TP Télévision Numérique

Université de Tlemcen

Dr. BELGACEM Nassima

Université de Tlemcen

Faculté de technologie

Département de Télécommunications

Email : *belgacem_nassima@yahoo.fr*

1.0

23/02/2024

Table des matières

I Chapitre II : La compression JPEG

La compression ***JPEG***, acronyme de Joint Photographic Experts Group, est une méthode largement utilisée pour réduire la taille des fichiers image tout en préservant une qualité visuelle acceptable. Cette technique, développée dans les années 1990, est devenue un standard incontournable pour le stockage et le transfert d'images numériques sur Internet et dans divers domaines tels que la photographie, la médecine et la conception graphique.

Le processus de compression ***JPEG*** repose sur la suppression sélective des informations superflues et la conservation des détails essentiels de l'image. Il utilise une combinaison de deux types de compression : la compression avec perte et la compression sans perte. La compression avec perte réduit la taille du fichier en éliminant les données qui sont moins perceptibles à l'œil humain, tandis que la compression sans perte préserve intégralement les données originales.

La compression ***JPEG*** divise l'image en blocs de pixels, puis applique des transformations mathématiques complexes pour réduire la redondance des données. Cette méthode permet d'obtenir des fichiers beaucoup plus petits tout en maintenant une qualité visuelle satisfaisante pour la plupart des utilisations courantes

Bien que la compression ***JPEG*** soit extrêmement efficace pour réduire la taille des fichiers, elle peut entraîner une perte de qualité perceptible, en particulier à des niveaux de compression élevés. Cependant, en ajustant les paramètres de compression, il est possible de trouver un compromis entre taille de fichier et qualité d'image pour répondre aux besoins spécifiques de chaque situation.

La compression JPEG est une technologie essentielle qui a révolutionné la manière dont les images sont stockées, transférées et affichées dans le monde numérique. Son impact est immense et son utilisation généralisée témoigne de son efficacité dans la gestion des ressources numériques tout en préservant une expérience visuelle satisfaisante.

1. Objectifs du TP

Les objectifs du travail pratique sont les suivants :

- Initiation à la compression des images fixes et vidéo.

-Comprendre les principes fondamentaux de la compression d'images et de vidéos

- Implémenter sous Matlab les différentes étapes de la compression JPEG

-Comprendre le fonctionnement de chaque étape de la compression JPEG.

-Identifier les algorithmes et les techniques utilisés dans la compression JPEG

-Acquisition de compétences pratiques dans l'utilisation de MATLAB pour implémenter des algorithmes de traitement numérique

2. Norme JPEG

La norme JPEG, acronyme de Joint Photographic Experts Group, représente un jalon majeur dans le domaine de la compression d'images fixes. Établie par un consortium d'experts renommés, cette norme définit non seulement le format d'enregistrement, mais aussi l'algorithme de compression* et de décodage, offrant ainsi une solution fiable et efficace pour la représentation numérique des images. Grâce à JPEG, les utilisateurs peuvent comprimer leurs images sans compromettre de manière significative laqualité visuelle*, ce qui les rend idéales pour une large gamme d'applications, allant de la photographie numérique** à la transmission rapide d'images sur Internet [4]*.*

Complément

https://www.youtube.com/watch?v=PVmh1zBcyVg

3. Principe de la compression JPEG

La norme ***JPEG*** est une norme qui définit le format d'enregistrement et l'algorithme de décodage pour une représentation numérique compressée d'une image fixe. JPEG est l'acronyme de Joint Photographic Experts Group. Il s'agit d'un comité d'experts qui édite des normes de compression pour l'image fixe.

L'algorithme ***JPEG*** pour une image à niveaux de gris (une image couleur est un ensemble d'images de ce type) repose sur plusieurs étapes .

1.On découpe l'image en blocs de taille identique LxL pixels (8x8 pixels en pratique). L'avantage est de travailler sur des matrices assez petites (64 valeurs) ce qui permet un traitement rapide. Le nombre de matrices à traiter est en revanche assez important.

