ON THE SIZE AND SHAPE OF MULTI-LEVEL CONTEXT TEMPLATES FOR COMPRESSION OF MAP IMAGES

Eugene Ageenko, Pavel Kopylov, Pasi Fränti

Department of Computer Science, University of Joensuu, PB 111, FIN-80101 Joensuu, Finland.

ABSTRACT

We present a method for estimating optimal context templates that are used for conditioning the pixel probabilities in context-based image compression. The algorithm optimizes the location of the context pixels within a limited neighborhood area, and produces the ordered template as a result. The ordering can be used to determine the shape of the context template for a given template size. The optimal template size depends on the size of the image, when the template shape depends on the image type. We apply the method for the compression of multi-component map images consisting of several semantic layers represented as binary images. We estimate the shape of the context-template for each layer separately, and compress the layers as generic regions using standard JBIG2 compression technique.

1. INTRODUCTION

In an image, pixels form geometrical structures with appropriate spatial dependencies. Dependencies can be localized to a limited neighborhood defined by a local template. Statistical context-based image compression utilizes spatial dependencies in the image by conditioning the pixel probabilities on the combination of neighboring pixel values. The compression consists of two distinct phases: statistical modeling and arithmetic coding [1].

In the modeling phase, the probability distribution of the pixel is dynamically estimated. The pixel configuration in the template determines the context, and in this way, the model to be used for pixel compression, see Fig. 1. Here, the pixel configuration 1110010010_2 gives the model index 914 of 1024, which is the number of contexts possible with this template. In adaptive image compression, the model is constructed dynamically during the encoding. It starts from equal initial distribution and is updated after each pixel has been processed.
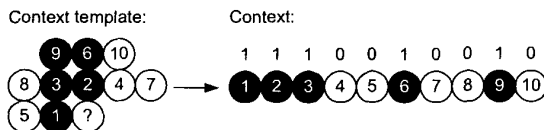
Context template: Context:

Fig. 1. Example of a 10-pixel template and context.

Arithmetic coding assigns optimal code for the pixels in regards to the given statistical model [2]. The code size can be estimated by the information content of the model measured as the entropy [3]. The examples of context-based image

compression are the latest international standards, JBIG (*Joint Bi-level Image Group*) and JBIG2 [4-7].

Theoretically, better probability model can be constructed using a larger context template. In practice, however, the use of larger templates does not always result in compression improvement [8]. The number of contexts grows exponentially with the size of template; adding one more pixel to the template doubles the size of the model. This can lead to excessive memory consumption. Another disadvantage is *learning cost* problem, which is the coding deficiency in the early stage of compression. It is because the model must adapt to the statistics of the image before becoming efficient. In addition, *context dilution* problem may occur if the statistics are distributed over too many contexts, thus affecting the accuracy of the probability estimates.

Optimal template size depends on the image size. The location of the template pixels, on the other hand, has no direct effect on the learning cost but, if properly designed, may greatly improve the accuracy of the model. It is therefore feasible to optimize the location of the template pixels for the compressed images. Usually, the pixels are distributed in the neighborhood using the principle of minimal distance to the current pixel. Standard 1-norm or 2-norm distance functions define two different templates shown in Fig. 2 [9]. These templates are well suited for mixed type images. However, they are not necessary the best choices for images of a specific type. The images in question are the multi-component map images. The images consist of several binary layers with different semantic content. Each layer consists of geometrical structures that do not necessarily match to the structures of another layer. The layers can be combined and displayed to the user as a color image.

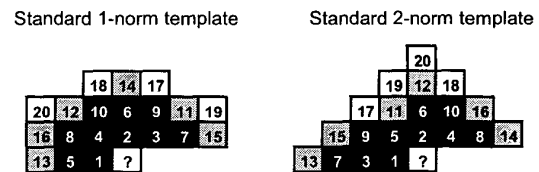Standard 1-norm template   Standard 2-norm template

Fig. 2. Standard orderings for the context templates.

