Accélération du codage fractal d'images

S. Douda*, A. A. El Imrani**, M. Limouri**

* Département de Mathématiques et Informatique

Faculté des Sciences et Techniques, Settat, Maroc

** Laboratoire de Conception et Systèmes

Faculté des Sciences, Rabat, Maroc

sofia_douda@yahoo.fr

RÉSUMÉ. La compression fractale d'images présente l'avantage de permettre un décodage rapide et une indépendance de la résolution mais souffre d'une lenteur dans le codage. En effet durant le codage, la recherche du meilleur bloc source parmi l'ensemble des blocs sources (dictionnaire) pour chaque bloc cible est longue. Pour réduire le temps de cette recherche, plusieurs méthodes ont été proposées. La plupart de ces méthodes sont basées sur la classification de l'ensemble des blocs sources en un certain nombre de classes puis chaque bloc cible est comparé aux blocs sources appartenant à la même classe. Le présent travail présente une approche visant à réduire le temps de calcul en utilisant deux dictionnaires par une approche originale. Cette approche peut être combinée avec les méthodes de classification et ainsi réduire davantage le temps de codage. Elle a en effet permis de réduire le temps de codage et la réduction a pu atteindre un gain de temps de plus de 65% en comparaison avec celui obtenu par la classification de Fisher et de plus de 72% avec celui obtenu par une recherche exhaustive.

ABSTRACT. The Fractal based image compression has the advantage of presenting fast decoding and independent resolution. But, it suffers of slow encoding phase: due to a time-consuming similarity search between image blocks. One way to remedy at this problem is to classify image blocs into a number of classes, and comparing only the domain blocs falling in the same class as the range bloc. In the present study, we propose to reduce the computational complexity by an original approach using two domain pools. This approach can be combined with classification methods leading to more reduction in encoding phase. Indeed, experimental results showed that the proposed fast encoding approach speed up the encoding time. The time reduction obtained reached a percentage of more than 65% in comparison with that obtained with Fisher classification and more than 72% in comparison with that obtained with exhaustive search.

MOTS-CLÉS : Fractale, compression, IFS, Codage/décodage, classification.

KEYWORDS: Fractal, compression, IFS, Coding/decoding, classification.

1. Introduction

La compression d'images basée sur la théorie des fractales a été proposée pour la première fois par Barnsley [1]. Cette méthode, basée sur le théorème du collage [2], montre qu'il est possible de coder des images fractales à l'aide de quelques transformations contractantes définissant un système de fonctions itérées (IFS). Barnsley proposa un algorithme pour construire, à partir d'une image donnée, un ensemble de transformations contractantes la représentant. Les signaux naturels ne possédant pas forcément la propriété d'auto-transformabilité globale, Jacquin [8] proposa de rechercher des auto-transformabilités locales ou partielles ce qui a conduit au premier algorithme de compression par IFS et à la notion de Local Iterated Function Systems (LIFS).

Pour coder une image réelle par fractales, on effectue un partitionnement adapté de l'image où chaque partie élémentaire (bloc cible) est mise en correspondance avec une autre partie d'échelle différente (bloc source) recherchée dans toute l'image [8]. Les blocs peuvent être de taille variable, de forme carrée (partition en quadtree [5]) ou de forme rectangle (partition H-V [6]) ou triangulaire [3]. Chaque correspondance est décrite par une transformation affine locale et l'union de ces transformations (LIFS) forme le code de l'image. Le calcul du LIFS pendant le processus de codage est très long car pour chaque bloc cible, le bloc correspondant est recherché parmi tous les blocs sources. L'ensemble de ces blocs sources est appelé dictionnaire. De nombreux travaux sur l'accélération de la génération des LIFS ont été réalisés. Les principales approches de réduction du temps de calcul des LIFS se basent sur la classification des blocs. Dans ce schéma de classification, les blocs cible et source sont groupés en un nombre fini de classes selon leurs caractéristiques communes. Durant la phase du codage, seul les blocs sources ayant la même classe que le bloc cible, seront comparés à ce dernier. Jacquin [9] a proposé un schéma de classification basé sur les caractéristiques discrètes des blocs. Seul trois classes de blocs sont distinguées : les blocs homogènes, blocs avec contour et blocs texturés. Dans la méthode de classification de Fisher [7], un bloc (cible ou source) est divisé en quatre quadrants. Pour chaque quadrant, la moyenne et la variance sont calculées. Selon une certaine combinaison de ces valeurs, 72 classes sont construites. Une autre méthode de classification des blocs, basée sur la DCT, consiste à classer les blocs en trois classes : bloc homogène, bloc avec contour diagonal/sous-diagonal et bloc avec contour horizontal/vertical [4]. Elle est basée seulement sur le calcul de deux coefficients de la DCT à savoir les coefficients horizontaux et verticaux de plus basses fréquences.

