Content Layer progressive Coding of Digital Maps

Forchhammer, Søren; Jensen, Ole Riis

Published in:
I E E E Transactions on Image Processing

Link to article, DOI:
10.1109/TIP.2002.806236

Publication date:
2002

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

Content Layer Progressive Coding of Digital Maps

Søren Forchhammer and Ole Riis Jensen

***Abstract*—A new lossless context based method is presented for content progressive coding of limited bits/pixel images, such as maps, company logos, etc., common on the World Wide Web. Progressive encoding is achieved by encoding the image in content layers based on color level or other predefined information. Information from already coded layers are used when coding subsequent layers. This approach is combined with efficient template based context bilevel coding, context collapsing methods for multilevel images and arithmetic coding. Relative pixel patterns are used to collapse contexts. Expressions for calculating the resulting number of contexts are given. The new methods outperform existing schemes coding digital maps and in addition provide progressive coding. Compared to the state-of-the-art PWC coder, the compressed size is reduced to 50–70% on our layered map test images.**

***Index Terms*—Contexts, digital maps, graphics, image coding, image communication, progressive coding.**

I. Introduction

In this paper, a lossless coding method is introduced for progressive or layered coding of images with a limited number of distinct color or grey scale values (referred to as levels). Examples of such images are typically seen on web pages on the Internet depicting icons, company logos, maps or palletized images. We have chosen to focus on coding of maps which is an important application area. Street maps appear among other places on Web sites as the Yellow Pages and for tourist bureaus. Images of this type are often limited to relatively few colors, usually computer generated and often coded with lossless GIF or lossy JPEG.

For internet and other interactive applications efficiency and progressive coding are important issues. Maps are often presented in response to a query, which means certain parts and certain symbols of the map will be more important than others. Therefore progression by contents becomes interesting. In addition to this the resolution of monitors is limited while it is desirable to have the option of a highly detailed high resolution version for printing.

A number of efficient lossless image coding schemes are based on context coding for bilevel images (JBIG [1], [2]) and for grey scale images [3], [4]. In context based coding, arithmetic coding is performed for each pixel based on probabilities conditioned on a context derived from the causal data. In bilevel image coding the context is efficiently specified by a template. Aside from bilevel images the increase in the number of contexts with increasing template size is a problem in practice.

To reduce the number of contexts, the context may be obtained by a mapping of the template pixels. For natural images, prediction and differencing followed by quantization is an efficient way to reduce the number of contexts [3], but linear prediction is not suited for graphic data such as maps. Recently, methods aimed at these limited bits/pixel images have been presented. PWC [5] and RAPP [6] both code whether the current pixel is equal to one of the causal 4-neighbors. PWC uses edge maps and other techniques to reduce contexts. RAPP uses relative pixel patterns within a four pixel template. JBIG may be applied to the bit-planes of the image. It is recommended to Gray code the pixel values before the bit-plane coding [1]. EIDAC [7] uses a bit-plane representation and selects bits from the current and previous bit-planes. It provides progression by bit-planes. In Table I, results are given for some of the coding methods for the map of Fig. 1. PWC achieves the best coding result among these prior art methods. Our content layer progressive coding scheme revisits some of these existing techniques extending them especially by ways of efficiently using cross layer information.

Images are usually represented as *composite images* in a *single* layer of pixels, each represented by a number of bits needed to describe the color or grey tone level of the pixel. Images such as maps often originate from data which are generated as separate (bilevel) layers such as text, roads, buildings, etc., each of which may overlap other layers. A *layered image* consists of a number of layers each representing one level. The layers may be combined by some composition rule to form a composite image.

We consider the range of coding problems that arise when the input may be either a composite or a layered image and the desired representation after decoding may be ordinary sequential or progressive by content layer. The aim is to present efficient and flexible coding tools to accomodate a range of applications rather than to devise specific progressions. We address the problem of efficiently coding layered images facilitating progression by layers in a flexible manner. This includes issues of random access of layers and the number of coding passes in the progressive coding.

In Section II, the layered image representation is formally introduced. Methods for coding bilevel and multilevel layers are introduced in Section III, as the components of the new progressive schemes. In Section IV, we analyze the number of contexts when relative pixel patterns are used. Section V presents results for coding digital street maps.

II. Layered Images

The layers of a layered image may be combined to form a composite image, e.g., by stacking the layers in a predefined

Manuscript received September 18, 2000; revised September 6, 2002. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Lina Karam.

The authors are with the Research Center COM, 371, Technical University of Denmark DK-2800 Lyngby, Denmark (e-mail: sf@com.dtu.dk; Ole.Riis.Jensen@dnk.xerox.com).

