Codebook Clustering by Self-organizing Maps for Fractal Image Compression

Institut für Informatik — Report 75[1]

Raouf Hamzaoui
Institut für Informatik, Universität Freiburg
Am Flughafen 17, 79110 Freiburg, Germany
Email: hamzaoui@informatik.uni-freiburg.de.

January 11, 1996

[1]This paper is based on a talk held at the NATO Advanced Study Institute *Fractal Image Encoding and Analysis*, in Trondheim, Norway, in July 1995. It also appears in a special issue of the journal *Fractals*.

Abstract

A fast encoding scheme for fractal image compression is presented. The method uses a clustering algorithm based on Kohonen's self-organizing maps. Domain blocks are clustered, yielding a classification with a notion of distance which is not given in traditional classification schemes.

1 Introduction

The time complexity of the encoding is one of the major drawbacks of fractal image compression. Each of a large number of image subsets, called ranges, has to be compared sequentially to a large number of other image subsets called domains. In his original approach, Jacquin [1] used a classification scheme to reduce the number of comparisons. Blocks were classified according to their perceptual geometric features. For a given range block, only domain blocks within the same class were considered. But since only 3 classes were differentiated, the encoding was still very slow. A more elaborate classification technique based on the intensity and the variance of the blocks was used by Fisher et al. [2, 3]. It provided a variable number of classes (3-24-72). However, no notion of distance between classes was defined so that one cannot search in a neighboring class if a good match was not found in the selected class. After pointing out that the best match for a range block is the domain block whose projection in a given subspace is mostly parallel to it, Lepsøy and Øien [4, 5] proposed a clustering algorithm for classifying blocks. A set of cluster centers was designed with the LBG algorithm [6]. Blocks were then classified according to their degree of parallelity to the cluster centers. Although the results were encouraging (a speed up factor of 8 at a loss of 0.68 dB with regard to full search is reported), the method presents some problems. The LBG algorithm is very sensitive to the initial cluster centers. Moreover, a cluster center was not updated by the centroid of the cluster as required normally in the LBG algorithm.

In this paper, we present a similar approach. However, we use neighborhood relationships instead of angular ones. This allows us to simplify the problem considerably. We show, for example, that the method is well suited to a large class of clustering algorithms.

It has been shown by Saupe [7] that the matching problem in fractal image compression can be expressed as a nearest neighbor queries in a feature space. Thus, clustering algorithms in which neighboring vectors are grouped in the same (or neighboring) clusters are the natural way to provide the best solution to the matching problem. Clusters are formed by grouping vectors around their corresponding nearest neighbors in the set of cluster centers. The encoding consists of searching matches inside the same cluster or in the neighboring ones. Finding optimal cluster centers is solved by a design of a vector quantization codebook. Since in our main application the initial cluster centers are computed from the test image and trained with vectors coming also from the test image, we used Kohonen's self-organizing maps [8, 9] for their ability to achieve good results when using a small number of training vectors, and when given an arbitrary initial codebook [10].

In the next section, we explain Kohonen's algorithm. In Section 3, the notation used in the paper is presented and the fundamental result of Saupe is recalled. Section 4 explains our implementation and compares it to previous schemes. We conclude with some remarks and indicate how our results could be improved.

2 Kohonen's self-organizing maps

Kohonen's self-organizing maps may be used to find clusters in a set of data vectors stemming from a stochastic source $x \in \mathbf{R}^n$. The algorithm starts by assigning to every node i of a regular array an initial cluster center $m_i(0) \in \mathbf{R}^n$. At step t, a vector $x(t)$ is taken from a training set $\{x(t) \in \mathbf{R}^n,\ t = 0, \dots, T\}$ and compared to all cluster centers $m_i(t)$. Let $c = \arg\min_i ||x(t) - m_i(t)||$. Then $m_c(t)$ and all cluster centers associated to nodes in the neighborhood of c are moved closer to the input vector $x(t)$ by the following learning rule

$$m_i(t+1) = m_i(t) + \alpha(t) h_{ci}(t)[x(t) - m_i(t)], \tag{1}$$

where the function $\alpha(t),\ 0 \leq \alpha(t) < 1$ is monotonically decreasing and $h_{ci}(t)$ is a decreasing function of both the grid-distance between nodes i and c and step t. The algorithm returns cluster centers m_i obtained as convergence limits of the learning process. These limit vectors are hoped to globally minimize the average expected distortion measure[1]

$$E = \int \sum_{i \in L} h_{ci} ||x - m_i||^2 p(x) dx, \tag{2}$$

where L denotes the set of indices of the array nodes and $p(x)$ the probability density function of the source x. Note that if h_{ci} is taken as the Kronecker

[1]We assume in the following that $h_{ci}(t)$ is independent of t. Thus, $h_{ci}(t)$ is denoted by h_{ci}.

delta, then E is the average squared error distortion used in classical vector quantization. The exact optimization of (2) is an unsolved problem. However, the best approximative solution is based on the Robbins-Monro stochastic approximation and is given precisely by (1) [9].

