FRACTAL-BASED TECHNIQUES FOR A GENERALIZED IMAGE CODING METHOD

Mohammad Gharavi-Alkhansari and Thomas S. Huang

Beckman Institute for Advanced Science and Technology
University of Illinois at Urbana-Champaign, Urbana, Illinois, U.S.A. 61801

ABSTRACT

This paper presents a new generalized image block coding algorithm which covers fractal techniques, block transform techniques, and vector quantization as its special cases. The coding is performed by approximating each image block with a linear combination of a series of (not necessarily orthogonal) blocks selected from a pool of basis blocks. This pool is made up of (1) a set of fixed basis blocks, (2) a set of blocks taken from the filtered, subsampled image, and (3) a set of blocks taken from the image without any change of scale. The index of the selected basis blocks and their corresponding coefficients make the code for each range block. Methods are proposed for making the pool and selecting blocks from the pool.

1. INTRODUCTION

Block-based methods are the most commonly used methods for image compression. The most well-known of these are block transform, vector quantization, and fractal-based coding.

In block transform coding methods like JPEG, each block in an image is encoded by approximating it with a linear combination of a series of linearly independent (and usually orthogonal) fixed basis blocks (FBBs). In these methods, the set of basis blocks is the same for all of the blocks being encoded and is also usually independent of the image. These methods are usually not adaptive in the sense that they cannot take into account image structures like repetitions or similarities between blocks that are located at different parts of the image. The basis blocks cannot change from one image to another or from one block to another.

On the other hand, vector quantization methods are designed for a class of images. They have codebooks that are designed such that they contain blocks that can suitably approximate any block in the class of images for which it is designed. In vector quantization, each block of an image is approximated by a single block from the codebook.

So, in contrast to block transform coding, which uses a linear combination of blocks to approximate a given block, VQ uses a single block (and possibly two if we use dc component and consider it as a separate block), which is of course chosen from a larger set of blocks. Although in standard VQ the codebook is designed for a class of images, it is typically not adaptive to single images and blocks because it is not efficient to send a new codebook for every image or every block in an image.

Fractal-based methods are typically based on the work by Barnsley [1] who proposed to use fractal properties of natural images for image compression. Based on Barnsley's work, Jacquin [2, 3] developed an algorithm for automatic compression of images. The work of Barnsley and Jacquin on *Iterated Function Systems (IFS)* and *Recurrent Iterated Function Systems (RIFS)* [1, 2, 4, 5] has made a basis for development of a series of fractal-based methods by other researchers, for compression of both still images [6, 7, 8, 9, 10, 11, 12] and image sequences [13, 14, 15].

The essence of most fractal-based coding methods is to approximate a range block by a linear combination of up to a few FBBs (for dc and low frequency components) [3, 6, 7, 14], and a single other block made by applying some contractive transformation on a larger domain block in the same image (for encoding details of the range block). Fractal-based methods are similar to VQ[1] in the sense that they use a single block from a large pool of blocks to approximate the details of the block. But fractal-based methods are adaptive because the codebook can be different from one image to another, and in some methods even different from one range block to another range block, without the need for sending the codebook each time. This adaptability brings the better potential for image compression. However, fractal-based methods have the problem that the codebook, although very adaptive, is not very controllable, and the codebook is limited to what exists in the image (and possibly to the neighborhood of the range block) or simple transformed versions of them. This pool is not always capable of providing a good approximation to highly detailed or unique range blocks in the image.

It is notable that as fractal-based methods approximate each part of an image by another part, they can use global relationships better than VQ or block transform coding (e.g. DCT).

At each step of encoding, if the fractal-based methods selected the domain blocks from the parts of the image that were already encoded, then they could be classified as a kind of adaptive block prediction coding where the prediction parameters are being sent to the receiver, and there was no contractivity restriction on the transformations that were applied to the domain blocks. However, the encoding process is typically non-causal, i.e., the domain block can be selected from parts of the image that are not yet encoded. This makes a more variety of blocks available for encoding a range block, and also makes the encoding quality of all

[1] For a comparison between VQ and Jacquin's method, see [3].

the range blocks uniform. But for the image to be reconstructable at the decoder side, this non-causality needs to be accompanied with the followings,

- The transformations that are applied on the domain blocks need to be contractive.
- The decoding process needs to be iterative.
- Some additional error is introduced during the decoding process. (An upper limit to this error is given by the Collage Theorem.)

Then it is possible to reconstruct the image at the decoder side by an iterative method [3] where the convergence is guaranteed by the Contraction Mapping Theorem.

