Fast Low Bit-Rate 3D Searchless Fractal Video Encoding

Vitor de Lima, William Robson Schwartz, Helio Pedrini
Institute of Computing
University of Campinas
Campinas-SP, Brazil

***Abstract*—Video encoding techniques play an important role in data reduction. Fractal compression has received considerable attention in the past decades. While early methods presented prohibitively large encoding times, recent searchless fractal encoders reduced this problem. A fast 3D purely fractal video encoder based on a flexible adaptive spatial subdivision data structure is proposed in this work. The method completely avoids any kind of search for a matching domain block and is capable of performing fast compression and decompression with high visual fidelity. Experimental results show that the developed approach outperforms the state-of-the-art x264 video encoder at very low bit rates in high motion video sequences in both structural dissimilarity measure and encoding time.**

***Keywords*-Video compression; fractal encoding; video codecs; low bit-rate encoder.**

I. Introduction

Earlier compression methods based on fractal coding [1] suffered from extremely slow encoding times to find an appropriate representation of the image content. Their performance was inferior than more conventional approaches based on invertible transforms, such as the discrete cosine transform (DCT). The main problem is that fractal encoders must find a transform that constructs an approximation of the original image, given the image itself, by looking for similar blocks between different regions using either brute force or a heuristic to reduce the number of elements considered in the search.

Ten years after the introduction of the original fractal encoding method, other methods were proposed by Furao and Hasegawa [2] and Wu et al. [3] aiming to completely avoid any kind of search by imposing a fixed relationship between a block of the original image (called a range block) and its correspondent similar block (called a domain block). This solution produced results comparable to the original JPEG standard and even surpassed it in higher compression ratios while being faster than most of the state-of-the-art encoders.

This paper proposes a fast low bit-rate 3D searchless fractal video compression method for encoding chunks of consecutive frames based on [2], [3]. According to the experimental results, it is perceptually superior to the state-of-the-art x264 [4] at high compression ratios in video sequences with large amount of motion. Additionally, it also presents an encoding time lower than that of x264.

An encoder that can operate at extreme compression ratios is needed by applications that require either low bandwidth communication channels, such as sensor networks and mobile devices, or that must transmit multiple video streams simultaneously, such as surveillance equipments.

The text is organized as follows. Section II presents a brief review of fractal encoding and related work available in the literature. The proposed method is described in Section III. Experimental results and a comparison between the proposed method and another fractal encoder are presented in Section IV. Finally, the conclusions of the work are given in Section V.

II. Background

This section initially reviews fractal image and video encoding, including searchless techniques, then describes heuristics for fast rate-distortion and structural dissimilarity measure.

A. Fractal Image Coding

Unlike other compression algorithms, fractal encoders [1] do not explicitly store an approximation of the image, but they create and transmit a collage, which is a series of instructions that indicate how to partition the image and, for each resulting partitioned region (called a range block), how to generate its content given another block with larger dimensions (called domain block) of the same image. To generate the range blocks, the collage resizes the domain block, applies an affine transform (such as rotation or mirroring), and modifies the gray level values using an equation such as the one proposed by Øien and Lepsøy [5]

$$G(D) = \alpha(D - \bar{D}I) + \bar{r}I \qquad (1)$$

where G is the gray level transform, D is the downsampled domain block, $\bar{D}$ is the mean value for the domain block, $\bar{r}$ is the mean value for the block in the original scale (the range block), I denotes a matrix filled with ones, and α is the scaling parameter.

The collage is capable of transforming the image into itself given its definition, but it is also capable of transforming any arbitrary signal into an approximation of the original image. In order to decode the image, it is only necessary to apply the collage to any initial image several times until it reaches a fixed point.

The most remarkable characteristics of these methods are the fast image decompression and the extremely slow and complex encoding process, since most approaches construct the collage exhaustively matching each range block with a large number of domain blocks and selecting the best match by certain criteria.

B. Searchless Fractal Image Coding

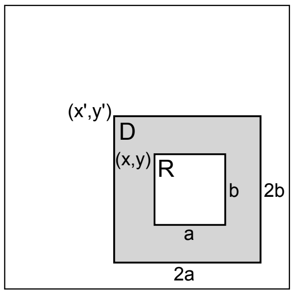


Fig. 1. Relationship between range (R) and domain (D) blocks in a searchless fractal encoder.

