Progressive Coding of Palette Images and Digital Maps

Forchhammer, Søren; Salinas, J. Martin

Published in:
IEEE Data Compression Conference

Link to article, DOI:
10.1109/DCC.2002.999974

Publication date:
2002

Document Version
Publisher's PDF, also known as Version of record

Progressive Coding of Palette Images and Digital Maps

Søren Forchhammer and Javier Martín Salinas
Research Center COM, 371, Technical University of Denmark
e-mail: sf@tele.dtu.dk, jmelenas@mixmail.com

Abstract

A 2-D version of PPM (Prediction by Partial Matching) coding is introduced simply by combining a 2-D template with the standard PPM coding scheme. A simple scheme for resolution reduction is given and the 2-D PPM scheme extended to resolution progressive coding by placing pixels in a lower resolution image layer. The resolution is increased by a factor of 2 in each step. The 2-D PPM coding is applied to palette images and street maps. The sequential results are comparable to PWC. The PPM results are a little better for the palette images with few colors (up to 4-5 bpp) and a little worse for the images with more colors. For street maps the 2-D PPM is slightly better. The PPM based resolution progressive coding provides a better result than coding the resolution layers as individual images. Compared to GIF the resolution progressive 2-D PPM the coding efficiency is significantly better. An example of combined content-layer/spatial progressive coding is also given.

1 Introduction

Widely employed image coders as JPEG and GIF are not very effective for palette images as recognized eg. in [1]. The reason being that these coders do not use a model matching the characteristics of palette images. In this paper we examine other techniques for coding palette, color-mapped and graphics images. These images are characterized by having a limited number of bits per pixel and that the pixel values are indices to a color. We shall use the term palette images. We are especially interested in street maps which is one example within the class of palette images. One goal is to devise efficient (resolution) progressive coding techniques. Progressive coding may be of benefit both when browsing over connections having a relatively low bandwidth as well as in heterogenous networks where a variaty of combinations of bandwidth and display size are available. Mobile communication is one example. Progressive coding of palette images were presented in [2], [3], performing progression over bit-planes (amplitude) in two different ways and in [4] where progression over content-layers was presented. GIF offers resolution progression in one dimension over the lines of the image. JBIG [5] provides resolution reduction by powers of 2 for

Proceedings of the DATA COMPRESSION CONFERENCE (DCC'02)
1068-0314/02 $17.00 © 2002 IEEE

bi-level images. We shall generalize the JBIG resolution reduction scheme to provide resolution progression for multi-level palette images. A staged scheme using contexts of different size was applied to bi-level images in [6]. In Section II we extend the popular PPM coding technique to 2-D simply by using a 2-D template. In Section III, the 2-D PPM is further extended to progressive coding by also placing template pixels in a low-resolution image. A simple resolution reduction scheme is also introduced. Results on the PWC corpus [1] are given in Sections II and III. Results for street maps are given in Section IV as well an example of combined content-layer/spatial progressive coding.

2 PPM using a 2-D template

PPM [7],[8] is a very effective coding scheme for 1-D data, eg. text files. PPM defines a context based on previous symbols. In the basic versions as [7],[8] it has a fixed maximum context length. If necessary the context length is subsequently reduced by one in each step until the symbol to be coded has been seen in the context considered. The reduction of context length for a context which has been seen before is coded by an 'escape' symbol. Probability estimates are updated for the contexts based on occurrence counts and based on these estimates, arithmetic coding is performed. We generalize this scheme to 2-D simply by defining the context pixels by a 2-D template along with an ordering of the pixels within the template. This generalization is at the expense of not being able to use the same efficient trie-structure implementation as in 1-D, but without changing the coding principle in any other way than the choice of context pixels. Much work has been carried out on the PPM scheme. A recent introduction to PPM is given in [9]. We shall just briefly outline the experiments we have conducted leading to a choice of a few efficient schemes for 2-D. Initial experiments were conducted trying out some of the variations in 2-D. The results of four sequential versions tested on the PWC palette image corpus is given in Table 1. Our basic 2-D PPM applies the selective updates of [8] only updating statistics until the symbol is coded as well as the exclusion principle introduced in [7], excluding colors present in longer contexts when escaping to the shorter contexts. We tried the type A [7], B [7], C [8] and D [10] estimators for escape probabilities (Table 2). Type D gave the best result overall whereas type A gave the best result on the images with few colors ($N \leq 10$) as well as for the maps we tested (Table 3). As we focus on street maps, the type A estimator is used throughout unless otherwise stated. In the type A probability estimator, escapes are given a count of 1 and otherwise treated as a symbol. The probability estimate for a symbol seen in the given context is $p_i = n(i)/(N_s + 1)$, where $n(i)$ is the number of occurrences of symbol i and N_s is the total number of occurrences in the context. The escape probability is $p_{esc} = 1/(N_s + 1)$. The type D estimator is given by $p_i = (n(i) - 1/2)/N_s$ and $p_{esc} = m_s/(2N_s)$, where m_s is the number of seen colors in the context so far. For the basic PPM version the template size and thereby the fixed maximum context length, M is derived based on the number of colors N in the image. One of three template sizes is chosen: $M = 6$ for 1 bpp (bits per pixel) images, $M = 4$ for (2-)4 bbp and $M = 2$ for (5-)8 bpp. The ordering

