in the integer implementation of the transform, so all error in a lossy system can be controlled by quantization.

Of the four filters participating in a wavelet transform, the low-pass synthesis filter is the most important for image quality because it combines the quantized coefficients and also smooths the artifacts. This fact has led to the choice of a relatively long (10-tap) and particularly well-behaved filter for the low pass synthesis filter in CREW. The 10-tap low pass synthesis filter makes the TT-transform an overlapped transform, contributing substantially to its high quality in both lossless and lossy compression.

The ten-tap filter can be computed by scaling and adding two previous and two future two-tap filters. This scaling can be achieved with only shifts and adds, further reducing computation. These factors, combined with pure integer arithmetic, make this filter pair very low in computational complexity.

Reasons such as implementation ease, memory cost, random access, etc., often make it desirable to have image tiles. To reduce artifacts caused by tiling, symmetric mirroring of pixels and coefficients is used. Also, the filter characteristics enable simple post-processing reconstruction techniques for removing tile boundary effects for lossy reconstruction. At the tile boundary, it is possible to estimate the value of coefficients to a more fine degree than the quantization step size by using the two-tap filter coefficients from the neighboring tile.

3.4 *Alignment and importance levels*

The transform coefficients of a frequency band are aligned with respect to coefficients of other frequency bands into “importance levels.” That is, the coefficients are shifted (multiplied by factors of two) with respect to each other

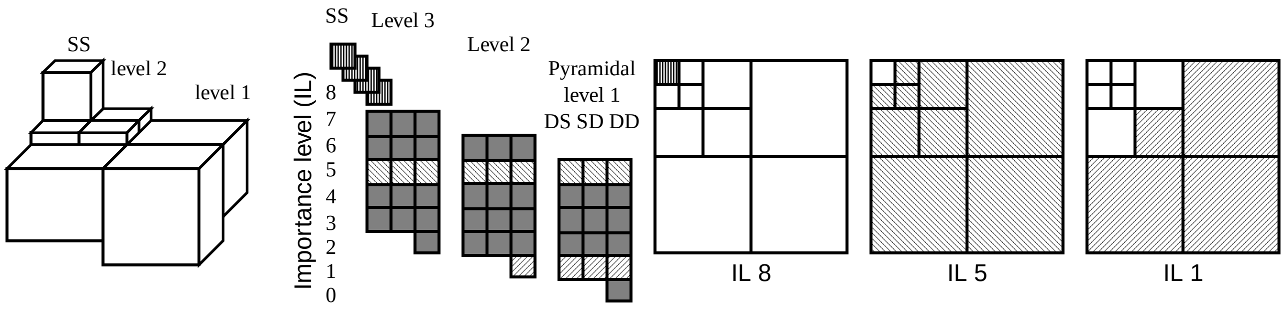


Fig. 3 Normal alignment and example importance levels

according to some user defined criteria, such as the best rate-distortion performance with respect to MSE.

In some sense alignment is a precursor to quantization. (Actual quantization occurs when the stream is truncated, or extracted, at encode, decode or both. The unnecessary portions of the file are not transmitted.) A given alignment determines the order data is written to the stream and, thus, the quantization which occurs if the file is truncated. (It will be shown below that simple truncation is not the only method of quantizing or extracting a lossy image from the codestream.) Consider a tile-component of an image with 4 bits per pixel. The mathematics of the filters lead to a maximum magnitude of Detail-Detail (DD) coefficients of 6 bits plus a sign bit, and the Detail-Smooth (DS) and Smooth-Detail (SD) coefficients of 5 bits plus a sign bit, while the single level of Smooth-Smooth (SS) coefficients are 4 bits unsigned.

The SS coefficients are placed in the codestream uncoded and packed as a number rather than bit-planes. This enables fast retrieval of thumbnail images. If enough levels of pyramidal decomposition are used, e.g. the SS coefficients are few in number, there is little or no effect on compression efficiency.