The searchless fractal image coding was first proposed by Furao and Hasegawa [2] to completely avoid searching for the best fit between range and domain blocks. Within this approach for a range block with dimensions a and b located at the coordinate (x, y), there is only one possible domain block with dimensions $2a$ and $2b$ that must be located at (x', y'), as shown in Equation 2 and illustrated in Figure 1. Each region of the image is encoded only by the α and $\bar{r}$ coefficients of the gray level transform.

$$\begin{aligned} x' &= x - a/2 \\ y' &= y - b/2 \end{aligned} \tag{2}$$

The image is initially partitioned into a uniform grid and the collage error (difference between the range block and the transformed domain block) is measured for each region of this partition. If this error is above a certain threshold T_{error}, such region is recursively divided into four subregions and the process continues recursively with the subregions until the block size reaches a minimum size or the collage error drops below T_{error}. Since each block of the image is always recursively divided into another four subregions with equal area, the resulting positions and sizes of each range region can be encoded in a quadtree and α and $\bar{r}$ are transmitted for each region.

A superior method was proposed by Wu et al. [3], which does not impose any limit on the size of the range blocks. To achieve a better compression ratio, there are no scaling parameters for the regions covering one or two pixels and the $\bar{r}$ value is more coarsely quantized in smaller regions. Instead of using a quadtree, this encoder uses a more flexible structure, called binary-tree partition (an example image subdivided by this structure is shown in Figure 2), which only divides a region into two sub-blocks with the same size by selecting between splitting it in half in the vertical or horizontal direction. The method presented here uses a volumetric generalization of this data structure.

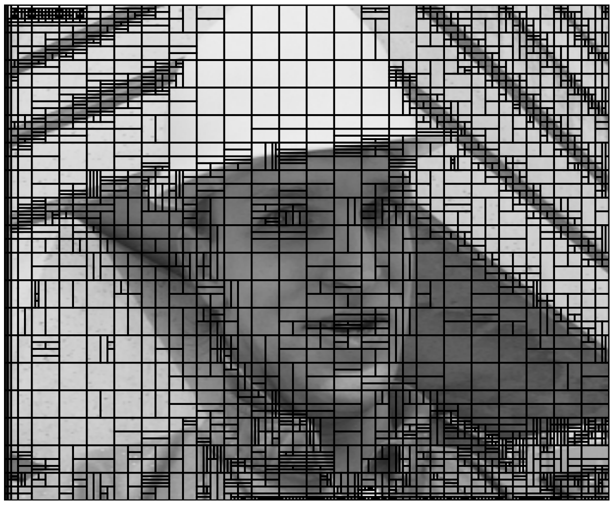


Fig. 2. An example image subdivided by a recursive binary tree partition.

C. Fractal Video Coding

The first fractal video encoder was proposed by Hurd et al. [6], which encodes each frame by using blocks from the previous frame as a domain pool either at the same scale as the range blocks or at higher scales, making it a generalization of the usual motion compensation techniques. This approach was later refined by Fisher et al. [7] through the use of quadtrees. More advanced variations of these algorithms were designed later [8], [9].

Another extension of the fractal coding for video sequences was proposed by Lazar and Bruton [10] and Li et al. [11]. In these methods, a chunk of consecutive frames is grouped into a single volumetric image where the x and y axes are the spatial position and the z axis is the time when that pixel value occurred. The collage consists of a spatial subdivision of that volume and, for each resulting range block, the position and the parameters necessary to transform the volumetric domain block. The proposed method is based on this specific generalization of the fractal methods to encode video sequences.

A fast volumetric encoder was later proposed by Chabarchine and Creutzburg [12] for real-time video encoding by simplifying the gray scale transform to use a constant α parameter, resampling every frame to 64×64 pixels, grouping 16 consecutive frames and dividing them into blocks of $16 \times 16 \times 16$ voxels. Each block is represented by an octree and the only possible domain block for each range block is its parent block on the spatial subdivision. The volume is subdivided until a target error threshold is reached. This method is simple and fast, however, its rate-distortion performance is extremely poor.

The volumetric video compression approach [12] was refined by Yao and Wilson [13] through a hybrid method that employs both vector quantization and collages to approximate the original signal. Such hybrid method can encode videos at low bit rates achieving a fair visual quality while being as fast as some MPEG-2 encoders. Unfortunately, this implementation suffers from convergence problems during the decoding stage.

