Traitement des images numériques

TP 2 : Fréquences associées à une image

Université Paris 13, Institut Galilée
Master Ingénierie et Innovations en Images et Réseaux - 1ère année

2017-2018

Consignes

- Récupérer le fichier `TP2.zip` sur le site

  `http://www.laurentoudre.fr/tin.html`

- Ouvrir MATLAB et créer un répertoire de travail. Dézipper le fichier `TP2.zip` dans ce répertoire.
- A la fin de la séance, récupérer les scripts que vous avez écrits et les envoyer par e-mail au chargé de TP ainsi qu'à vous même afin de les conserver pour la prochaine séance.

Rendu

- Cinq fichiers : `TP2Partie1.m`, `TP2Partie2.m`, `TP2Partie3.m`, `TP2Partie4.m` et `TP2Partie5.m` Chaque fichier doit contenir votre nom, votre prénom et la date.
- Compte-rendu succinct à rendre à la fin de la séance, contenant les observations, commentaires et réponses aux questions. Le compte rendu doit contenir votre nom et votre prénom.

Plan de l'étude

1 Rappels de cours

1.1 Transformée de Fourier d'une image

Soit $g(m,n)$ une image (renormalisée) de dimension $M \times N$, où $g(m,n)$ est l'intensité (valeur entre 0 et 1) du pixel à la position (m,n). On définit sa transformée de Fourier comme la transformée de Fourier discrète bidimensionnelle suivante :

$$G(k,l) = \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} g(m,n) e^{-j2\pi(\frac{km}{M} + \frac{ln}{N})}$$

k est ici l'indice associé à la composante fréquentielle u et correspond à la fréquence $\frac{k}{M} f_e$. l est l'indice associé à la composante fréquentielle v et correspond à la fréquence $\frac{l}{N} f_e$. f_e est ici la fréquence d'échantillonnage (que l'on suppose identique pour l'axe x et l'axe y). Le choix du coefficient $\frac{1}{MN}$ est dans une certaine mesure arbitraire, il garantit ici que la composante constante $G(0,0)$ est la moyenne du signal image.

Cette transformée de Fourier se calcule sous MATLAB avec la commande `fft2`, qui prend en paramètre une image renormalisée (à valeurs comprises entre 0 et 1) et renvoie une matrice **complexe** de même taille que l'image et qui contient les différents coefficients de la transformée de Fourier. En revanche, cette transformée ne tient pas compte du fait que les images sont à valeurs réelles et que leur spectre possède donc une symétrie hermitienne. Il faut donc comme en traitement du signal utiliser l'instruction `fftshift` pour placer la fréquence nulle au centre de l'image et ainsi mieux visualiser les propriétés de symétrie.

Exemple de commande pour calculer la transformee de Fourier d'une image :

```
% X : image renormalisee (valeurs entre 0 et 1)
Y = fft2(X); % Sans rearrangement
Y = fftshift(fft2(X)); % Avec rearrangement
```

1.2 Visualisation de la transformée de Fourier d'une image

Pour afficher la transformée de Fourier d'une image sous la forme d'une image, il y a deux problèmes : cette quantité est complexe, et ses valeurs ne sont pas comprises entre 0 et 1.

Pour répondre à la première problématique, on représente souvent non pas la transformée de Fourier mais son module. Il y a plusieurs façons de faire cela : tout comme en traitement du signal, on peut visualiser le module soit avec une échelle linéaire soit avec une échelle logarithmique (décibel).

- L'échelle linéaire rend compte effectivement de la proportion avec laquelle la forme de l'image résulte des différentes sinusoïdes élémentaires. Elle permet de mettre en évidence les phénomènes fréquentiels majeurs. En revanche, souvent l'essentiel du spectre est localisé sur la fréquence nulle (qui correspond à la moyenne de l'image) ce qui peut masquer certains phénomènes plus fins que l'on veut observer.
- L'échelle logarithmique permet de visualiser avec précision le spectre pour les fréquences où il prend des valeurs très faibles (ce qui est impossible avec l'échelle linéaire). En revanche, comme elle amplifie la dynamique de visualisation, elle peut aussi conduire à des erreurs d'interprétation. On introduit dans cette échelle un paramètre C qui permet de contrôler la dynamique et d'éviter de prendre des logarithmes de 0 !

Deux facons de calculer le module :

```
% Y : transformee de Fourier d'une image
Y1 = abs(Y); % Echelle lineaire
Y2 = 10*log10(C + abs(Y)); % Echelle logarithmique
```

Pour répondre à la deuxième problématique, il faut, après avoir calculé le module (en échelle linéaire ou logarithmique), ramener toutes les valeurs entre 0 et 1. Pour cela, on peut utiliser la fonction `mat2gray` de MATLAB, qui renormalise la matrice en fixant le minimum à 0 et le maximum à 1.