Dans ces approches de réduction de temps de codage citées, un seul dictionnaire est utilisé pour chercher les similarités entre blocs cible et source. Ce dictionnaire est construit avec un pas de chevauchement de blocs sources fixé au départ à P. Dans le but

d'accélérer davantage le codage fractal d'images, nous avons conçu dans notre travail, une approche qui utilise deux dictionnaires et une approximation de l'image en deux étapes.

2. Principe de base de la compression fractale

La plupart des algorithmes de compression fractale sont des variations de l'algorithme de Jacquin [9], [10]. Ils construisent, pour une image donnée I, un ensemble de transformations (simples) localement contractantes (LIFS) W_i qui, itéré sur une image de départ quelconque, donne une approximation de cette image (attracteur du LIFS : A = W_i (A)). La construction des transformations s'inspire du théorème du collage. En effet, elle minimise la distance entre l'image originale I et sa transformée par le LIFS $W_i(I)$ au lieu de celle entre I et l'attracteur du LIFS A.

Pour définir les transformations formant W_i, il faut préciser leurs domaines de départ et d'arrivée. Les domaines d'arrivées forment une partition $\{R_1, R_2, \ldots, R_N\}$ de l'image, sont appelés blocs cible et de tailles B x B. Les domaines de départ sont également des blocs de l'image de taille 2B x 2B, $\{D_1, D_2, \ldots, D_M\}$ qui peuvent se chevaucher et sont appelés blocs source et l'ensemble des blocs source est appelé dictionnaire. Chaque bloc cible est ensuite mis en correspondance avec un autre bloc transformé $W_i(D_j)$ lui ressemblant, au sens d'une mesure d'erreur sur les niveaux de gris. La complexité de la recherche du meilleur bloc source est linéaire en M, le nombre d'éléments du dictionnaire. Le code fractal est formé des paramètres des transformations (position du bloc source et coefficients de la fonction affine agissant sur les niveaux de gris). La distance utilisée dans le calcul des erreurs de collage est la distance euclidienne.

La transformation w_i des blocs source comprend une transformation géométrique qui déforme le support des blocs D_j et une transformation massique qui agit sur la luminance des pixels des blocs D_j. Chaque w_i exprime (code) la manière de passer d'un bloc D_j de l'image à un autre bloc R_i de taille plus petite lui ressemblant. La transformation affine d'un niveau de gris z=f(x,y) d'un pixel de position (x,y) de l'image A est du type de l'équation (1).

$$w_i \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} a_i & b_i & 0 \\ c_i & d_i & 0 \\ 0 & 0 & s_i \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} e_i \\ f_i \\ o_i \end{bmatrix} \quad (1)$$

Pour la compression, le nombre de transformations géométriques possibles est limité à 8 isométries applicables aux blocs carrés (identité, rotation de $\pi/2$, π, $3\pi/2$, symétrie horizontale, verticale et les 2 symétries diagonales). Ainsi, les coefficients a_i, b_i, c_i, d_i, qui décrivent la transformation géométrique, ne peuvent prendre qu'un nombre fini de valeurs. Le vecteur (e_i, f_i) permet de translater le support du bloc source pour le faire

coïncider avec celui du bloc cible. Ces deux transformations constituent la transformation spatiale définie par Jacquin. Les coefficients o_i et s_i concernent respectivement une différence de moyenne des niveaux de gris et une variation de contraste entre blocs cible et blocs source.

Le calcul d'une transformation w_i, constituant le code LIFS d'une image A, se fait par minimisation de l'erreur quadratique entre R_i et $w_i(D_j)$. Cette erreur s'écrit :

$$R_{MSE} = \frac{1}{N}\sum_{k=1}^{N}\left(sD_j(k)+o-R_i(K)\right)^2 \qquad (2)$$

N étant le nombre total de pixels dans le bloc cible. La minimisation de cette erreur, en fonction du facteur d'échelle s (contraste) et du rehaussement de niveau de gris o (offset), est assez immédiate puisqu'il suffit d'annuler les dérivées partielles en s et o. On obtient alors les valeurs optimales de s et o.

Pour le décodage, il suffit alors de commencer par n'importe quelle image initiale A_0 et d'appliquer les transformations w_i plusieurs fois (6 à 10 le plus souvent). L'algorithme tend alors vers le point fixe Ã qui constitue l'approximation fractale de l'image initiale A.

