EXPLOITING LOCAL COLOUR DEPENDENCIES IN BINARY TREE PREDICTIVE CODING

J A Robinson

University of York, UK.

INTRODUCTION

Binary Tree Predictive Coding (BTPC) is a general-purpose compression scheme for still images. Its design criteria are:

- Good rate/distortion performance for both lossy and lossless compression of all image types (including photographs, line graphics, text images, high-quality shaded graphics, and medical images).
- Fast operation, particularly for decoding
- Patent-free

When initially reported [1], BTPC was compared with JPEG, GIF and research coders of that time, SPIHT and CALIC. It outperformed JPEG and GIF (except for limited-palette colour graphics) but was inferior to SPIHT and CALIC in their respective areas of strength. Tested with an assortment of images of different modalities, BTPC was the most consistent overall performer, other coders faring poorly on image types outside their design limits.

An implementation of BTPC (version 4.1) has been available online since 1997. This has been widely downloaded, tested and applied.

On its initial release, two problems were identified with BTPC:

- The most visible coding artifact at low data rates is contouring. This effect should be reduced, even at the expense of objective quality (PSNR).
- BTPC performs significantly poorer than GIF for limited-palette, highly-structured colour graphics, where colour quantization allows very high compression.

Today, the most telling problem for BTPC is that the research systems against which it was compared have been superseded by standards. JPEG-LS and JPEG 2000 not only outcompress SPIHT and CALIC [2], but fast implementations are available. Moreover, GIF now has a standard patent-free replacement in PNG, and although JPEG 2000 is subject to many patents, the JPEG 2000 committee has negotiated royalty-free use of all the algorithms in the core method. Thus BTPC's lack of patents is no longer a telling advantage.

This paper reports developments aimed at solving the above technical problems and updated experimental comparisons showing that the latest version of BTPC remains competitive for general-purpose image coding.

BTPC METHOD

In common with a small number of other predictive coding schemes (e.g. [3-5]), BTPC divides an input picture into rasters of pels, evenly subsampled from the original, with the rasters coded in order of increasing density. In the example block of image pels shown in figure 1, the coding order is A1, A2, A3, A4, B1, C1, C2, … , E11, E12.

A1	E1	C1	E2	A2
E3	D1	E4	D2	E5
C2	E6	B1	E7	C3
E8	D3	E9	D4	E10
A3	E11	C4	E12	A4

Figure 1. Example block of image pels showing membership in five rasters.

A pel in the current raster is predicted from pels in the previous and current rasters that surround it spatially, for example E6 is predicted from E3, D1, E4, C2, B1 and D3. The method is therefore interpolative. BTPC differs from other such schemes in that its predictions are non-linear, and adaptive to luminance surface shape (and thereby to different types of picture content). Prediction errors are arranged into a binary tree structure for efficient coding of blocks of zeros. For example, B1 is the parent of C1 and C2, which in turn are parents of D1 and D2, and D3 and D4 respectively. A leaf codeword at B1 indicates that all descendents are predicted without error. A set of Huffman coders provides backend lossless coding of prediction errors and tree leaves. Details of the original method are provided in [1] and an improvement is documented in [6].

NEW DEVELOPMENTS

I have conducted systematic experiments aimed at improving BTPC's performance relative to the current standards. Some of this work has produced improvements for both monochrome and colour image coding:

- The predictor adaptation to local luminance variance has been reformulated to reduce the visibility of contouring on photographic images. Essentially this means

using more pels in the predictor in areas of low variation.
- The coding of Huffman tables has been improved so that small pictures are compressed more efficiently.
- The Huffman coders now incorporate runlength coding where this provides a benefit, so that large pictures are compressed more efficiently.

Other investigations resulted in small overall improvement at significant computational cost, so have not been incorporated in the next release version of BTPC. These include

- Adaptation of the Huffman code tables dependent on signal level and local activity.
- Vector predictions for simultaneous coding of three colour channels.

However, two investigations focused on exploiting the local dependencies between colour channels did yield significant improvements.