3 Notations and mathematical background

For the discussion in the paper let us assume that a sampled image is partitioned into non-overlapping square blocks of size $N \times N$ called range blocks. This is not a restriction since it will be clear how the principles described carry over to more general partitions.

We consider each range block as a vector R in the linear vector space $\mathbf{R}^n$ where $n = N \times N$. The conversion from a square subimage of side length N to a vector of length $n = N^2$ can be accomplished, e.g., by scanning the block line by line. Working with vectors in place of 2D-arrays simplifies the notation considerably without losing generality.

The domain pool is a collection of square blocks which are typically larger than the ranges and taken also from the image, called domain blocks. The domain pool is enlarged by including blocks obtained after applying the eight isometries of the dihedral group of the square to the domain blocks. Finally, by pixel averaging, the size of these blocks is reduced to the size of a range block. The resulting blocks are called codebook blocks. In the encoding process, for a range block $R \in \mathbf{R}^n$ a search through the codebook blocks $D_1, \ldots, D_{N_D} \in \mathbf{R}^n$ is required. We let $E(D_i, R)$ denote the least squares error of an approximation of the range block R by an affine transformation of the codebook block D_i. In terms of a formula, this is

$$E(D_i, R) = \min_{a,b \in \mathbf{R}} \|R - (aD_i + bC)\|,$$

where C is the block of constant intensity $C = (1, \ldots, 1)/\sqrt{n}$. The codebook block D_i which gives the smallest error $E(D_i, R)$ is selected on condition that the value of the scale factor a for the codebook block D_i ensures the convergence of the decoding process (e.g., by requiring $|a| < 1$). Now let O be the orthogonal projection operator which projects $\mathbf{R}^n$ onto the orthogonal complement $\mathcal{C}^\perp$, where $\mathcal{C}$ is the linear span of C. If $Z = (z_1, \ldots, z_n) \in \mathbf{R}^n \backslash \mathcal{C}$, we define the operator

$$\phi(Z) = \frac{OZ}{||OZ||}. \tag{3}$$

Keeping the same notations we have the theorem:

Theorem 1 [7]
Let $n \geq 2$ and $X = \mathbf{R}^n \backslash \mathcal{C}$. Define the function $\Delta : X \times X \to [0, \sqrt{2}]$ by

$$\Delta(D, R) = \min\left(||\phi(R) + \phi(D)||, ||\phi(R) - \phi(D)||\right).$$

For $D_i, R \in X$ the least squares error $E(D_i, R)$ is given by

$$E(D_i, R) = \langle R, \phi(R)\rangle\, g(\Delta(D_i, R))$$

where

$$g(\Delta) = \Delta\sqrt{1 - \frac{\Delta^2}{4}}.$$

The theorem states that the minimization of the least squares errors $E(D_i, R)$ for $i = 1, \ldots, N_D$ is equivalent to the minimization of the distance expressions $\Delta(D_i, R)$. It follows that we may replace the computation and minimization of N_D least squares errors $E(D_i, R)$ by the search for the nearest neighbor of $\phi(R) \in \mathbf{R}^n$ in the set of $2N_D$ vectors $\pm\phi(D_i) \in \mathbf{R}^n$.

4 Description of the method

4.1 The encoding algorithm

Let $\{\pm\phi(D_1), \ldots, \pm\phi(D_{N_D})\}$ be the set of projected codebook blocks. We want to partition this set into a finite number of disjoint subsets (clusters) defined by representatives (cluster centers) such that vectors in the same cluster are closer to each other than vectors in different clusters. The quality of the clustering can be measured by a criterion function that one tries to optimize. For example, one can choose to construct the cluster centers such that the sum of squared Euclidean distances $J = \sum_{i=1}^{2N_D} ||x_i - m(x_i)||^2$ is minimized. Here $m(x_i)$ denotes the cluster center closest to the projected codebook block x_i. A cluster of center m is formed by grouping around m all projected codebook blocks having m as their nearest neighbor. With

Kohonen's algorithm (see equation 2), more than one cluster center is considered for each vector x_i. Iterative optimization algorithms are frequently used to find optimal partitions. Unfortunately, they guarantee only local optimization.