In this paper, using fractal-based techniques, we propose a new generalized method for image compression, in which,

1. To get more control on how a range block is encoded, we allow linear combination of the FBBs with more than one domain block to approximate range blocks. (The use of multiple domain blocks has only recently been studied in [8, 9].)
2. We allow domain blocks to be of a size greater than *or equal to* that of the range blocks, and also remove the restriction of contractivity of transformations applied to domain blocks when all domain blocks are selected in a causal way.
3. We use a larger variety of FBBs with no restriction on their orthogonality.

These considerations, combined together, lead to a generalized method which treats FBBs and the adaptive basis blocks (ABBs) very similarly, and has the block transform methods, VQ methods, and most of previously developed fractal-based methods as its special cases. The following sections will describe this approach in more detail.

2. THE NEW METHOD

2.2. Encoding Process

In our method, the image is first partitioned into non-overlapping square range blocks of size $S_R \times S_R$. Each range block is considered as an $N = S_R^2$ dimensional vector. For each range block, a pool of M basis blocks[2] is made where M is typically larger than or equal to N, but in general it can be any positive integer. A small number of these basis blocks are then chosen such that their linear combination gives a good approximation of the range block. Then a least squares method is used to find the the coefficients of the chosen basis blocks. The index of these chosen basis blocks and their corresponding coefficients constitute the code for the range block.

It is notable that the pool of basis blocks can be different from one range block to another, and the range-block-dependent part of the pool is neither necessary nor sent for the decoding process and is not part of the code.

[2]In the literature, the expression "basis vectors" usually refers to linearly independent vectors that span a space. However, in this paper, basis blocks (vectors) are not necessarily linearly independent or complete, but we use the expression basis blocks because of the way we use them to approximate range blocks.

2.2.1 Making the Pool of Basis Blocks

For each range block, the pool of the basis blocks is made up of the following subsets,

1. *Adaptive Basis Blocks (ABB)*:
   This set contains basis blocks that are generated by applying some transformation T on domain blocks in a neighborhood of the range block. This set is range-block dependent, i.e., it may be different from one range block to another. The set of ABBs itself consists of two subsets,
   (a) *Higher Scale Basis Blocks (HSBB)*:
   These are blocks that are generated by shrinking domain blocks of size $S_D \times S_D$ of the image ($S_D > S_R$) to give $S_R \times S_R$ basis blocks. The set may also be complemented by adding rotated or reflected versions of the above blocks.
   (b) *Same Scale Basis Blocks (SSBB)*:
   These are blocks that are directly taken from the image (with no shrinking) and are restricted to be in parts of the image that are already encoded (i.e. selected causally). This subset may again be complemented by adding rotated or reflected versions of themselves.
2. *Fixed Basis Blocks (FBB)*:
   This set contains a series of blocks which are independent of the range block being encoded and is designed by the encoder designer. The set of FBBs is the same for all the range blocks in the image being encoded, and must be sent to the encoder offline. The presence of these basis blocks in the pool can serve the following purposes,
   - The encoder can use shorter codes for the indices of some of these blocks that are commonly present in the range blocks in the form of a strong component.
   - Enable the decoder to encode more accurately range blocks which are very different from other blocks in the image and therefore cannot be well-encoded by other subsets of the pool.
   - Allow the linear combination of basis blocks to be a contractive transformation.

2.2.2 Finding the Best Set of Basis Blocks

For each range block, after the set of basis blocks is constructed, we look for the smallest number of basis blocks that, when linearly combined, can approximate the range block within a given small error range. We look for the minimum number of basis blocks, because this results in the shortest code for the range block. In this selection, we may put a limit on the maximum number of basis blocks which may be used from either subsets of the basis blocks.

The above problem is an integer programming optimization, and as our basis blocks are not orthogonal, finding the absolute optimum seems to be a rather difficult problem[3].

[3]In fact this problem can be classified as a planar point location search in the field of combinatorial geometry.

However, we can use a suboptimal solution to this problem. We propose two different solutions:

1. Choose the basis block which has the strongest correlation coefficient (highest absolute value) with the range block. Then remove any component of its form from the range block and repeat this process for the residual of the range block with the rest of basis blocks until the residual becomes smaller than a threshold or until no other basis block has significant correlation with the residual range block.
2. Same as first method, with the difference that after a basis block is selected, remove any component of its form not only from the range block, but also from all other basis blocks before repeating the process.

   It is easy to prove that at each single step, this method can never perform worse than the first method in reducing the norm of the residual of the range block. However, this does not guarantee a better performance in the over all optimization.