2.On Calcule la transformée en cosinus discrète bi-dimensionnelle (DCT*) de chaque bloc. Cette transformée permet d'exprimer le contenu de chaque bloc en termes de fréquences horizontale et verticale, notées u et v, et d'amplitude C (u,v), selon l'équation suivante[5]*.*

$$C(u,v) = c(u).c(v).\sum_{x=0}^{n-1}\sum_{y=0}^{n-1} F(x,y).cos\left(\frac{\pi(2x+1)u}{2n}\right).cos\left(\frac{\pi(2y+1)v}{2n}\right)$$

La transformée en cosinus discrète inverse est donnée par

$$F(x,y) = \sum_{u=0}^{n-1}\sum_{v=0}^{n-1} c(u).c(v).C(u,v).cos\left(\frac{\pi(2x+1)u}{2n}\right).cos\left(\frac{\pi(2y+1)v}{2n}\right)$$

Dans les deux cas :

$$c(w) = \begin{cases} \sqrt{\frac{1}{n}} & si\ w = 0 \\ \sqrt{\frac{2}{n}} & si\ w = 1 \dots n-1 \end{cases}$$

Avec n, la taille du bloc, et F (x, y), la couleur au pixel (x, y) de l'image de départ.

La DCT retourne, pour chaque bloc, une matrice de 8x8 nombres. On obtient alors des coefficients représentant les différentes fréquences de l'image pour un canal donné. En haut à gauche se trouve les basses fréquences, en bas à droite les hautes fréquences. La transformée en cosinus discrète (DCT) est utilisée dans la compression jpeg parce qu'elle permet de concentrer l' information pertinente sur un petit nombre de coefficients [6]*.

3. On quantifie différemment les coefficients de la DCT suivant la fréquence. On utilise une table de quantification de 64 éléments définissant les pas de quantification. Dans la norme ***JPEG*** il existe 4 tables de quantification différentes suivant la compression voulue. Une table type est fournie par le standard mais n'est pas imposée.

La quantification consiste à diviser la matrice obtenue par la DCT par la matrice de quantification. Le but est d'atténuer les hautes fréquences, c'est-à-dire celles aux quelles l'œil humain est très peu sensible. Ces fréquences ont des amplitudes faibles, et elles sont encore plus atténuées par la quantification certains coefficients sont même souvent ramenés à 0 [6]*.

La table de quantification permet de choisir un pas de quantification important pour certaines composantes jugées peu significatives visuellement, car les informations pertinentes d'une image, caractérisée par son signal bidimensionnel Img(x,y) sont concentrée dans les fréquences spatiales les plus basses.

On introduit ainsi un critère perceptif qui peut être rendu dépendant des caractéristiques de l'image et de l'application (taille du document). L'intérêt est qu'une longue suite de zéros nécessite très peu de place pour être stockée dans un fichier.

Les pertes d'information (et donc de la qualité visuelle) sont dues à la quantification où l'on est obligé d'arrondir à l'entier le plus proche, lorsque l'on fait le décodage, les valeurs de l'image d'origine ne sont donc pas retrouvées. Mais c'est la quantification qui permet de gagner le plus de place (contrairement à la DCT, qui ne compresse pas). C'est principalement la table de quantification qui permet de modifier le taux de compression.

4. Suite à la quantification, de nombreux zéros apparaissent dans chaque bloc, essentiellement dans les hautes fréquences. Il s'agit alors, avant le codage, de réorganiser les valeurs du bloc dans un vecteur. Cette étape représente la lecture en zigzag du bloc (Figure 1).

Elle permet de trier les valeurs par ordre croissant de fréquences, donc placer d'abord les coefficients correspondant aux fréquences les plus basses. La plupart des zéros apparaissent alors à la fin du vecteur obtenu. Cela étant réalisé, il est possible de tronquer le vecteur de sorte à éliminer le dernier regroupement de zéros successifs.

5. On procède à un codage entropique de Huffman sur le vecteur résultant, c'est à dire que l'on passe d'une suite de coefficients à une suite de zéros et de uns : séquence de bits.