After the cluster centers have been designed, the set of projected codebook blocks $\{\pm\phi(D_1), \ldots, \pm\phi(D_{N_D})\}$ is clustered by mapping each vector $\pm\phi(D_i)$ to its nearest cluster center. A range block R is encoded in two steps. First, we map its feature vector $\phi(R)$ to its closest cluster center $m(\phi(R))$. Then the range block R is compared only to the codebook blocks whose feature vectors are in the cluster of center $m(\phi(R))$. This corresponds to a 1-class search. We can evidently search in more classes by considering the next nearest cluster centers of $\phi(R)$. This will yield more accurate encodings at the expense of increased time. The reason why the method works is obvious. Suppose that both $\phi(D_i)$ (or $-\phi(D_i)$) and $\phi(R)$ are close enough to cluster center m. Then the triangular inequality ensures that $\Delta(D_i, R)$ is small enough. Thus, by Theorem 1, codebook block D_i will provide a good match for range block R.

4.2 A comparison to Fisher's classification scheme

Our implementation is based on Fisher's adaptive quadtree algorithm [3] and Kohonen's self-organizing map program package [11]. Fisher's classification scheme works as follows. For every square block $B \in \mathbf{R}^n$ there exists a unique isometry $I_B : \mathbf{R}^n \rightarrow \mathbf{R}^n$ (corresponding, when block B is seen as a 2D-array, to one of the 8 isometries of the dihedral group of the square) that transforms B such that the average pixel intensities $A_i,\ i = 1, 2, 3, 4$ of the four quadrants (upper left, upper right, lower left and lower right) of its 2D-array representation are ordered in one of the three canonical positions:

$$\begin{aligned}
&\text{Major class 1: } A_1 \geq A_2 \geq A_3 \geq A_4,\\
&\text{Major class 2: } A_1 \geq A_2 \geq A_4 \geq A_3,\\
&\text{Major class 3: } A_1 \geq A_4 \geq A_2 \geq A_3.
\end{aligned}$$

Once the isometry I_B has been determined, there are 24 different possible orderings of the variances of the four quadrants which define 24 subclasses for each major class. Thus, there are 72 classes in all. If the scaling factor a is negative, then the orderings in the classes have to be modified accordingly. Hence, for a given range, each domain is classified in two orientations.

This means that two subclasses need to be searched, one for positive scaling factors and one for negative scaling factors. This classification allows one to search in 1, 3, 24 or 72 classes, the latter corresponding to a full search encoding. Contrary to the usual approach, a domain is compared to a given range in only two orientations. For example, for positive scaling factors, the orientation corresponds to the isometry $I_R^{-1} \circ I_D$.

Now, since we wanted a fair comparison of our classification to the one presented above, we had to use the same codebook and the same number of classes. The first requirement was fulfilled by using the same domain pool and determining the major class of the ranges and domains because for a given range the orientation of the domain is only known if the major class of each of the two blocks is specified. The second requirement was simply met by designing with the SOM program a set of 72 cluster centers corresponding to the nodes of a 12×6 rectangular array. The initial cluster centers were formed from the set $G = \{\phi(I_{D_i}(D_i)), i = 1, \ldots, N_D\}$ by the linear initialization procedure [9, 11]. The use of a random initialization is also possible. The training phase consisted of 10000 steps in which α decreased linearly from 0.1 to 0. The training vectors were taken randomly from the set G. The function h_{ci} was equal to 1 if the distance between the winning node c and node i was less than 1 in the topology of the array, and h_{ci} was equal to 0 otherwise. Finally, each vector $\phi(I_{D_i}(D_i))$ was mapped to its nearest cluster center. To encode a range block R, we defined three options:

1. Option 1: Only positive scalings. If we wanted to obtain only positive scaling factors and if doing a 1-class search, we looked for the nearest cluster center of $\phi(I_R(R))$ and retained the domain block D minimizing the least squares error $\min_{a,b\in\mathbf{R}} \|R - (aI_R^{-1}(I_D(D)) + bC)\|$ inside the cluster. For an s-class search, the clusters corresponding to the s nearest neighbors of $\phi(I_R(R))$ were considered.

2. Option 2: Both positive and negative scalings. We also considered negative scalings by looking for the nearest cluster center of $\phi(I_{-R}(-R))$. Thus, as in Fisher's classification scheme, for a 1-class search two searches are carried out for each range: one search in the cluster whose center is the nearest neighbor of $\phi(I_R(R))$ and a second one in the cluster whose center is the nearest neighbor of $\phi(I_{-R}(-R))$.