   Also it is interesting to note that in this method, the computations needed for selecting the basis blocks from the pool, include most of the computations that are necessary for finding the coefficients of the selected basis vectors after they are selected. The least squares can be done by a QR decomposition of the matrix whose columns are the selected basis vectors. If we denote the vector of the range block to be encoded by b and the vector of the coefficients of the basis blocks by x, then x can be found from

$$Rx = Q^T b$$

   (see p. 238 in [16]). The upper triangular matrix R and the vector $Q^T b$ are computed directly in the process of selecting the basis blocks. Therefore x can be found with a small number of computations after the basis blocks are selected. This is specially helpful when the number of selected basis blocks is large.

We refer to the above methods as *decomposition without basis orthogonalization* and *decomposition with basis orthogonalization* accordingly.

The following points are notable in our proposed method for encoding process with selection method of either decomposition with or without basis orthogonalization.

- Most other fractal-based image coding methods are close to special cases of this method where maximum number of FBBs is set to 1 (for dc component), or a small number, and no SSBBs are allowed.
- If the maximum number of ABBs is set to 0 and the maximum number of FBBs is set to $N = S_R^2$, and the FBBs are selected orthogonal, this method reduces to a block transform coding.
- On the other hand if the maximum number of ABBs is again set to 0 and the maximum allowed number of FBBs is set to 1, then this method reduces to a vector quantization method.

2.3. Decoding Process

The decoding algorithm, in the most general case, is similar to the one proposed by Jacquin [2, 3] and is based on the Contraction Mapping Theorem. We begin with any initial image, and for each range block bring the blocks from the image that have the same address as the selected domain blocks, and apply the corresponding transformations on these blocks to make an approximation of the selected ABBs. Then, these approximated basis blocks and the selected FBBs are multipled by their corresponding coefficients and are added together to make an approximation of the range block. This process is repeated for all range blocks until the resulting image does not change significantly with iterations.

The convergence of the decoding process is proven only for the case where the maximum allowed number of ABBs is 1 and their combination coefficients are less than 1. However, experimental results suggest that the decoder converges even when the maximum number of allowed ABBs is greater than 1 [8].

As mentioned before, the SSBBs are restricted to be chosen causally from the image, but the HSBBs are not. An interesting case occurs if we restrict the HSBBs to be also chosen causally. Then the whole encoding system becomes causal and the decoding process needs only a single iteration to converge and there are no restrictions on the coefficients of the basis blocks (except due to possible numerical stability issues).

2.4. Application to Image Sequences

The above method can be easily extended to encoding of image sequences. The extension can be done in either of the following two ways,

1. By considering a sequence of images as a three dimensional set and use 3D blocks instead of 2D blocks. In this case, 3D transform coding becomes a special case of the proposed method where only FBBs are allowed for encoding image 3D blocks.
2. By considering a sequence of images as a series of two dimensional images and for the range blocks of any image, allow the domain blocks to be chosen from the neighborhood of the range block which includes parts (or possibly all) of neighboring frames.

   In fact, in this case, block prediction methods like DPCM, and adaptive block prediction methods like block motion compensation [17], become special cases of our algorithm, where only one SSBB (with its coefficient restricted to 1) and no HSBBs are allowed. Also hybrid coding methods [17] like motion compensation combined with transform coding of residual errors (which is equivalent to DPCM of transform coefficients of blocks shifted according to motion vectors), is equivalent to using the basis blocks of the transform coding as FBBs and allowing one SSBB.

3. EXPERIMENTAL RESULTS

The encoding algorithm proposed in this paper was applied to natural images. The settings of the parameters of encoder are as follows,

Table 1: Parameter settings and results for the second set of experiments (CPU times are in seconds)

