# Iterative-Parametric Image Models For Video Coding


## Author

Le, XS, Gonzalez, R



## Published

2005

## Conference Title

ISSPA 2005: THE 8TH INTERNATIONAL SYMPOSIUM ON SIGNAL PROCESSING AND ITS APPLICATIONS, VOLS 1 AND 2, PROCEEDINGS

## DOI

10.1109/ISSPA.2005.1580193

# ITERATIVE-PARAMETRIC IMAGE MODELS FOR VIDEO CODING

*Xuesong Le, Ruben Gonzalez*
School of Computing and Information Technology, Griffith University
Xuesong.Le@student.griffith.edu.au, r.gonzalez@griffith.edu.au

**ABSTRACT**

This paper proposes a novel method for compressing digital image sequence based on iterative and parameterisable image models. By using proposed image models, the reconstructed image approximates the input image progressively by repeatedly segmenting each residual image obtained from previous iteration and the input image into objects or domains. Parameters characterising segmented domains are then encoded and transmitted. The proposed image model utilises product codes in an attempt to further improve the quality of the input image. Finally, applications of proposed images models are extended to video coding, which exploit similarity between successful frames, in addition to self-similarity within each frame.

## 1 INTRODUCTION

One of the major problems in current video coding standards, ITU-T H.261, H263, ISO/IEC MPEG-1 and MPEG-2 is that the image model used in motion compensation (MC) is inflexible. For each block in current frame, a matching block has to be searched in the corresponding reference frame and if suitable; its motion vector is substituted for the block during transmission. If not, those uncompensated blocks have to be transmitted in their entirety [1, 2] .

We try to improve the video compression by using a combination of iterative refinement and parameterisation of the image model. The transmission of entire blocks in motion compensation is replaced with transmission of parameters which characterise those blocks. The approach we used shares many similarities with vector quantisation [3] and fractal coding [4].

This paper is organized as follows. In section 2, we present the proposed image model with its simplicity and application. In section 3, the application of proposed image models in attempt to solve the redundancy of residual signals in video coding is discussed. Finally, we summarise some of the results presented in this paper along with contributions.

## 2 PROPOSED IMAGE MODELS IN INTRA-FRAME CODING

Image compression generally relies on being able to predict the image based on the models of the image. Some of these models are unable to capture all the necessary details of the image so that a separate residual signal must be encoded to make up for the failure of the model. In this work, we attempt to use a component based model that can be iteratively applied to reduce the amount of the residual that needs to be encoded.

We first establish a generalized fractal-like image model (GI) with iterative and parameterisable features, where the reconstructed image approximates the original image progressively over iterations.

The design of encoding scheme is described as follows:

Encoder:

1. *Generate a codebook of image vectors that will be used to reconstruct image. C= {$c_0$, $c_1$,…, $c_{n-1}$}.*

2. *Create an initial reconstructed black image $R_i$, and set iteration counter i=0.*

3. *Generate a difference image $E_i$ from the reconstructed image $R_i$ in previous iteration and original image I. Difference image $E_i$ is defined as*

$$E_i = \begin{cases} 1, & \text{if } |R_i(x,y) - I(x,y)| > T \\ 0, & \text{otherwise} \end{cases} \quad \textbf{Equation 1}$$

   *where T is predetermined threshold.*

4. *Determine boundary rectangles of areas, or domains D= {$d_0$, $d_1$,…, $d_{n-1}$} in difference image with quad-tree split techniques.*

5. *Map each domain $d_i$ with affined transformed codeword $W_j(C_k)$.*

6. *Create a reconstructed image $R_{i+1}$ with a set of inverse affine transformations,* $W^{-1} = \{ w_0^{-1}, w_1^{-1}, ..., w_n^{-1} \}$ .

7. *Calculate PSNR of reconstructed image $R_{i+1}$, if PSNR is higher than reconstructed image $R_i$ in previous iteration , then encode current affine transformations W, and go back to step 3 with increment of iteration counter i by 1 .Otherwise, stops.*

Decoder:

1. *Create an initial reconstructed black image $R_i$, and set iteration counter i=0.*

2. *For each domain $D_i$, map each inverse affine transformation $W_j^{-1}$ to reconstructed domain $R(D_j)$ and paste each reconstructed domain back to $R_i$, by which the new reconstructed image $R_{i+1}$ is created. The process continues repeatedly until iteration counter, i, reaches the same value of iteration counter in encoding.*

In this work, we developed a vector space partitioning method to generate the codebook, where the number of codewords is pre-determined. Our space partitioning is applied on a space consisting of vectors of pixels or codewords. We iteratively split the space into smaller spaces along the dimension of highest variance.

