

Crypto-Compression d'Images Fixes Par la méthode de Quadtree optimisée et AES

S. Ftérich¹ C. Ben Amar¹

¹ REGIM (Groupe de REcherche sur les Machines Intelligentes)

Ecole Nationale d'Ingénieurs de Sfax (ENIS)
Route de Soukra, B.P. W, 3038 Sfax – Tunisie

{souhir.fterich, chokri.benamar}@ieee.org

Résumé :

L'utilisation des technologies de l'information et des télécommunications dans la vie quotidienne a évolué ces dernières années d'une façon notable. La compression et le cryptage de données sont deux technologies dont l'importance croît d'une manière exponentielle dans une myriade d'applications.

En outre, l'usage excessif des réseaux informatiques pour le transfert des données doit évidemment obéir à un double objectif: la réduction du volume des données afin de désencombrer le maximum possible les réseaux publics de communication et la confidentialité en vue de garantir un niveau de sécurité optimum. Dans ce sens et afin d'assurer l'optimisation et la sécurisation de la transmission et du stockage des images fixes, nous proposons dans ce travail une nouvelle approche hybride de crypto-compression qui applique un cryptage à base de l'algorithme AES sur les paramètres de la compression par la technique de Quadtree optimisée.

Mots clefs :

Compression, cryptage, quadtree, crypto-compression, distorsion.

1 Introduction

Les approches de crypto-compression classiques ont toutes tendance à réaliser les techniques de cryptage et de compression de manière disjointe; ceci posait un problème au moment des étapes de décryptage et de décompression, surtout pour le cas de certains domaines d'applications du type temps réel comme l'émission des images par satellite ou encore la télémédecine où le facteur temps est primordial.

Ainsi de nouvelles approches mixtes de crypto-compression commencent à prendre de l'essor. Le concept visait à combiner à la fois les deux techniques de cryptage et de compression de manière à ce qu'elles soient effectuées de manière jointe. Le challenge est toujours de procurer pour toutes nos applications un volume de données de taille réduite ainsi qu'une confidentialité robuste. Il s'agit alors de parvenir à trouver une approche efficace de crypto-

compression. C'est dans cette thématique que notre travail se situe; nous essayons de décrire le processus d'une technique hybride de crypto-compression à base de la méthode de Quadtree optimisée et AES.

2 Principe de l'approche

2.1 Problématique

Nous avons appliqué le principe de cryptage immédiatement sur les paramètres appelés au moment de l'étape de la compression; le concept mixte serait ainsi fructueux et permettra d'atteindre de bonnes performances.

Nous aurons recours en un premier volet à l'étude du principe de la technique de compression appelée « Quadtree » [1]; nous décelons ensuite les étapes primordiales en vue de leur faire appliquer un algorithme de cryptage puissant : l'AES.

2.2 Quadtree et Compression par Quadtree

Les Quadtrees sont une sorte d'arbres pratiques et utiles pour représenter et manipuler des images [2]. D'un point de vue structure de données, les quadtrees sont des arbres dont les nœuds contiennent soit quatre fils soit une information utile. Chaque nœud correspond à une zone de l'image à représenter. La racine correspond à l'image complète. Soit elle contient une information qui est relative à l'image entière, soit elle possède quatre fils qui correspondent aux quatre sous images obtenues en divisant en deux horizontalement l'image puis encore en deux verticalement. Chacun de ces fils peut alors soit contenir une information relative à sa sous image, soit posséder à son tour quatre fils obtenus aussi par divisions successives et ainsi de suite jusqu'à ce que les nœuds correspondent à des zones formées d'un seul pixel, auquel cas ils ne peuvent être divisés et ne peuvent donc que contenir une information relative à ce pixel, son niveau de gris par exemple.

De manière habituelle, nous commençons par développer totalement l'arbre, c'est à dire jusqu'au niveau des nœuds correspondants à un seul pixel, puis nous stockons dans ces nœuds l'information de chaque pixel: 0 ou 1 pour une image binaire, un niveau de gris ou un triplet (r, g, b) pour

des images en niveaux de gris ou en couleur respectivement. Ensuite, lorsqu'un nœud possède quatre fils qui sont porteurs de la même information, nous supprimons les quatre fils et nous remontons l'information au niveau de leur père. Cela permet d'arriver à une représentation plus compacte d'une image, surtout si elle contient des zones uniformes. [2,3]