In this paper, we propose a method for optimizing the context template for a given image. The method optimizes the location of the template pixels within a limited neighborhood area, and produces the ordered template as the result. The ordering can then be used to derive the context template for any given template size. We apply the method in a static manner for the compression of multi-component map images. The template is optimized for each layer separately using a training image.

```
ConstructTemplate (int k_MAX, int array SearchTemplate[ ])
begin
    k ← 0;
    repeat
        k ← k + 1;
        for each i that SearchTtemplate [i] ≠ OCCUPIED
        begin
            CollectStatistics(i);
            l(i) ← CalculateCodeLength (i);
        end
        Choose j that l(j) = min_i l(i);
        ContextTemplate [k] ← j;
        SearchTemplate [j] ← OCCUPIED;
    until (k = k_MAX);
    return ContextTemplate [ ];
end.
```

Fig. 3. Algorithm for estimating an optimal context template.

Fig. 4. The neighborhood area used for optimizing the location of the template pixels. The 'x'-marked position corresponds to the compressed pixel location, which is marked by '?'

Fig. 5. Illustration of the context template construction.

2. TEMPLATE CONSTRUCTION

2.1. Single-layer template

The optimal context template can be solved for a given template size k by compressing the image using all possible templates and selecting the one with the best compression performance. However, this is not computationally possible as there is huge number of different template configurations to be tested. Therefore we take a more practical approach. We construct the template stepwise, optimizing the location of one pixel at a time.

The sketch of the algorithm is shown in Fig. 3. The algorithm starts with an empty context template and expands it by one pixel at a time. We repeat by adding a new pixel to each unoccupied location in the neighborhood area. We use the 40-pixel neighborhood shown in Fig. 4 on left. For each candidate pixel location, we make a pass over the input image, construct the statistical model and calculate image entropy. The image entropy is computed as the average entropy for all image pixels:

$$H_{image} = -\frac{1}{n}\sum_{i=1}^{n}\log_2 p_i ,$$

where p_i is the probability of i-th pixel and n is the total number of pixels in the image [3]. The entropy gives the optimal number of bits required for encoding a single pixel with a given model. The probability of a pixel is calculated on the basis of the observed frequencies using a Bayesian sequential estimator:

$$p_i = \begin{cases} p_W^i(C) = \dfrac{n_W^i(C)+\delta}{n_W^i(C)+n_b^i(C)+2\delta}, & \text{if } i^{th} \text{ pixel is } white \\ p_B^i(C) = 1 - p_W^i(C), & \text{if } i^{th} \text{ pixel is } black \end{cases}$$

Here $p_W^i(C)$, $p_B^i(C)$ are the probabilities for white and black colors of i-th pixel, respectively, conditioned on the pixel context C; $n_W^i(C)$, $n_B^i(C)$ are the pixel counters, which start
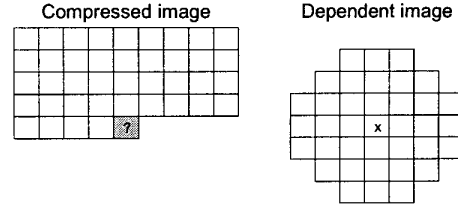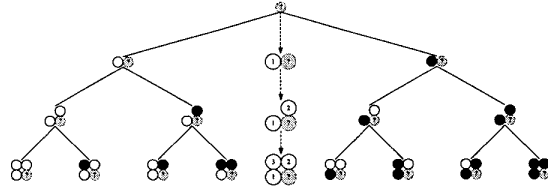
from zero and are updated after the pixel has been processed; and $\delta = 0.45$ due to JBIG [4].

After computing the entropies, we select the pixel location providing minimum entropy, and add it permanently to the context template. The process continues until the context template size reaches the predefined maximum k_{MAX}. The template construction process is illustrated in Fig. 5.