The algorithm of our space partitioning is described as follows.

1. *Begin with a set of spaces, $D = \{d_1, d_2, d_3.. , d_m\}$ . Initially, only one space E is in the set, which contains all the training vectors x in space E and number of spaces, m =1.*

2. *For each space d in set D, find the space $d_i$, which has the largest variance $R(d_i)$ at dimension l among all the sub-spaces in the set D.*

3. *Split the space $d_i$ into two subspaces $d_j$ and $d_k$, based on the median value among all vectors in space $d_i$ at dimension, l. If pixel value of input vector at specified dimension l is less than or equal to the median, then current vector is placed into first partitioned space $d_j$. Otherwise, current vector is placed into second partitioned space $d_k$.*

4. *Remove $d_i$ from the set D and add two split spaces $d_j$ and $d_k$ to the set D. increment number of spaces m by 1.*

5. *If number of spaces m in set D is greater than the desired number of codewords, stop, else, go to step 2.*

6. *Calculate the centroid $\bar{x}$ in every space $d_i$ of set D as a codeword.*

Fractal compression suffers from long encoding time. In our proposed encoding algorithm, we have organised the codebook into a binary tree structure to perform a quick search for each given domain block. The common sequential search in fractal compression has been replaced by the binary search within a small codebook, which runs in logarithmic time. Thus, the sum of all search times, is O(KNLog(C)),where K, N and C are number of iterations, domains found in each iteration and codewords respectively. Similar to fractal compression, proposed algorithms allow fast decoding simply by mapping each inverse affine transformed domain back to previous reconstructed image over iterations.

To improve the quality of image, two types of product codes, the mean removed product code, and the mean-gain removed product code are utilised with proposed image model [5]. The algorithm of the proposed image coding with product code is actually very similar to the no-product code one (GI) except for the following differences.

In the case where mean removed product code is used (MPCI), the mean of each domain $d_i$ is removed to match against each mean-removed codeword. If we define a domain as a k-dimensional input vector $D = \{d_1, d_2, \ldots d_k\}$, we compute the vector $m_d$, and shape $s_d$, as

$$m_d = \frac{1}{k}\sum_{i=1}^{k} d_i = \frac{1}{k} D\,1 \quad \text{where } 1 = \{1, 1, \ldots, 1\}^T$$

$$s_d = D - m_d 1$$

**Equation** 2

While in the case where the mean-gain removed product code is used (MGPCI), the mean $m_d$, and the gain $\sigma_d$ of each domain are successively removed to match against each mean-gain removed codeword. Note the shape domain $s_d$ has zero mean and unit gain.

$$m_d = \frac{1}{k}\sum_{i=1}^{k} d_i = \frac{1}{k} D\,1 \quad \text{where } 1 = \{1, 1, \ldots, 1\}^T$$

$$\sigma_d^2 = \sum_{i=1}^{k} (d_i - m_d)^2$$

$$s_d = \frac{d_i - m_d 1}{\sigma_d}$$

**Equation 3**

| | bird | | house | | Lenna | | peppers | |
|---|---|---|---|---|---|---|---|---|
| | **BPP** | **PSNR** | **BPP** | **PSNR** | **BPP** | **PSNR** | **BPP** | **PSNR** |
| **GI** | 1.28 | 31.89 | 1.50 | 29.91 | 1.84 | 26.45 | 1.91 | 24.39 |
| **MPCI** | 1.66 | 35.61 | 2.02 | 33.47 | 2.59 | 29.03 | 2.80 | 27.19 |
| **MGPCI** | 2.53 | 33.94 | 2.98 | 32.17 | 3.58 | 29.24 | 3.76 | 26.64 |

**Table 1.** Iterative-Parametric Image Coding Performance

Table 1 presents experimental results of 256x256 grey scale test images, which are encoded with three types of proposed iterative algorithms in terms of compression rate, bpp and image quality, PSNR. Typical fractal-like image compression methods can achieve compression ratio between 4:1 to 25:1 for grey scale images and 8:1 to 50:1 for colour images. The bit rates in bpp listed in table 1 are high since we are storing uncompressed codebook in the compressed file.

The effect of varying the number of codewords upon the PSNR of a reconstructed image, lenna, as shown in figure 1(each individual PSNR-bpp point in figure 1 is obtained by incrementing codebook size by 32 each step, from 64 to 256, given fixed codeword size 4x4 and threshold 8) illustrates two points. The first, is that utilization of product codes can achieve higher image quality than no-product code one at the cost of lower compression rate. The second point, is that increasing the number of codewords beyond a certain point, yields little improvement in the PSNR of reconstructed image, which can be seen in figure 1 that slopes of curves decrease.