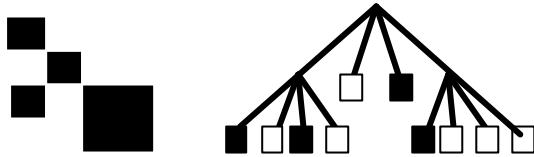


Figure1- Exemple de correspondance entre image binaire et Arbre Quaternaire

2.3 La compression par la technique de Quadtree Optimisée

Nous conservons toujours le principe de la compression à base des Quadtrees, pour introduire la technique de Quadtree Optimisée. Elle consiste en un codage toujours basé sur la notion de segmentation selon une structure d'arborescence ("tree") et qui parvient à atteindre un comportement asymptotique correct en terme de taux de distorsion : $R-D$ ("rate-distorsion") ou encore la distorsion en fonction du nombre de bits alloué. Cette technique est initialement conçue pour une famille bien distinguée de signaux mono-dimensionnels connue sous le nom: "piecewise polynomials functions" [4], puis une extension à permis de généraliser le processus pour le cas bidimensionnel, autrement le cas des images fixes. L'innovation vient du fait que la segmentation est plutôt guidée par un critère de distorsion : Le coût Lagrangien, cela permet finalement de réaliser une compression avec une distorsion minimale et donc optimale. Le coût Lagrangien J est décrit selon la formule : $J = D + \lambda R$ où R est le nombre de bits alloué théorique, D la distorsion correspondante et λ un facteur multiplicateur Lagrangien. Le processus alloue pour chaque nœud de l'arborescence un nombre de bits théorique permettant d'atteindre le $R-D$ optimal. Il serait important de rappeler que les performances atteintes par cette technique découlent directement de deux étapes appelées au cours de l'algorithme:

❖ **Le taillage des nœuds dans une structure de Tree**

❖ **Le codage joint des nœuds similaires**

Algorithme :

L'algorithme de compression par la technique de Quadtree Optimisée est décrit à travers les étapes suivantes :

a- Taillage de l'arbre

1^{ère} Etape : Initialisation

- 1- Segmentation de l'image entière en un arbre Quaternaire.
- 2- Approximation de chacun des nœuds obtenus après la décomposition à un modèle géométrique P .
- 3- Génération du couple (R,D) (taux de bits alloué, distorsion) théorique relatif à chaque nœud par une quantification scalaire des coefficients du polynôme P .
- 4- Le coût Lagrangien est ensuite calculé pour tous les nœuds d'un niveau donné ainsi que pour leur nœud père (bloc original): Le Coût Lagrangien est donné par

$$J = D + \lambda R$$

2^{ème} étape : Taillage de nœuds

- 5- L'objectif est toujours de parvenir à une structure d'arbre quaternaire ayant une valeur optimale voir même idéale en terme de $R-D$. Ainsi pour une valeur donnée du multiplicateur Lagrangien $(-\lambda)$, le critère optimal pour le taillage des nœuds est obtenu de la façon suivante :

"Tailler les nœuds fils si la somme de leurs coûts Lagrangiens est supérieure ou égale au coût Lagrangien du nœud père".

$$(D_{n+1} R_{n1}) + \dots + (D_{n4+1} R_{n4}) \geq (D_{n_joint+1} R_{n_joint})$$

Ce critère est utilisé récursivement pour tous les nœuds fils ainsi que leurs parents dans la structure d'arbre. Comparé à la structure d'arbre originale obtenu directement après la décomposition, le sous-arbre récupéré comme résultat à la fin est celui doté d'un taux de distorsion $R-D$ idéal.

- 6- Les blocs récupérés à travers l'arborescence subiront ensuite un codage, il s'agit de la prise en compte de la similarité des paramètres dans un bloc homogène, et coder uniquement l'information que porte le bloc et son ampleur.

- 7- Après taillage, les nœuds parents deviennent tous des feuilles (sans descendance) et possèdent donc pour une valeur donnée de λ le taux de distorsion idéal. La sommation à travers l'arbre entier des bits alloués aux feuilles permet de déterminer le taux de bits total de la structure de Tree : $R^*(\lambda)$, de même la sommation des taux de distorsion au niveau de toutes les feuilles déterminent le taux de distorsion net de l'arbre en entier: $D^*(\lambda)$ estimé donc idéal.

3^{ème} étape : Recherche du paramètre l

8- L'objectif est de déterminer de manière itérative le paramètre l qui correspond à une valeur désirée pour le taux de bits total R_0 .