Total # of basis blocks	Number of FBBs	Number of SSBBs	Number of HSBBs	Lena			Peppers			Sailboat on Lake		
				rms error	Av. # blocks	CPU time	rms error	Av. # blocks	CPU time	rms error	Av. # blocks	CPU time
64	6	25	33	4.91	5.59	17	5.46	5.61	16	6.45	17.79	44
64	64	0	0	4.96	5.15	15	5.11	5.94	17	5.46	13.09	36
128	6	0	122	4.87	5.49	34	5.05	5.09	31	5.69	15.94	86
128	6	61	61	4.83	4.19	28	5.08	4.43	27	5.77	12.67	75
128	6	122	0	4.90	4.62	31	5.27	5.57	34	5.88	12.33	71
128	64	0	32	4.90	4.42	20	5.27	5.57	35	5.40	11.99	51
128	64	16	16	4.90	4.22	20	5.05	4.69	21	5.39	11.73	52
128	64	32	0	4.93	4.47	21	5.10	5.23	25	5.41	11.77	51
256	6	0	250	4.82	4.25	59	4.96	4.24	55	5.39	12.16	147
256	6	125	125	4.78	3.46	52	4.95	3.90	53	5.28	10.42	131
256	6	250	0	4.86	3.92	57	5.17	4.77	64	5.76	10.20	131
256	64	0	192	4.82	3.78	54	4.95	3.85	53	5.36	10.12	131
256	64	96	96	4.79	3.43	51	4.94	3.68	52	5.32	9.59	126
256	64	192	0	4.86	3.75	56	5.03	4.51	62	5.31	9.77	128
512	6	0	506	4.77	3.66	222	4.92	3.79	214	5.35	10.03	373
512	6	253	253	4.74	3.22	220	4.91	3.49	211	5.26	8.95	354
512	6	506	0	4.81	3.53	229	5.09	4.36	234	5.61	8.95	355
512	64	0	448	4.78	3.49	220	4.92	3.61	215	5.34	9.23	417
512	64	224	224	4.75	3.13	213	4.90	3.37	145	5.61	8.95	399
512	64	448	0	4.80	3.39	222	4.98	4.13	231	5.26	8.73	352

- $S_R = 8$ and $S_D = 16$.
- Two different sets of FBBs were used in the experiments, only one of which is used in each experiment.

  The first set is made up of six mutually orthogonal blocks basically of the form $z = 1$, $z = x$, $z = y$, $z = x^2 - a$, $z = y^2 - a$, and $z = xy$, where the z variable represents the pixel value and the origin of the x and y coordinates is the center of the block. a is a constant and its value depends on the size of the block.

  In the second set, the 64 orthogonal basis blocks of DCT were used.

  In both cases, no restriction is put on the maximum number of FBBs used.
- SSBBs are taken from a square with the range block located at its center, while avoiding parts of this square that are not encoded yet. The size of this square depends on the number of SSBBs. If parts of this square fall out of the image, those parts are not used.
- HSBBs are again taken from a square (with possibly a different size) with the range block located at its center. The size of the square depends on the number of HSBBs. If parts of this square fall out of the image, those parts are not used.
- No pixel shufflings are applied on the ABBs.
- Step size for bringing basis blocks from the image is 1.
- Minimum rms of the residual of the range block for stopping selection of basis blocks is set to 6.

In one set of experiments, the encoding process was applied to the 256×256 standard "Lena" image with both methods of decomposition without basis orthogonalization, and decomposition with basis orthogonalization for choosing basis blocks. In these experiments, the set of 6 FBBs were used, with 81 HSBBs and no SSBBs. For the case of decomposition without basis orthogonalization, an average of 10.14 basis blocks were needed for encoding range blocks, resulting in an encoding rms error of 5.16 (PSNR 34 dB). For the case of decomposition with basis orthogonalization, an average of 8.88 basis blocks were needed for encoding range blocks resulting in an encoding rms error of 5.10 (PSNR 34 dB). These results show that the method of decomposition with basis orthogonalization gives a shorter code for almost the same PSNR for the 256×256 Lena image.

In another set of experiments, the encoding process was applied to 512×512 "Lena", "Peppers", and the "Sailboat on the Lake" standard images using only decomposition with orthogonalization. The settings for these experiments and their results are shown in Table 1. In this table, the "Total # of basis blocks" represents the size of the pool (M) from which the basis blocks were selected, the "Av. # blocks" represents the average number of basis blocks that were selected from the pool of basis blocks and were used for approximating range blocks. The CPU time is the processing time (in seconds) used for performing the encoding on a Hewlett-Packard Apollo Series 735 workstation. As it can be seen from the table, in the experiments, for cases with same total number of basis blocks, the differences in rms error resulting from the encoding process is small (due

to a fixed rms thresholding of 6). But considering the average number of basis blocks used for encoding each range block, the results suggest that,

- In most of the cases, using a combination of HSBBs and SSBBs gives a better performance compared to using HSBBs alone or SSBBs alone.
- Complementing the 64 DCT FBBs with the HSBBs and SSBBs reduces the average number of basis blocks that are needed for encoding range blocks, when compared with using DCT FBBs alone. But when the total number of basis blocks is fixed at 64, using the 64 orthogonal DCT FBBs gives a better performance compared to combining 6 FBBs with HSBBs and SSBBs.
- Except when the total number of basis blocks is small (64), there is no clear advantage in using either of the two sets of 6 FBBs or 64 DCT FBBs over the other.

