Université de la Rochelle
Master 1 ICONE - Codage de l'information

TP - Algorithme de compression JPEG

© Renaud Péteri

Année universitaire 2019-2020

Vous ferez un rapport sur ce TP à déposer sous Moodle: vos réponses aux questions seront détaillées et vos codes Python commentées. Vous pourrez utiliser Pweave pour générer votre rapport (en pdf ou html)

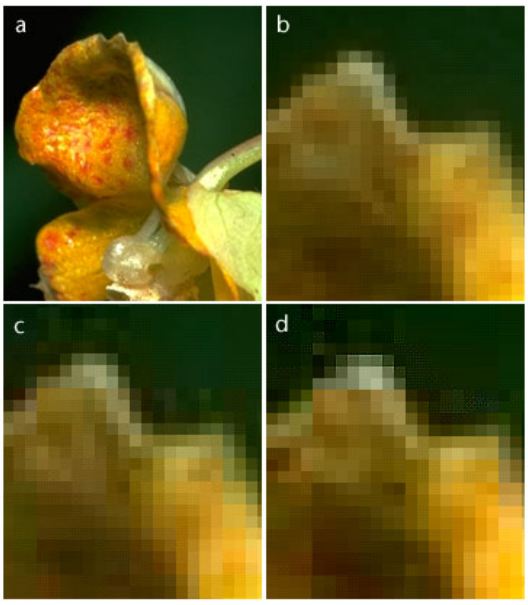

FIGURE 1 – Image JPEG pour différents taux de compression

1 Transformée DCT (en cosinus discret)

On rappelle que dans la norme JPEG, les fonctions de projection utilisées ne sont pas des exponentielles complexes (cas de la TFD) mais des cosinus (la DCT=Discrete Cosine Transform). La DCT donne donc des coefficients réels.
Plusieurs expressions de la DCT existent, nous utiliserons la DCT-II qui est la plus courante. Son expression est en $[i,j] \in [0, N-1]^2$ pour une image `Im` de taille $N \times N$:

$$\mathrm{DCT}[i,j] = \frac{2}{N} C[i] C[j] \sum_{n=0}^{N-1} \sum_{m=0}^{N-1} \mathrm{Im}[n,m] \cos\left[\frac{(2n+1)i\pi}{2N}\right] \cos\left[\frac{(2m+1)j\pi}{2N}\right]$$

avec $C[x] = \begin{cases} \frac{1}{\sqrt{2}} & \text{pour } x = 0 \\ 1 & \text{pour } x > 0 \end{cases}$

La transformée DCT utilisée dans le codage JPEG s'effectue sur des sous-images de taille 8×8 pixels, que l'on appelle des macroblocs.

Nous nous proposons dans ce TP de simuler une transformation JPEG avec perte et de visualiser les résultats de compression obtenus. Les fonctions DCT de décomposition `DCT` et de reconstruction `iDCT` pour des blocs de pixels de taille 8×8 sont fournies sous Moodle (`JPEG.zip`). Elles se trouvent dans le module `Functions` du package `JPEG` : `import JPEG.Functions as jpeg`

1. Générez une image de taille 8×8 remplie de 1 et visualisez le résultat de sa transformée DCT. Commentaires ?
2. Inversez la DCT précédente, et vérifiez que vous retrouvez bien votre image.