Digital Object Identifier 10.1109/TIP.2002.806236

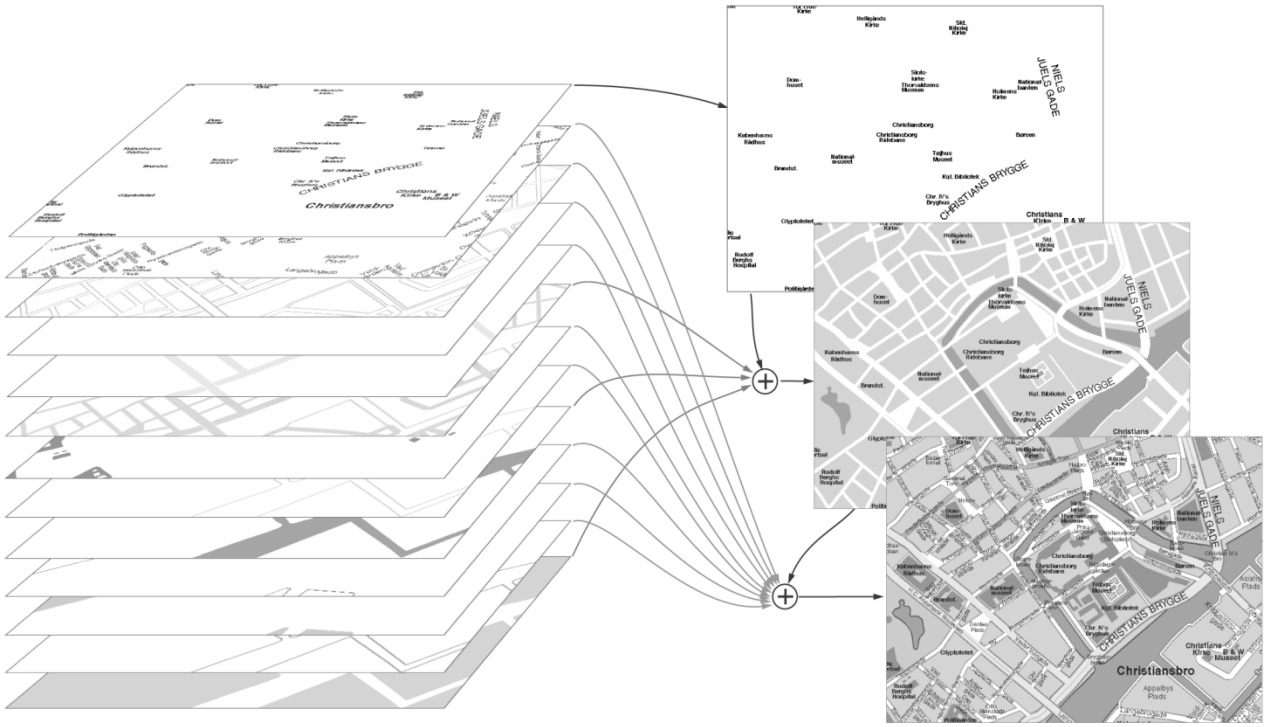

Fig. 1. Copenhagen map (city center). Left, the layered image. Right, progressive images leading to the composite image. Courtesy of Geo Vision and Tele Danmark.

order, such that image data of layers with higher priority are opaque with respect to image data of lower priority layers. In this case, a composite image is obtained from a layered image by merging layers from the top layer to the bottom layer into a multilevel image with the same visual appearance (Fig. 1). These concepts are introduced formally below.

Let y_t denote a pixel of a composite image y^T with T pixels. Each pixel y_t takes on one of N values, $\{1,\dots,N\}$, specifying the distinct color level. Let $\mathbf{x}_t$ denote a vector of binary pixels, $\mathbf{x}_t = (x_t(1),\dots,x_t(N))$, $x_t(i) \in \{0,1\}$, defining the layered image $\mathbf{x}^T$. (In this definition all pixels are assumed to be specified in at least one layer. Layer N may be a background layer or a layer in its own (Fig. 2, right) containing all pixels not specified in any other layer as a subset.) A composition rule specifies the (many-to-one) mapping from $\mathbf{x}^T$ to y^T. The example of mapping by priority given at the beginning of this section may be expressed by

$$y_t = l_0, \quad l_0 = \min_l\{l | x_t(l) \neq 0\}. \tag{1}$$

A composite image with N distinct color levels may on the other hand be split into a layered image by separating it into N non overlapping bilevel images each representing a distinct level from the composite image, i.e.,

$$y_t = i \Rightarrow \begin{cases} x_t(j) = 1, & j = i \\ x_t(j) = 0, & j \neq i \end{cases} \tag{2}$$

where $x_t(j)$ refers to the pixels of split layer j obtained from the composite image (Fig. 2).

A *hybrid image representation* is introduced as a combination (or generalization) of the layered and the composite image. The hybrid image is represented by L image coding layers. Let $\mathbf{y}_t$ denote a vector pixel of the layered hybrid image $\mathbf{y}^T$. $\mathbf{y}_t = (y_t(1),\dots,y_t(L))$, where the value of each $y_t(i)$ is taken from some subset of the N color levels. Image coding layer k has n_k distinct levels. Therefore, $\sum_{k=1}^{k=L} n_k = N$. Besides the last layer (L) all layers have a "to-be-coded" value in addition to the n_k levels. We shall restrict the compositions we consider to be by priority (1). Having multiple coding layers the default progression after k layers is defined by composing these k layers according to the composition rule. The layered images, $\mathbf{x}^T$, and the hybrid image representations, $\mathbf{y}^T$, we consider shall map to the same composite image, y^T. Thus, the progressions considered shall always end up with a unique composite image.

III. Context Coding of Layered Images

We shall code the pixels of the composite image, y^T, the binary layers, $x^T(i)$, and multilevel hybrid layers, $y^T(i)$, applying arithmetic coding based on conditional probabilities. For the binary pixels, $x_t = x_t(i)$, arithmetic coding may be performed for each pixel, x_t, based on probabilities $p(x_t|c)$ conditioned on the context c derived from the causal data $x^{t-1} = \{x_1,\dots,x_{t-1}\}$. In addition to the causal data of the current layer also pixels of layers already sent are causal and may therefore be included in the context, i.e., pixels of $x^T(j)$ may be used in coding $x_t(i)$, if layer j is available at the decoder when decoding layer i.

In hybrid image coding the use of binary layers may be combined with multilevel layers. One example is obtained by coding $n_b < N$ bilevel image layers, together with one or more *residual (multilevel) image(s)* of the $N - n_b$ remaining levels. Residual layer i represents n_i levels so that each level is coded in exactly one coding layer.

Fig. 2. Copenhagen map, layer 11 (land). Left, from composite image. Right, from layered image.