Ce codage permet d'utiliser les propriétés statistiques des images. Le code de Huffman consiste à représenter les symboles les plus probables par des codes comportant un nombre de bits le plus petit possible[7].**

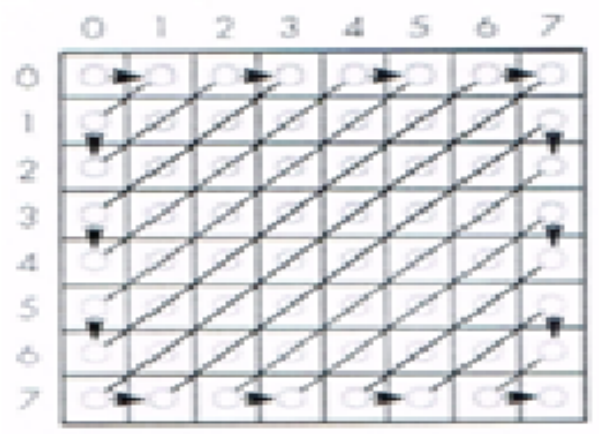
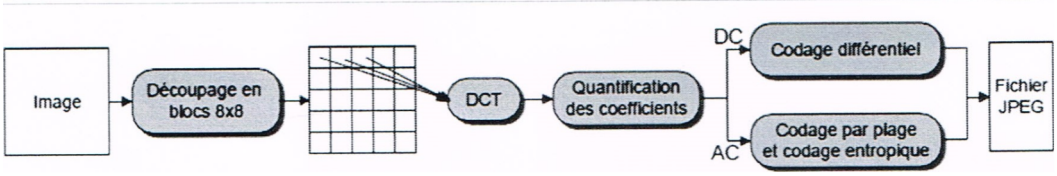


Figure 1 : Lecture en zigzag

Compréssion [8]*



Décompréssion [8]*.

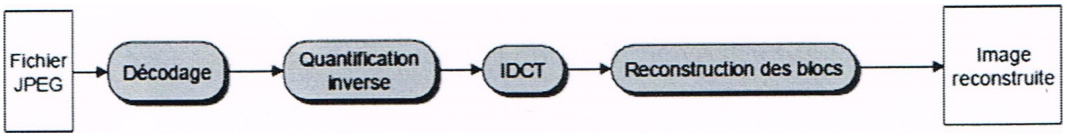


Figure 2 : Etapes de la compression et de la décompression JPEG

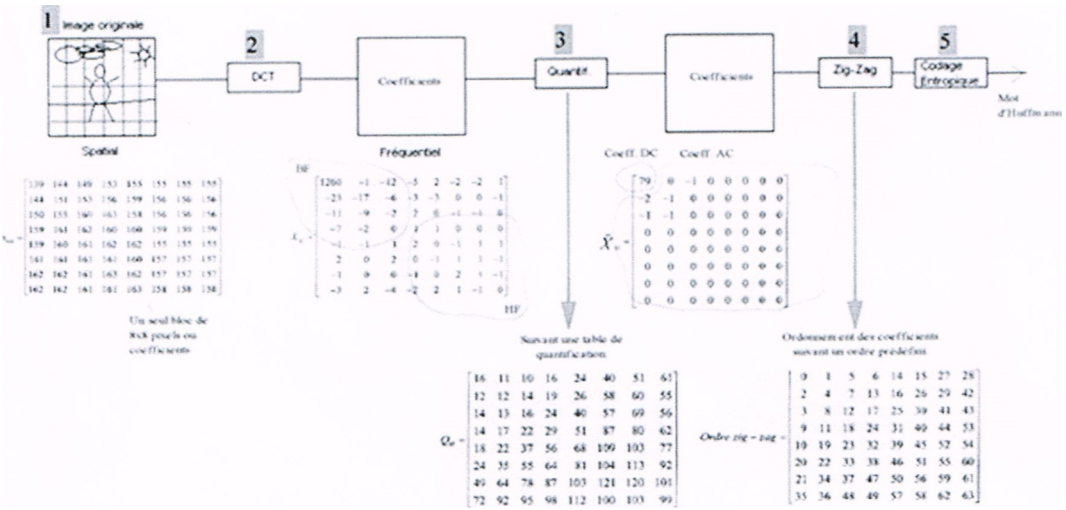


Figure 3 : Exemple de compression d'image

Lors du décodage ***JPEG***, il s'agit d'exécuter les étapes inverses : décodage de Huffman, puis, regroupement par vecteurs de blocs, et, passage de vecteurs à matrices de bloc [9]*.