D. Fast Rate-Distortion Heuristic

Every fractal encoding method must decide how to partition the image into regions that have a similar domain block and the total number of regions must satisfy the restriction on the number of bits set by the user. Most encoders (such as the one described in Section II-B) subdivide each region recursively until a certain threshold for the collage error is reached. This heuristic tries to guarantee a minimum reconstruction quality for the resulting decoded image, but it is difficult to efficiently satisfy any restriction on the total size of the collage.

A solution to this problem was proposed by Saupe et al. [14] after investigating optimal partitions in fractal encoding. In this method, the image is divided into a uniform grid, where each region has its collage error calculated and inserted into a priority queue. At each iteration, the region with the highest error is removed from the queue and subdivided, then its subregions are inserted into the queue.

The size of the collage increases slightly at each iteration, so it is possible to achieve a certain size by stopping the heuristic after a certain number of iterations, resulting in an approximation of the desired size. The heuristic is also intuitive since the most distorted regions of the image have priority over the other ones.

E. Structural Dissimilarity

Most comparisons between video and image encoders are based on metrics derived from the sum of squared differences (SSD) or the mean squared error (MSE). The SSD and the MSE between two images A and B with size $W \times H$ is given by

$$\text{SSD}(A,B) = \sum_{x=0}^{W-1} \sum_{y=0}^{H-1} (A_{x,y} - B_{x,y})^2 \tag{3}$$

$$\text{MSE}(A,B) = \frac{\text{SSD}(A,B)}{W \times H} \tag{4}$$

A critical issue with the MSE is that it does not measure the resulting image quality directly and it can attribute similar scores to images with large differences in psychovisual quality. As illustrated in Figure 3, the psychovisual quality degradation between the images is measured by the SSIM, while the MSE does not reflect that fact, as indicated in the figure captions.

The structural similarity index (SSIM) [15] was proposed as a metric to compare images which correlates more appropriately with the human perception. It maps two images into an index in the interval $[-1, 1]$, where higher values are given to more similar pairs of images, calculated as

$$\text{SSIM}(A,B) = \frac{(2\mu_A\mu_B + c_1)(2\sigma_{AB} + c_2)}{(\mu_A^2 + \mu_B^2 + c_1)(\sigma_A^2 + \sigma_B^2 + c_2)} \tag{5}$$

where μ_A, μ_B, σ_A^2 and σ_B^2 are the averages and variances of A and B, σ_{AB} is the covariance between A and B, and both c_1 and c_2 are predefined constants. This metric is calculated as the average of the score between several blocks using a sliding window of 11×11 pixels.

The structural similarity scores are also shown in Figure 3. In this example, it is possible to notice that the image with the lowest MSE is the least similar to the original. In addition, all three images have almost the same MSE, but the structural similarity is more coherent to what one would expect from a comparison metric.

The structural dissimilarity is a derived metric from the structural similarity that results in more distinct values, since a small variation in the original SSIM indicates a large difference in image quality. It is given by

$$\text{DSSIM}(x,y) = \frac{1}{1 - \text{SSIM}(x,y)} \tag{6}$$

III. Proposed Method

The proposed 3D video encoder constructs a volumetric image composed of 32 consecutive frames and transmits a collage that is used to reconstruct them (using the same definition as the volumetric encoders described in Section II-C). This image is divided into a uniform grid of blocks with $16 \times 16 \times 16$ voxels and each one of these blocks has its own spatial subdivision tree. In this binary tree, each block has two subblocks with equal volume which are created splitting their parent into half in the horizontal, vertical or temporal direction. The blocks are subdivided according to the heuristic presented in Section II-D, using the SSD of the collage error as the distortion metric.

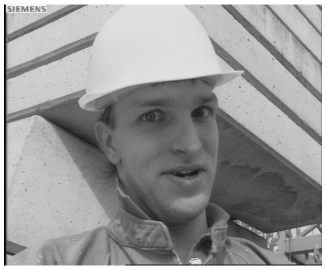
The SSD was chosen as the distortion metric since the MSE disregards the size of the block, giving the same score to equally distorted blocks with large differences in volume. The SSIM was not created to evaluate volumetric images, it must be calculated using sliding windows since it cannot properly compare two isolated subblocks of an image and, contrary to other usual metrics, it is impossible to estimate the final SSIM of the image given the SSIM of each range block. In the case of using SSD, the final score is the sum of the score of all the encoded blocks. Given these observations, if the SSIM was employed the heuristic would not estimate which block can result in the largest reduction of distortion, since the distortion metric itself cannot be efficiently measured in either the entire image or the range blocks.