By encoding each of the bilevel and residual images separately one by one (all in the usual raster-scan order), we obtain a *layered* or *progressive coding method*. As the layers are associated with specific contents of the image the coding is *content progressive*.

A. Inter-Layer Context Mappings

Efficient coding of a layered image is in our scheme achieved by using the inter-layer dependence by sharing information across the layers. One approach is to choose template pixels from already coded layers (e.g., as was done for bit-planes in [7]), thus coding a layer relative to one or more other layers. We use this approach for bilevel layers using one prior bilevel layer as reference. We also introduce another approach, which we call SKIP pixel coding. It may be used with split layers (2) or layers with priority composition. In a particular layer, if a given pixel has already been coded in another layer of higher priority it does not need to be coded in the current or any of the lower layers when the full layered information is not required. Such pixels are denoted *skip pixels* and they are skipped by the encoder in our hybrid coding scheme. Skipped pixels will appear in the context of subsequent pixels. As the value is known at the decoder, we may readily use it. To reduce the number of contexts we may instead apply one of two context mappings to the skip pixel. The skip pixels may be labeled by a distinct 'skip value' (s) in the subsequent contexts, thus quantizing the context pixel values of all higher layers into a single *skip value* in the current layer (i):

$$y_t < i \Rightarrow c(y_t) = s \tag{3}$$

where $c(y_t)$ denotes the value which y_t is mapped to as a context pixel. The context alphabet becomes one larger than the alphabet of the pixels being coded in the current layer (including the "to-be-coded" value if this is not the final layer). We use this mapping for multilevel SKIP coding. This increase of alphabet size may be avoided by alternatively labeling skip pixels in the context as one of the colors in the current coding layer, e.g., the "to-be-coded" value or for the last layer the background color, N:

$$y_t < i \Rightarrow c(y_t) = N. \tag{4}$$

The 'to-be-coded' pixels can be displayed as the background color until the last residual. The skip pixel approach is combined with both bilevel and multilevel coding.

Skip pixel coding is especially of benefit when coding split layers, as we avoid coding (and confusing the statistics with) the 'holes' left by levels of higher priority (Fig. 2).

B. Coding Bilevel Layers

Our default choice, for coding the bilevel layers independently of other layers, is a 16 pixel template with up to four adaptive template pixels. This is the largest template in JBIG2 [2], [8]. We shall also be using free tree coding [9]. (In free tree coding, the context is specified by a variable number of pixels taken from a large causal region. The second context pixel chosen depends on the the value of the first, the third on the values of the two first etc.) To utilize dependencies between two bilevel layers a template with nine pixels from a previously coded bilevel layer and four from the current layer is used. This template is the same as for JBIG2 refinement coding, but here used to code a new layer instead of refining the old. We refer to this as template coding with a reference layer. To utilize dependencies from multiple levels of higher priority, skip pixel coding is applied. A ten-pixel template is used for skip pixel coding (3) having a ternary context alphabet. This is denoted SKIPt. Context mapping by labeling skip pixels as background (4) maintains a bilevel context alphabet and therefore a 16 pixel template is used. This is the default bilevel SKIP setting.

C. Coding Multilevel Layers

For coding of multilevel layers, PWC [5] and RAPP [6] may be applied. For each pixel PWC applies arithmetic coding to a sequence of binary questions until the value is determined. Very briefly described, first PWC codes in four steps, whether the current pixel has the same value as one of the four causal neighbors. If not it resorts to a list of good guesses and finally if nec-

essary codes the actual value. In RAPP (Runs of Adaptive Pixel Patterns) contexts are quantized, substituting the pixel values within the template, specified by the 4 causal 8-neighbors, by labels A and up to D assigning a new label to each new color in the context. We shall apply the straightforward generalization for templates larger than four pixels introducing labels E, F, etc. if necessary. We refer to this as template relative pixel patterns (TRPP) and choose a default template size of 9. (In TRPP we do not use the runs which may be applied in RAPP.) As in RAPP coding, the first step is to code if the pixel has the value of one of the context pixels, conditioned on the relative pixel pattern context. If not, an escape character is coded and then in the second step the pixel value. We use the same context in both steps. Splitting the coding in two steps is advantageous for images where the colors are spatially clustered. The first step is sufficient most of the time and coding is performed on a reduced alphabet improving both time and compression performance.

For multilevel residual coding, we introduce SKIP pixels in TRPP. One distinct value is assigned to skip pixels when appearing in a context. We also modify the coding and statistics update, as there is no need to assign any probability to coding SKIP values even though they may be present in the template. We refer to this as TRPP-SKIP.

For both bilevel and multilevel coding we apply the binary arithmetic coding also used in [9] as the entropy coding. The probabilities are obtained directly from the occurrence counts.

While being presented as a multiple pass algorithm the approach may, with the same final compression result, be implemented in a one-pass (multiple context) algorithm. For each pixel the layered coding is applied until the pixel value is actually coded. Processing the image in this way results in a sequential nonprogressive method.

IV. Number of Contexts for Relative Pixel Patterns

The reason for mapping contexts is to reduce the number of contexts for a given template size. In order to achieve good code lengths the right combination of a large template and context mapping is sought for. This section analyzes the number of contexts using relative pixel patterns as in TRPP. The number of contexts determines the number of parameters which is important to restrict both with respect to code length performance and memory demand.