4. Manipulations

Pour l'image considérée, effectuer les opérations suivantes :

1. Lire l'image « kératose séborrhéique .bmp » et l'enregistrer dans la variable I.

2. Effectuer un découpage de l'image I en blocs de 8*8 pixels.

3. Effectuer ensuite, une transformation DCT de chacun des blocs. En utilisant la commande dct2.

4. Quantifier les coefficients DCT obtenus pour chaque bloc. Cette technique a pour but de supprimer les hautes fréquences, qui sont moins visibles par l'œil humain. Pour cela, créez un filtre de taille (8,8) est donné sous la forme suivante :

Q=[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]

5. Utiliser la fonction norm2huff pour générer le code de Huffman d'une matrice ainsi que la fonction huff2norm pour effectuer l'opération inverse.

6. Reconstruire l'image originale en effectuant des opérations inverses, la quantification inverse et la transformation DCT inverse.

7. Evaluer le taux de compression, en utilisant la formule suivante :

$$EQM = \frac{1}{m\,n}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}\|I_0(i,j) - I_r(i,j)\|^2$$

Le PSNR est donné par l'expression

$$PSNR(I_o, I_R) = 10log_{10}\frac{D^2}{EQM(I_o, I_R)}$$

Où D est la dynamique couverte par les variations de l'intensité lumineuse. Les pixels des images à traiter sont codés sur 8 bits, donc D = 255. L'unité du PSNR* est le décibel et est noté dB.

Plus celui-ci est important, plus les images comparées sont semblables.