3. Option 3. With this option, and for an s-class search, we considered the s clusters corresponding to the s smallest values of the set

$$\{d(\phi(I_R(R)), m_1), d(\phi(I_{-R}(-R)), m_1), \ldots, d(\phi(I_R(R)), m_{72}), d(\phi(I_{-R}(-R)), m_{72})\},$$

   where m_i denotes a cluster center.

Figure 1 shows the results of encoding the 512 by 512 Lenna image with a fixed range size (4×4). The domain pool consisted of 4096 nonoverlapping blocks having twice the range size. The PSNR was computed without post-processing. All the times reported are measured on an SGI Indigo2 running an R4400 processor. The two curves correspond to options 2 (Both) and 3 (Best) and the boxes to Fisher's classification scheme. We make the following observations:

1. As expected, increasing the number of classes searched improved the PSNR. The increase was largest when a 2-class search replaced the 1-class search. Thus, most of the range blocks had their best matches in either the first or the second nearest cluster.

2. Our classification was clearly more accurate than Fisher's one. For example, searching in one class with option 2 made us lose 0.70 dB as compared to full search while a 1-class search with Fisher's classification scheme lost 1.91 dB to full search.

3. To approach the full search fidelity, we needed to search in only a few number of classes, namely, 3 or 4. A comparable performance was attained with a 24-class search with Fisher's classification scheme.

4. The compression ratio was equal to 4.74 . Higher compression ratios can be obtained by using blocks of different sizes as explained below.

Figure 2 shows the PSNR as a function of encoding time for the same tests as in Figure 1. We remark the following:

1. Our encoding scheme needed less time to find good matches. For example, an encoding fidelity of 37.05 dB was obtained in 95 seconds with option 2 while it took 758 seconds to reach 37.04 dB with Fisher's scheme.

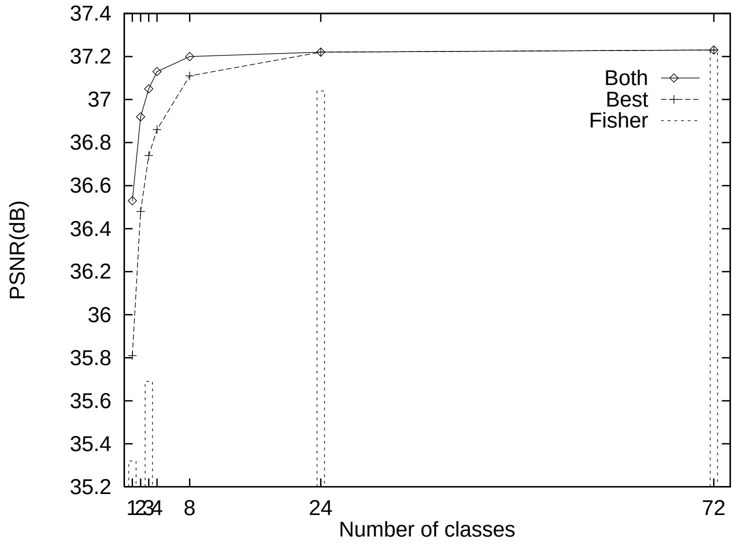

Figure 1: PSNR vs. number of classes for the 512 by 512 Lenna image.

2. For the considered domain pool, option 2 was able to reduce the complexity of the full search by a factor of 20.07 at a loss of 0.18 dB.

Figure 3 shows the number of domains in each of the 72 clusters. Figures 4 and 5 show the results of similar tests for the 256 by 256 San Francisco image.

In our study the optimal number of clusters has not been investigated. Figure 6 gives the PSNR vs. time curve when varying the number of cluster centers.

Our classification can be applied to blocks of variable sizes enabling a quadtree-based encoding scheme. Ranges and domains of all quadtree levels are put in their canonical position. Then, the size of the obtained blocks is reduced to 16 by pixel averaging. The cluster centers are computed from the set of projected domain blocks $\bigcup_{p=l}^{l'} F_p$, where $F_p = \{\phi(I_{D_i^p}(D_i^p)), i = 1, \ldots, N_{D_p}\}$. Here l and l' denote respectively the minimum and the maximum quadtree level, and N_{D_p} denotes the number of domain blocks D_i^p at quadtree level p. Finally, at every level of the quadtree, we encode the range blocks as explained before. Thus, searching is carried out for the 16-dimensional feature vectors but computations of the least squares errors are done for blocks at the real size. Down-filtering all domains and ranges to di-

