Énoncé de TP module : TNM2
Master I ILM

Mr Z. MAHROUK

1. Introduction

Le principe des méthodes de codage d'image par transformation est de transformer un bloc de pixels statistiquement dépendants en un bloc de coefficients dont l'interdépendance statistique est nulle ou faible. Pour cela, des transformations orthogonales sont souvent utilisées. Elles rassemblent la plus grande partie de l'énergie du signal sur un nombre réduit de coefficients. La compression est obtenue par :

– la quantification d'un certain nombre de coefficients qui sont transmis,

– l'abandon des autres coefficients.

La sélection des coefficients à retenir et la manière de les quantifier met en jeu des critères psychovisuels.

Les valeurs quantifiées sont ensuite considérées comme les symboles d'entrée d'un codeur entropique.

La transformation qui concentre au mieux l'énergie sur un faible nombre de coefficients est la transformée de Karhunen-Loeve. Cependant, l'utilisation de cette transformée repose sur un modèle d'image parfois peu conforme à la réalité, et ses fonctions de base dépendent de la matrice de covariance de l'image sur laquelle est appliquée : elles doivent donc être calculées et transmises pour chaque image. Le gain en compression est alors faible voir plutôt nul. Enfin, il n'existe pas d'algorithme rapide comme pour les transformées suivantes :

– la transformée de Fourier (DFT) ;

– la transformée en cosinus discrète (DCT) ;

– la transformée de Hadamard ;

– la transformée de Haar.

La DCT offre de bonnes performances pour un coût de calcul modéré. Par rapport à la transformée de Fourier, elle présente l'avantage, de par sa parité, d'introduire moins de discontinuités artificielles aux frontières des blocs. Pour ces raisons elle est largement utilisée, en particulier en codage d'image. De plus, elle n'implique que des calculs réels, ce qui permet des implantations VLSI relativement aisées. La norme de compression JPEG est fondée sur ces principes.

2. Codage par transformée

En notant f la matrice représentant un bloc (un carré de côté N) extrait de l'image et F le bloc transformé, la transformation T et son inverse T^{-1} s'expriment sous forme de produits matriciels :

$$T(f) = F = A.f.A^T$$

$$T^{-1}(F) = f = A^T.F.A$$

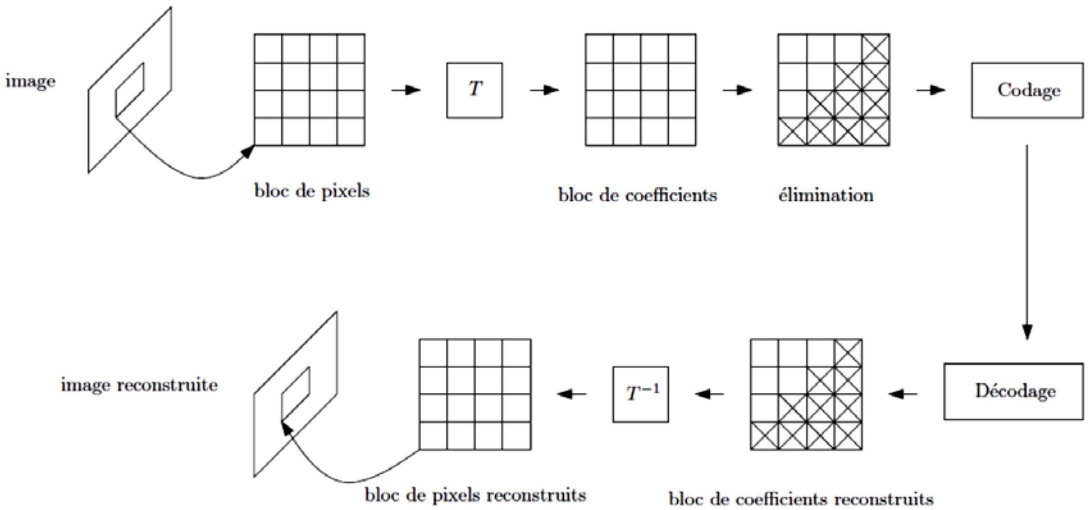

FIG. 1 – Étapes du codage par transformée

Pour la DCT, la matrice A est donnée par

$$A^{\mathsf{T}} = (a_{k,l})_{0\le k,l<N} \quad \text{avec} \quad a_{k,l} = \frac{c(l)}{\sqrt{N}} \cos\left(\frac{(2k+1)l\pi}{2N}\right)$$

où

$$c(l) = \begin{cases} 1 & \text{si} \quad l = 0 \\ \sqrt{2} & \text{si} \quad l \neq 0 \end{cases}$$

Les différentes étapes d'un schéma de codage par transformée sont récapitulées sur la figure 1.