5. Solution de TP N°02

Le programme suivant illustre le processus de compression d'image en utilisant l'algorithme JPEG, comprenant des étapes telles que la conversion en espace colorimétrique YCbCr, le sous-échantillonnage, la transformation DCT, la quantification, le balayage zigzag et le codage de Huffman."

```
close all
clc
% Étape 1 : Charger l'image
img = imread('peppers.png');
figure(1)
imshow(img);
% Étape 2 : Conversion en YCbCr
ycbcr = rgb2ycbcr(img);
Y_d = ycbcr(:, :, 1); % Canal de luminance Y
imgCb = ycbcr(:, :, 2); % Canal de chrominance bleue Cb
imgCr = ycbcr(:, :, 3); % Canal de chrominance rouge Cr
figure(2)
subplot(331);imshow(ycbcr);
subplot(334);imshow(Y_d);
subplot(335);imshow(imgCb);
subplot(336);imshow(imgCr);
% Étape 3 : Sous-échantillonnage sous format 4 :2 :2 pour Cb et Cr
Y_d=Y_d(1:2:end,1:2:end);
Cb_d=imgCb(1:2:end,1:2:end);
Cr_d=imgCr(1:2:end,1:2:end);
subplot(337);imshow(Y_d);
subplot(338);imshow(Cb_d);
subplot(339);imshow(Cr_d);
% Étape 4 : Découpage en blocs 8x8 et transformation DCT
Cf = dctmtx(8);
[height, width] = size(Y_d);
% Initialiser des matrices pour stocker les coefficients DCT
YF = zeros(height, width);
CbF = zeros(height, width);
CrF = zeros(height, width);
% Diviser l'image en blocs 8x8 et appliquer la DCT
for i = 1:8:height
    for j = 1:8:width
        block_Y = Y_d(i:i+7, j:j+7);
        block_Cb = Cb_d(i:i+7, j:j+7);
        block_Cr = Cr_d(i:i+7, j:j+7);
        YF(i:i+7, j:j+7) = double(Cf) * double(block_Y)*double(Cf');
        CbF(i:i+7, j:j+7) = double(Cf)* double(block_Cb)*double(Cf');
        CrF(i:i+7, j:j+7) = double(Cf) * double(block_Cr) * double(Cf');
    end
end
figure(3)
subplot(131);imshow(YF);
subplot(132);imshow(CbF);
subplot(133);imshow(CrF);
% 5 Quantification
% Maintenant, vous avez les coefficients DCT pour chaque bloc de Y, Cb et Cr.
Qy = [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];

Qc = [17 18 24 47 99 99 99 99;
      18 21 26 66 99 99 99 99;
      24 26 56 99 99 99 99 99;
      47 66 99 99 99 99 99 99;
      99 99 99 99 99 99 99 99;
```

```
      99 99 99 99 99 99 99 99;
      99 99 99 99 99 99 99 99;
      99 99 99 99 99 99 99 99];
% Appliquer la quantification à chaque bloc
YQ = blkproc(YF ,[8 8],'round(x./P1)',Qy);
CbQ = blkproc(CbF,[8 8],'round(x./P1)',Qc);
CrQ = blkproc(CrF,[8 8],'round(x./P1)',Qc);
figure(4)
subplot(131);imshow(YQ);
subplot(132);imshow(CbQ);
subplot(133);imshow(CrQ);
% 6. Balayage zigzag et Codage de Huffman
[DC_YQ, AC_YQ_zig] = DC_diff_and_zigzag(YQ);
[DC_CbQ, AC_CbQ_zig] = DC_diff_and_zigzag(CbQ);
[DC_CrQ, AC_CrQ_zig] = DC_diff_and_zigzag(CrQ);
zig=[DC_YQ AC_YQ_zig DC_CbQ AC_CbQ_zig DC_CrQ AC_CrQ_zig];
% Compression RLE
rleCode = run_length_encode(zig)
% Codage Huffman
huffmanCode= huffman_encode(rleCode);
disp(huffmanCode)
% 7. Le taux de compression
originalSize = numel(img)*8
compressedSize = numel(huffmanCode)
compressionRatio = originalSize / compressedSize
[DC_YQ, AC_YQ_zig] = DCdif_fn(YQ);
[DC_CbQ, AC_CbQ_zig] = DCdif_fn(CbQ);
[DC_CrQ, AC_CrQ_zig] = DCdif_fn(CrQ);
DC_YQ_Huff = DC_Huff_fn(DC_YQ);
DC_CbQ_Huff = DC_Huff_fn(DC_CbQ);
DC_CrQ_Huff = DC_Huff_fn(DC_CrQ);
AC_YQ_Huff = AC_Huff(AC_YQ_zig);
AC_CbQ_Huff = AC_Huff(AC_CbQ_zig);
AC_CrQ_Huff = AC_Huff(AC_CrQ_zig);
% Balayage zigzag et codage de Huffman pour les composantes Y, Cb, et Cr
[DC_Y, AC_Y_zig] = zigzag_and_huffman(YQ);
[DC_Cb, AC_Cb_zig] = zigzag_and_huffman(CbQ,huffmanCodes);
[DC_Cr, AC_Cr_zig] = zigzag_and_huffman(CrQ, huffmanCodes);
% Vous devez implémenter la fonction zigzag_and_huffman pour le balayage zigzag et le codage
Huffman.
% 7. Le taux de compression
originalSize = numel(img) * 8;
compressedSize = numel(DC_Y) + numel(AC_Y_zig) + numel(DC_Cb) + numel(AC_Cb_zig) + numel(DC_Cr
) + numel(AC_Cr_zig);
compressionRatio = originalSize / compressedSize;
% Affichage du taux de compression
fprintf('Taux de compression : %.2f\n', compressionRatio);
% Appliquer le balayage zigzag aux blocs YQ, CbQ et CrQ
DC_Y = zeros(1, numel(YQ));
AC_Y_zig = cell(1, numel(YQ));
for i = 1:numel(YQ)
block = YQ{i};  % Bloc DCT de la composante Y
AC_Y_zig{i} = zigzag_scan(block(2:end));  % Appliquer le balayage zigzag aux coefficients AC
DC_Y(i) = block(1);  % Coefficient DC
 end
% Appliquer le balayage zigzag aux blocs CbQ
%  (Répéter le même processus pour la composante CbQ)
%  Appliquer le balayage zigzag aux blocs CrQ
%  (Répéter le même processus pour la composante CrQ)
%  Appliquer l'algorithme de Huffman pour les coefficients DC
DC_Y_Huff = huffenco(DC_Y, huffmanCodes);
%  Appliquer l'algorithme de Huffman pour les coefficients AC
```

```
AC_Y_Huff = huffenco(AC_Y_zig, huffmanCodes);
% Faites de même pour les composantes CbQ et CrQ.
%  7 Le taux de compression
Tailleorigin= m*n*d*8;
originalsize=numel(img)*8;
compressesize=numel(DC_Huff)+numel(AC_Huff);
tauxcomp=originalsize/compressesize;
```

Le codage de Huffman est un algorithme de compression de données utilisé pour représenter efficacement un ensemble de symboles en assignant des codes de longueurs variables, où les symboles les plus fréquents sont représentés par des codes plus courts, ce qui permet une compression sans perte de données

```
function huffmanCode = huffman_encode(data)
    symbols = unique(data);
    probabilities = hist(data(:), symbols) / numel(data);

    huffmanDict = huffmandict(symbols, probabilities);
    huffmanCode = huffmanenco(data(:), huffmanDict);
end
```