Au début, nous déterminons l_{Min} et l_{Max} de façon à ce que : $R^*(l_{Max}) = R_0 = R^*(l_{Min})$

9- l_{New} est la nouvelle valeur de pente supposée satisfaisante.

$$l_{New} = (D^*(l_{Min}) - D^*(l_{Max})) / (R^*(l_{Max}) - R^*(l_{Min}))$$

10- Nous recalculons le nouveau coût Lagrangien décrit dans l'étape 2 pour la nouvelle valeur l_{New} .

Si $(R_0 = R^*(l_{New}))$ alors l'optimum converge
Sinon

Si $(R_0 < R^*(l_{New}))$ Alors

$$l_{Min} = l_{New}$$

$$\text{Sinon } l_{Max} = l_{New}$$

b- Codage joint de l'arbre taillé

Jusqu'ici nous avons essayé de bien guider le processus de segmentation pour pouvoir garantir un taux de distorsion idéal. Une reproche est immédiatement évoquée : pourquoi ne pas fusionner les nœuds voisins portant une information identique, même si le prétexte est le fait que ces nœuds ne dérivent pas de même parents ?. Pour corriger un tel comportement du R-D, nous proposons une extension de la méthode citée ci-dessus. Il s'agit d'une technique de Codage Joint qui agit sur les feuilles de l'arbre taillé, qui exploite bien la dépendance entre les nœuds même s'ils n'appartiennent pas au même nœud père. Une étape de recherche de nœuds voisins est alors effectuée [4] ; et une évaluation des coûts Lagrangiens est réalisée. Ainsi deux nœuds $n1$ et $n2$ seront codés ensemble de manière jointe si la somme de leurs coûts Lagrangiens est supérieure ou égale au coût Lagrangien du bloc Joint : autrement dit si :

$$(D_{n1} + l_{Rn1}) + (D_{n2} + l_{Rn2}) = (D_{n_joint} + l_{Rn_joint})$$

Dans le cas où les voisins sont codés de manière jointe, une variable de codage joint sera mise à « 1 » et l'information contenu dans le nœud joint est stockée à la place de l'un des nœuds, sinon la variable de codage joint sera mise à « 0 » et les informations des nœuds feuilles seront gardées telles qu'elles étaient. Rappelons qu'une fois un bloc joint est construit, il serait traité au sein de l'arbre comme une feuille à la place de ses deux feuilles d'origine, et elle peut servir par suite pour une éventuelle opération de codage joint. Les étapes déjà citées se voient changer l'ordre : nous commençons par l'initialisation, puis le codage joint et finalement la recherche du paramètre l.

3 le cryptosystème AES

La technique fait appel, pour la phase de cryptage, à l'algorithme AES ("Advanced Encryption Standard"); il s'agit du successeur du fameux DES ("Advanced Encryption Standard") qui a été implémenté dans un grand nombre de modules cryptographiques à une échelle mondiale depuis son apparition en 1977 [5].

L'algorithme AES conserve toujours le haut niveau de sécurité proposé par le DES ; effectivement le processus est toujours basé sur une fonction d'expansion E, des boîtes de substitution S, appelées au niveau de la diversification de clé K ; en outre le processus conserve toujours le principe des étages au moment de l'étape d'expansion. L'innovation apportée par l'AES est notée au niveau de la taille de la clé secrète ainsi que la taille des données traitées en entrée ; justement nous passons d'une clé de chiffrement de taille 64 bits (8 octets) pour le cas du DES vers une clé de taille double 128 bits (16 octets) pour l'AES. La taille des données à crypter est aussi notablement plus grande que celle du DES puisque nous passons de 56 bits vers 128 bits. En outre, comme pour le DES, l'AES est un système cryptographique à clé secrète ; ce qui rend l'opération de cryptage-décryptage assez légère. La taille des données traitées par l'AES (16 octets) nous donne bien la possibilité d'exploiter l'algorithme dans des applications supportant des fichiers de données de taille grande [6].

4 Schéma de Principe de l'approche de Crypto-Compression proposée

4.1 Principe

L'idée primordiale est de combiner compression et cryptage au cours de la procédure, il s'agit donc d'appliquer immédiatement le cryptage sur les données de la compression. Nous allons pour cela travailler directement sur la structure d'arbre récupéré après taillage et après décision du codage joint si possible ; nous nous trouvons alors dans la seconde étape de l'algorithme.

Le résultat est une arborescence taillée comportant des blocs de tailles variables, dont quelques uns demandent un codage joint. Le premier pas de cryptage est de choisir un ordre d'exploration de l'arbre qui soit le plus difficile à dévoiler ; la meilleure solution est de suivre un chemin aléatoire pour éviter le maximum possible son obtention par une recherche heuristique. Les blocs ainsi obtenus, tels qu'ils sont lus dans l'ordre de parcours proposé de l'arbre ne permettront pas une reconstruction évidente de l'image originale.