Proceedings of the DATA COMPRESSION CONFERENCE (DCC'02)
1068-0314/02 $17.00 © 2002 IEEE

is chosen by distance (2-norm) as specified in [11]. We also implemented a PPMstar variation (Table 1 and 2) which initially examines a longer context (the length is specified by a parameter, M^*) and otherwise codes using the PPM procedure given above. (This PPMstar version does not consider whether the contexts are deterministic.) If the long context has not appeared before or an escape is coded in this context i.e. the symbol had not been seen in the this context before, the normal PPM procedure is initiated starting at a context of shorter length ($M < M^* - 1$). The concept of deterministic contexts in PPM*, inspiring PPMstar, was also tested on map images but the results were not as good as the PPMstar results. Including ideas of PPMZ may improve the results, but this has not been tested yet.

The results on the PWC corpus [1] in Table 1 show that a comparison of PWC(-SI) and 2-D PPM by code length is highly correlated with the number of colors, N in the image. PWC-SI is the best PWC version reported in [1]. PWC-SI uses 'Skip-Innovation' to efficiently code pixels in uniform areas. The SI technique could be combined with other coding schemes as PPM. The 2-D PPM coding provides the best results up to 4-5 bpp (with the exception of the small 'music' image.) The total code length for the images of the PWC corpus (Table 1 and 2) up to 5 bpp ($N \leq 32$) is 40.345 and 47.796 bytes for the basic PPM (type A escape) and PWC, respectively. In the range 6-8 bpp, PWC provides the best results. The figures for images with $N \geq 46$ are 178.390 and 166.909 bytes for basic PPM (type D esc.) and PWC, respectively. The basic PPM versions were, for the chosen 2-norm ordering, improved by optimizing the context length. This only reduced the overall code length for the PWC corpus compared to the basic versions (both type A and D esc) by about 700 bytes. The PPMstar version provides a 3 % improvement compared with the basic PPM (type A esc) on the PWC corpus (Table 1), but only 1 % on the street maps (Table 3).

For the street map images in Table 3, PPM provides the best results for each of the four maps. Three of these require 7 bpp (N in the range 75-79) in raw form. PWC applies (JPEG-LS) prediction in the last coding step (for $N > 16$, [1]) if a match has not been found before. This is probably one of the reasons for the better performance of PWC as 8 bpp is approached in Table 1.

In [1] a comparison on the PWC corpus is made with GIF, png, bzip2, BW-MTF, CALIC (with ordering), EIDAC and RAPP. Both PWC and the sequential PPM versions outperform all of these on the PWC corpus.

3 Resolution progressive coding based on PPM

As in JBIG [5], we consider resolution reduction by a factor of 2 in each dimension in each step, which applied recursively provides resolution reduction by powers of 2. An example of such a resolution reduction scheme is given in the following subsection. The progressive coding consists of coding a high(er) resolution layer using information from the low(er) resolution layer having half the resolution. For bi-level images, JBIG [5] did this by placing template pixels in both the low resolution layer (4 pixels) and the causal part of the high resolution layer (6 pixels). Besides the 10 image pixels the