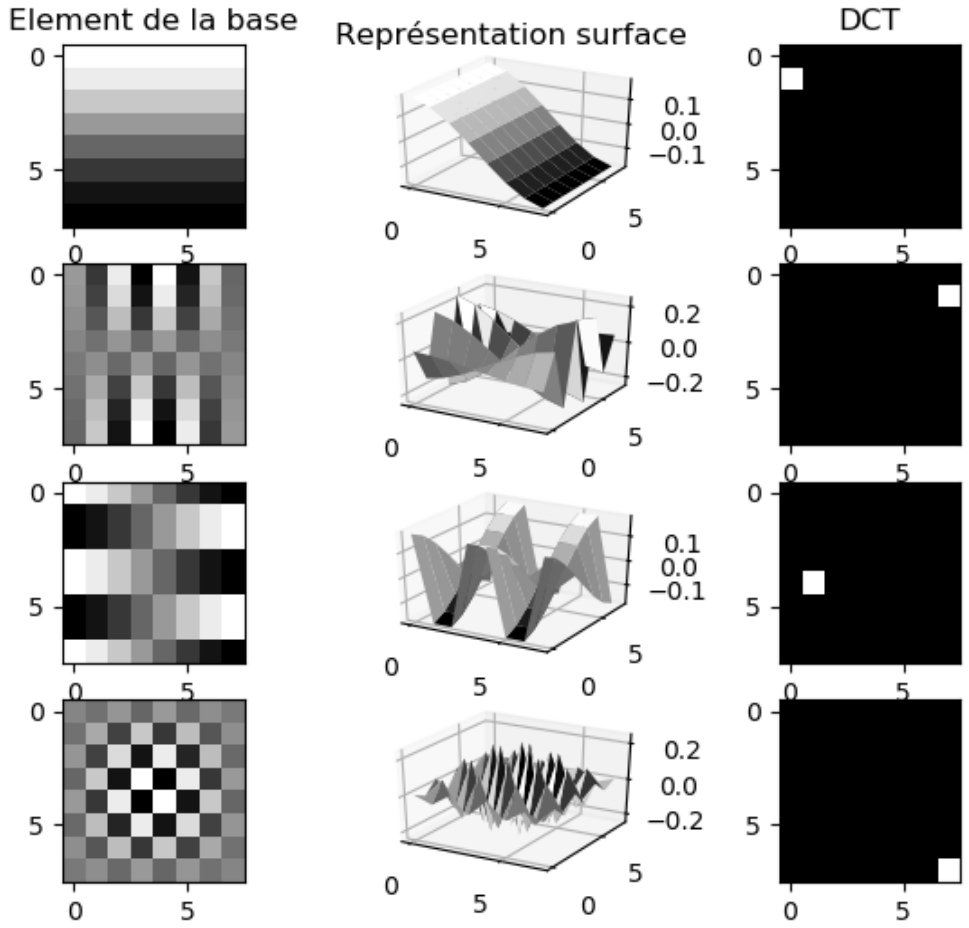

FIGURE 2 – Répresentation dans l'espace direct et DCT de 4 éléments de la base DCT

3. On veut visualiser les fonctions cosinus de la base de projection (voir figure 2). Pour cela, générez dans l'espace DCT une matrice nulle sauf en un point de valeur 1 à l'endroit correspondant à la fréquence de l'élément de la base à visualiser.
   Prenez ensuite la DCT inverse, et représentez la fonction pour les fréquences $(1; 0)$, $(1; 7)$, $(4; 1)$ et $(7; 7)$. On représentera ces fonctions avec `subplot` avec une colonne pour leur représentation 2D, une autre colonne pour leur représentation surface et une dernière colonne pour la DCT.
   Aide pour le tracé de surface :

```
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

xx, yy = np.mgrid[0:8,0:8]
fig = plt.figure()
ax = fig.add_subplot(436, projection='3d')
ax.plot_surface(xx,yy,Surface_a_tracer,...)
```

2 Transformée DCT d'une image

On souhaite maintenant appliquer la DCT non plus à des blocs de taille 8×8 pixels mais à des images de taille quelconque (prendre l'image `ulr.png` sous moodle).

1. Charger l'image `ulr.png` dans la variable `img` et initialiser à 0 une image `dct_img` de même taille que `img`.
2. Effectuez une boucle sur les macroblocs 8×8 de l'image `img` pour appliquer sur chaque macrobloc la DCT et stocker le tout dans `dct_img`.
   Aide : `im[i*8:8+i*8,j*8:8+j*8]` réprésente le macrobloc extrait de l'image `im` de taille 8×8 au point de coordonnées `[i*8,j*8]` (coin haut à gauche du macrobloc).
3. Prendre la DCT inverse sur chaque macrobloc, et vérifier en l'affichant que vous retrouvez l'image originale. Prendre aussi la norme de la différence des 2 images pour une confirmation quantitative.