Fig. 3a show the magnitudes of the coefficients in each frequency band can be aligned with respect to each other. Notice that the SS coefficients are shifted above all the rest. This alignment is called “normal” alignment.

Fig. 3b is a different representation of the same alignment. Each bit-plane of each frequency band coefficient is shown by a box. This represents the maximum magnitude of the coefficients in that particular frequency band. This special alignment has the property of being progressive by fidelity in a linear coded bit stream. That is, normal alignment leads to a nearly orthogonal transform. Truncating importance levels is equivalent to uniform scaler quantization, which is optimal in terms of MSE or SNR for an orthogonal transform. This truncation can be at a target bit-rate, transmission time, or quality level.

Fig. 3c shows three example importance levels, 8, 5, 1. The gray and hatched areas represent coefficients that have magnitude bits at that level. The white areas are deterministically known to contain no data and are not encoded.

Another example of alignment shown in Fig. 4 called “pyramidal” alignment. Here the coefficients are grouped into pyramidal levels (the preferred alignment). Empirical experiments show the compression efficiency of these two different alignments is essentially the same.

3.5 *Context models*

Coefficients are represented in bit-significance form [5]. The magnitudes of the coefficients are coded from most significant to least significant. The leading zero value bits and the highest magnitude one value bit are called the head bits. After encoding the last head bit a coefficient has a non-zero magnitude for the first time in the codestream. Therefore, the sign bit is coded immediately after the last head bit. The remaining bits are called tail bits. Both the sign and tail bits have unique context models.

The importance levels are encoded as if they were bit-planes in a binary image, much like JBIG [16]. The highest importance level comes first and the lower ones follow in order. To code the importance levels several context models are used. These context models use previously coded (causal) information to estimate the probability of an event occurring. Each importance level is encoded and decoded in the same order and is the minimum size of an independently coded segment of data. If 9 of the last 10 times a zero value bit followed some event, or context, the probability of the bit

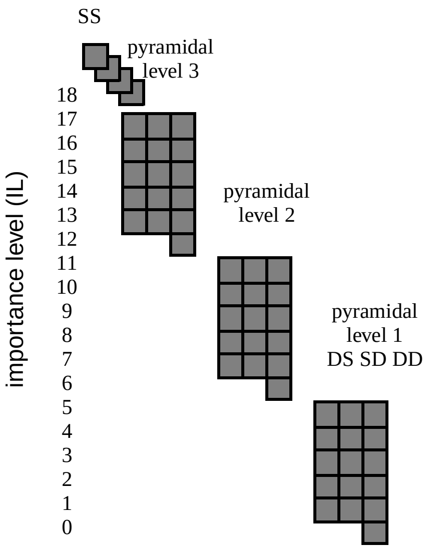


Fig. 4 Pyramidal alignment

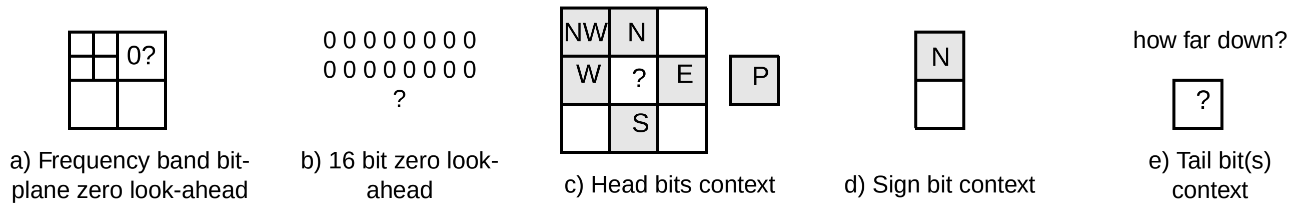


Fig. 5 Transform-style context model

following the next occurrence of this event having the value zero is estimated be 0.9, or 90 percent. Again, much like JBIG, CREW uses a state machine instead of explicit counts to estimate probabilities. This reduces the amount of information that must be stored for each context.