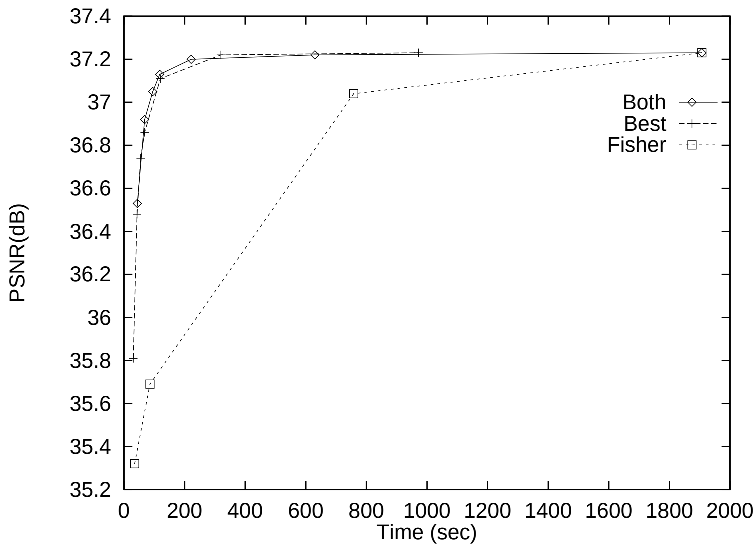

Figure 2: PSNR vs. time for the 512 by 512 Lenna image. Dots are obtained by varying the number of classes searched.

mension 16 has the advantage of reducing the memory requirements for the feature blocks and speeding up the computations involved in the algorithm. However, it implies that the success of the method is no more rigorously justified by Theorem 1.

As suggested by Øien [12, 5], one may use fixed cluster centers computed from a set of several images to avoid the preprocessing time involved in the computation of adaptive cluster centers. Figure 7 shows the rate-distortion curve of the following schemes: Fisher, option 2, option 3 and option 3 with precomputed cluster centers (Fix). All curves correspond to a 3-class search. The maximum range size was 32×32 and the minimum was 4×4. The domain pool consisted of nonoverlapping subsquares having twice the range size. The compression ratio was varied by changing the rms tolerance [3]. Figure 8 shows the time vs. compression curves for the same series of tests. Our method was also successful when blocks of variable sizes were used. Since our classification was more accurate than Fisher's one, good matches were found at a higher level of the quadtree permitting better compression ratios. Note how encoding with fixed cluster centers outperformed all other encodings in speed efficiency while keeping the same image quality. The fact

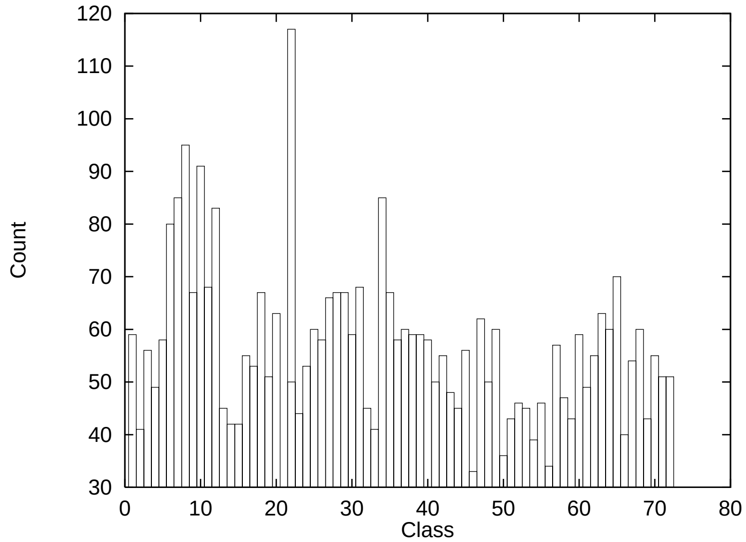

Figure 3: Number of domains in each of the 72 clusters for the 512 by 512 Lenna image.

that these centers were not computed from the test image did not seem to hinder the success of the method. Further tests with other images confirmed this result.

4.3 Some comments on previous schemes

The first approach to clustering with self-organizing maps for fractal image compression was proposed by Meadows and Bogdan [13]. The reported results were, however, not satisfying. The offset b was not used, maybe because it would have required the normalization given by Theorem 1, which has not been known by the authors. A large network consisting of 1600 nodes was employed. This resulted in long preprocessing time. The training step, done on line, took 210 minutes on a Sun Sparc 2!