One of the key factors which limit the performance of iterative algorithms is the number of parameters which characterise each domain. In GI, four parameters, the x, y locations, scaling factor s, and codeword index i are required to characterise a domain. If each input vector in VQ is treated as a not scaled, pre-ordered domain, there is only one parameter associated with each domain. However, in motion compensated video compression, the benefits of encoding less parameter in VQ based coding in comparison to iterative-parametric coding do not exist any more since the two extra parameters, x, y coordinates must be associated with each vector and encoded.

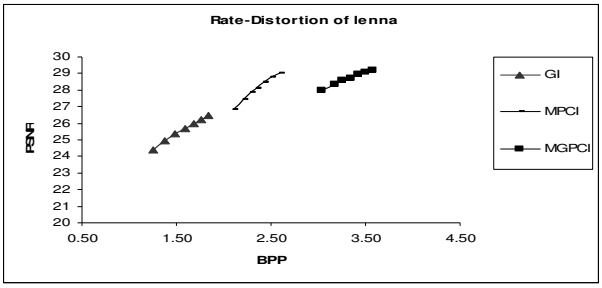


**Figure 1.** The effect of varying codebook size upon PSNR for the "Lenna"

## 3 PROPOSED IMAGE MODELS IN INTER-FRAME CODING

Our video coder implementation based on the image model proposed in section 2 overcomes the shortcoming of transmitting entire uncompensated blocks in MC compression. Similarly to other fractal-like inter-frame compression techniques [6, 7], we segment the frame difference between the current frame and previously reconstructed frame into the background where the prediction is successful and a foreground where the prediction failed. As the background is already known to the decoder, only the foreground is encoded using the iterative and parameterised intra frame coding discussed in section 2.

**Encoding Process**

1. *Begin with an initial reconstructed frame $R_i$ (i = 0) with each pixel value at zero.*
2. *For each frame $I_i$ in the image sequences $I = \{I_1, I_2, \ldots, I_n\}$,*

   *Generate a difference image $D_i$ between current frame $I_i$ and previous reconstructed frame $R_i$ and $D_i$ is defined as*

   $$D_i = \begin{cases} 1, & if\ |R_i(x,y) - I_i(x,y)| > T \\ 0, & otherwise \end{cases}$$ **Equation 4**

   *where T is predetermined threshold.*

   *Encode a set of affine transformations $W_i$ from difference image $D_i$ with proposed iterative-parametric intra frame approach.*

   *Reconstruct a frame $R_{i+1}$ with affine transformations W.*

**Decoding Process**

1. *Begin with an initial reconstructed frame $R_m$, with each pixel value at zero and frame counter, m = 0.*
2. *While the counter m is less than n, the number of frames in original sequence*

   *Reconstruct frame $R_{m+1}$ with a set of affine transformations $W_m$, based on previous reconstructed image $R_{m-1}$. Increment frame counter m by 1.*

Similarly to the intra frame coding, two types of product codes, mean-removed product code (MPCI) and mean-gain-removed product code (MGPCI) are utilised in inter frame coding.

| | GI | | MPCI | | MGPCI | |
|---|---|---|---|---|---|---|
| | **BPS** | **PSNR** | **BPS** | **PSNR** | **BPS** | **PSNR** |
| Miss_am | 39k | 31.41 | 100k | 33.40 | 148k | 34.50 |
| Salesman | 37k | 26.88 | 179k | 28.84 | 228k | 29.28 |
| Carphone | 73k | 26.26 | 246k | 28.75 | 351k | 29.74 |
| Container | 26k | 24.42 | 114k | 26.38 | 146k | 26.71 |

**Table 2.** Iterative-Parametric Video Coding Performance.

Table 2 gives general information and overall results how the respective sequences perform with three type iterative proposed algorithms. The video test sequences are coded with 256 4x4 codewords with threshold at 8.

The average bps is calculated as:
Average bps =
$\sum$ size of compressed frames in bits / duration in seconds **Equation 5**

Table 2 demonstrates that the performance of product code iterative-parametric coding in terms of PSNR is much better than no-product code one, GI. On average, the visual quality of frames in MPCI is 2dB higher than that in GI and in MGPCI, it is 3dB higher than that in GI. Similar to image coding, one of the reasons that utilisation of product code in proposed iterative-

parametric video coding is that in no-product code algorithm, such as GI, beyond a certain codebook size, any further increase in number of codewords has the only effect of increasing the bit rate with almost no improvement in the PSNR. MPCI and MGPCI can improve average PSNR significantly over frames at the cost of higher data rates.