Renormalisation des coefficients d'une matrice :

```
% Y : Matrice ayant des coefficients quelconques
Y_norm = mat2gray(Y);
% Y_norm : Matrice ayant des coefficients entre 0 et 1
```

1.3 Transformée de Fourier inverse

La transformée de Fourier discrète inverse est :

$$g(m,n) = \sum_{k=0}^{M-1} \sum_{l=0}^{N-1} G(k,l) e^{j2\pi(\frac{km}{M}+\frac{ln}{N})}$$

Cette formule exprime le fait que les images sont des combinaisons linéaires d'images formées de sinusoïdes.

Etant donnée une transformée de Fourier (avant `fftshift` !), on calcule la transformée de Fourier inverse grâce à la fonction `ifft2` de MATLAB. Etant données les erreurs de précision de MATLAB, il est plus prudent de prendre ensuite la partie entière pour s'assurer que l'image reconstruite est bien réelle.

Transformée de Fourier inverse

```
% Y : transformee de Fourier
X = real(ifft2(Y));
```

1.4 Interprétations

En observant le spectre (réarrangé) d'une image, on peut dire les choses suivantes :

- Les fréquences situées proche du centre du spectre (donc proches de la fréquence nulle) sont appelées **basses fréquences** : elles correspondent aux zones uniformes de l'image ou contenant peu de détails
- Les fréquences éloignées du centre du spectre correspondent aux **hautes fréquences**, c'est à dire aux détails de l'image
- Si l'on voit apparaître des lignes sur le spectre, cela signifie qu'on peut retrouver des éléments assimilables à des lignes ou des directions privilégiées dans l'image, dans la direction perpendiculaire à celle donnée par le spectre.

2 Etude sous MATLAB

2.1 Quelques peintures célèbres

Nous allons travailler dans cette partie sur trois peintures très célèbres (Monet, Picasso et Mondrian) et observer leurs spectre.

1. Créer sous MATLAB un script vide nommé `TP2Partie1.m`
2. Ouvrir l'image `image1.bmp`, la stocker dans une matrice `X1`, la renormaliser et l'afficher.
   (a) Diriez vous que cette image contient plutôt des basses ou des hautes fréquences ? Y'a-t-il dans l'image des directions privilégiées ?
   (b) Calculer la transformée de Fourier `Y1` de l'image. Réarranger cette transformée (grâce à `fftshift`) et calculer son module.
   (c) En utilisant la commande `mat2gray` pour renormaliser les valeurs, afficher le spectre d'abord en échelle linéaire, puis en échelle logarithmique (on prendra $C = 100$).
   (d) Quelle échelle vous semble la plus appropriée ici ?
   (e) Vérifiez vous vos observations en regardant le spectre ? Commenter.
3. Réaliser le même processus pour `image2.bmp` (stockée dans `X2`) et `image3.bmp` (stockée dans `X3`). Comparer les résultats et commenter.

2.2 Images synthétiques

1. Créer sous MATLAB un script vide nommé `TP2Partie2.m`
2. Pour chacune de ces images, générer une image de taille 256×256 (grâce à ce qui a été fait au TP1) et afficher sur la même figure l'image, son spectre en échelle linéaire et son spectre en échelle logarithmique (On prendra $C = 100$). Commenter et interpréter les résultats obtenus.
   (a) `X4` Carré gris clair centré sur un fond uniforme gris foncé (vous choisirez le côté)

(b) `X5` Cercle gris foncé centré sur un fond uniforme gris clair (vous choisirez le rayon)

(c) `X6` Image d'équation $(x, y) \to 0.5 + 0.5\cos(6\pi x)$ avec $\Delta x = 0.05$ et $\Delta y = 0.05$

(d) `X7` Image d'équation $(x, y) \to 0.5 + 0.5\cos(2\pi(3x + 4y))$ avec $\Delta x = 0.05$ et $\Delta y = 0.05$

Pour afficher plusieurs images de même taille sur une même figure sous MATLAB, on peut utiliser

```
% X, Y, Z : images renormalisees de meme taille
figure
imshow([X Y Z])
```

2.3 Seuillage : sélection des basses/fréquences

Afin de mieux comprendre à quoi correspondent les basses et hautes fréquences, nous allons faire l'expérience suivante. Nous avons vu que les basses fréquences étaient proches de la fréquence centrale (sur un spectre réarrangé). Nous allons essayer de ne conserver que les basses fréquences (ou hautes) dans le spectre et de reconstruire l'image par transformée de Fourier inverse. Pour sélectionner les basses (ou hautes fréquences), nous allons considérer un cercle de rayon R centré sur la fréquence nulle, et ne garder que les coefficients de Fourier correspondant aux fréquences comprises (ou non comprises) dans le cercle.