Fig. 5 shows five of the context models used in CREW. (Note that there are slight differences between the binary-style and transform-style context models. The transform-style is described here.) A context model can be thought of as a series of questions. The first context, or question, (Fig. 5a) is "Are all the bits in the current frequency band zero?" If yes, code a one bit and go to the next frequency band. If no, code a zero bit and go to the next context.

Next, the context model (Fig. 5b) determines whether it is worth examining a group of bits (8 wide and 2 high) for a zero run. If the 16 bits are all head bits, the "northern" 8 coefficients and the "parent" coefficient are examined. If they are all zero head bits then the next context asks, "Are all the bits in this group zero?" If yes code a one bit and go to the next 16. If no, go to the next context model.

The context model for an individual coefficient first determines whether the current bit of the current coefficient is a head bit or tail bit. If it is a head bit, the neighbor coefficients and the parent coefficient are examined and a context is generated as a function of all previous bit-planes and the current bit-plane for causal pixels (Fig. 5c). This context and the bit value are sent the entropy coder. If the bit is a one value head bit, then the sign bit is immediately coded using the northern neighbor sign value as the context (Fig. 5d). Finally, tail bits are coded with a context that is a function of how many tail bits have already been coded (Fig. 5e).

3.6 *Binary entropy coding with the Finite State Machine*

The Finite State Machine (FSM) keeps a probability estimate for every context used by the context model. Different probability estimates are used to compress the outcome bits from the context model. After using the estimate for coding, it is updated based on the bit just coded[17].

The FSM is actually quite simple to implement. It consists of a memory to hold estimates and logic to update the context probabilities and a look-up table to transition between states and output coded bits. Fig. 6 shows these components. The values of the context model are updated after each bit is coded. The bit generation state table has 61 static entries or states. The table is not dependent on the image, coding style, or anything else. The values for this table are available through the CREW web page (www.crc.ricoh.com/CREW).

The FSM machine is similar to the Q-coder in compression performance. However, since there are a finite number of possible states, there is no carry-over problem. It is also very simple to implement and is low in complexity.

Fig. 7 shows the relative compression performance of the FSM coder versus the Q-coder for coding symbols with a fixed probability. The y-axis is the amount over the entropy limit in bits per pixel and the x-axis is probability between 0.5 and 1.0. Notice that the FSM performs better at lower probabilities. At the highest probabilities the Q-coder has better performance because it has more states that it can transition through before outputting a coded bit. The CREW context model (described above) is designed to collect runs of highly probable results and send less probable decisions to the entropy coder. Thus, within the CREW system, the FSM coder outperforms the Q-coder. In addition to better compression results, collecting bits also speeds the execution time of the coder.

3.7 *Entry points*

The context model and binary coder can be reset at the beginning of any importance level, allowing random access

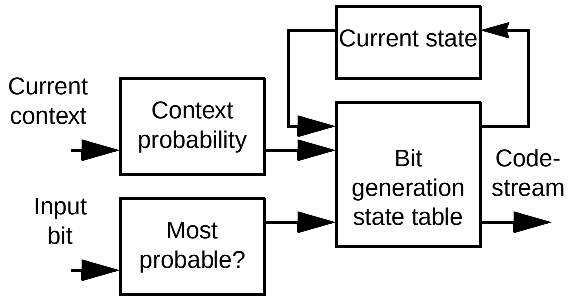


Fig. 6 Finite State Machine (encoding)