This heuristic requires a given volumetric block to be encoded and split in case it is chosen during an iteration. The block encoding method uses a relationship between range and domain blocks similar to the one used in bidimensional searchless fractal encoders, but generalized to three dimensions. For each range block with dimensions a, b and c located at (x, y, z), there is only one possible domain block that can be used to represent it. The position of the domain block can be expressed in Equation 7, with dimensions equal to $2a$, $2b$ and $2c$. This block is illustrated in Figure 4.

$$\begin{aligned} x' &= x - a/2 \\ y' &= y - b/2 \\ z' &= z - c/2 \end{aligned} \tag{7}$$

(a) Original image (SSIM=1.0, MSE=0)

(b) Multiplied by 1.072 (SSIM=0.995837, MSE=145.96)


(c) Subtracted by 12 pixels (SSIM=0.994981, MSE=143.97)

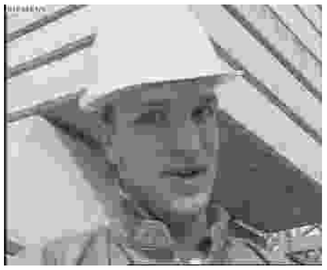

(d) Compressed by JPEG (SSIM=0.742805, MSE=142.91)

Fig. 3. Several distorted versions of the same image with different perceptual qualities and approximately the same MSE

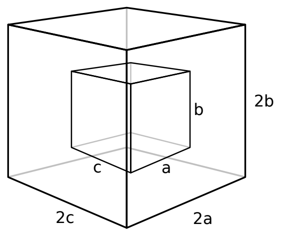


Fig. 4. An example of the relationship between range and domain blocks in the proposed 3D encoder.

Some blocks cannot use this equation since applying it would result in a domain block that is not completely inside the volumetric image. In these cases, if a coordinate is negative, it is set to zero and if any pixel inside the block has a coordinate larger than the dimensions of the image, the coordinate of the domain block is set to be the dimension of the image minus the dimension of the block.

Each range block is matched to its only respective domain block using Equation 1 by assigning the α parameter as 0.25, 0.5, 0.75 or 1.0, as suggested by Tong and Pi [16].

Decision Table for the Quantization of $\bar{r}$		
Volume	Quantization step	Number of used bits
1	16	4
2	16	4
4	16	4
8	8	5
16	8	6
32	4	6
64	4	6
128	2	7
256	2	7
≥ 512	1	8

TABLE I
QUANTIZERS APPLIED ACCORDING TO THE VOLUME OF THE RANGE BLOCK.

The heuristic used to decide how to properly split a block divides it into all the three possible directions and the direction resulting in the smallest sum of the SSD for both resulting subblocks is chosen.

All the required symbols and parameters are encoded using a context-adaptive arithmetic coder [17]. Each range block is encoded by its α parameter, which occupies 2 bits in the worst case, along with $\bar{r}$, which is quantized according to the range block volume as shown in Table I. For range blocks with one or more dimensions smaller than 2 pixels, the only transmitted parameter is $\bar{r}$. Along with these parameters, the spatial subdivision tree for each block in the initial uniform subdivision is coded by a sequence of symbols pointing to the decoder, in a depth-first order, whether a certain region was subdivided or not and in which direction it was split. The α parameter and the binary decision symbols in the spatial subdivision tree have their own high order adaptive contexts, one for each possible value of $\lfloor \log V \rfloor$, where V is the total volume of the encoded block. The direction which each block is split is encoded by another set of 3 high order adaptive contexts chosen according to the direction used to split its parent.

The $\bar{r}$ parameter is encoded as a difference between a quantized prediction calculated as the average of $\bar{r}$ of the neighboring blocks located at the top, to the left and behind the encoded block and the real quantized value. This difference is encoded by the Adaptive Goulomb-Rice code described in [18], using one context for each possible $\lfloor \log V \rfloor$ in the same manner as the other parameters.

IV. EXPERIMENTAL RESULTS

All experiments were conducted on an Intel Core 2 Duo E6750 processor, 2.66 GHz with 3GB of RAM running Linux operating system. The method was implemented using the C++ programming language without any additional optimizations.