3. Travail à réaliser

Lors de ce TP, vous allez coder les trois premières étapes de base d'un schéma de compression par transformation DCT :

– l'implantation de la transformée,

– la quantification des coefficients,

– le codage des coefficients,

3.1 Algorithme de base : DCT et DCT inverse

Pour une transformée *DCT 2D* de taille $N \times N$, le coefficient *DCT Fi,j* d'indice *(i, j)* s'écrit

$$\begin{aligned} F_{i,j} &= T(f_{i,j}) \\ &= \frac{c(i)c(j)}{N} \sum_{k=0}^{N-1} \sum_{l=0}^{N-1} \left[\cos\left(\frac{(2k+1)j\pi}{2N}\right) \cos\left(\frac{(2l+1)i\pi}{2N}\right) \times f_{l,k} \right]. \end{aligned}$$

De la même façon, le coefficient spatial peut se retrouver à partir des coefficients transformés:

$$f_{i,j} = T^{-1}(F_{i,j})$$

$$= \frac{1}{N} \sum_{k=0}^{N-1} \sum_{l=0}^{N-1} \left[c(k)c(l) \cos\left(\frac{(2j+1)k\pi}{2N}\right) \cos\left(\frac{(2i+1)l\pi}{2N}\right) \times F_{l,k} \right]$$

Remarquez que la fonction cosinus est continument appellée. Il est donc possible de réduire sensiblement le temps de caclul en précalculant la matrice de terme courant :

$$\mathrm{cosp}_{p,q} = \cos\left(\frac{(2p+1)q\pi}{2N}\right)$$

Question 1. Écrivez une fonction ***dct_cospre_calc*** qui renvoie la matrice $N \times N$ des coefficients DCT précalculés.

Question 2. Implantez l'algorithme qui effectue la transformation par bloc d'une image ***(dct),*** puis la transformation par bloc inverse ***(dct_inv).***

Dans la suite de ce TP et en particulier pour les tests, vous prendrez une taille de bloc N = 8.

Question 3. Comparez avec l'image d'origine en utilisant le MSE.

3.2 Aspects psycho-visuels

À ce point, aucune compression n'est encore réalisée dans la représentation transformée. Pour commencer, vous allez annuler un certain nombre de coefficients de chaque bloc transformé.

Question 4. Pour voir les effets visuels de l'élimination de certains coefficients transformés de l'image, utilisez dans le programme la fonction ***appl_masque*** qui annule un certain nombre de coefficients. Ce masque sera appliqué uniformément sur l'image. Les masques à utiliser sont dans la figure suivante :

1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	0
1	1	1	1	1	1	0	0
1	1	1	1	1	0	0	0
1	1	1	1	0	0	0	0

1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	0
1	1	1	1	1	1	0	0
1	1	1	1	1	0	0	0
1	1	1	1	0	0	0	0
1	1	1	0	0	0	0	0
1	1	0	0	0	0	0	0

1	1	1	1	1	1	1	0
1	1	1	1	1	1	0	0
1	1	1	1	1	0	0	0
1	1	1	1	0	0	0	0
1	1	1	0	0	0	0	0
1	1	0	0	0	0	0	0
1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

1	1	1	1	1	0	0	0
1	1	1	1	0	0	0	0
1	1	1	0	0	0	0	0
1	1	0	0	0	0	0	0
1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

1	1	1	0	0	0	0	0
1	1	0	0	0	0	0	0
1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

À chaque fois, reconstruisez l'image et notez le ***PSNR.***

3.3 Quantification

Dans cette partie, vous allez poursuivre le but de réduire le nombre de coefficients, mais d'une manière plus fine. La norme JPEG propose pour cela une matrice Q définissant le pas de quantification à appliquer à chaque coefficient de chaque bloc DCT :

$$Q = \begin{pmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{pmatrix}$$

Un coefficient *DCT* $F_{i,j}$ d'indice (i, j) dans son bloc donnera donc un coefficient DCT quantifié défini par :

$$F^Q_{i,j} = \text{arrondi}\left(\frac{F_{i,j}}{Q_{i,j}}\right)$$

Question 5. Implantez la quantification proposée (***quantdct***) et la fonction ***iquantdct*** permettant d'estimer le coefficient DCT original. Affichez un bloc quantifié.

Question 6. On vous a proposé au cours une méthode simple (paramétrée par un réel α) pour régler "l'âpreté" de la quantification (i.e. : la qualité de la compression). Implantez cette méthode et affichez le même bloc que précédemment en faisant varier α.

3.4 Codage des coefficients

Dans cette partie, vous allez vous intéresser au codage des coefficients quantifiés. Ces coefficients sont usuellement classés en catégories :

– les coefficients DC, c'est-à-dire les coefficients de basse fréquence $F_{0,0}$;

– les coefficients AC, i.e. tous les autres.

Question 7. Le type de parcours choisi est le codage en zig-zag. Implantez-le.

Question 8. Implantez la méthode Cod_Huff, permettant de coder une suite de symboles en utilisant l'algorithme de Huffman. Affichez la table de codage.

Question 9. Utilisez le codage de Huffman pour coder les coefficients non nuls issus de l'étape de quantification

Question 10. Les coefficients nuls sont codés par le couple symbole spécial (# par exemple), Nombre d'apparitions. Avec Nombre d'apparition est codé sur le minimum de bits possibles.

Exemple : 00000000 → #8

Question 11. Évaluez le taux de compression obtenu dans les deux cas :

1. Tous les symboles (nuls est non nuls) sont codés avec le codage de Huffman.
2. Les symboles non nuls utilisent le codage de Huffman, alors que les symboles nuls utilisent le couple ***#, fréquence***