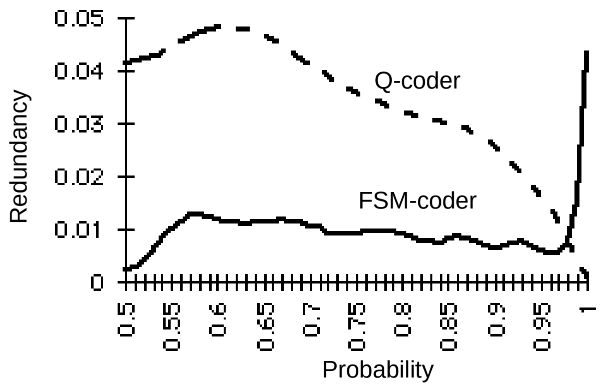


Fig. 7 FSM versus Q-coder

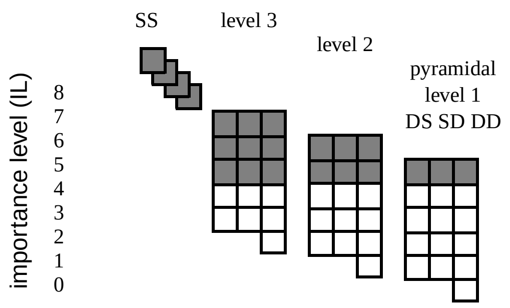
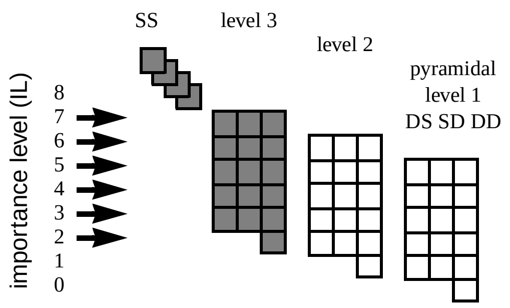


Fig. 8 Progressive pixel fidelity, normal alignment



Entry points at 7, 6, 5, 4, 3, and 2.

Fig. 10 Pyramidal resolution, normal alignment

or an "entry point." Thus, any importance level, or group of consecutive importance levels, can be independently coded. Pointers are signalled in the headers to allow direct access to these entry points. By careful selection of the levels with entry points, a number of features of CREW are enabled. This is discussed further below.

Each entry point usually requires at least 16 bits of signaling. With common tile sizes (256x256) and pyramidal alignment (6 level, 5 entry points), the signaling cost of these tags is about 0.001 bits per pixel. For some images, alignments, and entry points there is a coding benefit for resetting to coder.

3.8 *Binary-style coding*

Many images, or components or regions of images, contain sharp edges such as text regions or noiseless graphics. This type of imagery is poorly modeled by smooth transform coefficients typical of wavelets. Since CREW is already a grey-level bit-plane coding system, it is a simple matter to remove the wavelet transform from the compression path for these image components. Instead of the wavelet transform, the bits of the component magnitude are Gray coded with respect to each other. The Gray coded components can be sub-sampled, if desired, and then coded with the same alignment and entry points, a variation on the context model, and the same FSM as with the wavelet coefficients.

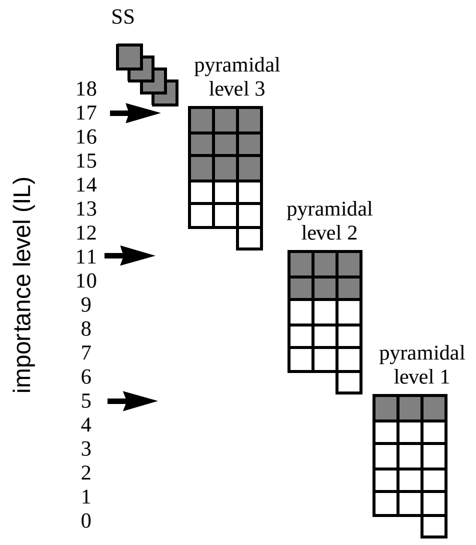
4 Decoding with CREW

CREW is unique in that several types of reconstruction or quantization decisions can be made by the user or client at decode time. Tiling, coding style, decomposition levels, and the location of entry points are all decided at encode time. Quantization and resolution decisions can be made at decode, encode or both. This allows one compressed image to serve multiple dissimilar target devices like printers and monitors. Also, the image may be browsed, panned and zoomed without decoding any more data than what is needed for these operations. Below are some examples.

4.1 *Progressive pixel fidelity or fixed rate*

Sometimes it is desirable to limit the number of bits transmitted or limit the storage size of an image. With CREW an exact fixed bit-rate can be achieved at encode or decode time. Other times it is desirable to achieve a target quality or fidelity. The same mechanism applies.