The proposed approach is compared to the state-of-the-art H.264 encoder, called x264 [4], using the standard grayscale benchmark sequences 'Foreman', 'Car phone', 'Bus', 'Football', 'Akiyo', 'Miss America', 'Bowing', and 'Hall monitor' in the CIF format [19]. The length of each sequence is shown in Table II.

The x264 was configured to closely match the behavior of the proposed encoder by forcing it to insert a keyframe at every 32 frames and compiling it without any CPU specific optimizations. The command line used to invoke this encoder was

```
x264 --tune ssim --preset medium \
```

Sequence	# Frames
Foreman	300
Carphone	457
Bus	150
Football	260
Akiyo	300
Hall monitor	300
Bowing	300
Miss America	179

TABLE II
NUMBER OF FRAMES FOR THE USED VIDEO SEQUENCES.

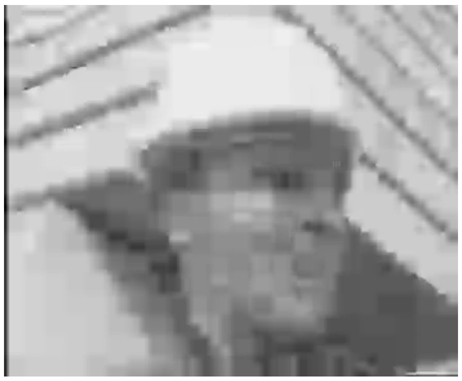
```
--profile baseline --keyint 32 \
--bitrate 'target bitrate'
```

The video sequences in the following experiments were encoded at low bit rates, which implies that the results had high distortions when measured by Equation 3. As shown in [15], the ambiguity of the metrics derived from the SSD from a perceptual point of view is high and becomes even larger as the distortion increases. It is important to observe that both α and $\bar{r}$ are quantized (i.e. they must assume one of a small set of possible values instead of being continuous) which causes a mean shift and a contrast change in every range block, even though the effect of these quantizations is perceptually negligible. In order to ensure a proper comparison between both methods, the mean structural dissimilarity for both encoders is used in the experiments. This metric is widely accepted for its simplicity and reasonably accuracy, being employed in the design of several image encoders, such as [20], and has a built-in implementation in the x264.

The bit rate was varied to closely match the same values in both encoders. As observed in the first and second rows of Figure 6, the proposed encoder outperforms the x264 codec at very low bit rates in these high motion sequences. This is due to the motion compensation algorithm of the H.264 encoder cannot operate properly in these conditions given that an accurate prediction of each frame would require a large amount of bits. An example of this case is shown in Figure 5, where the motion of smooth regions is ignored or poorly represented by the x264, then generating temporal artifacts. In this sequence, the proposed method accurately represents most of the moving regions, causing high distortions only on instantaneous movements such as eye blinking. The 'Car phone' and 'Foreman' sequences have transitions between high and low motion scenes giving an advantage to x264 at bit rates larger than 60 kbps.

The third and fourth rows of Figure 6, which are related to scenes with a static background and one or two moving objects, show that in these cases the highly efficient transform coding of the x264 has a significant advantage over the proposed fractal encoder and the motion compensation is achievable due to the localized motion of a few regions in each frame.

The total encoding time of the proposed method is remarkably low as evidenced in Figure 7. Both methods, the proposed fractal encoder and the x264 codec, were implemented in high level languages without excessive optimizations to ensure a fair comparison between them. The proposed method takes between one third and one fifth of the total time needed by the x264 to encode the same content.


(a) A frame from the 'Foreman' sequence encoded

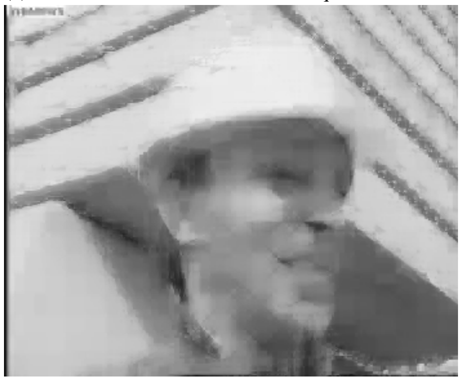

(b) The same frame encoded by the proposed method

Fig. 5. Differences in the accuracy of frame prediction in both methods at 50 kbps.