Lepsøy and Øien [4, 5] showed that the codebook block that minimizes the least squares error $E(D_i, R)$ is the one that maximizes $\langle OR, \phi(D_i)\rangle^2$. The criterion function for the clustering was defined as

$$J = \sum_{k=1}^{M} \sum_{i=1}^{I_k} \langle OR_{k,i}, m_k \rangle^2, \tag{4}$$

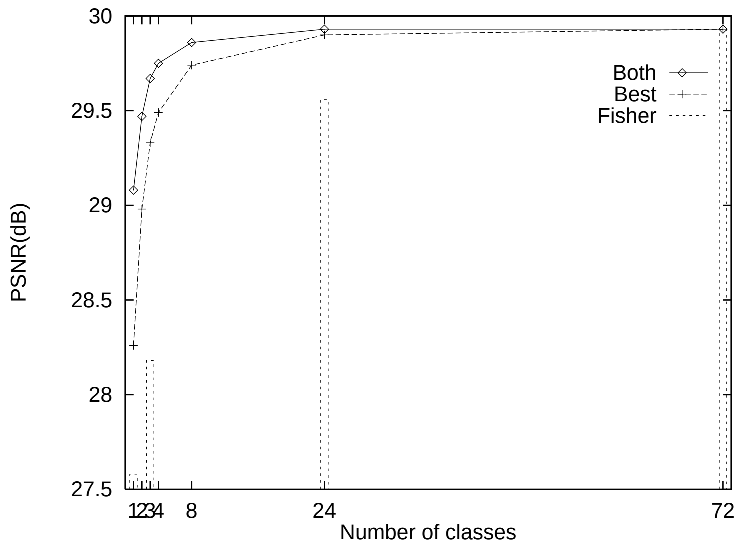

Figure 4: PSNR vs. number of classes for the 256 by 256 San Francisco image. Ranges had the fixed size (4×4). The domain pool consisted of 1024 nonoverlapping blocks.

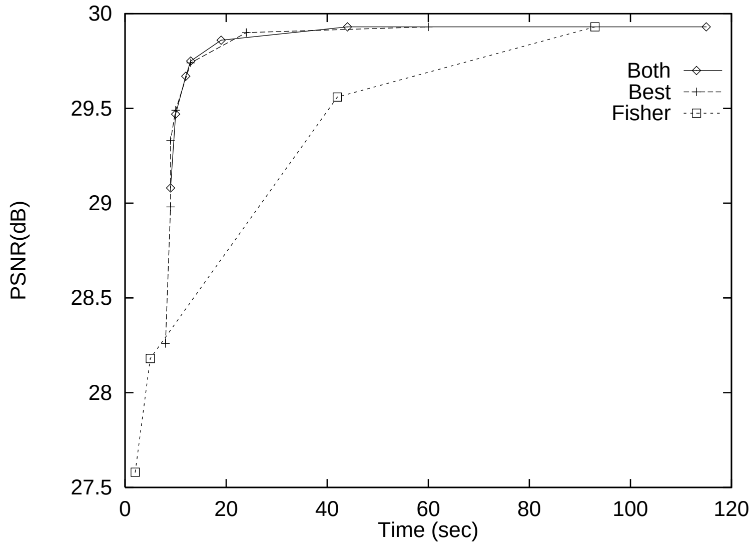

Figure 5: PSNR vs. time for the 256 by 256 San Francisco image.

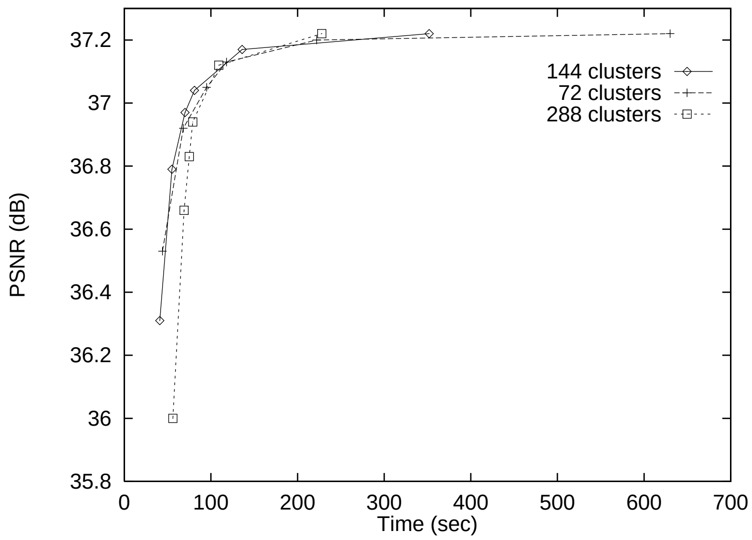

Figure 6: PSNR vs. time for the 512 by 512 Lenna image. Maps of various sizes are considered (72, 144, 288). Encoding is with option 2. Dots correspond to the number of classes searched (1,2,3,4,8,24).