Fig. 8 and Fig. 9 show the quantization on importance



Entry points at 17, 11, and 5

Fig. 9 Progressive pixel fidelity, pyramidal alignment

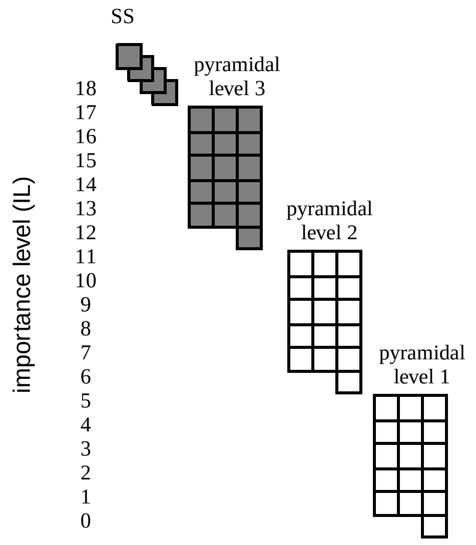


Fig. 11 Pyramidal resolution, pyramidal alignment

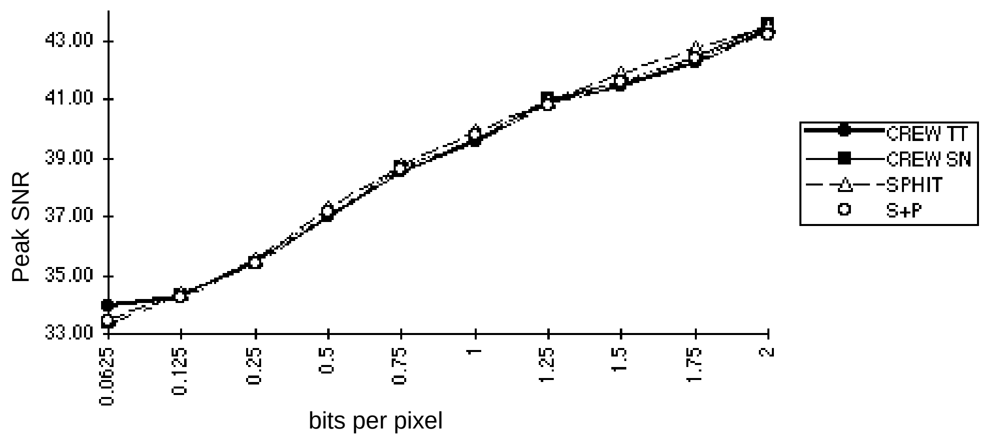


Fig. 12 CREW with the two-ten filter versus CREW with seven-nine, SPIHT with seven-nine and S+P filters

levels for the two alignments that have been described. Fig. 8 shows the normal alignment with the lowest four importance levels (shown in white) truncated. This quantization assures the best possible image, with respect to MSE or SNR, for that bit-rate. Also, because the coefficients are linearly related to the pixels, a quality bound, in terms of MSE or SNR, can be assured.

Fig. 9 shows the pyramidal alignment with the exact same quantization. This is achieved by placing entry points at the first importance level of each pyramidal level. Thus, after the appropriate number of importance levels in one pyramidal level, the system can skip to the beginning of the next importance level.

4.2 *Progressive pyramidal resolution*

Fig. 10 and Fig. 11 show the two types of aligned images quantized for lower pyramidal resolution. Once again, the regions shown in grey are either encoded or decoded and the white regions are quantized. The pyramidal alignment (Fig. 11) uses trivial truncation to achieve this end. The normal alignment (Fig. 10) has entry points that allow the decoder to decode the part of the importance level that is necessary and skip to the next important level. This type of quantization provides images with reduced spatial resolution.

4.3 *"Custom" alignments*

The CREW syntax allows any custom alignment to be used. This flexibility allows alignments for purposes not thought of by the authors.

