Context based Coding of Binary Shapes by Object Boundary Straightness Analysis

Aghito, Shankar Manuel; Forchhammer, Søren

Published in:
Data Compression Conference, 2004. Proceedings. DCC 2004

Link to article, DOI:
10.1109/DCC.2004.1281485

Publication date:
2004

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

Citation (APA):
Aghito, S. M., & Forchhammer, S. (2004). Context based Coding of Binary Shapes by Object Boundary Straightness Analysis. In *Data Compression Conference, 2004. Proceedings. DCC 2004* (pp. 399-408). IEEE. https://doi.org/10.1109/DCC.2004.1281485

Context Based Coding of Binary Shapes by Object Boundary Straightness Analysis

Shankar Manuel Aghito and Søren Forchhammer
Research Center COM, 345v, Technical University of Denmark
e-mail: sma@com.dtu.dk sf@com.dtu.dk

Abstract

A new lossless compression scheme for bi-level images targeted at binary shapes of image and video objects is presented. The scheme is based on a local analysis of the digital straightness of the causal part of the object boundary, which is used in the context definition for arithmetic encoding. Tested on individual images of binary shapes and binary layers of digital maps the algorithm outperforms PWC, JBIG and MPEG-4 CAE. On the binary shapes the code lengths are reduced by 21%, 25%, and 42%, respectively. On the maps the reductions are 34%, 32%, and 59%, respectively. The algorithm is also more efficient than the state-of-the-art and more complex Free Tree coder for most of the binary shape and map test images.

1 Introduction

Coding of bi-level images representing object boundaries (binary shapes) is used in object based video coding. In MPEG-4 this coding is performed by the binary CAE (context-based arithmetic encoding). The symbol probabilities are determined by a predefined table with an index defined by pixels within a template. This algorithm is simple, but not so efficient in comparison with state-of-the-art context based arithmetic encoders. The recent video compression standard, H.264, has reduced the bit rate (at a given PSNR quality) to about half the rate required by previous video coding standards, at the cost of increased complexity. It is therefore natural to consider a more complex and efficient encoder for binary shapes, for use in future object based video encoders. A rate control analysis of MPEG-4 video object coding showed that the shape information has high priority in terms of operational rate-distortion [14].
This paper presents different techniques for improving template based coding of binary image/video shapes and binary layers of digital maps. The presented work is restricted to single images (or I-frames in video terminology). The most novel contribution is a new approach for defining the coding context based on analysis of the digital straightness of the causal part of the object boundary. The proposed algorithm is also expected to provide efficient coding for other types of bi-level images such as layers of digital maps and computer generated material. It is not targeted at halftone images and text, for which efficient context based coding techniques already exist [9] [4]. The new algorithm should easily be modified to provide efficient coding of line drawings in general.
Several proposals for lossless intra coding of binary shapes are found in the literature. The vertex-based method described in [5] provides an average reduction of 7.8% with respect to CAE. The skeleton-based method proposed in [15] gives bitrates $8-18\%$ smaller than CAE. The differential chain code method proposed in [12] gives reductions on CAE up to 10%, but also increments up to 13%, for an average reduction of less than 1%. All these results where obtained using different MPEG-4 test sequences, at QCIF, CIF and SIF resolutions. Our proposal was tested on standard TV material, i.e. higher resolution, so a direct comparison with the algorithms above is not possible.

Binary shapes and their characteristics are described in Section 2. In Section 3 we review some of the known properties of ideal digital straight lines. The proposed algorithm is described in details in Section 4. The complexity of the proposed algorithm is briefly discussed in Section 5. Coding results are shown in Section 6.

2 Properties of binary shapes

Binary shapes are used in object based video. There are basically two types: binary shapes obtained from natural video with techniques such as blue-screening or segmentation, and binary shapes which are part of a video composed using computer generated video objects. In both cases, the majority of binary objects are large connected uniform regions of pixels. For halftone material the primary information is the local density of the image. For binary shapes and most other bi-level images the information to be used for efficient coding lies along the boundary of the objects. The task of the encoder may be viewed as predicting whether the current pixel locally lies before or after the boundary on the current line. The curvature of the (underlying) boundaries is typically continuous, and can locally be represented by a digital straight line segment, as shown in Fig. 1. The figure illustrates why the coding of pixels in the proximity of the boundaries may easily be improved taking the causal part of the boundary into consideration.