Having a template of τ pixels, we may encounter up to τ different relative values. Let $C(\tau)$ denote the total number of different contexts for τ template pixels. Let $C_j(\tau)$ denote the number of these $C(\tau)$ context patterns which have exactly j different colors. Let N denote the size of the alphabet. The number of different contexts can recursively be calculated as follows: $C_j(\tau+1)$ is obtained by adding a new color to one of the $C_{j-1}(\tau)$ contexts having $j-1$ different colors or adding one of the j colors already present in each of the $C_j(\tau)$ contexts. For a given N, the recursion is given by

$$C_j(\tau+1) = C_{j-1}(\tau) + jC_j(\tau), \quad 1 < j \leq N, \ j \leq \tau+1 \tag{5}$$

which, consistent with $C_\tau(\tau) = 1$, is initialized by

$$C_1(\tau) = 1, \quad 1 \leq \tau \tag{6}$$

$$C_j(\tau) = 0, \quad j > \tau \vee j > N. \tag{7}$$

Summing $C_j(\tau)$, the total number of contexts for τ template pixels can be calculated by

$$C(\tau) = \sum_{j=1}^{\tau} C_j(\tau). \tag{8}$$

Consider the situation where we code one of the j colors within the template or an escape character if the color was not in the template. In this case, the number of free parameters, $P_j(\tau)$ for the class of contexts specified by τ and j is given by $jC_j(\tau)$. Summing over j gives the total number of free parameters $P(\tau)$ for the τ pixel template (excluding the parameters used to code the value after an escape).

$$P(\tau) = \sum_{j=1}^{\tau} jC_j(\tau). \tag{9}$$

Using the recursion (5) we can calculate the number of contexts. The results are given in Table II. With respect to the number of contexts it is quite feasible to work with 9–10 template pixels independently of the alphabet size.

Relative pixel patterns may be combined with the actual color of some pixels as well as a limitation of the number of colors. As described in Section V, the best result for the composite image of Fig. 1 was achieved combining these three techniques. Based on (5) the number of contexts may also be calculated for these combinations [10]. The idea of imposing a maximum on the number of colors in the template, by truncating the template when the number is exceeded, is a very direct way to keep the number of contexts down while allowing for large templates when only few colors are present locally.

A. Implementation Issues

The recursions and expressions for calculating the number of contexts may also be used to allocate memory and devise fast indexing methods when implementing context mappings using relative pixel patterns. In our implementation of relative pixel patterns, we have stored context statistics sequentially in a one-dimensional array, which is allocated when the codec is initialized based on (5), allocating space for all possible contexts. The indexing of the array is done by a recursive procedure, so that statistical information related to a specific pattern can easily be retrieved and updated. If memory is not available for all the parameters (9) of the possible contexts a slower dynamic structure may be used.

V. Results

Our main interest is efficient progressive coding of street map data. The algorithms were tested on a number of street maps. The first is a map of Copenhagen (Fig. 1). The map has 723 $\times$ 546 pixels and it is represented both as a layered image $\mathbf{x}^T$ with 12 overlapping layers, and as its composite verion y^T, consistent under priority composition (1). Tests were also conducted on four layered digital maps from which a composite street map for printing can be composed. These maps were provided by KRAK, Denmark. They were chosen as representa-

TABLE I
LOSSLESS CODE LENGTHS (BYTES) FOR THE COMPOSITE MAP SHOWN IN FIG. 1

GIF	PNG	JPEG-LS	BIT-PLANE DIRECT	BIT-PLANE GRAY	RAPP (M=5)	PWC (FLIP, COLLAPSE)
49248	43075	66778	33298	30426	26472	20600

TABLE II
NUMBER OF POSSIBLE CONTEXTS $C(\tau)$ AND FREE PARAMETERS $P(\tau)$ FOR TEMPLATE SIZE τ USING RELATIVE PATTERNS ($N \geq \tau$)

τ	1	2	3	4	5	6	7	8	9	10
$C(\tau)$	1	2	5	15	52	203	877	4140	21,147	115,975
$P(\tau)$	1	3	10	37	151	674	3263	17,007	94,828	562,785

tive maps of Copenhagen and suburbs, representing city center (Cph148), suburbs (Cph117), coastal region (Cph038), and outskirts (Cph144).

For content progressive transmission it is decided in which order the layers of the image are presented to the decoder. We shall give a few examples of different progressions to demonstrate the flexibility of our approach. On a map, the layers that first enables a client to identify a location is probably the text and the roads. Fig. 1 depicted a progressive version with (a subset of) the text layer, the roads and the water. To restrict the number of passes over the image, the progression may be over the desired layers followed by coding the residual in one coding layer.

A. Composite Images

Results on the composite versions are presented as a starting point. Table I showed that the PWC [5] and RAPP [6] algorithms outperform prior methods on the composite image. These two algorithms are able to obtain a reduction in code length of 45–55% compared to the GIF file size. Table III compares a number of existing methods for the KRAK images. The tendency is even more pronounced here, e.g., PWC compresses these four images 3.8 times better than GIF.

Table IV shows the benefit of increasing the template size for relative pixel patterns for the Copenhagen map as well as the four KRAK maps. The code lengths obtained using the chosen nine-pixel templates are reduced to 75–80% of the result for the 4 pixel template. For the four KRAK images it is 3.7 times better than GIF.

The best one-pass result for the composite Copenhagen map was achieved by a 7 pixel template combining relative pixel patterns with actual pixel values for the four closest template pixels and imposing a maximum of two different colors in the template. This yielded a code length of 18 700 bytes. The gain is obtained by giving most emphasis to nearby template pixels using their actual pixel values. Template pixels further away are mapped using relative pixel patterns while large templates only are allowed when few colors are present within the template.

The composite image may be coded by bit-planes which also provides progression by bit-planes as in [7]. Each bit-plane is a mixture of several content layers, though (Fig. 3). Instead we may use bit-plane coding as the residual coding. Coding the text layer as a separate layer and coding the residual as (Gray coded) bit-planes yields a code length of 25 698 bytes for the Copenhagen map. Reordering the layers before Gray coding and bit-plane coding reduced the total code length to 23 360 bytes. This is a fair improvement over the 30 426 bytes for bit-plane coding of the Gray coded composite image (Table I). Having text in a separate layer may also be seen as replacing the 4 bit index values with a 5 bit representation which may be bit-plane coded. This provides a simple and efficient content progressive coding based on bilevel image coding only including the residual coding. The results of the more elaborate schemes are presented below.