3 Entropie de Shannon

Pour quantifier la répartition des valeurs entre un bloc dans le domaine spatial et sa transformée DCT, nous allons estimer l'entropie de Shannon de ces deux entités.
L'entropie de Shannon, est une fonction mathématique qui, intuitivement, correspond à la quantité d'information contenue ou délivrée par une source d'information. Dans notre cas, la source d'information correspondra aux pixels dans un bloc du domaine direct ou du domaine DCT. Pour schématiser, plus la source émet d'informations différentes, plus l'entropie (ou incertitude sur ce que la source émet) est grande. Inversement, si la même information est toujours émise, l'entropie sera nulle.
Mathématiquement, l'entropie de Shannon est définie comme suit :

$$H_2(X) = -\mathbb{E}[\log_2 P(X)] = \sum_{i=1}^{n} P_i \log_2 \left(\frac{1}{P_i}\right) = -\sum_{i=1}^{n} P_i \log_2 P_i$$

où P_i est la probabilité d'apparition du symbole i de la source. Pour nous un symbole sera ici un pixel de l'image ou d'un macrobloc.

1. Ouvrir l'image `Barbara.jpg` se trouvant sous Moodle à la fin de la séance. En se servant de ce que vous avez fait précédemment, créer un bloc 8×8 `sum_dct` qui sera la somme de tous les macroblocs DCT de l'image (on prendra les valeurs absolues des coefficients DCT de chaque bloc).
2. Diviser ensuite `sum_dct` par la somme de ses valeurs pour obtenir la densité de probabilité de répartition des valeurs de la DCT dans un macrobloc. La visualiser sous forme d'une surface.
3. Calculer ensuite son entropie (fonction SCIPY.STATS.ENTROPY).

4. Calculer ensuite l'histogramme des valeurs des pixels de l'image et le normaliser pour obtenir une densité de probabilité.
   On utilisera `np.histogram(I, bins=256,range=(0,255))`.
5. Calculer l'entropie de l'image dans le domaine direct et conclure...

4 Compression JPEG

La DCT permet donc une représentation d'une image de manière plus parcimonieuse que l'espace "direct", c'est à dire qu'il faut moins de coefficients dans l'espace DCT que dans l'espace de départ pour représenter l'information *visible* par l'oeil humain.
Dans cette partie, nous allons mettre en avant cette propriété, base de la norme JPEG.

1. Créer une fonction `seuillage_DCT(coeff,pourcent_max)` permettant de ne garder que les coefficients DCT qui sont au dessus de *pourcent_max* du maximum d'un bloc DCT, et de mettre à 0 les autres. On pourra s'inspirer du code suivant :

   ```
   thres=pourcent_max*np.max(coeff)
   coeff_seuil  = np.where(np.abs(coeff) > thres, coeff, 0.0)
   ```

2. Appliquer votre fonction de compression en ne gardant pour l'image `ulr.png` que les coefficients DCT au dessus de 1%, 5% et 30% du max dans chaque bloc. Affichez le tout dans une seule figure et commentez.
3. Tester aussi votre fonction de compression sur l'image `barbara.jpg`

5 Extension aux images couleurs

JPEG est capable de coder les couleurs sous n'importe quel format, toutefois les meilleurs taux de compression sont obtenus avec des codages de couleur de type luminance/chrominance car l'œil humain est assez sensible à la luminance (la luminosité) mais peu à la chrominance (la teinte) d'une image. Afin de pouvoir exploiter cette propriété, l'algorithme convertit l'image d'origine depuis son modèle colorimétrique initial (en général RVB) vers le modèle de type chrominance/luminance YCbCr. Dans ce modèle, Y est l'information de luminance, et Cb et Cr sont deux informations de chrominance, respectivement le bleu moins Y et le rouge moins Y.

Effectuez la compression JPEG pour l'image couleur `fox.jpg` présente sous Moodle.