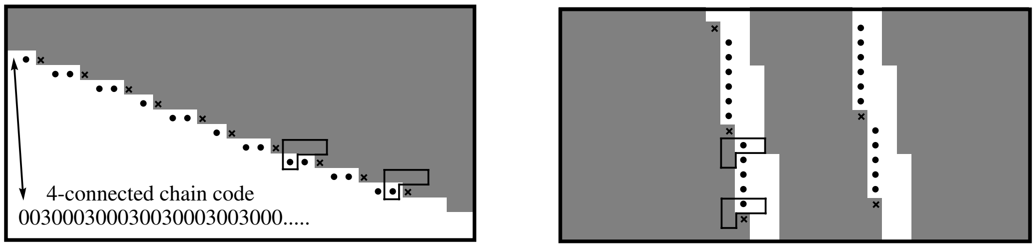

Figure 1: Excerpts of the binary shape sequences Akiyo and Rain, respectively. The boundary pixels indicated with ● are within the object, while the boundary pixels indicated with × belong to the background. Within each image the context defined by a 4 pixel template is the same for the boundary pixels marked.

3 Digital straight lines and boundaries

As illustrated in Fig. 1 tracking the causal part of a boundary to predict the next element should enable efficient coding in many cases. For computer generated binary images the local appearance and the apparent uncertainty of a boundary is related to the digitization of the (geometric) objects. In [11] digitization of straight lines at all angles were used to analyze the rate for the individual chain elements. Template coding increases the order of the probability estimates but even viewed through the (small) window of a template the digitization leads to significant uncertainty (Fig. 1). If the boundary indeed is given by an ideal digital straight line the rate may be greatly reduced by considering the long range higher order dependencies. The ideal Digital Straight Line Segments are treated below.

3.1 Digital Straight Line Segments (DSLS)

Looking at Fig. 1, the boundary may be described in terms of a 4-connected set of edges (right, up, left, down) separating the black and white pixels or in terms of an 8-connected set (including also the diagonal directions) of boundary pixels. There is a one-to-one correspondence between the two sets. Without loss of generality we consider the 8-connected case in the first octant based on the digitization of the straight line

$$y(x) = \alpha x + e, \quad 0 \le \alpha < 1. \tag{1}$$

A digital straight line segment with n elements is defined [2] by the set of grid points (x_i, y_i) resulting from the digitization of (1) described by

$$y_i = \lfloor \alpha i + e \rfloor, \quad x_i = i, \quad i \in 0, 1, ..., n. \tag{2}$$

The $n+1$ grid points (x_i, y_i) are 8-connected by n chain elements. Each chain element, d, is defined as 0 $(y_{i+1} = y_i)$ or 1 $(y_{i+1} = y_i + 1)$.
In his classic paper on chain codes Freeman stated three properties of digital straight lines (cited from [13]): "(F1) at most two elements can be present, and these can only differ only by unity modulo eight; (F2) one of the two elements values always occurs singly; (F3) successive occurrences of the element occurring singly are as uniformly spaced as possible." The last imprecise property has later been formally formulated. In [6] the order of the number of (8-connected) lines segments of length n with fixed starting point was determined as n^3/π^2. Thus clearly the per symbol rate asymptotically approaches zero as $(3\log n)/n$. In Appendix A it is shown that applying LZ78 universal coding [16] to an infinite DSLS sequence, the per symbol rate also converges to zero. The properties of a DSLS have been studied intensively (see [13] for an overview). These properties include a relation between the value of the slope, α, of the line and continued fractions as well as a characterization of the preimage geometric lines e.g. in a parameter space given by the slope α and the offset e (2) [2]. We refer to the set of possible preimages of a DSLS as the preimage domain described by a set of (α, e) values.