V. CONCLUSIONS AND FUTURE WORK

This paper proposed a 3D searchless fractal video encoder that is comparable to the x264 encoder [4] at very low bit rates. As it has been observed from the experimental results, while the proposed encoder outperformed the x264 in the high motion video sequences, for scenes with a static background and few moving objects, the x264 presented advantage over the proposed method. Furthermore, contrary to most fractal-based methods, it presents a very low encoding time even when compared to x264 for all tested sequences.

Suggestions for future enhancements in the proposed approach include a better lossless encoding of the gray level parameters and of the symbols used in the spatial subdivision, the use of fractal interpolation to encode the content at a lower frame rate and super-sample it to the original rate, the implementation of more complex gray level transforms such as the one used in [21] and, finally, the use of rate-distortion optimization methods [22] to choose which quantizers to use and which regions must be split.

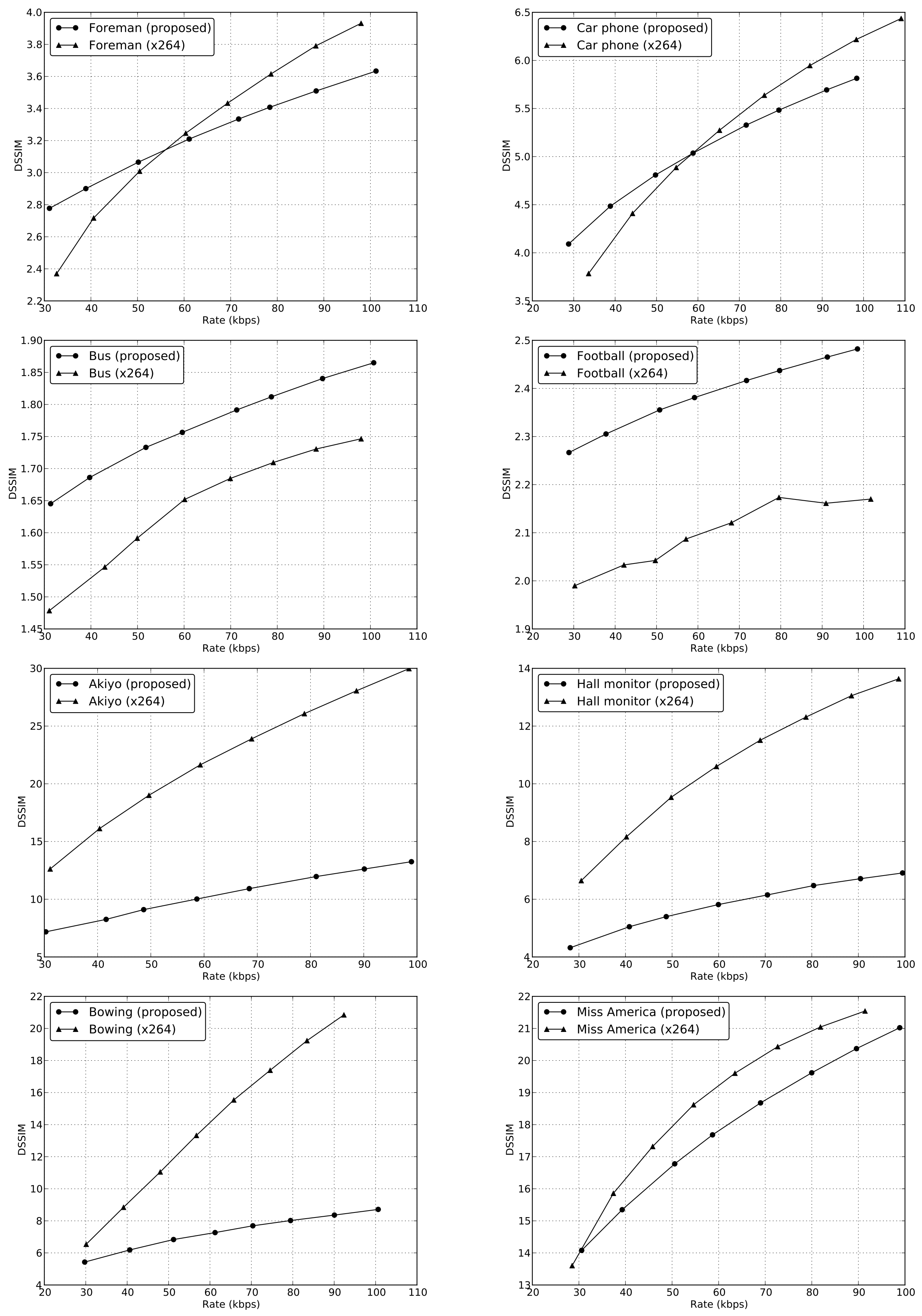


Fig. 6. Mean structural dissimilarity at different rates for the proposed video encoder and the x264 encoder.