Linking prediction across colour channels

Most colour image coding schemes exploit statistical correlations between colour components through a global transformation of the colour space – typically a fixed transform from R,G,B to Y,C1,C2. BTPC 5 uses a global transform which is based on the statistics of the image. However, an adaptive predictive coding scheme is also able to exploit local dependencies between components. In BTPC, this works as follows: The first component recovered from a prediction and a non-zero prediction error is compared, after decoding, with the prediction region and used to select a predictor that would have been better than the one used. This modified predictor is then used for the remaining two components if their local prediction surfaces are of similar shape. This method means that prediction of components 2 and 3 (typically the chrominance) is based on the spatial structure of component 1, which normally yields a significant reduction in the prediction error for those components.

Integrated coding of zero prediction errors

An integrated binary tree over all three colour channels provides more efficient coding of blocks of zeros. I investigated a number of structures for simultaneous coding of zeros in colour images, including a hex-tree structure wherein codewords could signify that any combination of the three colour channels had all zeros below the current pel. However, the relatively simple binary tree proved to be almost as data-rate efficient and far faster. This modification sends a leaf codeword at the current pel if all pels below it in the tree, in all three components, are predicted without error. The use of the modified predictor, described above, increases the number of zeros in the second and third components, so the leaf-codeword overhead is only slightly greater than that for coding the first component only.

The improvements described above plus numerous smaller changes have been incorporated into BTPC version 5.

EXPERIMENTS

BTPC 5 was compared against the following systems:
JPEG-LS (LOCO I) -- The Hewlett-Package locoe/locod implementation [7]. Experiments used all three sample, line and plane interleaving options, and the one that compressed most was chosen for each image. (Sometimes this was not the option recommended for a particular image type.)
JPEG 2000 -- The Kakadu V3.4 implementation [8]. Equal weighting of colour channels was used to maximize PSNR.
GIF encoding was independently done with the ImageMagick convert tool and the image viewer xv. These palletize differently and give different compression rates. The better was chosen in each case.
PNG encoding was done with the Image Magick convert tool.
Baseline JPEG -- The Independent JPEG Group's implementation. This rescales input images to [0,255] so images were pre- and post- processed to ensure fair PSNR results.
BTPC 5 -- The new version of BTPC incorporating the improvements described above.

The experiments are summarized with representative and BTPC worst-case examples in figures 2 to 8. For lossless coding, only the better of GIF and PNG is shown in each case.

DISCUSSION AND CONCLUSIONS

BTPC, even with its improvements, underperforms its best competitors in their areas of strength. However, the degree of underperformance is, arguably, minor, compared to BTPC's advantage of good performance across *all* image types.

For lossless compression of photographs, JPEG-LS outperforms BTPC (and PNG) by 0-10%. For lossy compression of photos, JPEG 2000 beats BTPC by between 0 and 2 dB over all usable bitrates. Although subjective tests have not been performed, figure 2 allows com-

parison of JPEG 2000 and BTPC at equal bitrates and at equal PSNRs. This illustrates that PSNR may favour JPEG 2000 relative to subjective judgement. BTPC's relative performance improves for smaller images: below 100 x 100 pels it is usually competitive with JPEG 2000, even on PSNR terms, making it equally efficient for any application where large images are sliced (e.g. in web-based interactive applications). For lossless compression of graphics, BTPC is inferior to the better of GIF and PNG, and where palettization is possible, it can still be significantly worse. However, it achieves lossy coding of graphics without visible artifacts at well below the GIF and PNG rates, and is also much better than any other lossy system for graphics. JPEG 2000 typically requires at least 50% more bits than BTPC for high-quality graphics coding. BTPC performs equally well with JPEG 2000 over composite (mixed text, graphics and photograph) images, and better than the lossless alternatives.

BTPC 5 encoding and decoding times have not been optimized. The comparison implementation of JPEG-LS also identifies itself as unoptimized, and its decoding takes about twice as long as BTPC decoding. The comparison implementation of JPEG 2000 is a commercial system which claims to be "heavily optimized" and decodes at about the same speed, on average, as BTPC 5.