Ces blocs sont par la suite codés ; Ainsi La technique du cryptage sélectif appliquée à la compression par Quadtree peut être utilisée à la fois pour la compression avec et sans perte.

Dans le cas de compression sans perte, au moment du cryptage les valeurs d'intensité sont tous représentées avec

le même nombre de bits. Par contre dans le cas de compression avec perte, le nombre de bits utilisé pour représenter l'intensité est variable [4]. Une formule est proposée pour l'allocation de bits dans ce cas, soit :

$$b_i = \frac{1}{2} \log(s_i^2 L / 4^i D)$$

- ❖ **b_i** : le nombre de bits alloué pour représenter chaque valeur de feuille d'un niveau i
- ❖ **s_i²** : la variance des valeurs des feuilles au niveau i
- ❖ **L** : le nombre total des nœuds feuilles
- ❖ **D** : une constante spécifiée par l'utilisateur pour contrôler le taux de bits

C'est à ce moment où nous avons recours à l'algorithme AES ; il s'agit d'une opération d'adaptation de l'algorithme AES de cryptage à des données bidimensionnelles de tailles variables relativement à la taille inchangée de données en entrée que supporte l'AES au moment de cryptage (128 bits toujours). Les blocs crypto-compressés sont disposés linéairement selon l'ordre choisi du parcours de l'arbre, puis insérés matriciellement en vue de retrouver finalement une image crypto-compressée.

Nous tacherons avant cryptage, de sauvegarder les blocs codés dans un fichier à part pour une éventuelle évaluation de la qualité de compression de la technique proposée. Le chemin inverse est toujours réalisable, il suffit de récupérer les blocs crypto-compressés à partir de l'image crypto-compressée, de les décrypter et les decoder; puis les lire dans le chemin correct de parcours de l'arbre initial.

4.2 Schéma de Principe

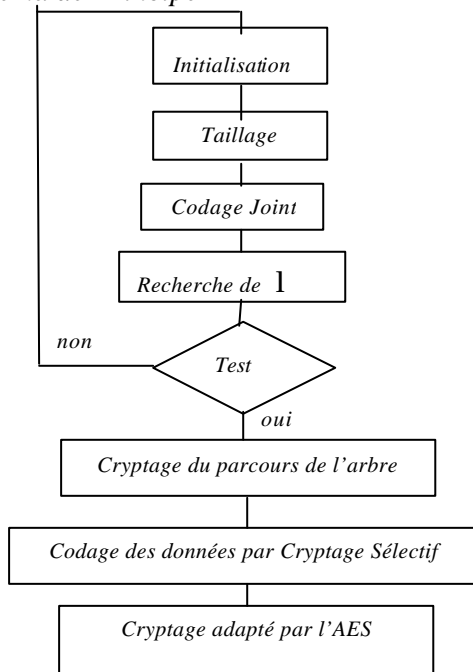


Figure2 - Organigramme de l'approche hybride

5 Implémentation et Résultats

Nous avons eu recours au schéma précédent de principe de la technique et nous avons ensuite dégagé les grandes étapes de travail. Le travail a été réalisé sur une large panoplie d'images pour pouvoir noter le maximum de remarques, comprendre les détails et les origines de quelques problèmes rencontrés en vue de pouvoir à la fin construire un jugement objectif à propos de la technique. Une étape d'évaluation des performances est d'une importance primordiale afin de pouvoir situer le niveau de la robustesse du système de crypto-compression proposé ; nous évaluons la technique mixte globale de crypto-compression ainsi que la technique de compression utilisée, ceci pour pouvoir déceler le cadre correct d'un problème en cas d'un échec. Nous avons, au début, traité le cas des images binaires mais nous nous sommes ensuite plutôt concentrés sur le cas des images à niveaux gris. Plusieurs critères de performance sont pris en considération ; nous citons le calcul du PSNR et d'entropie ; nous proposons ici quelques résultats.