The instantaneous data rates of all algorithms for video test sequence "Miss America", are shown in figure 2. Curves in figure 2 have very similar shapes. The peaks in graphs can be explained by a significant movement and some blurring. It can be found that data rates in motion compensation are higher than others, given same test settings. Besides, the variations of bps in motion compensation are larger than those encoded with iterative algorithms. The MPCI is preferred to others for its low data rate, small variation of data rate, and moderate average image quality.

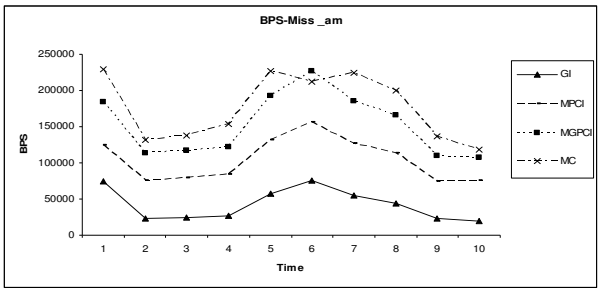


**Figure 2.** Instantaneous BPS for "Miss America" Sequence.

The instantaneous bps is calculated as:
Instant bps = $\sum$ size of compressed frame in bits per second **Equation 6**

The effect of varying threshold upon PSNR of reconstructed video sequence "Miss America" is shown in Figure 3. The curves are plotted by varying threshold (the maximum pixel difference) from 8 to 24, with increment by 2 each step, given fixed codeword size 4x4, and fixed codebook size 256. The RMSE-value threshold in block-based motion compensation is the square of each threshold.

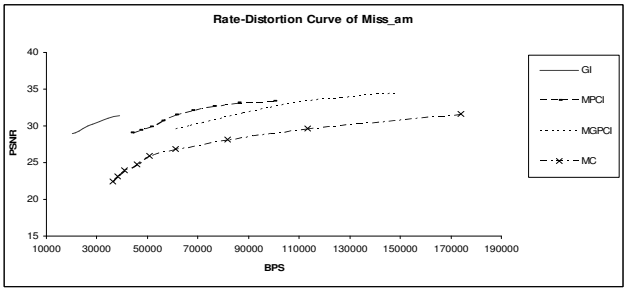


**Figure 3.** The effect of varying threshold upon the PSNR for the encoded sequence of "Miss America".

Figure 3 also indicates that proposed iterative-parametric algorithms are more suitable than motion compensation type algorithms due to the higher quality encoded sequence in iterative-parametric video coding.

## 4 SUMMARY

In this paper, a novel iterative-parametric image model has been proposed, which shares the self-similarity feature in fractal image coding but reduces the complexity of domain-range mappings with domain-codeword mappings over iterations. The proposed image models also utilises product code techniques, which improves the quality of images.

With the proposed image model and product codes, the main deficiency of transmitting uncompensated blocks in motion compensated video compression diminishes by iteratively fractal-like and parametrically encoding the frame difference. After presenting the details of the proposed algorithms, the experimental results obtained from video sequences are given. The results indicate the proposed algorithms achieve lower bit rate and less variation of bit rate in comparison to the motion compensation technique. In addition, the proposed algorithm is simple and easy to implement.

## 5 REFERENCES


[1] Sikora, T, "MPEG digital video-coding standards," *Signal Processing Magazine, IEEE*, Vol. 14, Issue 5, Sept. 1997 pp. 82 - 100.

[2] Rijkse, KLeslie Lamport, "H.263: video coding for low-bit-rate communication," *Communications Magazine, IEEE*, Vol. 34, Issue 12, Dec. 1996 pp. 42 - 45.

[3] A.Gersho; R. M. Gray. "*Vector Quantization and Signal Compression,"* Kluwer Academic Publishers, Boston, MA, 1992.

[4] Elsherif, M.; Rashwan, M.; Elsayad, A, "Matching criteria in fractal image compression," *Circuits and Systems, 2000. Proceedings of the 43rd IEEE Midwest Symposium*, Vol. 2, 8-11 Aug. 2000 pp. 612 – 615.

[5] Hamzaoui, R.; Saupe, D.; Wagner, M, "Rate-distortion based video coding with adaptive mean-removed vector quantization," *Image Processing, 1998. ICIP 98. Proceedings. 1998 International Conference*, 4-7 Oct. 1998 pp. 958 - 961 vol.3.

[6] Zhen Yao; Wilson, R., "Hybrid fractal video coding with neighbourhood vector quantisation," *Data Compression Conference, 2004. Proceedings. DCC 2004*, 2004 pp.573 – 573.

[7] B.Hurtgen and P.Buttgen. "Fractal Approach to Low Rate Video Coding," *Proceedings of the SPIE*, Boston, Massachusetts, Nov.1993, Vol. 2094, pp. 120-131.