The improvements to BTPC described here suggest that it is a good choice (though not an official standard) for application-independent image coding. It can substitute for JPEG, GIF, JPEG 2000, JPEG-LS and PNG without incurring the risk of substantial underperformance. Although it does not have all the features of JPEG 2000 (such as ROI coding), work is continuing to produce a competitive progressive version and improve its error resilience. BTPC remains patent-free.

REFERENCES

[1] J A Robinson, "Efficient General-Purpose Image Compression with Binary Tree Predictive Coding", IEEE Trans on Image Processing, Vol 6, No 4, April 1997, pp 601-607.

[2] D Santa-Cruz, R Grosbois, T Ebrahimi, "JPEG 2000 performance evaluation and assessment", Signal Processing: Image Communication, Vol 17 Number 1, pp 113-130, 2002.

[3] P Roos, M A Viergever, M C A can Dijke, J H Peters, "Reversible Intraframe Coding of Medical Images", IEEE Trans Med Imag, Vol 7, pp 328-336, Dec 1988.

[4] L Arnold, "Interpolative Coding of Images with Temporally Increasing Resolution," Signal Processing, Vol 17, pp 151-160, 1989

[5] R L de Quieroz, D A F Florencio, R W Shafer, "Nonexpansive Pyramid for Image Coding using a Nonlinear Filterbank", IEEE Trans Im Proc, Vol 7, No 2, pp 246-252, 1998.

[6] J A Robinson, "In-Band Redundancy Removal for Binary Tree Predictive Coding", Proc First Internat Symposium on Communication Systems and Digital Signal Processing, Sheffield, UK, 6-8 April 1998, pp 52-55.

[7] M Weinberger, G Seroussi, G Sapiro, "LOCO-I: A Low Complexity, Context-Based, Lossless Image Compression Algorithm, " Proc IEEE Data Compression Conference, Snowbird, Utah, March April 1996. http://www.hpl.hp.com/loco

[8] "Kakadu Software – A Comprehensive Framework for JPEG2000", http://www.ee.unsw.edu.au/~taubman/kakadusoftware/index.html.