4.4 *Region-of-interest, zooming and panning*

Regions of interest can be decoded independently with the use of tiling. As discussed before, tiles are rectangular segments of the image that are independently coded. A set of pointers allow the decode to select the right tile or tiles.

By repeatedly selecting the correct adjacent tiles, an application can pan through an image. Also, by using the pyramidal resolution (or the reversible wavelet transform's ability to interpolate higher resolution images) zooming can be achieved.

4.5 *Parsing a compressed image*

Parsing refers to extracting the right data from a CREW file to reconstruct a quantized image without decoding. Imagine using this feature for a WWW client-server protocol. A high resolution lossless or near-lossless image is stored on the server. Perhaps a thumbnail image is extracted from the server, sent in a minimal bit stream, and decoded at the client. Then, according to the instructions from the user or client, a pyramidal segment of the image is sent to fill the window area of a particular screen. Finally, the full resolution image could be sent for printing. All of this is done without decoding at the server or sending redundant data. Nearly all of the quantization that can be performed by a decoder can also be performed by a parser.

5 Comparison of CREW with other systems

Fig. 12 shows the relative performance, with respect to peak SNR, of

- CREW with the two-ten reversible filter,
- CREW with the classic seven-nine filter,
- SPIHT system with the seven-nine filter,
- SPIHT with the S+P reversible filter.

SPIHT [6] is widely considered to be one of the best compression systems and uses a variant of Wavelet Embedded Zerotree. The image is the grey-level version of the SCID "cafe" image used for the JPEG 2000 experiments. (Similar performance was found for other images.)

Note that the performance of all the systems are very similar. CREW with the two-ten filter performs almost as well as the seven-nine filter and generally slightly better than SPIHT with the S+P filter. CREW with the seven-nine filter has generally the same performance as SPIHT with the seven-nine filter.

Although the SPHIT with the seven-nine filter slightly outperforms CREW with the two-ten reversible filter in terms of PSNR, CREW generally performed better in subjective tests run by the JPEG committee [18]. This suggests that the two-ten filter is actually better when viewed subjectively than when measured objectively.

The CREW systems achieves this performances without sacrificing any features. CREW allows progression in both the resolution and pixel fidelity axis. With SPIHT a choice of one or the other must be made at encode time. If the choice is made for progression in resolution, the Set Partitioning Hierarchical Trees method cannot be used. Also, CREW is capable of lossless compression.

6 Future plans and research

CREW has been submitted to the ISO/IEC JTC1/SC29/WG1 (JPEG/JBIG) committee for the new JPEG 2000 next generation still image compression standard effort. In fact, the CREW technology was key in prompting this standardization effort [19].

The flexibility of the CREW system suggests a number of areas of research to optimize its use for specific applications. For example medical imaging, facsimile, digital cameras, scanners, printers, copiers, CD-ROM archival systems, image browsing, remote sensing, and the World Wide Web all have different image types and system requirements. All can benefit from the flexibility of CREW.

7 Conclusions

Accessibility and flexibility of an image compression system is of more importance as the size and pixel depth of images in modern digital imaging systems increases. Functions such as access to different resolutions and pixel fidelities, zooming and panning, region of interest, and fixed-size can be achieved with CREW without sacrificing compression performance or image quality.

8 Bibliography

[1] O. Rioul and M. Vetterli, “Wavelets and Signal Processing,” IEEE SP Magazine, pp. 14-38, October 1991.

[2] W. B. Pennebaker and J. L. Mitchell, *JPEG still image data compression standard*, Van Nostrand Reinhold, New York, NY, 1992.

[3] JPEG, *Digital Compression and Coding of Continuous-tone Still Images, Part 1: Requirements and guidelines*, ISO/IEC DIS 10918-1, 1992.

[4] A. Lewis and G. Knowles, “Image compression using the 2-D wavelet transform,” *IEEE Trans. Image Proc.*, vol. 1, pp. 244–250, April 1992.