The result of the algorithm is not only the final template of k_{MAX} pixels but also the ordering of the pixels. From the ordering we can derive all possible templates of the size up to k_{MAX}. The size of the context template is a parameter of the compression method, and it mainly depends on the size of the image. For example, the map images are very large and therefore relatively large templates can be applied without the risqué been weighed down by the learning cost and context dilution problems.

The proposed method can be applied in two alternative manners: *static* and *semi-adaptive*. In the static approach, as taken here, we optimize the template using a priori knowledge of the image type. This is possible, as we know the type of the images to be compressed. The advantage of this approach is that the time-consuming optimization can be done off-line. In the semi-adaptive approach, the template is optimized for the compressed image and is stored in the compressed file. This would be a better solution when the image type is not known beforehand. The compression process, however, would be very slow, which makes this approach not suitable for real-time applications.

2.2. Multi-Layer template

The idea of *multi-layer* context template is to utilize the information from additional image, referred here as to *dependent image*. The restriction on use of dependent image in compression is that it must have already been coded, so that both encoder and decoder would have the same information. The difference
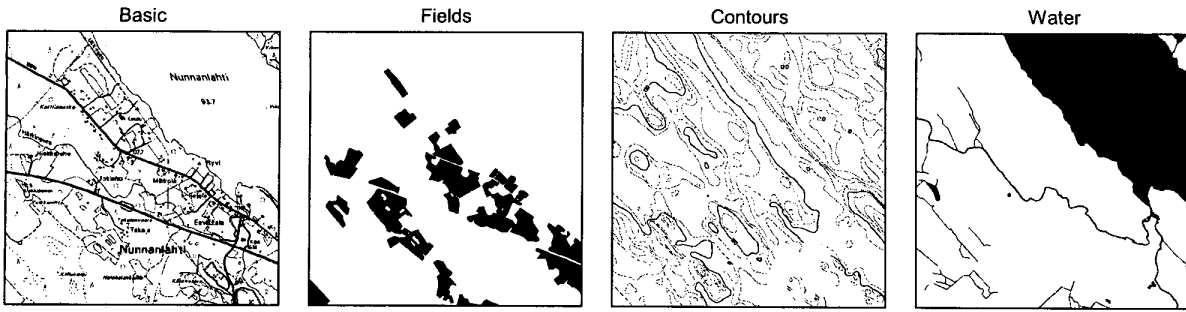
459

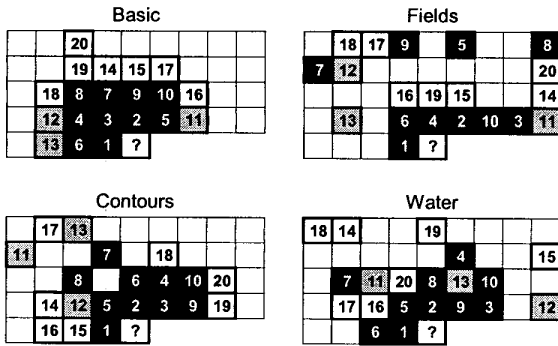**Fig. 6.** Sample 1000 x 1000 pixels fragments of the layer images.

Fig. 7. Optimized context templates for the semantic layers.
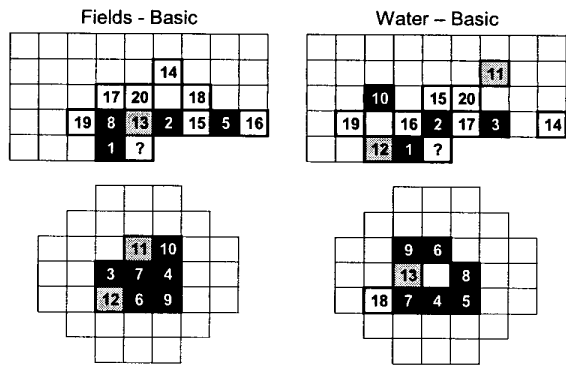