3.2 Parsing a DSLS

A simple and fast algorithm for sequential parsing or recognizing an DSLS from a sequence of chain elements is given below [7]. The algorithm, formulated for the 4-connected edge description, is based on the calculation of the narrowest strip defined by the positive and negative bases, as shown in Fig. 2. If a sequence of elements contains only one distinct

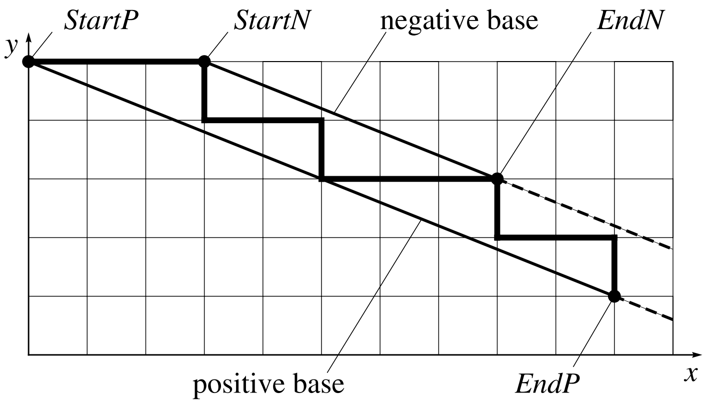

Figure 2: Notation for the DSLS parsing algorithm [13].

value then the sequence is a DSLS. When a second distinct value occurs for the element with index t the algorithm is initialized, setting the four limiting points $StartP = (0, 0)$, $StartN = EndN = (t-1, 0)$ and $EndP = (t-1, -1)$. The first chain value corresponds to the positive x direction, the second value corresponds to the negative y direction. Assume that a sequence of n (4-connected) grid points is a DSLS. We want to know if the chain obtained adding a new grid point (x_n, y_n), called $Point$, is still a DSLS. The information about the DSLS is, besides the length, captured by the four limiting points. Let $Tang = (u, v)$ be the vector parallel to the two bases, with relatively prime integer coordinates. Introduce $w = uy - vx$ for any point (x, y) on the negative base. Calculate $h(x_n, y_n) = vx_n - uy_n + w$. We can have the following cases:

$0 \leq h(x_n, y_n) \leq |u| + |v| - 1$: the new point is on the given DSLS.

$h(x_n, y_n) = -1$: The n vertices form the DSLS with new parameters $EndN = (x_n, y_n)$, $StartP = EndP$, $Tang = Point - StartN$.

$h(x_n, y_n) = |u| + |v|$: The n vertices form the DSLS with new parameters $EndP = (x_n, y_n)$, $StartN = EndN$, $Tang = Point - StartP$.

otherwise : the $n + 1$ vertices do not form a DSLS.

3.3 Prediction by DSLS

For a DSLS with a given prior on the $(\alpha, e)-$preimage domain it is possible to perform optimal compression sequentially applying arithmetic coding to the conditional probabilities derived from the DSLS analysis. The following is derived using the analysis in [2]. For many elements the choice is deterministic. For increasing length n, non-deterministic elements occur at increasing distance bisecting the preimage domain into two. Without loss of generality the case with a slope $0 \leq \alpha < 1$ is considered. Given the t chain elements d^t, assume for the next chain element d_{t+1} that two values, denoted a and b, are possible i.e. both $d_t a$ and $d_t b$ constitute a DSLS extension of d_t. Let $A(d^t a)$ denote the area of the (α, e) preimage domain of the DSLS $d^t a$. Assuming a uniform prior on the (α, e)-preimage domain, the conditional probability for the next element is simply expressed by the ratio of the areas,

$$P(a|d^t) = \frac{A(d^t a)}{A(d^t a) + A(d^t b)}, \tag{3}$$

where $P(a|d^t)$ denotes the probability of element a at position $t+1$ given d^t. The range of slopes of the preimages are given by, $p^-/q^- < \alpha < p^+/q^+$. The preimage domain is given by a quadrangle (or triangle) with vertices at $\alpha = p^-/q^-$ and $\alpha = p^+/q^+$ and maximum width in the e dimension at $\alpha = p/q$ [2]. All values of p and q are integers in the range given by $0 \leq p \leq q \leq n$. (There is a one-to-one mapping between the limiting points of the parsing algorithm described in Sec. 3.2 and the edges of the preimage domain.) Thus the area of the preimage is

$$A(d^t a) = \frac{1}{2q(a)} \left(\frac{p^+(a)}{q^+(a)} - \frac{p^-(a)}{q^-(a)} \right), \tag{4}$$

where the argument (a) specifies that it is the values of the DSLS preimage of $d^t a$. Inserting (4) for a and b in (3) gives the conditional probability. A simplified approximation is given by just considering the ratio of the q values in (4).

$$\hat{P}(a|d^t) = \frac{q(b)}{q(a) + q(b)}. \tag{5}$$

The next section describes how the DSLS analysis may be combined with context based coding as part of the context formation.