TABLE III
CODE LENGTHS (BYTES) FOR THE COPENHAGEN KRAK MAPS, 3939×2760 PIXELS

MAP	GIF	PNG	PWC (COLLAPSE)	RAPP M=8	TRPP 9 PIX
Cph038	613692	447448	146901	202650	159602
Cph117	1015170	687673	254243	320810	268098
Cph144	353332	272725	76322	99242	83966
Cph148	1340185	1077940	394339	490430	397022
TOTAL	3322379	2485786	871805	1113132	908688

TABLE IV
CODE LENGTHS (BYTES) FOR THE COPENHAGEN MAPS FOR DIFFERENT TEMPLATE SIZES USING TEMPLATE RELATIVE PIXEL PATTERNS (TRAPP)

TEMPLATE SIZE	4	5	6	7	8	9
Copenhagen	28140	26533	25457	22835	21973	21002
Cph038	211002	203622	195122	172042	164366	159602
Cph117	334222	322690	311194	289342	275322	268098
Cph144	105346	102210	97482	90478	85738	83966
Cph148	513762	493866	480862	437466	411634	397022

The template based bit-plane coding, PWC and RAPP, as well as the techniques of our progressive scheme are all context based techniques. One difference among these schemes lies in the decomposition and order of coding of the data. The other difference is the way they define the contexts.

B. Bilevel Content Layers

Tables V and VI show the results on the Copenhagen map using the bilevel techniques of Section III-B. Table V shows the results for the image coded as the original 12 overlapping bilevel layers. The total code lengths show, that by having access to the original layered bilevel data results better than those of PWC can be achieved. The best result using templates including a reference layer for some layers is almost three times better than GIF. The default setting with a 16 pixel template was used for single layer templates and the default setting with a $9+4$ pixel template was used with the reference layer. Using one reference layer was very beneficial when coding the (white) roads and for utilizing the contours when coding the corresponding interior layers (Table V).

Table VI presents the results coding the 12 nonoverlapping bilevel layers extracted from the composite equivalent (2). Table VII gives the results for our content progressive scheme using hybrid image representations of the image. The residuals are coded by TRPP-SKIP introduced in Section III-C using the default nine-pixel TRPP template. Skip pixel coding is generally superior to ordinary template coding on the extracted layers (Table VI), as the holes from the higher priority layers increases complexity (Fig. 2).

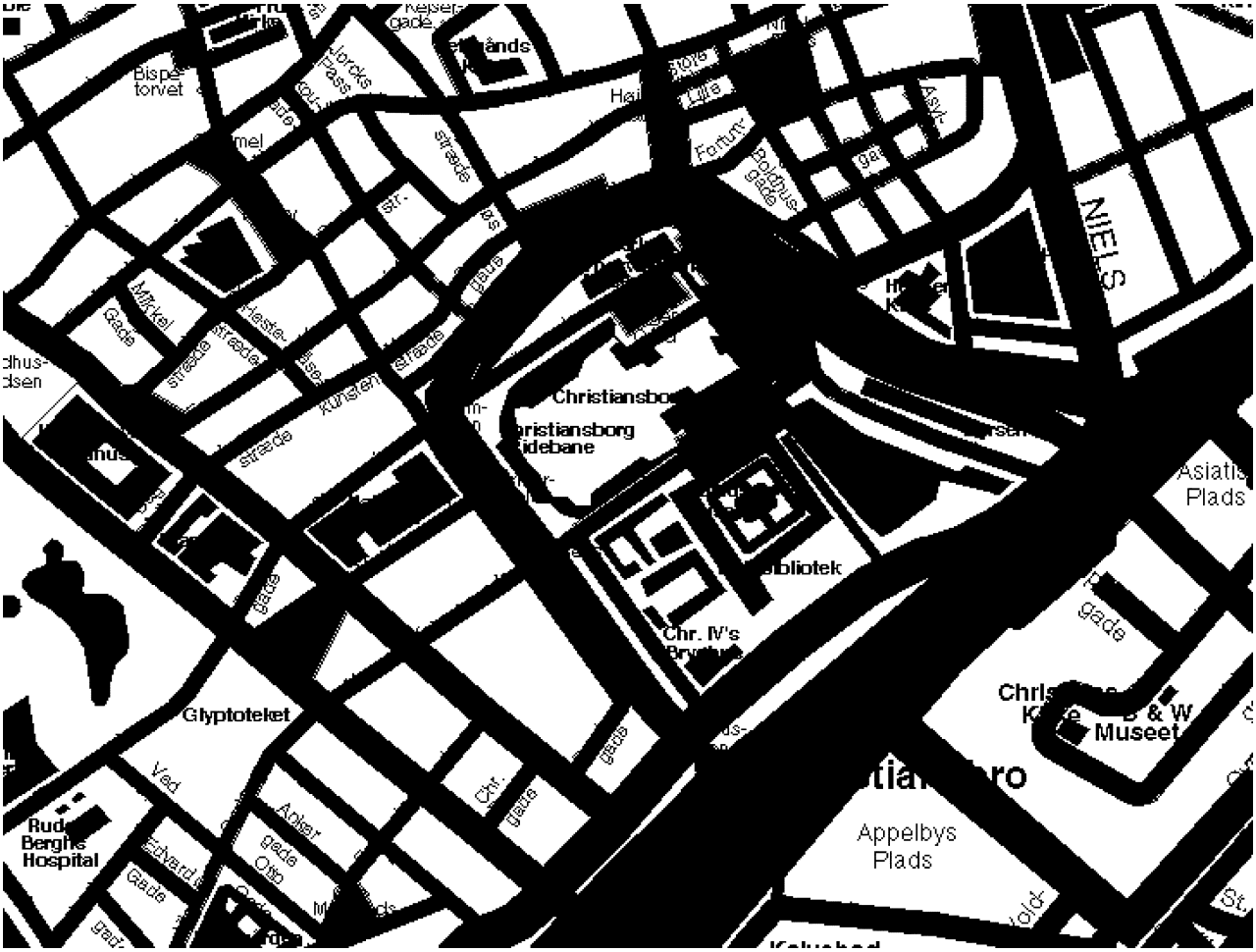

Fig. 3. Bitplane of the copenhagen map. Bitplane (msb) of representation with high priority layers having high index values.