Image	N	PWC-SI [1]	PPM esc A	PPM esc D	PPMstar esc A	PPMstar esc D	PPM prog	PPM layers	PPM prog, $rl0$
ccitt01	2	12,683	6,845	6,856	**6,754**	6,763	7,733	8,672	5,900
pattern	2	1,099	**1,080**	1,081	**1,080**	1,081	1,175	1,176	1,093
flax	3	142	62	58	60	**56**	171	171	60
stone	3	3,637	2,505	2,522	**2,349**	2,522	3,125	3,510	2,224
books	7	8,153	7,832	7,974	**7,681**	7,816	8,847	9,810	6,899
music	8	**696**	848	863	713	701	1,098	1,127	714
winaw	10	10,853	10,639	10,780	**10,495**	10,620	12,720	14,372	9,173
netscape	32	10,533	10,534	10,435	10,272	**10,205**	12,643	13,952	9,405
sea dusk	46	**678**	761	722	746	722	1,257	1,260	761
benjerry	48	**2,399**	2,663	2,647	2,486	2,448	3,216	3,476	2,342
gate	84	15,282	15,969	**15,111**	15,699	**15,111**	19,542	20,906	14,683
sunset	204	**51,623**	58,361	56,603	56,710	55,490	74,162	90,553	45,643
cmpndn	223	**53,021**	60,858	56,370	57,776	55,552	71,824	77,344	57,344
yahoo	229	**4,350**	4,860	4,633	4,749	4,544	6,380	6,768	4,555
cmpndu	246	**39,556**	45,266	42,304	44,368	42,199	55,813	60,337	42,140
Total		**214,705**	229,083	218,959	221,938	215,830	279,706	313,434	202,936

Table 1: PWC and 2-D PPM coding of the PWC Palette Image Corpus sorted by the number of colors, N. The first four PPM columns are variations of sequential coding. The last three PPM columns are related to progressive coding.

No. of colors N	PWC	PPM esc A	PPM esc B	PPM esc C	PPM esc D	PPMstar esc A	PPMstar esc D
2-32	47,796	40,345	40,658	40,620	40,569	39,404	39,764
46-246	166,909	188,738	181,164	178,685	178,390	182,534	176,066
Total	214,705	229,083	221,822	219,305	218,959	221,938	215,830

Table 2: PWC and 2-D PPM coding of the PWC Palette Image Corpus accumulated over 2-5 bpp ($2 \leq N \leq 32$) and 6-8 bpp ($46 \leq N \leq 246$).

phase value taking one of four values was included in the context. The phase value specifies the position of the current pixel in the high resolution image relative to the low-resolution pixels. The same approach to resolution progression is applied here for multi-level images based on the 2-D PPM of the previous section (Fig. 1). We tried to include the phase value in the context or to disregard it. The latter option gave the best result so defining the contexts without phase information was chosen.
Resolution progressive coding by placing context pixels in a low-resolution layer was also combined with an extension of the RAPP coder [12] (see later). It could also be combined with PWC [1], especially if speed is of concern. Combining it with PWC is less straightforward though, as several contexts are used in a series of steps (when necessary).

3.1 Resolution reduction

The resolution reduction scheme developed is presented here. The 2 by 2 pixels of the high resolution layer at the position of the low resolution layer are examined first (Fig. 1). If a color is dominating, i.e. it has more occurrences than any of the other colors

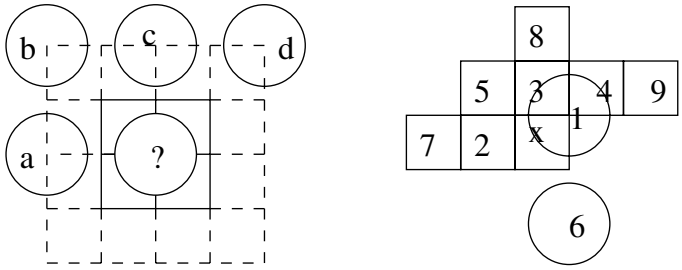

Figure 1: Templates for resolution reduction (left) and progressive coding (right). Low resolution pixels are marked with ◯ and high resolution pixels are marked with □. The numbers indicate the pixel ordering.