4 Proposed algorithm

The proposed algorithm is a combination of several techniques which are suited for binary shapes and similar material: DSLS context based coding, modified skip-innovation codes, distance based template quantization, and locally adaptive statistics update. These techniques may also be useful for a larger class of images. Scanning the image in raster order, the selection between the different coding modes is done based on the values within the template defined by the 4 causal neighbors closest to the coding pixel.

4.1 DSLS based context coding (DSLSC)

We shall not rely on actually having straight lines in the image but rather use the digital straightness of the boundary as part of the context formation. If the current pixel to be coded is in the proximity of an object boundary, the idea is to use the theory introduced in Section 3 to get a better estimate of the symbol probabilities. If the 4 pixel template is one of the 5 cases in Fig. 3 (plus the inverted instances) indicating an edge, we track the causal part of the boundary as long as the chain code represents a DSLS. The maximum length for tracking the chain code is fixed by the parameter *maximum line length* (default value 50). If the length of the chain l_{DSLS} is not less than the parameter *minimum line*

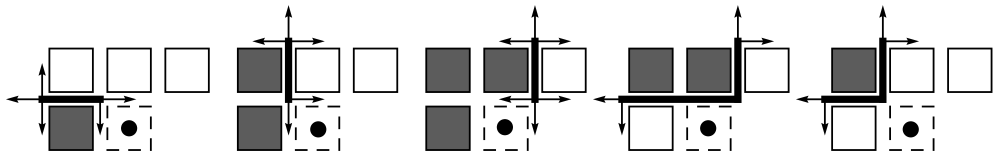

Figure 3: Possible cases for DSLS analysis. The arrows indicate the allowed directions for extending the DSLS around the current pixel marked with •.

length (default value 10) we check for how many directions the DSLS can continue around the coding pixel. There are two interesting cases.

Deterministic case. The DSLS can continue in 1 or 2 directions, corresponding to only one possible predicted value for the coding pixel.

Non-deterministic case. The DSLS can continue in 2 directions, corresponding to two different predicted values of the coding pixel.

Given the deterministic case, a right decision r is encoded if the current pixel is consistent with the predicted value, and a wrong decision w is encoded otherwise. The probabilities used for the encoding are

$$P(r) = \frac{n_r + 0.9}{n_r + n_w + 1} \quad \text{and} \quad P(w) = \frac{n_w + 0.1}{n_r + n_w + 1}, \tag{6}$$

where n_r and n_w are the numbers of right and wrong decisions encoded so far. Different contexts can be defined for this decision. With the default settings the context is defined by l_{DSLS}, dividing the interval defined by *minimum line length* and *maximum line length* into a specified number, *length intervals* (default value 1), of equally spaced intervals. The context is further specified according to the 4 pixel template (one of the 5 cases in Fig. 3, each lumped together with the inverted configuration) and to the length of the last run of non-singular elements in the chain code of the DSLS (by dividing the interval between 0 and the length of the longest non-singular run in the DSLS in the specified number of *runs intervals*, default value 5). The number of parameters with the default settings is 25.
Given the non-deterministic case we can estimate the probabilities that the line will continue in the two directions using equation (3) (default choice) or (5).
In all the other cases, we code the pixel as described in 4.2 to 4.4. (These cases are: the template is not one of the configurations in Fig.3, the l_{DSLS} is not in the selected interval, the DSLS can continue in 3 different directions, the DSLS cannot continue in any of the allowed directions.)

4.2 Modified skip-innovation codes (MSI)

Skip-innovation (SI) codes were introduced in PWC [1] as a fast and efficient way for coding large uniform regions. When a uniform 4 pixel template is found, there is a high probability that a run of pixels of the same color will occur. The run length is estimated from the length S of the corresponding run in the row above. If the run is not shorter than S then a 1 is coded, otherwise a 0 is coded followed by a binary representation of the length I of the run before the innovation (shorter codes are assigned to long runs