TABLE V
LAYERED IMAGE. CODE LENGTHS (BYTES) FOR THE LAYERS OF THE COPENHAGEN MAP. (rl) MARKS THAT THIS LAYER IS CODED RELATIVE TO LAYER l. * THE RESULT IS OBTAINED BY FREE TREE CODING ON THE TEXT SPLIT IN TWO LAYERS

LAYER DESCRIPTION	OVERLAPPING LAYERS JBIG1	TEMPLATE	TEMP+REF	FREE TREE	BEST
0, TEXT	9514	8952	8952	8772	* 8280
1, ROAD CONTOURS	5103	4512	4512	3556	3556
2, ROADS, PINK	382	340 (r4)	291	248	248
3, ROADS, YELLOW	656	616 (r4)	287	456 (r4)	287
4, ROADS, WHITE	3587	3216 (r1)	447	2140 (r1)	447
5, BUILDINGS	966	924	924	792	792
6, WATER, CONT.	522	492	492	416	416
7, WATER	510	480 (r6)	124	392 (r6)	124
8, PARKS, CONT.	188	176	176	156	156
9, PARKS	189	172 (r8)	76	132 (r8)	76
10, LAND, CONT.	735	516	516	388	388
11, LAND	409	408 (r10)	108	320 (r10)	108
TOTAL	22761	20804	16905	17768	14878
COMPOSITE	22352	20396	16797	17448	14770

TABLE VI
LAYERS EXTRACTED FROM COMPOSITE IMAGE. CODE LENGTHS (BYTES) FOR BI-LEVEL SPLIT LAYERS EXTRACTED FROM THE COPENHAGEN MAP. SKIPt USES AN EXPLICIT SKIP SYMBOL IN THE CONTEXTS

LAYER DESCRIPTION	NON OVERLAPPING LAYERS TEMPLATE	SKIPt	SKIP	FREE TREE
0, TEXT	8952	9470	8952	8772
1, ROAD CONTOURS	6008	6286	5488	5184
2-4, ROADS	9055	3030	3928	8248
5, BUILDINGS	1683	1478	1240	1572
6-7, WATER W. CONT.	1384	1316	756	1264
8-9, PARKS W. CONT.	328	836	152	284
10-11, LAND W. CONT.	6760	820	1396	6164
TOTAL	34170	23236	21912	31488
COMPOSITE	27682	22914	20700	25544

TABLE VII
HYBRID CODING. CONTENT PROGRESSIVE CODE LENGTHS (BYTES) FOR THE COPENHAGEN MAP IN DIFFERENT PROGRESSIVE VERSIONS. LAST COLUMN IS FOR (NON-OVERLAPPING) SPLIT LAYERS. * SKIP CODING WAS USED FOR CODING THE ROAD CONTOUR LAYER

CODING METHOD	LAYER DESCRIPTION	SIZE	SIZE	SIZE	SIZE
TEMPLATE	TEXT	8952	8952	8952	8952
TEMPLATE	ROAD CONTOURS	4512		4512	* 5488
TEMPLATE	ROADS, WHITE		3216	(r1) 447	
TEMPLATE	ROADS, PINK			340	
TEMPLATE	ROADS, YELLOW			616	
TRPP-SKIP (9 pix.tem.)	RESIDUAL	4669	10912	3016	4669
TOTAL		18133	23080	17883	19109

The results of Tables V–VII all represent content layer progressive coding. We can extract and calculate the best results depending on the setting we are aiming at, i.e., whether the composite or layered image is used as input to the encoder and what the (desired) attributes of the output are: The best result for coding the complete layered image, $\mathbf{x}^T$ is 14 878 bytes (Table III). For five of the layers, context pixels were chosen from other layers, introducing a dependency. This figure is reduced to 14 770 if we are only required to reproduce the composite image, y^T, as the last layer is not necessary in this case. If we want to have random access to the layers and be able to choose any ordering of the layers after coding, all layers should be coded independently. The best result in this case is 17 276 bytes obtained by the free tree [9] and coding

TABLE VIII
CODE LENGTHS (BYTES) FOR CODING MAP CPH117 BY THEMES. (n) REFERS TO THE NUMBER OF BI-LEVEL LAYERS IN THE THEME

LAYER THEME DESCRIPTION	CODING METHOD GIF	PNG	PWC	TRPP
SYMBOLS (7)	36907	24750	4007	4062
TEXT (1)	225095	218105	84506	87254
GRID (1)	60967	24521	1355	430
LINES (4)	48406	40920	6252	7022
ROADS (7)	397342	284782	86140	79722
AREAS (7)	283047	187875	64138	68930
TOTAL (27)	1051764	780953	246398	247420

TABLE IX
CODE LENGTHS (BYTES) FOR BI-LEVEL LAYERS ACCUMULATED BY THEMES FOR MAP CPH117. (n) REFERS TO THE NUMBER OF BI-LEVEL LAYERS IN THE THEME