Le codage RLE est une technique simple mais efficace de compression de données qui consiste à représenter des séquences répétitives de symboles par une seule occurrence du symbole suivi du nombre de répétitions de ce symbole

```
% Fonction de Run-Length Encoding (RLE)
function rleCode = run_length_encode(data)
    rleCode = [];
    currentRun = data(1);
    runLength = 1;

    for i = 2:length(data)
        if data(i) == currentRun
            runLength = runLength + 1;
        else
            rleCode = [rleCode, currentRun, runLength];
            currentRun = data(i);
            runLength = 1;
        end
    end

    rleCode = [rleCode, currentRun, runLength];
end
```

Le codage zigzag est une méthode utilisée dans la compression d'images, notamment dans le format JPEG, où les coefficients de fréquence des blocs de données sont réorganisés de manière à maximiser la quantité de zéros consécutifs, facilitant ainsi la compression

```


function [DC_Y, AC_Y_zig] = zigzag_and_huffman(YQ)
% Définir les tables de Huffman pour les coefficients DC et AC en dehors de la fonction
% huffmanCodes.DC = {'00', '01', '10', '110', '1110', '11110', '111110', '1111110',
'11111110', '111111110'};
% huffmanCodes.AC = {'00', '01', '100', '101', '110', '1110', '11110', '111110', '1111110',
'11111110'};
    % Appliquer le balayage zigzag
    zigzag = zigzag_scan(block);

    % Extraire le coefficient DC (le premier élément du zigzag)
    DC = zigzag(1);
```

```
12
13    % Extraire les coefficients AC (tous les éléments sauf le premier)
14    AC = zigzag(2:end);
15
16    % Appliquer le codage de Huffman aux coefficients DC
17    DC_Huff = huffenco(DC, huffmanCodes.DC);
18
19    % Appliquer le codage de Huffman aux coefficients AC
20    AC_Huff = huffenco(AC, huffmanCodes.AC);
21
22    % Retourner les coefficients DC et AC codés
23    AC_zig = [DC_Huff, AC_Huff];
24 end
```

Le scan Zigzag est une technique utilisée dans la compression d'images où les coefficients de fréquence des blocs de données sont parcourus dans un motif en zigzag, permettant d'organiser les données de manière à réduire les longueurs des coefficients non nuls, ce qui facilite la compression sans perte.

```
1  function zigzag_sequence = zigzag_scan(block)
2    [rows, cols] = size(block);
3    zigzag_sequence = zeros(1, rows * cols);
4    idx = 1;
5
6    for sum = 1:rows + cols - 1
7        if mod(sum, 2) == 1  % direction du haut vers le bas
8            for j = 1:cols
9                i = sum - j + 1;
10               if i >= 1 && i <= rows
11                   zigzag_sequence(idx) = block(i, j);
12                   idx = idx + 1;
13               end
14           end
15       else  % direction du bas vers le haut
16           for i = 1:rows
17               j = sum - i + 1;
18               if j >= 1 && j <= cols
19                   zigzag_sequence(idx) = block(i, j);
20                   idx = idx + 1;
21               end
22           end
23       end
24   end
25
26   end
```

6. Exercice

[solution n° 1 p.17*]*

Exercice

1-Qu'est-ce que l'acronyme JPEG signifie ?

- ☐ Joint Photographic Experts Group
- ☐ Justified Photo Encoding Graphics
- ☐ Jumbled Picture Editing Guide

Exercice

2-Quel est l'objectif principal de la compression JPEG ?

- ☐ Augmenter la taille des images pour une meilleure qualité
- ☐ Réduire la taille des fichiers images tout en préservant une qualité acceptable
- ☐ Convertir les images en différents formats de fichiers

Exercice

3-Quelle est la principale caractéristique de la compression JPEG ?

- ☐ Elle utilise la compression sans perte pour réduire la taille des fichiers
- ☐ Elle utilise la compression avec perte pour réduire la taille des fichiers
- ☐ Elle ne permet pas de compresser les images

Exercice

4-Quel est le type de compression utilisé par la compression JPEG ?