present in the 2 by 2 square, then the low-resolution pixel is assigned this color. In case of a tie between two colors the four high resolution pixels are examined to decide whether there is a horizontal edge, vertical edge or diagonal cross in accordance with the connectivity of the pixels of the same color. The four causal pixels (Fig. 1) of the low-resolution image are weighted as follows: a horizontal edge, $(a, b, c, d) = (3, 0, 2, 0)$, a vertical edge, $(a, b, c, d) = (2, 0, 3, 0)$, and a diagonal cross, $(a, b, c, d) = (1, 3, 1, 3)$. In all these cases each of the surrounding 16 nearest pixels of the high-resolution is given a weight of 1. The color with the highest accumulated weight is chosen. The weights above emphasize the causal low-resolution pixel(s) in the direction of the edge. In the case of the four high-resolution pixels having four different colors, the weights of the causal low-resolution pixels (Fig. 1) are given by $(a, b, c, d) = (2, 2, 2, 2)$. For this case only the 4 center and the 4 diagonal out of the 16 high resolution pixels are considered, having a weight of 1. Again the color with the highest accumulated weight is chosen. In the very rare cases where there, still after considering the low resolution pixels and surrounding high resolution pixels, is no dominating color, the first occurring of the remaining candidate colors having the highest accumulated weight is chosen. This resolution reduction scheme was applied twice to yield three resolution layers in all for the images of the PWC corpus. These resolution layers were coded both using the progressive PPM and by applying PPM sequentially to each resolution layer (Table 1). Both variations produced a significantly shorter code length than the 376.208 bytes which GIF requires [1].

The resolution layers (rl) are numbered by the number of resolution steps i.e $rl0$ is the full image, $rl1$ has half the resolution etc. For the street maps this simple resolution reduction scheme gives good results. The main problem being the text (Fig. 2). A way to improve the performance on text is given below.

The resolution scheme above was designed primarily to provide visually acceptable reduced images. Besides that some preference is given to continuity in the colors of the reduced image, the scheme was not especially designed for good progressive coding performance.

Proceedings of the DATA COMPRESSION CONFERENCE (DCC'02)
1068-0314/02 $17.00 © 2002 IEEE

Figure 2: Progressive coding. Test image of map (city center of Copenhagen). Upper left) Resolution layer (rl) 1 (8262 bytes to this point). Upper right) Background rl 2 and text sub-layer (sl) 1+2 of rl 1 (3305 bytes to this point). Lower left) Background rl 1 and text sl 1+2 of rl 0 (8281 bytes to this point). Lower right Background rl 1 and full text rl 0 (15483 bytes to this point).

3.2 Content-layer/spatial progression

In [4] a content-layer progressive scheme for street maps was presented. One setting is that the content layers are given beforehand along with a composition rule defining the final composite image, i.e. the palette image. Street maps are generated in layers so the layers are easily extracted in the process of generating the maps. Another setting is that the layers have to be extracted from a composite image. For text written on (top of) street maps this is easily done. The focus in [4] was on the binary layers the street map was composed of. Here we propose to combine the content-layer progression with spatial progression. The scheme is confined to a combination of JBIG coding of bi-level text layers and 2-D PPM coding of the background. JBIG [5] provides resolution progressive coding for bi-level images and suggests a resolution reduction scheme suited for text. Text has the characteristic that it is not very useful unless you can read it. Separating the text and the background gives the option of displaying the text and background in different resolutions. Segmenting the text eg. by size into sub-layers (sl) gives the option of selectively deferring the reduced text

(sub-layers) in the progression until you can actually read it. This is illustrated in Fig. 2.

3.3 Progressive skip coding.

In content layered coding the different content layers define pixels within the image. If the (final) value of a given pixel is already defined by a previous content layer it is not necessary to code it in the current layer. In [4] skip coding was introduced, simply skipping the pixels already coded with their (final) value. When content layers are extracted by color from a composite image each pixel is defined in exactly one layer, thus already coded pixels may be skipped. This skipping technique may also be applied in content-layer/spatial progressive coding. We implemented resolution progressive skip coding with a modification of the RAPP coding scheme [12]. The RAPP principle is to label the colors within each template, labelling the first color 'a', the next 'b' and so on, thereby reducing the number of different contexts. In [12] a four pixel template was used. In [4] the pixel pattern technique was extended to larger templates. Here an extra extension is made. The coding template has a given size M specifying the pixels which define the pattern. A larger (superset) template of size $M^* > M$ defines the candidate colors which are considered in the first coding step. We call this variant for MAPP (Modified Adaptive Pixel Pattern) coding. Placing pixels in a low-resolution layer gives a progressive coding. In both the sequential and progressive version the skip pixel technique may be applied. (This skip pixel technique could also be combined with PPM or PWC.)