for $||m_k|| = 1$, where M is the number of cluster centers m_k, and I_k is the number of range blocks $R_{k,i}$ in the cluster with index k. Contrary to our approach, the range blocks and not the codebook blocks are involved in the computation of the cluster centers. The LBG algorithm is used to find cluster centers that maximize the criterion function. Given a range block R, the *nearest neighbor condition* consists of maximizing $\langle OR, m_k \rangle^2$. For a given cluster, the *centroid condition* requires finding cluster center m that maximizes $\sum_{i=1}^{I_k} \langle OR_{k,i}, m \rangle^2$. As the computations necessitated in fulfilling the centroid condition were too heavy, l_2-norms were replaced by l_1-norms. Furthermore, even for l_1-norms, optimal solutions were not found. In the following, we show how using neighborhood relationships makes the computation of the centroid straightforward when the LBG algorithm is used for the clustering. If we assume that the range blocks are the training vectors, then the criterion function can be taken as

$$J = \sum_{k=1}^{M} \sum_{i=1}^{I_k} ||\phi(R_{k,i}) - m_k||^2. \tag{5}$$

The nearest neighbor condition consists of minimizing the distortion measure

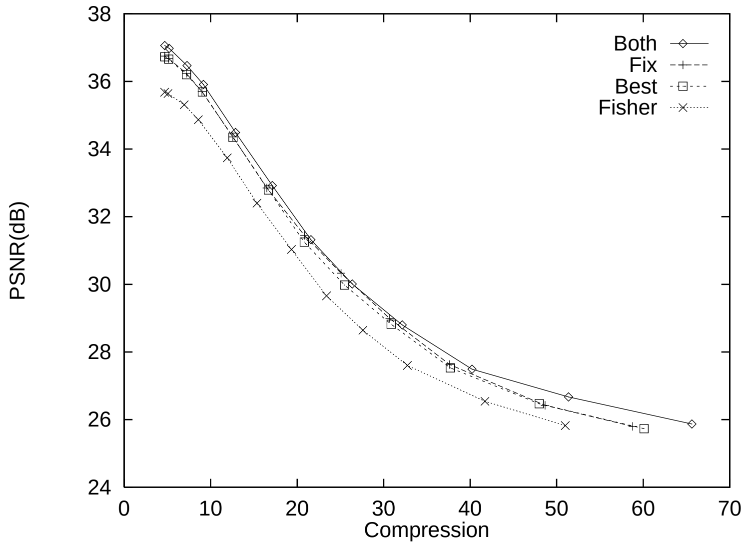

Figure 7: PSNR vs. compression for the 512 by 512 Lenna image. The curve denoted by Fix corresponds to encodings with a set of 72 fixed cluster centers designed from the four images San Francisco, Peppers, Baboon and Collie. The training consisted of 100000 steps.

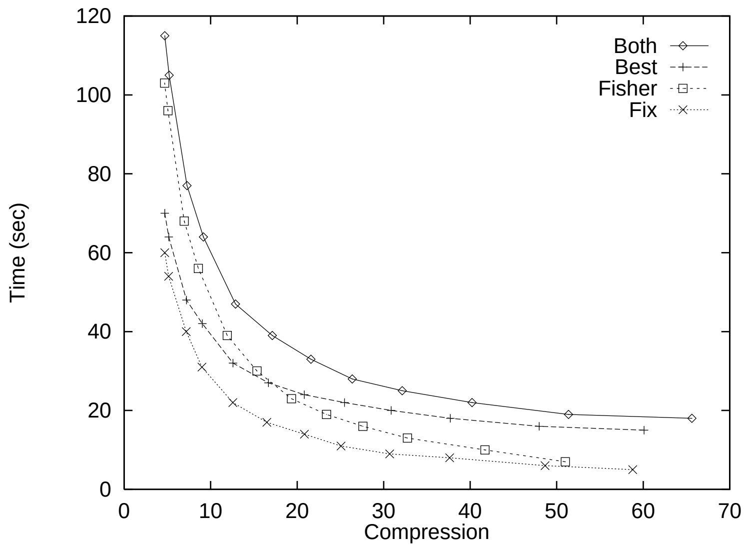

Figure 8: Time vs. compression for the 512 by 512 Lenna image.