Fig. 8. Optimized two-level context templates.

in construction of *multi-layer* and *single-layer* context templates is in additional neighborhood mask used for selection of the pixels from dependent image. The combined neighborhood area amounts to 77 pixels and is shown in Fig. 4

3. EXPERIMENTS

We evaluate the proposed method by compressing a set of map images from the NLS topographic database, in particular Basic map series 1 : 20,000 [10]. Each image is of the size 5000×5000 pixels, and represents a 10×10 km² area. The image consists of four binary layers with different semantic meaning: *basic* (topographic data), *fields* (solid polygonal regions), *contours* (thin lines representing elevations levels), *water* (solid regions and poly-lines representing water areas and ways), see Fig. 6.

The image layers are compressed using JBIG2 technique in generic mode [6]. One image not participating in compression tests is used as a training image for estimating the context templates. Objectives of the evaluation are to determine the compression performance using the constructed context templates in comparison to the standard, 1-norm and 2-norm templates.

The templates constructed for the different semantic layers using the proposed algorithm are shown in Fig. 7. Two-level context templates using *basic* layer as dependent image are shown in Fig. 8. The pixel ordering is illustrated by numbering. The first ten context pixels are tinted in black, and the next six in gray. The compression results separated for each layer are

illustrated in Fig. 9. The results are obtained by varying the template size from 1 to 20.

The resulting templates have different shapes corresponding to the geometrical structures of the images. The *basic* layer includes wide variety of different elements: text, solid lines of different width, and single pixel dots. The optimized template and compression results are therefore close to ones of JBIG2.

The *fields* layer containing merely solid areas has a different template, in which the nearest pixels are enough to predict the existence of a field. Additional pixels are chosen far away from the compressed pixel location. The optimized template improves the compression of the fields by about 12 %, on average. The simplicity of the structures also means that relatively small template sizes are sufficient for this type of images.

Contours layer consists of elevation lines, which are one or two pixels wide solid or dashed contours. There are no single pixels or larger structures in these images. *Water* layer contains also contour lines but they are always two or more pixels wide. In addition to that, there are larger black areas representing water areas. The optimized context templates for these two types of images are similar, and they provide moderate improvement in the compression.