- ☐ Compression temporelle
- ☐ Compression spatiale
- ☐ Compression de couleur

Exercice

5-Quelle est la particularité de la compression JPEG par rapport à d'autres formats d'image comme le PNG ou le GIF ?

- ☐ Elle permet une transparence des pixels
- ☐ Elle est spécifiquement conçue pour les images animées
- ☐ Elle est optimisée pour les photographies et les images complexes

Exercice

6-Quel est le format de fichier le plus couramment associé à la compression JPEG ?

- ☐ jpg
- ☐ png
- ☐ gif

Exercice

7-Quel est le principal avantage de la compression JPEG par rapport à la compression PNG ?

- ☐ Meilleure qualité d'image
- ☐ Fichiers de plus petite taille
- ☐ Transparence des pixels

Exercice

8-Quelle est la principale différence entre la compression JPEG sans perte et avec perte ?

- ☐ La compression JPEG sans perte ne réduit pas la qualité de l'image
- ☐ La compression JPEG sans perte est plus efficace pour les images complexes
- ☐ La compression JPEG avec perte réduit la qualité de l'image pour obtenir une plus petite taille de fichie

7. Conclusion

La compression ***JPEG*** est un pilier de la manipulation d'images numériques, offrant un compromis entre la taille du fichier et la qualité visuelle. Son utilisation répandue dans divers domaines, tels que la photographie numérique et le web, témoigne de son importance. Malgré ses avantages, la compression JPEG peut entraîner une perte de qualité perceptible, surtout à des taux de compression élevés. Cependant, son évolution continue et l'émergence de nouveaux formats témoignent de son adaptabilité aux besoins changeants de la technologie . La gestion efficace du taux de compression est cruciale pour maximiser ses avantages tout en minimisant les artefacts visuels . En dépit de ses limitations, la compression ***JPEG*** demeure un outil indispensable pour la transmission et le stockage efficaces des images. Son rôle dans la réduction de la bande passante sur Internet contribue à une expérience utilisateur optimisée. Toutefois, il est important de reconnaître que d'autres formats peuvent être plus adaptés à certains types d'images. En somme, la compression ***JPEG*** reste un élément fondamental de la gestion des ressources numériques, offrant un équilibre entre efficacité et qualité.

8. Test final

1. Quelle est le type de cette image ? (expliquer)

2. Quelle est la différence entre sous-échantillonnage et sur-échantillonnage dans la compression JPEG ?

3.

a) Chargez et affichez l'image « peppers.png»et l'enregistrer dans la variable img1.

b) Extraire et affichez les matrices des couleurs R, G et B de l'image img1.

c)Sous-échantillonner l'image img1 ce format 4 :2 :2. enregistrer et afficher dans la variable img2

d) Calculez la taille de l'image img2 en bit.

8.1. Solution

1

- image en niveaux de gris
- Le nombre de couleurs n = 8 c'est-à-dire 2^8=256 donc on a 256 couleurs (les valeurs des couleurs variant de 0 à 255).
- L'intensité de chaque pixel variait de 0 à 255.

2

- Dans le sous-échantillonnage, les informations de chrominance sont réduites par rapport à la luminance avec un facteur de 4:2:2. On applique dans la compression JPEG
- Le sur-échantillonnage consiste à augmenter la résolution de la chrominance par rapport à la luminance. On applique dans la décompression JPEG.

Programme

```
% % Étape 1 : Charger l'image
img1 = imread('peppers.png');
figure(1)
imshow(img1);
R = img1(:, :, 1); % Canal de Rouge
G = img1(:, :, 2); % Canal de Vert
B = img1(:, :, 3); % Canal de Blue
figure(2)
```

```
subplot(131);imshow(R);
subplot(132);imshow(G);
subplot(133);imshow(B);
% Étape 2 : Sous-échantillonnage sous format 4 :2 :2 pour img1
img2=img1(1:2:end,1:2:end);
figure(3)
imshow(img2);
% Étape 3 : Calculer la taille en bit
Timg2=numel(img2)*8= 1179648 bits
```

Conclusion

En combinant les connaissances acquises dans ce TP sur l'initiation au traitement d'image avec la compréhension des techniques de compression ***JPEG***, on peut pleinement apprécier l'importance de la manipulation d'images dans différents domaines, tels que la photographie numérique, la vidéo, la vision par ordinateur, etc. Le traitement d'image fournit les outils nécessaires pour analyser, modifier et améliorer les images, tandis que la compression ***JPEG*** offre des moyens efficaces de stocker et de transmettre ces images de manière économique. Cette synergie entre le traitement d'image et la compression ***JPEG*** offre aux professionnels de l'informatique la possibilité de créer, manipuler et partager des contenus visuels de manière innovante et efficace

Solutions des exercices

> **Solution** n°1

Exercice p. 12

Exercice

1-Qu'est-ce que l'acronyme JPEG signifie ?