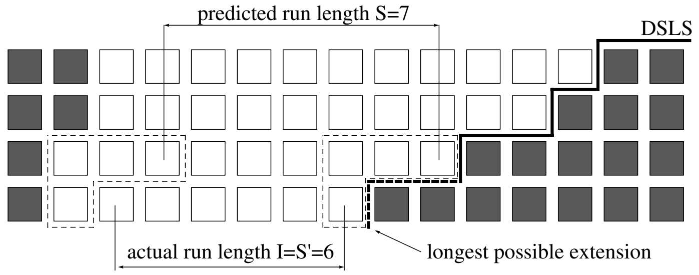

Figure 4: A failed skip followed by an innovation is coded with the basic version of SI codes. Analyzing the boundary we can reduce the predicted run length to S', which in this situation is the correct prediction and will be coded as a full skip.

if S is not a power of 2). The bits of the SI code words are coded using context based arithmetic encoding. We propose a modification based on modeling the situations which Ausbeck refers to as pseudo-innovations (Fig. 4). The goal is to reduce the probability of failed skips (which are expensive) by assuming that the edge of the structure occurring at the end of the run on the line above is a DSLS. Figure 4 illustrates the approach, when a pseudo-innovation occurs. In this case, it is clear that analyzing the boundary, a run length shorter than a full skip S is highly probable. Let S_{DSLS} be the maximum run allowed by all the possible straight extensions of the DSLS. If the causal part of the DSLS is not shorter than the specified value of *minimum line length for SI* (default value 5), then S is replaced by $S' = max\{S_{DSLS}, S - 10\}$. The lower bound $S - 10$ prevents a too large reduction of the skip, which might be undesired if the edge of the pseudo-innovation is not perfectly straight.
A second modification to the SI as used in PWC is about the upper bound for the counts in the statistics. In PWC this is set to 255 for speed optimization and because the use of SI codes has the effect of reducing the counts in the statistics, such that counts higher that 255 are not often seen. In our target material the number of successful skips is often larger than 255, so the upper bound was removed.

4.3 Distance based template quantization (TQ,TE)

It is well-known that the use of large templates leads to the context dilution problem, as well as the need for memory space to allocate the statistics. The problem can be partly solved assuming that, apart from the boundary, the importance of context pixels decreases with increased distance. Starting from the norm-2 ordered 16 pixels template [10] with pixel values $T(i)$, $\forall i \in [1, 16]$, we define the quantized template values

$$T_Q(i) = \begin{cases} T(i) & if \;\; 1 \le i \le K \\ T(K+1) \;\; OR \;\; T(K+2) & if \;\; i = K+1 \\ T(K+3) \;\; OR \;\; T(K+4) & if \;\; i = K+2 \\ T(K+5) \;\; OR \;\; T(K+6) & if \;\; i = K+3 \end{cases} \qquad (7)$$

where $K \le 10$ specifies the so-called internal template size. The first K pixels of the template contribute to the definition of the context as usual, while the following 6 pixels are coupled in pairs, assigning a value of 1 to each pair if at least one of the two pixels is within the object, and 0 otherwise. The number of parameters is 2^{K+3}.
For the case of uniform (quantized) templates, the following method can be applied as an alternative to SI codes. If the $K + 3$ values of the quantized template are all the same, then the quantized template lies completely inside or completely outside the object. These contexts are very frequent in binary shapes, and it may be a good idea to enlarge the coding context in these situations, in order to discriminate the cases where the current pixel is in the proximity of the border from the cases where the pixel is far from it. We define the extended template values T_E with the values of the remaining neighbors from $T(K + 7)$ to $T(16)$ plus the values of three additional pixels with relative position (horizontal and vertical, respectively) respect to the coding pixel $[-6, 0]$,$[0, -6]$ and $[+6, -1]$. We distinguish 4 cases for the uniform quantized template, as follow.

context A All the values in T_Q are 0 AND all the values in T_E are 0
context B All the values in T_Q are 0 AND at least one the value in T_E is 1
context C All the values in T_Q are 1 AND all the values in T_E are 0
context D All the values in T_Q are 1 AND at least one the value in T_E is 0

The number of parameters is $2^{K+3} + 2$.