[5] J. Shapiro, “An embedded hierarchical image coder using zerotrees of wavelet coefficients,” *Proc. IEEE Data Compression Conference*, Snowbird, UT, pp. 214–223, March 1993.

[6] A. Said and W. Pearlman, “Reversible image compression via multiresolution representation and predictive coding,” *SPIE Visual Communications and Image Processing*, vol. 2094, pp. 664-674, November 1993.

[7] D. Taubman and A. Zakor, “Multirate 3-D Subband Coding of Video,” *IEEE Trans. on Image Proc.*, Vol. 3 No.5, September 1994.

[8] FlashPix format, www.kodak.com/daiHome/flashPix/flashPix.shtml.

[9] A. Zandi, J. Allen, E. Schwartz, and M. Boliek, “CREW: Compression with reversible embedded wavelets,” *IEEE Data Compression Conference*, Snowbird, UT, pp. 212–221, March 1995.

[10] E. L. Schwartz, A. Zandi, M. Boliek, “*Implementation of Compression with Reversible Embedded Wavelets*,” *SPIE*, vol. 2564, San Diego, CA, 11 July 1995.

[11] M. J. Gormish, E. L. Schwartz, A. Keith, M. Boliek, and A. Zandi, “Lossless and near-lossless compression for high quality images,” SPIE, San Jose, CA, February 1997.

[12] M. Boliek, M.J. Gormish, E. L. Schwartz, and A. Keith, “Next Generation Image Compression and Manipulation Using CREW,” *Int. Conf. on Image Processing*, Santa Barbara, CA, October 1997.

[13] ITU, “Encoding Parameters of Digital Television for Studios,” *ITU CCIR Recommendation 601-1*.

[14] D. Le Gall and A. Tabatabai, “Sub-band coding of digital images using symmetric short kernel filters and arithmetic coding techniques,” *IEEE International Conference on Acoustics, Speech and Signal Processing*, New York, NY, pp. 761–765, 1988.

[15] j. Villasenor, B. Belzer, J. Liao, “Wavelet Filter Evaluation for Image Compression,” *IEEE Trans. Image Processing*, Vol. 4, No. 8, pp. 1053-1060, August 1995.

[16] JBIG, *Information technology - Coded representation of picture and audio information - Progressive bi-level image compression*, ISO/IEC IS 11544, 1993.

[17] M. J. Gormish and J. Allen, “Finite state machine binary entropy coding,” *Proc. Data Compression Conference*, Snowbird, UT, p. 449, March 1993.

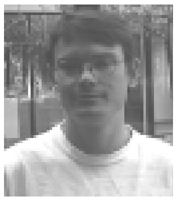
[18] JPEG committee, “JPEG 2000 Algorithm Submissions,” *ISO/IEC JTC1/SC29/WG1 N705*, November 1997.

[19] A. Zandi, M. Boliek, E.L. Schwartz, M.J. Gormish, J.D. Allen, “*CREW Lossless/Lossy Image Compression Contribution to ISO/IEC JTC 1.29.12*,” proposal submitted to the ISO/IEC JTC1/SC29/WG1 (JPEG/JBIG), CRC-TR-9524, 30 June 1995.

Martin Boliek received his bachelor’s degree in Physics from the University of California, Santa Cruz in 1984 and his masters in electrical engineering from the University of California, Davis in 1990. He has been working in the field of image processing for over twelve years, the last seven at RICOH.

Michael J. Gormish received his dual bachelors in mathematics and electrical engineering from the University of Kansas in 1989 and his masters and P.h.D. from Stanford University in 1991 and 1994 respectively. His focus has been image processing, compression, and entropy coding. He has been at RICOH for five years.

Edward L. Schwartz received his bachelors and masters in electrical engineering from the University of California, Davis in 1988 and 1990 respectively. He has focused in image processing and compression and the hardware implementation of algorithms. He has been at RICOH for six years.



Alexander Keith received his bachelor and masters degree in physics from the University of California, Santa Cruz in 1989 and 1991 respectively. He has focused on software architecture and development for image processing algorithms. He has been at RICOH for two years.