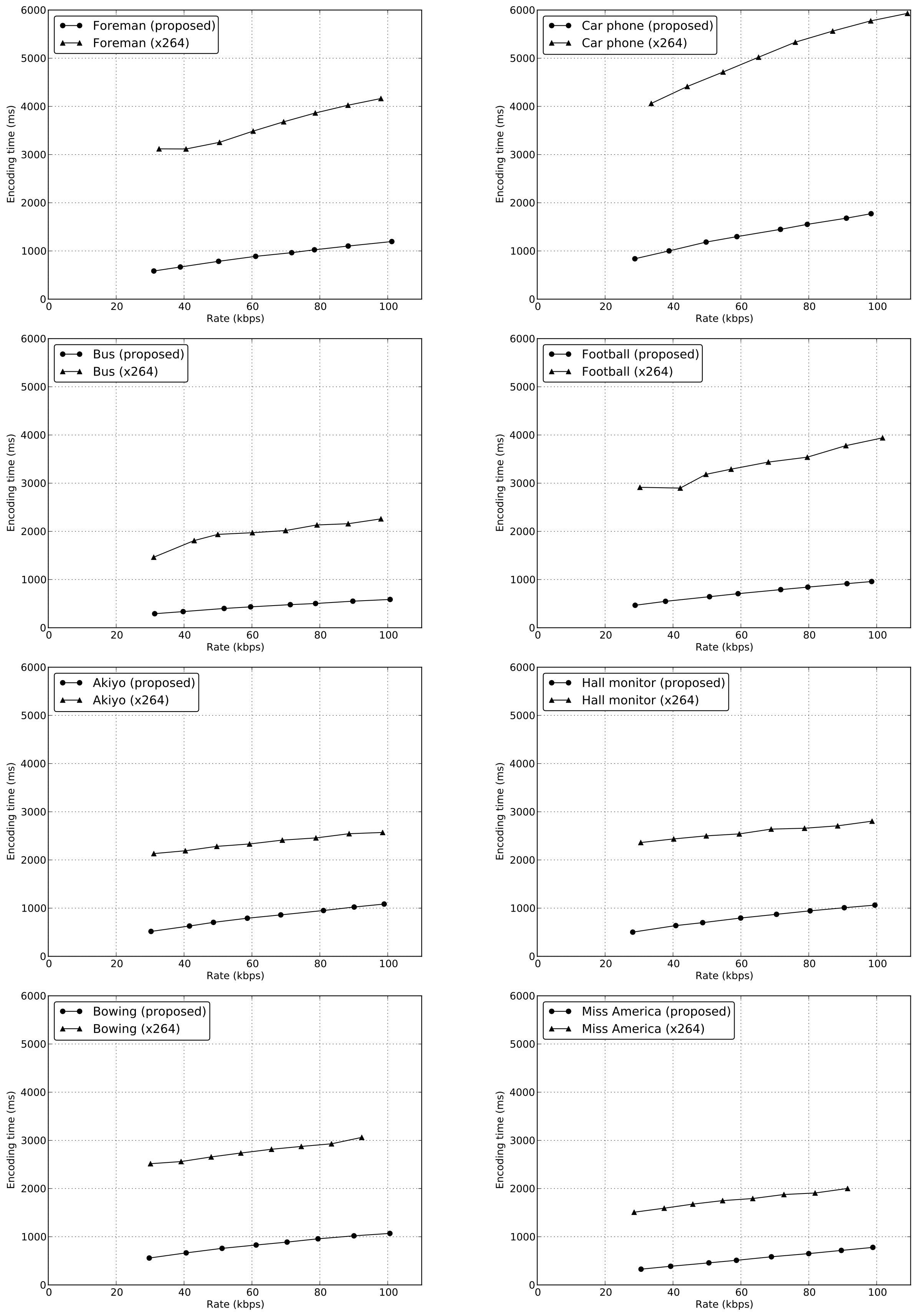


Fig. 7. Encoding time at different rates for the proposed video encoder and the x264 encoder.

Acknowledgments

The authors are thankful to FAPESP, CNPq and CAPES for the financial support.