4.4 Locally adaptive statistics update (LS)

Adaptive template based coding can be improved if the estimation of probabilities places more weight on recent occurrences than the older ones. This is a way of exploiting the local 'stationarity' of images. This is often done by periodically reducing all the counts in the statistics. In our implementation, besides the overall counts $n_i(c)$ of the symbol occurrences for each context index $c \in [1, 2^{K+3}]$ and each pixel value $i \in [0, 1]$, also the recent counts $n_{R,i}(c)$ and the very recent counts $n_{VR,i}(c)$ are accumulated. The counters $n_i(c)$ and $n_{VR,i}(c)$ are updated after each symbol is encoded. At the beginning of each row we set $n_{R,i}(c) = n_{VR,i}(c)$, and at the beginning of each even-numbered row we set $n_{VR,i}(c) = 0$. The probability estimate for the coding symbol being 0 is obtained with

$$P(0|c) = \frac{n_0(c) + 10n_{R,0}(c) + 10n_{VR,0}(c) + 0.5}{n_0(c) + 10n_{R,0}(c) + 10n_{VR,0}(c) + n_1(c) + 10n_{R,1}(c) + 10n_{VR,1}(c) + 1}. \quad (8)$$

The factor 10 weights the recent and the very recent statistics 10 times more than the overall statistics. Little effort has been put into this and we do not claim that this is the best way for tracking the changing statistics. Table 1 demonstrates that this method combined with template quantization improves the coding on our test material.

5 Fast implementation and fuzzy versions

The work of this paper has focussed on high compression performance and the DSLS theory, but fast implementation should be achievable. Using skip-innovation provides fast coding for most of the pixels. For binary shapes, the majority of the pixels are coded as full skips or long innovations. The proposed algorithm can be implemented as a single pass raster order process. The DSLS parsing algorithm utilized is an on-line implementation, meaning that each time a section of a boundary is tracked and modelled as a DSLS, the few parameters (see Section 3.2) defining the DSLS can be stored and passed to the following row, making the process quite fast. Internally in the encoder and decoder the edges could be stored using run-lengths to the next edge and the parameters of the edge. Storing these parameters is not a problem, because typical binary shapes do not contain a large number of boundaries (in the majority of the cases it is only 2 for each row). For bi-level images with many objects as dense small font text, the DSLS coding mode should be disabled.
Many variations of the presented algorithm may be realized, the main principle being to track the causal boundaries for prediction. Fuzzy versions of the proposed algorithm could be implemented relaxing the necessary conditions for a DSLS, e.g. only using Freemans first two conditions (F1+F2), which would make the implementation simpler, and eventually more suitable to a larger class of images without perfect digital straightness.

6 Results

We compared the proposed algorithm with JBIG[8], PWC [1], the Free Tree coder [10], and the MPEG-4 method for binary shapes CAE (results obtained with the MoMuSys reference software). We chose a selection of binary shapes from the MPEG-4 test set (720×486 pixels, 30 Hz): layer 1 of sequence akiyo, layer 1 and 2 of the sequence children (named kids and logo respectively), layers 1, 2, 3, 5 (robot), 6 (explosion), 7 and 8 (rain)

of the sequence destruction. The layers of destruction and logo are computer generated, while akiyo and kids are extracted from natural video. The algorithms were also tested on some binary layers of a digital map of Copenhagen (723×546 pixels) [3], and some of the CCITT fax images (2339×1728 pixels). For the proposed algorithm we report the ideal code lengths obtained by summing the negative value of the logarithm of the estimated probabilities. Actual code lenghts close to these values may be obtained by applying arithmetic coding. A small contribution for header information should be added in this case (also for Free Tree and CAE). Results are reported in Table 1.
The different combinations of the building blocks described in Section 4 are compared: DSLS based context coding (DSLSC), modified skip-innovation codes (MSI), distance based template quantization (TQ), template extension for the uniform context case (TE), and locally adaptive statistic update (LS).
TQ combined with LS and TE gives an average rate reduction of 5% on the binary shapes, and 2% on the maps, compared with normal template coding.
Compared with TE, MSI gives 1% rate reduction on the shapes, and 8% on the maps, while normal SI codes gives a rate increase of 1% and 2%, respectively.
The proposed algorithm which uses all the building blocks including DSLSC provides very good coding results for all the tested binary shapes, yielding code lengths of 0.00062 bpp up to 0.037 bpp. Measured by the reduction of the average code length, the algorithm with the default settings is as efficient as the state-of-the-art (and more complex) Free Tree coder, 21% better than PWC, 25% better than JBIG and 42% better than CAE. Setting *minimum length* = 3, *minimum length for SI* = 3 and *length intervals* = 5 the