$||\phi(R) - m_k||^2$, and the centroid of the cluster with index k reduces to the arithmetic average

$$\frac{1}{I_k}\sum_{i=1}^{I_k}\phi(R_{k,i}). \tag{6}$$

Another method using fixed cluster centers computed from a set of training images which does not include the test image is the archetype classification presented by Boss and Jacobs [14]. A cluster center (called archetype) for a set of training codebook blocks is defined as that particular codebook block that can best cover all others in the usual least squares sense. For a set of blocks D_i this is the block D_k,

$$D_k = \arg\min_{D_k}\sum_{i\neq k}\min_{a,b}||D_i - (aD_k + bC)||.$$

For a given test image to compress, the classification is done by assigning each range block and domain block to the class of the archetype that covers it best, i.e, for each block the least squares approximation is minimized over the set of archetypes. This differs from our approach in two aspects. Firstly, an archetype is a member of the training set whereas our cluster centers are not since they are optimized by a clustering algorithm. Secondly, classifying with our scheme is less expensive than with the archetypes because it requires computations of Euclidean distances instead of least squares approximations.

5 Conclusions and future work

In this paper we have introduced a new classification scheme based on a notion of distance between blocks and cluster centers designed with Kohonen's self-organizing maps. The use of Kohonen's algorithm is particularly advantageous due to its capability to rapidly generate a high quality clustering. Of course, other clustering algorithms may be employed as well.

The experimental results showed that when using blocks of fixed size, the maximum fidelity was almost retained while the encoding speed was drastically improved. The classification was also implemented in an adaptive quadtree fractal scheme. It provided rate-distortion performance superior to previously published ones [3].

The results described here were obtained with a code which is not optimal in terms of speed. Our main purpose was to show the benefits of our clustering. The two programs [3, 11] used in our tests were merged through a pipe command. Integrating the clustering in the main program will evidently speed up the encodings. An additional reduction of the time complexity can be obtained by applying a nearest neighbor search technique for both finding the nearest cluster centers and the best candidates inside the clusters. Efficient algorithms are well known in the literature [6, 15].

6 Acknowledgements

The author would like to thank Dietmar Saupe, Martin Müller and Klaus Ries for helpful discussions. Martin Müller adapted and merged the codes used in our simulations.

References

[1] A. E. Jacquin, *Image coding based on a fractal theory of iterated contractive image transformations,* IEEE Trans. Image Processing **1** (1992) 18–30.

[2] E. W. Jacobs, Y. Fisher and R. D. Boss, *Image compression: A study of the iterated transform method,* Signal Processing **29** (1992) 251–263.

[3] Y. Fisher, *Fractal image compression with quadtrees,* in: *Fractal Image Compression — Theory and Application,* Y. Fisher (ed.), Springer-Verlag, New York, 1994.

[4] S. Lepsøy, *Attractor Image Compression: Fast Algorithms and Comparisons to Related Techniques,* PhD Thesis, The Norwegian Institute of Technology, Trondheim, Norway, June 1993.

[5] S. Lepsøy and G. E. Øien, *Fast attractor image encoding by adaptive codebook clustering,* in: *Fractal Image Compression — Theory and Application,* Y. Fisher (ed.), Springer-Verlag, New York, 1994.

[6] A. Gersho and R. M. Gray, *Vector quantization and signal compression,* Kluwer Academic Publishers, 1992.

[7] D. Saupe, *Breaking the time complexity of fractal image compression,* Technical Report 53, Institut für Informatik, Universität Freiburg, 1994.

[8] T. Kohonen, *The self-organizing map,* Proceedings of the IEEE **78,9** (1990) 1464–1480.

[9] T. Kohonen, *Self-Organizing Maps,* Springer-Verlag, 1995.

[10] J. D. Mc Auliffe, L. E. Atlas and C. Rivera, *A comparison of the LBG algorithm and Kohonen neural network paradigm for image vector quantization,* in: *Proceedings of ICASSP-1990 IEEE International Conference on Acoustics, Speech and Signal Processing,* **Vol. 4**, pp. 2293–2296, 1990.

[11] The Self-Organizing Map, Version 3.1, Helsinki University of Technology, Laboratory of Computer and Information Science, April 7, 1995.

[12] G. Øien, L_2*-Optimal Attractor Image Coding with Fast Decoder Convergence,* PhD Thesis, The Norwegian Institute of Technology, Trondheim, Norway, April 1993.

[13] A. Bogdan and H. E. Meadows, *Kohonen neural network for image coding based on iteration transformation theory,* in: *Proceedings from SPIE Neural and Stochastic Methods in Image and Signal Processing,* **Vol. 1766,** pp. 425–436, 1992.

[14] R. D. Boss and E. W. Jacobs, *Archetype classification in an iterated transformation image compression algorithm,* in: *Fractal Image Compression — Theory and Application,* Y. Fisher (ed.), Springer-Verlag, New York, 1994.

[15] C.-H. Lee and L. H. Chen, *Fast closest codeword search algorithm for vector quantization,* IEE Proc.-Vis. Image Signal Process. **141,3** (1994) 143–148.