4. CONCLUSIONS

In this paper we presented a new generalized image compression method. The block transform coding, standard VQ, and most of the earlier fractal compression methods can be considered as special cases of this method. In this method, the fixed basis blocks are complemented with a set of adaptive basis blocks. These additional basis blocks are constructed from the image itself; some from larger scales and some from the same scale as of the range block. The proposed method exploits self-similarities of image both at different scales and at the same scale. Two methods were tested for solving the discrete optimization problem of choosing the smallest number of basis blocks that can closely approximate each range block. It was found that the method of decomposition with basis orthogonalization gives a better performance compared to the method of decomposition without basis orthogonalization. Also, results of using two different sets of fixed basis blocks, and also using a combination of same-scale basis blocks and higher-scale basis blocks were given.

5. ACKNOWLEDGEMENT

This work is supported in part by Joint Services Electronics Program Grant N00014-90-5-1270 and in part by a grant from Mitsubishi Electric.

6. REFERENCES

[1] M. F. Barnsley and H. Rising III, *Fractals Everywhere.* Boston: Academic Press Professional, second ed., 1993.

[2] A. E. Jacquin, *A Fractal Theory of Iterated Markov Operators with Applications to Digital Image Coding.* PhD thesis, Georgia Institute of Technology, Aug. 1989.

[3] A. E. Jacquin, "Image coding based on a fractal theory of iterated contractive image transformations," *IEEE Transactions on Image Processing*, vol. 1, pp. 18–30, Jan. 1992.

[4] M. F. Barnsley and A. E. Jacquin, "Application of recurrent iterated function systems to images," in *Proceedings of the SPIE, Visual Communications and Image Processing*, vol. 1001, pp. 122–131, 1988.

[5] M. F. Barnsley and L. P. Hurd, *Fractal Image Compression.* Wellesley, Massachusetts: AK Peters, Ltd., 1993.

[6] G. E. Øien, S. Lepsøy, and T. A. Ramstad, "An inner product space approach to image coding by contractive transformations," in *Proceedings of IEEE ICASSP-91*, pp. 2773–2776, May 14–17, 1991.

[7] D. M. Monro, "A hybrid fractal transform," in *Proceedings of IEEE ICASSP-93*, vol. V, (Minneapolis, Minnesota), pp. 169–172, Apr. 27–30, 1993.

[8] M. Gharavi-Alkhansari and T. S. Huang, "A fractal-based image block-coding algorithm," in *Proceedings of IEEE ICASSP-93*, vol. V, (Minneapolis, Minnesota), pp. 345–348, Apr. 27–30, 1993.

[9] G. Vines, *Signal Modeling with Iterated Function Systems.* PhD thesis, Georgia Institute of Technology, Atlanta, GA, May 1993.

[10] L. Thomas and F. Deravi, "Pruning of the transform space in block-based fractal image compression," in *Proceedings of IEEE ICASSP-93*, vol. V, (Minneapolis, Minnesota), pp. 341–344, Apr. 27–30, 1993.

[11] Y. Fisher, "Fractal image compression," in *SIGGRAPH '92 Course Notes: From Folk Art to Hyperreality* (P. Prusinkiewicz, ed.), pp. 1–21, 1992.

[12] Y. Fisher, ed., *Fractal Image Compression: Theory and Application.* New York: Springer-Verlag, 1995.

[13] L. P. Hurd, M. A. Gustavus, and M. F. Barnsley, "Fractal video compression," in *Digest of Papers. Thirty-Seventh IEEE Computer Society International Conference (COMPCON)*, pp. 41–42, Feb. 24–28, 1992.

[14] H. Li, M. Novak, and R. Forchheimer, "Fractal-based image sequence compression scheme," *Optical Engineering*, vol. 32, pp. 1588–1595, July 1993.

[15] Y. Fisher, D. Rogovin, and T. P. Shen, "Fractal (self-VQ) encoding of video sequences," in *Proceedings of the SPIE, Visual Communications and Image Processing*, (Chicago, Illinois), Sept. 25–28, 1994.

[16] S. J. Leon, *Linear Algebra with Applications.* New York: Macmillan Publishing Company, third ed., 1990.

[17] H. G. Musmann, P. Pirsch, and H.-J. Grallert, "Advances in picture coding," *Proceedings of IEEE*, vol. 73, pp. 523–548, Apr. 1985.