3. Réduction du temps de calcul par l'approche proposée

Dans une recherche exhaustive ou dans les méthodes de classification utilisées pour l'accélération du codage d'images, en l'occurrence la classification de Fisher, un seul dictionnaire D(P) est construit avec un pas de chevauchement de blocs source fixé à P au départ. Pour réduire le temps de calcul nous avons utilisé l'approche décrite ci-après qui peut être utilisée avec une recherche exhaustive ou combinée avec les méthodes de classification. Nous manipulons deux dictionnaires $D_1(2P)$ et $D_2(P)$. $D_1(2P)$ est construit avec un pas de chevauchement qui est multiple de 2 (2P). Le Dictionnaire $D_2(P)$ est crée en utilisant le pas de chevauchement égal à P. Nous cherchons une première approximation de l'image en comparant les blocs cible avec les blocs source du dictionnaire $D_1(2P)$. Quant aux blocs cible mal approchés résultant de cette première approximation, ils sont comparés aux blocs source du deuxième dictionnaire $D_2(P)$ en omettant de ce dernier les blocs source du dictionnaire D(2P). De cette manière, nous diminuons le nombre de comparaisons entre blocs cible et source, et par conséquent le temps de codage serait réduit.

En effet, si nous considérons une image de taille NxN et des blocs cible de taille nxn. Le nombre de blocs cible, N_{TC}, est donné par :

$$N_{TC} = \left(\frac{N}{n}\right)^2 \qquad (3)$$

Le nombre de comparaisons, NC_{RE}, entre blocs cible et blocs source appartenant au dictionnaire D(p), dans une recherche exhaustive est donné par :

$$NC_{RE} = N_{TC} \times \left(\frac{N - 2n + 1}{p} \right)^2 \quad (4)$$

D'autre part, le nombre de comparaisons, NC_{AP}, entre blocs cible et source dans l'approche proposée, est donné par :

$$NC_{AP} = \underbrace{N_{TC} \times \left(\frac{N - 2n + 1}{2p} \right)^2}_{(a)} + \underbrace{N_M \times \left(\left(\frac{N - 2n + 1}{p} \right)^2 - \left(\frac{N - 2n + 1}{2p} \right)^2 \right)}_{(b)} \quad (5)$$

N_m est le nombre des blocs cible mal approchés issus de la première approximation de l'image avec le dictionnaire D(2p). Le terme (a) correspond au nombre de comparaisons dans la première approximation et le terme (b) correspond au nombre de comparaisons des blocs cible mal approchés avec les blocs source issus de D(p) en omettant les blocs source issus de D(2p) puisqu'ils étaient déjà comparés avec ces derniers. Du fait que $N_{TC}=N_A+N_M$ (N_A est le nombre de blocs cible bien approché lors de la première approximation) et après simplification, (5) devient :

$$NC_{AP} = \frac{N_A}{4} \times \left(\frac{N - 2n + 1}{p} \right)^2 + N_M \times \left(\frac{N - 2n + 1}{p} \right)^2 \quad (6)$$

Il est évident, d'après (4) et (6), que NC_{AP} est très inférieur à NC_{RE}.

4. Résultats et discussion

Les différents tests sont effectués sur les trois images représentées sur la figure 3 avec 8 bpp sur un AMD Athlon XP 2000 à 1,6 Ghz et à 256 MO de RAM. Le partitionnement quadtree est adopté pour le codage fractal.

(a) (b) (c)

Figure 3. Images, de taille 256x256, de : Lenna (a), Collie (b) et San256 (c).

Une comparaison, en terme de temps de calcul et du PSNR, entre l'approche proposée et une recherche exhaustive a été établie. Les résultats de cette comparaison sont représentés sur le tableau 1. Ils montrent que les temps de codage obtenus par l'approche proposée sont plus courts que ceux obtenus par une recherche exhaustive. Cette réduction atteint des pourcentages supérieurs à 72%.

Image	Recherche exhaustive				Approche proposée				Gain en Temps
	Pas	Temps (s)	PSNR (dB)	Taux	Pas	Temps (s)	PSNR (dB)	Taux	
Lenna	2	152.43	31.31	10.32	4 après 2	41.21	30.92	10.46	72.97%
	1	670.83	31.57	9.87	2 après 1	168.14	31.31	10.32	74,94%
Collie	2	118.62	32.77	14.14	4 après 2	30.24	32.72	13.98	74.50%
	1	526.15	32.92	13.74	2 après 1	121.98	32.77	14.14	76.81%
San256	2	181.37	30.81	7.78	4 après 2	49.70	30.13	8.25	72,60%
	1	821.90	31.25	7.50	2 après 1	215.21	30.81	7.78	73.81%

Tableau 1. *Comparaison entre l'approche proposée et une recherche exhaustive*

Les résultats de la comparaison entre l'approche proposée et la classification de Fisher sont représentés sur le tableau 2. Ces résultats montrent aussi que les temps de codage obtenus par l'approche proposée sont plus courts que ceux obtenus par la classification de Fisher. Cette réduction a atteint plus de 65%.