- [x] Joint Photographic Experts Group
- [ ] Justified Photo Encoding Graphics
- [ ] Jumbled Picture Editing Guide

Exercice

2-Quel est l'objectif principal de la compression JPEG ?

- [ ] Augmenter la taille des images pour une meilleure qualité
- [x] Réduire la taille des fichiers images tout en préservant une qualité acceptable
- [ ] Convertir les images en différents formats de fichiers

Exercice

3-Quelle est la principale caractéristique de la compression JPEG ?

- [ ] Elle utilise la compression sans perte pour réduire la taille des fichiers
- [x] Elle utilise la compression avec perte pour réduire la taille des fichiers
- [ ] Elle ne permet pas de compresser les images

Exercice

4-Quel est le type de compression utilisé par la compression JPEG ?

- [ ] Compression temporelle
- [x] Compression spatiale
- [ ] Compression de couleur

Exercice

5-Quelle est la particularité de la compression JPEG par rapport à d'autres formats d'image comme le PNG ou le GIF ?

- ☐ Elle permet une transparence des pixels
- ☐ Elle est spécifiquement conçue pour les images animées
- ☑ Elle est optimisée pour les photographies et les images complexes

Exercice

6-Quel est le format de fichier le plus couramment associé à la compression JPEG ?

- ☑ jpg
- ☐ png
- ☐ gif

Exercice

7-Quel est le principal avantage de la compression JPEG par rapport à la compression PNG ?

- ☐ Meilleure qualité d'image
- ☑ Fichiers de plus petite taille
- ☐ Transparence des pixels

Exercice

8-Quelle est la principale différence entre la compression JPEG sans perte et avec perte ?

- ☐ La compression JPEG sans perte ne réduit pas la qualité de l'image
- ☑ La compression JPEG sans perte est plus efficace pour les images complexes
- ☑ La compression JPEG avec perte réduit la qualité de l'image pour obtenir une plus petite taille de fichie

Glossaire

Algorithme de compression

Un ensemble d'instructions mathématiques et de techniques de traitement des données utilisées pour réduire la taille d'un fichier d'image tout en préservant sa qualité visuelle.

Photographie numérique

La pratique de prendre des photographies à l'aide d'un appareil photo numérique, où les images sont enregistrées sous forme de données numériques.

Qualité visuelle

Le niveau de détail et de fidélité d'une image par rapport à l'original, souvent mesuré en termes de netteté, de résolution et de fidélité des couleurs.

Abréviations

DCT : Discrete Cosine Transform

PSNR : sigle de Peak Signal to Noise Ratio

Bibliographie

Cours "Compression par transformée de codage JPEG"

Emre ZEYBEK , cours "Compression multimodale du signal et de l'image en utilisant un seul codeur"

HAMIDOUCHE Naima, cours "COMPRESSION D'IMAGES HYPERSPECTRALES PAR LE STANDARD JEPG 2000"

Webographie

https://iast.univ-setif.dz/documents/Cours/CoursInformatiqueL1GAT21.pdf. Représentation d'une image numérique.

https://www.univ-constantine2.dz/files/Theses/Informatique/Magistere/Mohamed-Sandeli.pdf. Traitement d'images par des approches bio-inspirées(Application à la segmentation d'images)

https://efreidoc.fr/M1/Traitement%20d%27images/Cours/2011-12.cours.complet.img.pdf. La COULEUR dans les IMAGES Et Indexation par le Contenu

http://rdoc.univ-sba.dz/bitstream/123456789/2546/1/THESE.pdf. Sécurité des images Numériques compressées JPEG

https://elearning.univ-msila.dz/moodle/pluginfile.php/543996/mod_resource/content/2/Chapitre05%20Norme%20JPEG.pdf.Techniques de compression d'image (Norme JPEG)

https://www.techno-science.net/glossaire-definition/JPEG-page-2.html