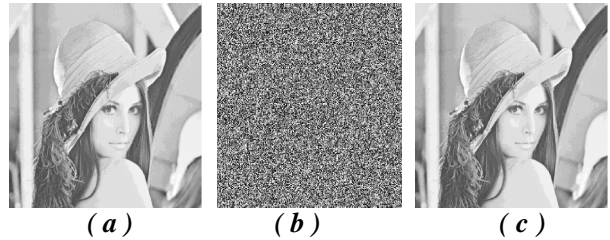


Figure3 - L'image test Lena : (a) image originale, (b) image crypto-compressée, (c) Image reconstruite

PSNR	MSE	Entropie originale	Entropie de l'image reconstruite
35.08	3.687	7.589	7.014

Tableau 1 - Analyse de résultats de l'image test Lena

Images de test	PSNR en dB
Lena	35.08
Alumgrns	36.03
Blood	37.1
Cameraman	35.6
Woman	34.5

Tableau 2 - PSNR des images reconstruites (après décryptage et décompression)

Dans ce qui suit nous proposons une évaluation des erreurs quadratiques moyennes MSE ainsi que le rapport Signal/bruit : PSNR des deux techniques de crypto-compression à base de Quadtree_AES et JPEG-DES. Nous allons plusieurs fois pour fixer le taux de bits alloué pour chaque pixel d'une image donnée et calculer le taux

de distorsion relatif à l'image reconstruite à partir de la technique à base de Quadtree-AES ainsi que celle à partir de JPEG-DES. Le résultat est exprimé dans la figure ci-dessus.

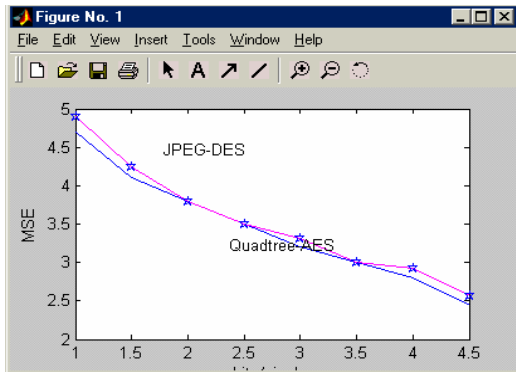


Figure 4 - Calcul du MSE en fonction de nombre de bits/pixel

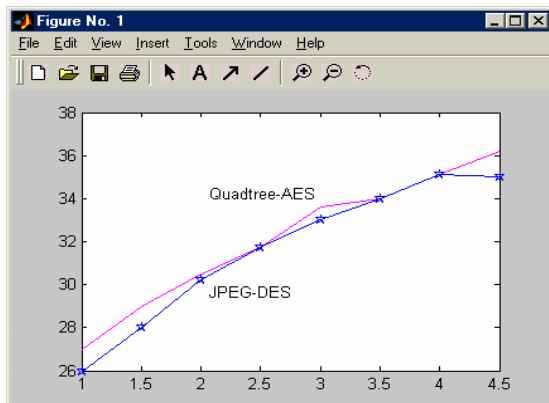


Figure 5 - Calcul du PSNR en fonction de nombre de bits/pixel

6 Conclusion

Nous nous sommes basés sur la technique de compression par la méthode de Quadtree pour élaborer une nouvelle technique mixte de crypto-compression, vérifiant un certain nombre de contraintes et dotée d'une bonne robustesse. Nous avons eu recours et testé une large panoplie d'images, et à chaque fois un test d'évaluation a été réalisé dans le but d'obtenir une idée plus objective sur la robustesse de l'approche proposée. Ces tests ont montré que la technique est performante.

Il serait bénéfique et nécessaire d'approfondir encore l'étude de cette technique, et de la comparer davantage avec d'autres méthodes existantes.

Comme l'application est orientée vers la crypto-compression des images fixes, une extension de la méthode dans le domaine de la vidéo (images animées) est tout à fait envisageable. Nous pouvons aussi combiner cette technique avec d'autres techniques de crypto-compression en vue d'améliorer encore la robustesse.

7 Références

- [1] Cheng et Xiaobo Li. Partial Encryption of Compressed Images and Videos. Howard, Août 2000.
- [2] H.K.C.Chang et J.L.liu .A linear quadtree compression scheme for image encryption. Signal processing: image communication, Taiwan 1997.
- [3] Xiabo Li, Jason Knipe et Howard Cheng. Image compression and encryption using tree Structures, Pattern Recognition Letters, Alberta et Canada 1997.
- [4] R.Shukla, P.Luigi. Dragotti, Minh Do. Rate-Distorsion Optimized Tree tructured compression Algorithms for Piecewise smooth images, Lausanne, Suisse 2000.
- [5] Douglas Stinson Cryptographie : théorie et pratique, Vuibert Informatique, 2001.
- [6] Xavier Marsault : Compression et cryptage des données Multimédias, Hermes, 1997.