LAYER THEME DESCRIPTION	CODING METHOD TEMPLATE	TEMP+REF	FREE TREE	SKIP	BEST
SYMBOLS (7)	6678	6678	4836	4398	3108
TEXT (1)	75363	75363	60556	81602	60556
GRID (1)	220	220	76	1222	76
LINES (4)	6542	5239	4308	6444	3658
ROADS (7)	104374	69042	55036	85122	46540
AREAS (7)	104435	67837	57824	50330	28702
TOTAL (27)	297612	224379	182636	229118	142640

the text layer as the two coding layers it was split into in Fig. 1. When the encoder only has access to the composite version, the best result is 18 589 bytes using the free tree for layers 0 and 1 (Table VI) and residue TRPP-SKIP (9 pixel template, Table VII) coding for the rest. It should be noted that free tree encoding is very complex. Using a 16 pixel template instead of the free tree the result is 19 109 bytes (Table VII). Comparing with the last column in Table VII, the cost of coding the hybrid image without knowledge of the original overlapping layers is only 976 bytes in this case.

For the settings considered above, the best results are all better than the PWC result while at the same time providing content progressive coding. The overall best result reduced the code length to 70% of PWC. Having only the composite image at the encoder the code length was reduced to 90% of PWC. When the layered image is coded more information ($\mathbf{x}^T$ or $\mathbf{y}^T$) is conveyed than just the composite image (y^T) which obviously has lower (or in special cases equal) entropy, given the rule of composition.

In Table VI, the results show that SKIP coding (4) using a 16 pixel binary context is more efficient than the 10 pixel ternary context of SKIPt coding (3).

C. Thematically Layered Maps

Tables VIII and IX list more results for one of the KRAK maps (Cph117). This map has 27 color levels. It was provided, as the other KRAK maps, as $\mathbf{y}^T$ in the six themes listed in Tables VIII and IX. Therefore these maps are actually hybrid images and not fully layered images by the definition in Section II. Coding this image theme by theme increases the code length for GIF and PNG, whereas it improves the performance of TRPP and PWC (Table VIII). In this case PWC is just slightly better than TRPP. The themes were also split using (2) to obtain a full bilevel layered description ($\mathbf{x}^T$). The results of coding these bilevel layers are given in Table IX accumulated by theme. The

TABLE X
CODE LENGTHS (BYTES) FOR BI-LEVEL LAYER CODING OF THE FOUR LAYERED COPENHAGEN KRAK MAPS

MAP	CODING METHOD TEMPLATE	TEMP+REF	FREE TREE	SKIP	BEST
Cph038	165826	132783	109996	133189	80707
Cph117	297612	224379	182636	229118	142640
Cph144	81031	62997	51984	68329	39302
Cph148	412929	346546	276176	344477	209938
TOTAL	957398	766705	620792	775113	472587

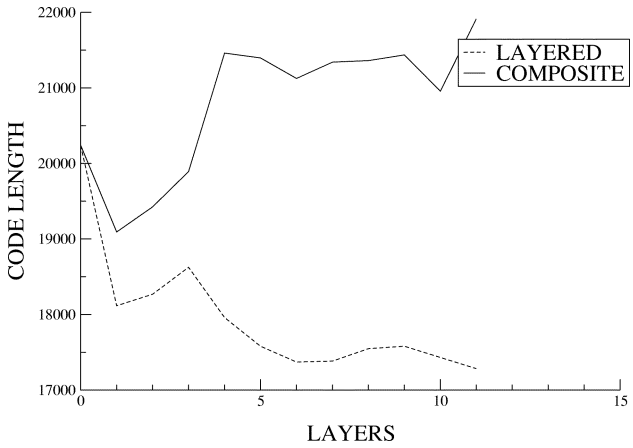

Fig. 4. Hybrid progressive total code lengths of the copenhagen map as a function of the number of bilevel layers (n_b). The bilevel layers are the original layers (- -) or extracted (—) from the composite image. The residual images are coded using TRPP-SKIP.

SKIP coding codes the composite image by split layers yielding a 7% shorter code length than PWC (Table III). Even better results are achieved by the free tree or by template coding using a reference layer for some of the bilevel layers (temp + ref). To evaluate the potentials, the best result of the free tree and SKIP coding was picked at a bilevel layer basis, yielding a hybrid code length of 142 640 bytes. This is 42% less than PWC and more than seven times less than GIF on the composite image (Table IX). Table X summarizes the results accumulated over the whole image for each of the four KRAK images coded by bilevel layers. The best combined result is 7 times better than GIF for the four maps and up to 9 times better on single maps. Overall it codes the four maps to 46% less than the PWC code length on the composite images.

These results demonstrate that we can achieve content layer progressive coding and at the same time improve the performance of the state of the art PWC coder. On the other hand the results also suggest that introducing skip pixels in PWC coding would provide an even better residual coder.

D. Hybrid Progressive Coding

Figs. 4 and 5 show the total code lengths for hybrid image coding with n_b bilevel layers followed by residual coding, i.e., using $n_b + 1$ coding layers in all. The results are shown as a function of n_b, for the bilevel layers coded in the order of their composition order. In Fig. 4, the results are given for the Copenhagen map both with the original bilevel layers available (layered input) at the encoder and coding split layers extracted from the composite image (composite input). In the first