References

[1] A. Jacquin, “Image Coding Based on a Fractal Theory of Iterated Contractive Image Transformations,” *IEEE Transactions on Image Processing*, vol. 1, no. 1, pp. 18–30, Jan. 1992.

[2] S. Furao and O. Hasegawa, “A Fast No Search Fractal Image Coding Method,” *Signal Processing: Image Communication*, vol. 19, no. 5, pp. 393–404, 2004.

[3] X. Wu, D. Jackson, and H. Chen, “Novel Fractal Image-Encoding Algorithm Based on a Full-Binary-Tree Searchless Iterated Function System,” *Optical Engineering*, vol. 44, no. 10, 2005.

[4] “x264 Video Encoder,” http://www.videolan.org/developers/x264.html.

[5] G. Øien and S. Lepsøy, *A Class of Fractal Image Coders with Fast Decoder Convergence*. London, UK: Springer-Verlag, 1995, ch. Fractal Image Compression, pp. 153–175.

[6] L. Hurd, M. Gustavus, and M. Barnsley, “Fractal Video Compression,” in *Thirty-Seventh IEEE Computer Society International Conference (Digest of Papers, COMPCON Spring 1992)*, Feb. 1992, pp. 41–42.

[7] Y. Fisher, D. Rogovin, and T. Shen, “Fractal (Self-VQ) Encoding of Video Sequences,” *Visual Communications and Image Processing*, vol. 2308, no. 1, pp. 1359–1370, 1994.

[8] C. Kim, R. Kim, and S. Lee, “Fractal Coding of Video Sequence using Circular Prediction Mapping and Noncontractive Interframe Mapping,” *IEEE Transactions on Image Processing*, vol. 7, no. 4, pp. 601–605, Apr. 1998.

[9] S. Zhu, Z. Wang, and K. Belloulata, “A Novel Fractal Monocular and Stereo Video Codec based on MCP and DCP,” in *IEEE International Conference on Industrial Technology*, Mar. 2010, pp. 168–172.

[10] M. Lazar and L. Bruton, “Fractal Block Coding of Digital Video,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 4, no. 3, pp. 297–308, Jun. 1994.

[11] H. Li, M. Novak, and R. Forchheimer, “Fractal-Based Image Sequence Compression Scheme,” *Optical Engineering*, vol. 32, no. 7, pp. 1588–1595, 1993.

[12] A. Chabarchine and R. Creutzburg, “3D Fractal Compression for Real-Time Video,” in *2nd International Symposium on Image and Signal Processing and Analysis*, 2001, pp. 570–573.

[13] Z. Yao and R. Wilson, “Hybrid 3D Fractal Coding with Neighbourhood Vector Quantisation,” *EURASIP Journal on Applied Signal Processing*, vol. 2004, pp. 2571–2579, Jan. 2004.

[14] D. Saupe, M. Ruhl, R. Hamzaoui, L. Grandi, and D. Marini, “Optimal Hierarchical Partitions for Fractal Image Compression,” in *IEEE Transactions on Image Processing*, 1998.

[15] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, “Image quality assessment: from error visibility to structural similarity,” *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600 –612, april 2004.

[16] C. Tong and M. Pi, “Fast Fractal Image Encoding based on Adaptive Search,” *IEEE Transactions on Image Processing*, vol. 10, no. 9, pp. 1269–1277, Sep. 2001.

[17] A. Said, *Lossless Compression Handbook*, ser. Communications, Networking, and Multimedia. Academic Press, 2003, ch. Arithmetic Coding.

[18] M. Weinberger, G. Seroussi, and G. Sapiro, “LOCO-I: A Low Complexity, Context-based, Lossless Image Compression Algorithm,” *Data Compression Conference*, vol. 0, p. 140, 1996.

[19] “CIPR Sequences,” http://www.cipr.rpi.edu/resource/sequences/.

[20] P. K. Krause, “ftc - Floating precision texture compression,” *Computers and Graphics*, vol. 34, pp. 594–601, October 2010.

[21] X. Wang, Y. Wang, and J. Yun, “An Improved No-search Fractal Image Coding Method Based on a Fitting Plane,” *Image and Vision Computing*, vol. 28, pp. 1303–1308, Aug. 2010.

[22] A. Ortega and K. Ramchandran, “Rate-distortion methods for image and video compression,” *IEEE Signal Processing Magazine*, vol. 15, no. 6, pp. 23 –50, nov 1998.