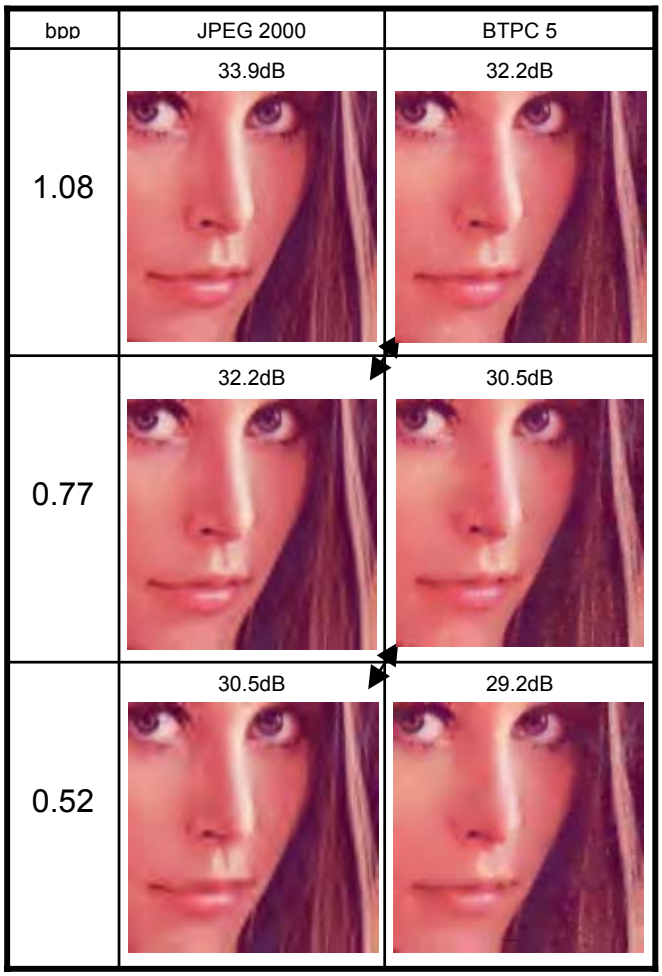

Figure 2: Objective vs. subjective measures. The graphs shown in figures 3-8 compare Peak Signal-to-Noise Ratios. Famously, PSNR does not necessarily correspond to subjective judgements. The relative visibility of coding artifacts in JPEG2000 and BTPC 5 is illustrated here at three bit rates, chosen so that the PSNR achieved by JPEG 2000 at a given rate is matched by BTPC at the next rate up. Comparison of the equal-PSNR cases (shown with arrows) in this and other photographic images suggests that PSNR does not unfairly advantage BTPC relative to JPEG2000, and the reverse may be true.

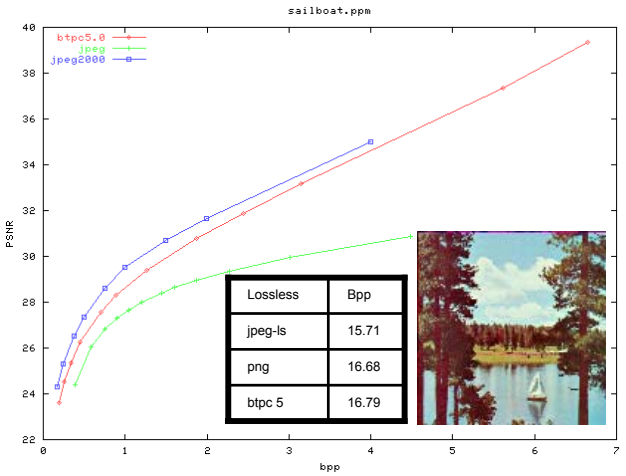

Figure 3: A representative example of coding a photographic image. In most cases BTPC 5's performance is inferior to JPEG2000, but the gap between BTPC and JPEG2000 has been closed by the exploitation of colour dependencies in version 5. For lossless coding of photographs BTPC uses 0-10% more bits than JPEG-LS.

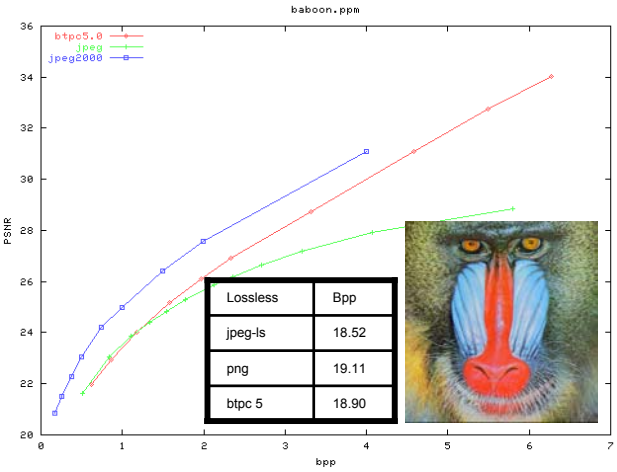

Figure 4: BTPC's worst case out of a test set of 80 standard photographic images. For this highly-textured colour image, BTPC 5 is also worse than BTPC 4.1 at low data rates! This is the only such case and illustrates where the assumption of local colour dependence fails.

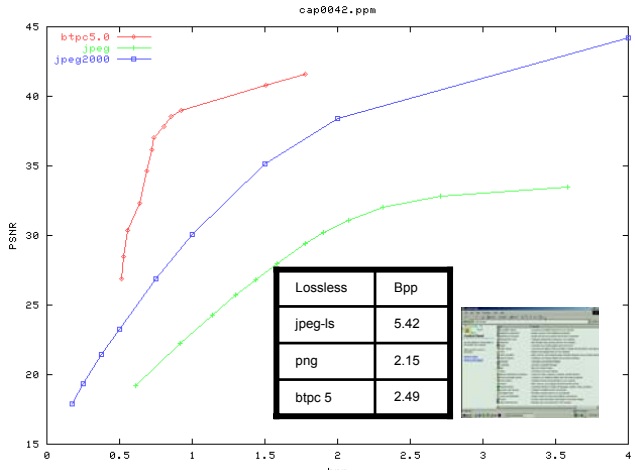

Figure 5: A representative example of coding a graphical/textual image, in this case a screen shot. Sometimes worse than PNG for lossless coding, BTPC's lossy coding causes little visible distortion even at half the PNG data rate. BTPC always outperforms the lossy alternatives, usually by a significant amount.