img/seq, no. frames	templ. coding (best)	TQ LS TE (best)	TQ LS SI (best)	TQ LS MSI (best)	TQ LS MSI DSLSC (best)	TQ LS MSI DSLSC (fixed)	Free Tree	PWC	JBIG	CAE
akiyo, 300	288	236	246	**219**	227	228	229	290	344	385
logo, 297	21	14	**13**	14	27	27	21	51	44	98
kids, 300	754	703	714	690	692	695	**660**	753	823	906
dest.l1, 114	237	222	217	**214**	223	225	218	262	283	237
dest.l2, 78	57	41	41	38	38	**38**	52	82	86	41
dest.l3, 124	759	751	757	756	558	558	**507**	781	792	1213
robot, 131	504	478	474	458	**410**	414	435	505	512	563
explosion, 18	203	180	178	157	**132**	134	148	201	200	360
dest.l7, 103	269	257	255	250	**245**	247	252	301	326	267
rain, 169	2184	2179	2199	2199	1628	**1628**	1695	2227	2248	3380
avg.shapes	557	531	537	525	452	453	**451**	574	602	781
buildings	943	950	976	896	**682**	683	792	962	893	1395
land	426	396	410	371	249	**249**	320	416	444	765
water	552	527	528	480	**288**	289	392	514	457	974
road,yellow	731	708	718	667	414	**414**	456	663	619	1354
road,white	3586	3577	3719	3340	2379	2379	**2140**	3492	3487	5206
road,pink	382	353	318	260	217	**217**	248	366	348	681
avg.maps	1103	1085	1112	1002	705	**705**	725	1069	1041	1729
CCITT01	13527	13592	13091	13144	13345	13345	**10724**	12829	12728	16878
CCITT02	7860	8002	7875	7759	7709	7710	**6420**	7878	7910	10136
CCITT08	13561	13939	13492	13344	13340	13363	**10260**	13049	13158	17045
avg.CCITT	11649	11844	11486	11416	11465	11472	**9135**	11252	11265	14686

Table 1: The proposed algorithm compared with other context based binary image encoders. Code lengths are expressed in bytes/frame. Ideal code lengths are used for the first 6 columns. The notation (best) indicate the best result among all the possible sizes of the template, while (fixed) indicates that the internal size of the template K is set to 5 for the binary shapes, and 10 for the maps and the CCITT images. Only not uniform frames are considered for the shapes. The number of these frames is indicated for each sequence, and used as weight in the final average.

proposed algorithm is very efficient on the bi-level layers of the digital map being 34% better than PWC, 32% better than JBIG, 59% better than CAE, and nearly 3% more efficient than the Free Tree coder.
As expected the proposed algorithm is fairly robust but not competitive on the tested CCITT images which are outside the scope of the method.

7 Conclusions

A new efficient algorithm for coding binary shapes is presented. Modeling the object boundaries as digital straight line segments is used for improving context based coding of binary shapes. An efficient variation of skip-innovation codes is also derived in a similar manner. The algorithm obtained provides efficient coding results on binary shapes and is also very efficient on binary layers of digital maps, outperforming PWC, JBIG and CAE. The respective bitrate reductions are 21%, 25%, 42% on binary shapes, and 34%, 32%, 59% on digital maps. Application of the straight line analysis is in part motivated by the fact that for a digital straight line the per symbol entropy asymptotically converges to zero, while the low order entropy generally is high.
Directions for future work are implementation of an encoder/decoder using arithmetic coding, fast and fuzzy versions, and the development of variants more specifically targeted to digital maps or to a larger class of images. The longer term perspective is the integration of the algorithm in an object based video encoder.