4 Results on digital maps

This section presents results on the sub-class of palette images composed of street maps. In Table 3 results for PPM coding (also using type D escape coding) of four street maps are given and compared to PWC. The Copenhagen map is 723 by 546 pixels [4]. The three other images are japanese street maps which are 560 by 560 [12]. For comparison, the Copenhagen map was coded using PPM* and the sequential MAPP yielding code lengths of 19,247 and 20,322 bytes, respectively. The 2-D PPM achieves a shorter code length than PWC for all these four images. For the total of the four images the reduction is 5 %. Resolution reduction and progressive coding were also applied to these images. The results are given in Table 4. Progressive coding in 3 layers gave a 32-36% longer code length overall compared to a single coding of the final image. Coding the resolution layers independently gave 12-13 % longer code length than the progressive coding. As the figures show, unfortunately, the progressive coding does not come for free. On the other hand, it is of some benefit when the progression is desired.

Proceedings of the DATA COMPRESSION CONFERENCE (DCC'02)
1068-0314/02 $17.00 © 2002 IEEE

Image	N	PWC	PPM esc A, opt M	PPM esc D, opt M	PPMstar esc A
Copenhagen	12	20,600	18,991	19,603	18,872
2b09cf-	79	18,096	17,253	17,331	17,087
2b09cq-	75	18,126	17,919	17,967	17,637
2b0avt-	78	13,083	12,547	12,609	12,460
Total		69,905	66,702	67,510	66,056

Table 3: PWC and 2-D PPM coding of street maps.

layer	Cph seq.	Cph prog.	2b0xyz- seq.	2b0xyz- prog.
2	2,640	2,640	6,397	6,397
1	6,565	5,622	18,533	16,371
0	18,872	16,678	47,719	42,025
Total	28,077	24,940	72,649	64,793

Table 4: 2-D PPM resolution progressive (prog.) coding of the Copenhagen (Cph) map the three maps from [12] (2b0xyz-). The results for sequential (seq.) of each of the three resolution layers are also given.

4.1 Content-layer/spatial progression

As mentioned, text which is not readable due to resolution reduction is not very informative. Therefore the text of the Copenhagen map is reduced and coded seperately using JBIG progressive coding (Table 5). For this image the text is written on top of the background so it is readily extracted even from the composite image. The standard JBIG coding is also applied to the text. Not surprisingly, JBIG outperforms the PPM coding on the (bi-level) text. A larger template as used in JBIG2 [13] may provide even better results. We stick to the original JBIG as JBIG2 does not define resolution progression. The text was also divided according to text size into three sub-layers (Fig. 2) and two different versions of sub-layer combinations and resolution reduction was tried out (Table 5). The text layer progression (Table 5) was combined with resolution reduction and progressive PPM coding of the background layer (Table 6). Some of the intermediate images are shown in Fig. 2. The division in text and background layers both gives improved coding performance, more steps and increased flexibility in the progression as well as a better visual result. Applying resolution progressive coding seperately to the text and background using JBIG and PPM, respectivly, brings the progression overhead down to 15 % compared to the code length of sequential PPM (Table 6). As for a sequential coding of the layers of the image [4], it is a bit more effective to code the original layers, if they are available, than coding the background when extracted from the composite image. Progressive MAPP with skip coding was also applied to the extracted background layers. Using this skip coding, skipping pixels which are defined in the text layer, the difference becomes very small though. The skipping can of course only be done for pixels already received in the progression. But this goes well with our recommendation to prioritize text layers in the progression.

Proceedings of the DATA COMPRESSION CONFERENCE (DCC'02)
1068-0314/02 $17.00 © 2002 IEEE

Resolution layer	JBIG seq	JBIG pr	PPM seq	PPM pr	JBIG pr	JBIG $sl1{+}2$	JBIG $sl1$
2	1,150	1,150	1,414	1,414			
1	4,062	3,102	4,155	3,469	4,096	1,132	409
0	9,543	6,924	9,824	8,323	6,924	1,541	520
0 (sub-layer 2)							1,513
0 (sub-layer 3)						7,202	7,202
Total	14,755	11,176	15,393	13,206	10,986	9,875	9,644

Table 5: Resolution progressive (pr) coding of text layers and sublayers (sl). The total for coding the full text of the Copenhagen map for each combination is given at the bottom.