The compression of the *fields* ate *water* layers can be further improved by utilizing the pixels of *basic* layer in the context template. It is explained by the presence of the contours of field and water regions in the basic layer. The effective compression improvement for these two layers is up to 50 %.
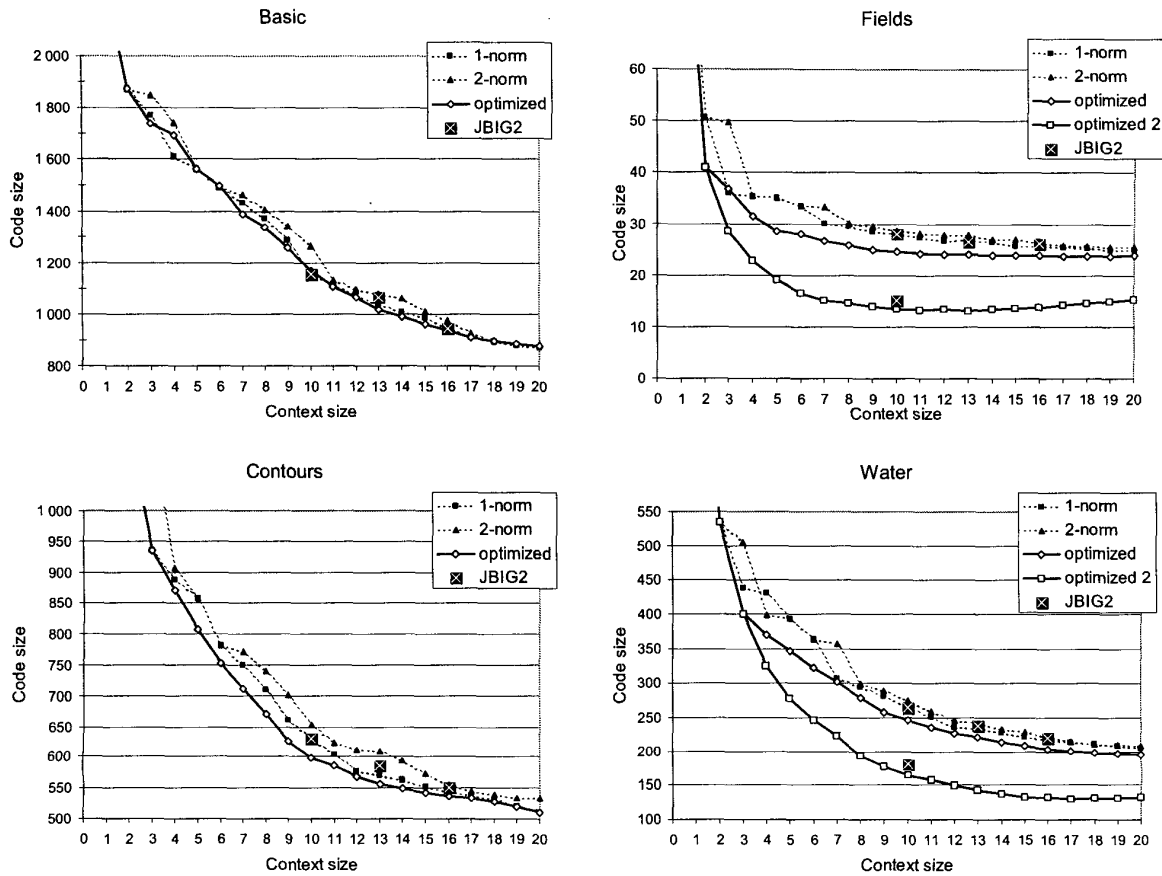
460

Fig. 9. The total sizes of the compressed files in kilobytes for map layers using JBIG2 compression in generic mode with various context templates: *1-norm*, *2-norm*, *optimized*, and standard JBIG2 templates (10, 13, and 16 pixels). The results for multi-level context template (marked as *optimized 2*) and 10-pixel two level template of JBIG2 are also given for *fields* and *water* images.

4. CONCLUSION

A method for optimizing context templates for a given image was introduced. The algorithm optimizes the location of the context pixels within a limited neighborhood area, and produces the ordered template as a result. It was shown that the optimized templates could be quite different for different types of images. The method can be applied for the compression of multi-component map images, and moderate compression improvement was obtained for a set of map images.

5. ACKNOWLEDGMENT

6. REFERENCES

[1] Rissanen J.J., Langdon G.G., "Universal modeling and coding," *IEEE Trans. Inform. Theory*, IT-27, 12-23, 1981.

[2] Rissanen J.J. and Langdon G.G. "Arithmetic coding," *IBM Journal of Research and Development*, 23, 146-162, 1979.

[3] Shannon C.E., "A mathematical theory of communication," *Bell System Tech Journal*, 27, pp. 398-403, 1948.

[4] ISO/IEC International Standard 11544, 1993.

[5] Haskell B.G. et. al. "Image and video coding – emerging standards and beyond," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 8 (7), pp. 814-837, 1998.

[6] Final committee draft for ISO/IEC International Standard 14492, 1999; http://www.jpeg.org/public/jbigpt2.htm

[7] P.G. Howard et. al., "The emerging JBIG2 standard," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 8 (7), pp. 838-848, 1998.

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

[9] Martins B., Forchhammer S., "Bi-level image compression with tree coding," *IEEE Trans. Image Processing*, vol. 7 (4), pp. 517-528, 1998.

[10] National Land Survey of Finland, Opastinsilta 12 C, PB 84, 00521 Helsinki, Finland. http://www.nls.fi/index_e.html.