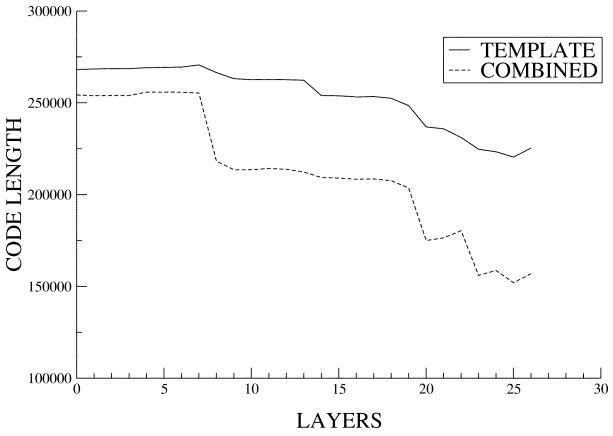

Fig. 5. Hybrid progressive code lengths for map Cph117 as a function of the number of bilevel layers (n_b). Template coding was applied to layers and TRPP-SKIP to the residual. The best bilevel result for each of the n_b bilevel layers was combined with the best of TRPP-SKIP and PWC for the residual.

case, the bilevel layers are encoded using template coding. For layers 4, 7, 9, and 11 reference pixels are included in the template (Table V). In the latter case SKIP coding is applied to the split layers (Table VI). In both cases the residual is coded using relative pixel patterns and skipping the pixels already coded (TRPP-SKIP). Adding the layers in the order of composition and coding, the progressions yield the same visual appearance at the decoder. In Fig. 5, hybrid coding is applied to the bilevel layers of map Cph117. The n_b bilevel layers are template coded using reference pixels for some of the layers (Table IX). The residual coding is by TRPP-SKIP as for the Copenhagen map. A combined result is also depicted selecting the best results. The best bilevel coding is chosen for each bilevel layer (Table IX). For each residual the best choice of TRPP-SKIP and PWC is used.

For the results in Figs. 4 and 5, there is a clear tendency that the total code length improves as more levels are coded by coding their bilevel layer and thereby using more coding layers. This is interesting as more information is available and the entropy is higher given the composition rule. When only the composite image is available (Fig. 4) the code length increases with the number of coding layers, though little for few layers and less than 10% for progression over all bilevel layers. In general the loss introduced by performing progression by levels derived from the composite image compared to coding the original layers can be kept at a small level.

VI. Conclusion

The presented content progressive lossless coding scheme achieves very good overall compression results for limited bits/pixel images of maps. The compression factor was improved up to a factor of nine compared to GIF. The code lengths for layered image data was reduced to as little as 50–70% in comparison with the state-of-the-art PWC. At the same time the new scheme provides progression. The progression is achieved by coding multiple content image layers as text, buildings, etc., for maps. Especially progression over a number of bilevel layers was investigated. When the full layered data set is available at the encoder, the progressive scheme actually provides more information than the ordinary composite map image. For efficient coding, information from pixels coded in layers of higher priority were skipped (skip pixels) in the current and all lower layers. For bilevel layer coding, an already coded layer was also used by providing pixels for the context of a proceeding layer. The introduced principle of skip pixel coding was combined with existing efficient template based context bilevel coding [8], context collapsing methods for multilevel images [6] and arithmetic coding. Analysis was carried out supporting that it is quite feasible to use a template size of 9–10 independently of the alphabet size when relative pixel patterns are applied to template pixels. The coding scheme was also applied to layers extracted (by color) from ordinary composite images providing efficient progressive coding by applying the skip pixel technique. All in all, the new schemes can be used as a flexible tool for efficient progressive coding of digital maps. Future work includes applying some of the techniques to other computer graphics material with limited bits/pixel.

References

[1] *Progressive Bi-Level Image Compression*, ISO/IEC Int. Std. 11 544, 1993.
[2] *Lossy/Lossless Coding of Bi-Level Images (JBIG2)*, ISO/IEC Int. Std. 14 492, 2000.
[3] M. J. Weinberger, J. Rissanen, and R. B. Arps, "Applications of universal context modeling to lossless compression of grey-scale images," *IEEE Trans. Image Processing*, vol. 5, pp. 575–586, Apr. 1996.
[4] *Lossless and Near-Losless Compression of Continous-Tone Still Images*, ISO/IEC Int. Std. 14 495, 1999.
[5] P. J. Ausbeck Jr., "A piecewise-constant image model," *Proc. IEEE*, vol. 88, no. 11, pp. 1779–1789, Nov. 2000.
[6] V. Ratnakar, "RAPP: Lossless image compression with runs of adaptive pixel patterns," in *Proc. 32nd Asilomar Conf. Signals, Systems and Computers*, Nov. 1998.
[7] Y. Yoo, Y. Kwon, and A. Ortega, "Embedded image-domain adaptive compression of simple images," in *Proc. 32nd Asilomar Conf. Signals, Systems, and Computers*, Nov. 1998.
[8] P. G. Howard, F. Kossentini, B. Martins, S. Forchhammer, and W. J. Rucklidge, "The emerging JBIG2 standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 8, pp. 838–848, Nov. 1998.
[9] B. Martins and S. Forchhammer, "Tree coding of bilevel images," *IEEE Trans. Image Processing*, vol. 7, pp. 517–528, Apr. 1998.
[10] S. Forchhammer and O. R. Jensen, "Content layer progressive coding of digital maps," in *Proc. Data Compression Conf.*, Mar. 2000, pp. 233–242.

Søren Forchhammer received the M.S. degree in engineering and the Ph.D. degree from the Technical University of Denmark, Lyngby, in 1984 and 1988, respectively.

Currently, he is an Associate Professor at Research Center COM, Technical University of Denmark, where is has been employed since 1988. His main interests include source coding, data compression, video coding, information theory, and image communications.

Ole Riis Jensen received the M.Sc. degree in electrical engineering and the Ph.D. degree from Department of Telecommunication, Technical University of Denmark, in 1992 and 1997, respectively.

From 1997 to 1999 he was a Research Assistant Professor with the same department, working in the fields of image and video coding and multimedia techniques. Since 1999, he has been with Xerox Corporation, Ballerup, Denmark. Since 2001, he has been a Technology Project Manager in Xerox Global Services, Ballerup.