Resolution layer	extracted	layered	layered sub-text	extracted skip
2	2,158	2,173	2,173	2,158
1	3,549	3,435	3,435	3,510
0	7,418	6,515	6,515	6,667
Background	13,125	12,123	12,123	12,335
Text	10,986	10,986	9,644	9,644
Total	24,940	23,109	21,767	21,979

Table 6: Resolution and content progressive coding of the Copenhagen map. The total for coding the whole Copenhagen map for each combination is given at the bottom. The three first columns are combined progressive PPM and JBIG. In the last column progressive MAPP with skipping is applied instead of the progressive PPM.

5 Conclusions

PPM coding was applied to images by using a 2-D template and ordering the pixels within the template. Tested on the PWC corpus, the PPM versions yielded a code length 0.5-7 % longer than PWC. For the images up to 4-5 bpp PPM generally yielded shorther code lengths. For this subset of the PWC corpus the basic PPM code length was only 84 % of the PWC code length. For images with 6-8 bpp the PWC produced the best results. On four street maps with 4-7 bpp the PPM code length was 5 % shorter. A resolution progressive coding of image data was introduced by also placing context pixels in a low resolution image. A simple scheme for resolution reduction by a factor of 2 was also presented. Progression with 3 layers in all required a 22 % longer code length than a single sequential coding, but with an 11 % shorter code length than sequential coding of each of the three layers. The latter figure was 12 % measured on the street maps. Resolution progression may be combined with other types of progression. For the street maps resolution progression was combined with content layer progression. On the sample street map, for a progression in 7 steps the code length was only 16 % longer than that of a single sequential coding. The example also showed the benefit of having progression with text and background in different resolutions both in terms of efficiency and visual quality in the progression. The resolution reduction and progressive coding could be combined with other context based coding schemes, eg. the PWC to obtain a higher speed, which has not been an objective of this study.

Proceedings of the DATA COMPRESSION CONFERENCE (DCC'02)
1068-0314/02 $17.00 © 2002 IEEE

6 Acknowledgement

We would like to thank Jakob D. Andersen for programming assistance in this research.

References

[1] P. J. Ausbeck Jr., "The Piecewice-Constant Image Model", *Proceedings IEEE*, vol. 88, no. 11, Nov. 2000, pp. 1779-1789.

[2] Y. Yoo, Y. Kwon, and A. Ortega, "Embedded Image-Domain Adaptive Compression of Simple Images", *Thirty-Second Asilomar Conference on Signals, Systems and Computers*, November 1998.

[3] U. Rauschenbach, "Comparison of palettized images with progressive coding of color information", *Proc. of SPIE VCIP*, vol. 4067, 2000, pp. 586-597.

[4] S. Forchhammer and O. R. Jensen, "Content Layer Progressive Coding of Digital Maps", *Proc. Data Compression Conference*, Mar. 2000, pp. 233-242.

[5] ISO/IEC Int'l Standard 11544, "Coded Representension of Picture and Audio Information - Progressive bi-level image compression", 1993.

[6] A. Moffat, "Two-level context based compression of binary images", *Proc. Data Compression Conference*, 1991, pp. 382-391.

[7] J. Cleary and I. Witten, "Data Compression using adaptive coding and partial string matching", *IEEE Trans. Communications*, vol. 32, April 1984, pp. 396-402.

[8] A. Moffat, "Implementing the PPM Data Compression Scheme", *IEEE Trans. Communications*, vol. 38, Nov. 1990, pp. 1917-1921.

[9] K. Sayood, *Introduction to Data Compression*, 2. ed. Morgan-Kaufmann. San Francisco, 2000.

[10] P. G. Howard, *The Design and Analysis of Efficient Lossless Data Compression Systems*, Report CS-93-28, Dept. Comp. Sci., Brown University, Providence, RI 1993.

[11] B. Martins and S. Forchhammer, "Tree Coding of Bilevel Images", *IEEE Trans. Image Processing*, vol. 7, no. 4, April 1998, pp. 517-528.

[12] Viresh Ratnakar, "RAPP: Lossless Image Compression with Runs of Adaptive Pixel Patterns", *Thirty-Second Asilomar Conference on Signals, Systems and Computers*, November 1998.

[13] P. G. Howard, F. Kossentini, B. Martins, S. Forchhammer, and W. J. Rucklidge, "The Emerging JBIG2 Standard", *IEEE Trans. Circuits and Systems for Video Tech.*, vol. 8, no. 7, November 1998, pp 838-848.