Appendix A. LZ78 coding of digital straight lines

Consider a digital straight line specified by given slope α and offset e (in a given octant). For 8-connected elements the number of different DSLS of length k is (at most) $k+1$ [2]. So obviously the rate should converge to zero asymptotically. A simple analysis is carried out for the classic universal LZ78 [16] coding scheme applied to the binary sequence of chain elements. A scenario providing an upper bound on the per symbol code length is provided by assuming that for all k all $k+1$ distinct DSLS of length k are entered into the dictionary before a string of length $k+1$ is coded. In this case the number of elements parsed when all the DSLS strings up to length k are entered in the dictionary is

$$n(k) = \sum_{i=2}^{k} i(i+1) > \frac{k^3}{3}, \tag{9}$$

evaluating the sum by a corresponding integral. The corresponding number of entries when all the DSLS strings up to length i are entered in the dictionary is

$$N(i) = \sum_{j=1}^{i} j + 1 = \frac{(i+2)(i+1)-2}{2}. \tag{10}$$

Combining these equations, the code length for the entries is bounded by

$$L(k) \leq \sum_{i=2}^{k}(i+1)(\log(N(i))+2) = 2\sum_{i=2}^{k} i\log(i) + o(i\log(i)). \tag{11}$$

Evaluating the sums by integrals it is seen that the code length bound increases as $k^2 \log k$ for $k \to \infty$ and therefore the upper bound for the rate, $\frac{L(k)}{n(k)}$, converges to 0 for $n \to \infty$ as

$$\frac{k^2 \log k}{n} < \frac{\log n}{(3n)^{1/3}}. \tag{12}$$

This convergence (of the upper bound) is slower than what the number of DSLS lines gives but the coding is sequential and without the use of knowledge of the DSLS properties in the coding process.

References

[1] P. J. Ausbeck Jr., "The piecewice-constant image model", *Proceedings IEEE*, vol. 88, no. 11, Nov. 2000, pp. 1779-1789.

[2] L. Dorst and W.M. Smeulders, "Discrete representation of straight lines", *IEEE Trans. Pattern Analysis Machine Intell.*, vol. 6, no. 4, July 1984, pp. 450-463.

[3] S. Forchhammer and O.R. Jensen, "Content layer progressive coding of digital maps", *IEEE Trans. Image Proc.*, vol. 11, no. 12, Dec. 2002, pp. 1349-1356.

[4] 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, Nov. 1998, pp. 838-848.

[5] A.K. Katsaggelos *et al.*, "MPEG-4 and reate-distortion-based shape-coding techniques", *Proc. IEEE*, vol. 86, no. 6, June 1998, pp 1126-1154.

[6] J. Koplowitz *et al.*, "The number of digital straight lines on an NxN grid", *IEEE Trans. Inform. Theory*, vol. 36, no. 1, Jan. 1990, pp. 192-197.

[7] V.A. Kovalevsky, "New definition and fast recognition of digital straight segments", *Proc. Pattern Recognition, 10th Int'l. Conf. on*, 1990, pp. 31-34.

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

[9] ISO/IEC 14492 CD, "Coded Representension of Picture and Audio Information - Lossy/Lossless coding of bi-level images (JBIG2)", *ISO/IEC JTC1/SC29/WG1*

[10] B. Martins and S. Forchhammer, "Tree coding of bilevel images", *IEEE Trans. Image Processing*, vol. 7, no. 4, Apr. 1998, pp. 517-528.

[11] D. L. Neuhoff and K. G. Castor, "A rate distortion analysis of chain codes for line drawings", *IEEE Trans. Inform. Theory*, vol. 31, no. 1, Jan. 1985, pp. 53-68.

[12] P. Nunes, F. Marques, F. Pereira, A. Gasull, "A contour-based approach to binary shape coding using a multiple grid chain code", *Signal Processing: Image Communication*, vol. 15, no. 7-8, 2000, pp. 585-599.

[13] A. Rosenfeld and R. Klette, "Digital straightness", *Electronic Notes in Theoretical Computer Science* vol. 46, 2001, 32 pp. (http://www.elsevier.nl/locate/entcs/volume46.html)

[14] H. Wang, G.M. Schuster, A.K. Katsaggelos, "Object-based video compression scheme with optimal bit allocation among shape, motion and texture", *Proc. ICIP'03*, Barcelona, Spain, Sept. 2003, 4 pp.

[15] H. Wang, G.M. Schuster, A.K. Katsaggelos, T.N. Pappas, "An efficient rate-distortion optimal shape coding approach using a skeleton-based decomposition", *IEEE Trans. Image Processing*, vol. 12, no. 10, Oct. 2003, pp. 1181-1193.

[16] J. Ziv and A. Lempel, "Compression of individual sequences via variable-rate coding", *IEEE Trans. Inform. Theory*, vol. 24, no. 5, Sept. 1978, pp. 530-536.