Image	Classification de Fisher				Approche proposée				Gain en temps
	Pas	Temps (s)	PSNR (dB)	TAUX	*Pas*	Temps (s)	PSNR (dB)	TAUX	
Lenna	2	3.78	30.58	9.67	4 après 2	1.12	30.09	9.62	65.60%
	1	16.37	31.04	9.48	2 après 1	4.45	30.58	9.67	72.81%
Collie	2	3.82	32.77	12.40	4 après 2	1.07	32.58	12.03	72.99%
	1	15.54	32.80	12.56	2 après 1	4.12	32.77	12.40	73.48%
San256	2	5.14	29.68	7.57	4 après 2	1.54	28.78	7.99	70.03%
	1	22.11	30.34	7.26	2 après 1	6.33	29.68	7.57	71.37%

Tableau 2. *Comparaison entre l'approche proposée et la classification de Fisher*

Le tableau 3 montre que le nombre de comparaisons a diminué de manière importante par l'approche proposée. Cette diminution est la conséquence de l'approche qui utilise deux dictionnaires $D_1(2P)$ et $D_2(P)$ au lieu d'un seul. En effet, dans la première approximation, le cardinal du dictionnaire $D_1(2P)$ est inférieur au cardinal de D(P). Les blocs mal approchés, résultant de cette première approximation qui sont comparés aux blocs sources de $D_2(P)$, sont en nombre limité.

Image	Classification de Fisher		Approche proposée	
	Pas	Nombre de comparaisons entre blocs cible et source	Pas	Nombre de comparaisons Entre blocs cible et source
Lenna	2	1 521 161	4 après 2	802 577
	1	5 735 113	2 après 1	2 800 848
Collie	2	1 420 230	4 après 2	512 251
	1	5 168 830	2 après 1	1 694 572
San256	2	2 352 518	4 après 2	1 421 847
	1	9 166 638	2 après 1	5099213

Tableau 3. *Effet de l'approche proposée sur le nombre de comparaisons entre blocs cibles et sources*

La réduction du temps obtenue est accompagnée simultanément d'une légère augmentation du taux de compression et une faible dégradation de la qualité de l'image.

5. Conclusion

Dans cette étude, nous avons réduit le temps de codage fractal d'images en utilisant deux dictionnaires et une approximation d'images en deux étapes. Le premier

dictionnaire, construit avec un pas égal à 2P, sert à une première approximation. Les blocs mal approchés, résultant de cette dernière, sont comparés aux blocs sources appartenant au dictionnaire construit avec un pas égal à P. Nous avons comparé les résultats de notre approche à ceux obtenus par les approches avec un seul dictionnaire à pas égal à P. La comparaison, en terme de temps de calcul, montre que l'approche proposée a permis une réduction de plus de 65% par rapport à la classification de Fisher et de 72% par rapport à une recherche exhaustive. Cette réduction s'explique par la réduction du nombre de transformations induite également par notre approche.

6. Bibliographie

[1] Barnsley M. F., Sloan A. D., A better way to compress images, *BYTE magazine*, 1988, pp. 215-223.

[2] Barnsley M. F., *Fractal every where*, New-york: Academic Press, California, 1988.

[3] Davoine F., Antonini M., Chassery J. M., Barland M., Fractal Image compression Based on Delaunay Triangulation and Vector Quantization. IEEE Trans. *image Processing*, 1996, Vol. 5, N° 2, p. 338-346.

[4] Duh D. J., Jeng J. H., Chen S.Y. DCT based simple classification scheme for fractal image compression, *Image and vision computing* 23 , 2005 P. 1115-1121.

[5] Fisher Y., Fractal encoding with quadtree, Chapter 3 in *Fractal Image Compression: Theory and Applications to Digital Images*, Yuval Fisher, Ed, New York: Springer-Verlag, 1995, pp. 55-77.

[6] Fisher Y., Menlove S., Fractal encoding with HV partitions, Chapter 6 in *Fractal Image Compression: Theory and Applications to Digital Images*, Yuval Fisher, Ed, New York: Springer-Verlag, 1995, pp. 119-136.

[7] Fisher Y., *Fractal Image Compression : Theory and Application,* Springer-verlag, 1995 , New York: Springer-Verlag, 1995, 341 P.

[8] Jacquin A. E., A fractal theory of iterated Markov operators on spaces of measures with applications to digital image coding, PhD Thesis, Georgia Institute of Technology, 1989.

[9] Jacquin A. E., Image coding based on a fractal theory of iterated contractive image transformations, IEEE Trans. *Image Processing*, 1992, Vol. 1, N°.1, pp.18-30.

[10] Jacquin A. E., Fractal image coding : A review, in *Proceedings of IEEE*, 1993, Vol. 81, N°.10, pp.1451-1465.