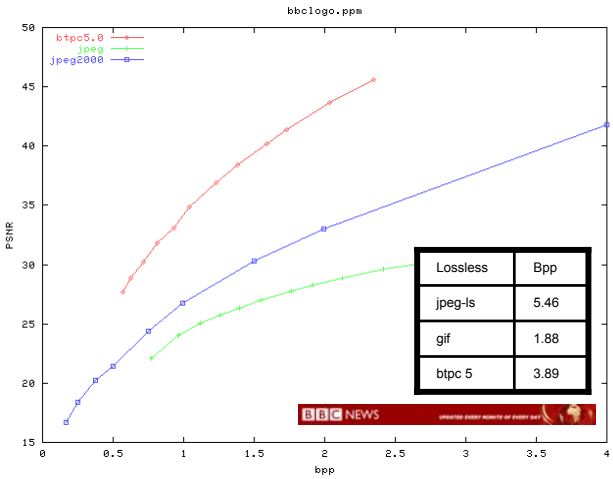

Figure 6: BTPC's worst case out of a test set of 40 graphical images relative to the better of PNG and GIF. GIF performs best on small, palettized, highly-structured images. In such cases, lossy BTPC coding at the rate achieved by GIF produces very good graphics

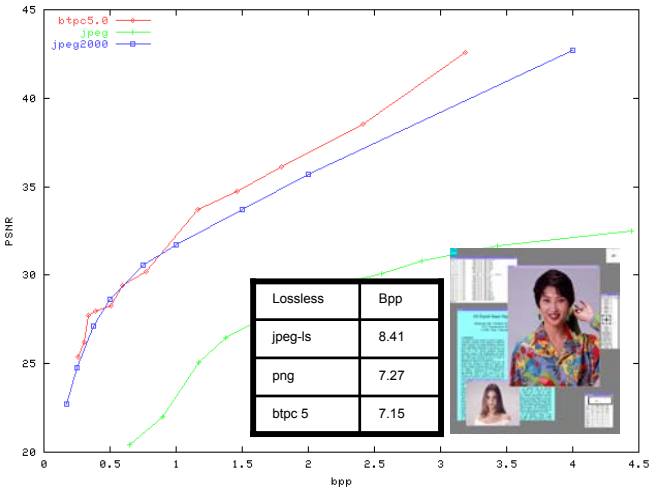

Figure 7: BTPC usually matches JPEG 2000 and outperforms the other alternatives for mixed images.

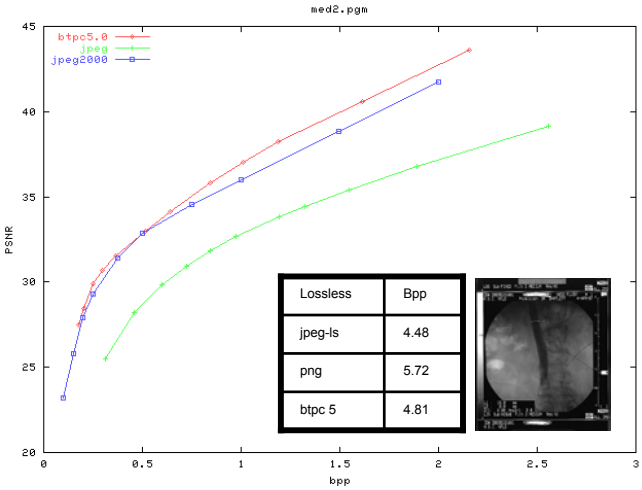

Figure 8. BTPC 5 is competitive with JPEG 2000 and JPEG-LS across a wide range of image modalities.