La fonction `SeuillageFrequencesFourier(X,R,param)` fournie réalise les étapes suivantes :

- Prend en entrée une image renormalisée `X`, un rayon `R` (entier), et un paramètre `param` (égal à 0 ou 1)
- Calcule la transformée de Fourier réarrangée
- Deux cas :
  - Si `param = 0`, annule toutes les coefficients de Fourier associées aux fréquences situées à une distance de plus de `R` pixels de la fréquence centrale
  - Si `param = 1`, annule toutes les coefficients de Fourier associées aux fréquences situées à une distance de moins de `R` pixels de la fréquence centrale
- Reconstruit une image par transformée de Fourier inverse
- Affiche sur la même image, de gauche à droite et de haut en bas :
  - L'image originelle *(en haut à gauche)*
  - Le spectre de l'image originelle en échelle linéaire *(en haut au milieu)*
  - Le spectre de l'image originelle en échelle logarithmique *(en haut à droite)*
  - L'image reconstruite *(en bas à gauche)*
  - Le spectre de l'image reconstruite en échelle linéaire *(en bas au milieu)*
  - Le spectre de l'image reconstruite en échelle logarithmique *(en bas à droite)*

1. Créer sous MATLAB un script vide nommé `TP2Partie3.m`
2. Ouvrir l'image `cameraman.tif`, la stocker dans une matrice `Z1` et la renormaliser.
   (a) Lancer la fonction `SeuillageFrequencesFourier.m` avec `R = 20` et `param = 0`. Commenter.
   (b) Relancer la même simulation avec `param = 1`. Commenter.
   (c) Faire varier `R` et commenter.
3. Recommencer ces opérations avec une image naturelle ou synthétique de votre choix. Commenter.

2.4 Seuillage : sélection des zones d'énergie faibles/élevées

Afin de comprendre à quelles parties de l'image correspondent les zones d'énergies élevées dans le spectre, nous allons faire une deuxième expérience. Nous n'allons garder que les fréquences où l'énergie est élevée (ou faible), et reconstituer l'image ainsi obtenue par transformée de Fourier inverse. A priori, si l'on sélectionne effectivement les coefficients de Fourier concentrant le plus d'énergie, la reconstitution devrait donner une bonne approximation de l'image. Nous allons donc définir un seuil T et mettre tous les coefficients de la transformée de Fourier dont le module est inférieur (ou supérieur) à T à 0. On reconstruira ensuite l'image en prenant la transformée de Fourier inverse de ce spectre modifié.

La fonction `SeuillageEnergieFourier(X,T,param)` fournie réalise les étapes suivantes :

- Prend en entrée une image renormalisée `X`, un seuil `T` (réel), et un paramètre `param` (égal à 0 ou 1)
- Calcule la transformée de Fourier
- Deux cas :
  - Si `param = 0`, annule toutes les coefficients de Fourier dont le module est inférieur à T
  - Si `param = 1`, annule toutes les coefficients de Fourier dont le module est supérieur à T
- Reconstruit une image par transformée de Fourier inverse
- Affiche sur la même image, de gauche à droite et de haut en bas :
  - L'image originelle *(en haut à gauche)*
  - Le spectre de l'image originelle en échelle linéaire *(en haut au milieu)*
  - Le spectre de l'image originelle en échelle logarithmique *(en haut à droite)*
  - L'image reconstruite *(en bas à gauche)*
  - Le spectre de l'image reconstruite en échelle linéaire *(en bas au milieu)*
  - Le spectre de l'image reconstruite en échelle logarithmique *(en bas à droite)*

1. Créer sous MATLAB un script vide nommé `TP2Partie4.m`
2. Ouvrir l'image `cameraman.tif`, la stocker dans une matrice `Z1` et la renormaliser.
   (a) Lancer la fonction `SeuillageEnergieFourier.m` avec `T = 0.001` et `param = 0`. Commenter.
   (b) Relancer la même simulation avec `param = 1`. Commenter.
   (c) Faire varier `T` et commenter.
3. Recommencer ces opérations avec une image naturelle ou synthétique de votre choix présentant des directions privilégiées. Commenter.

2.5 Quelques transformations de l'image

Nous allons ici tester rapidement l'influence de quelques transformations de l'image sur le spectre : redimensionnement, rotation, translation...

Pour afficher les spectres, on utilisera la fonction `AffichageSimultane(X1,Y1,X2,Y2)` fournie où `X1` et `X2` sont deux images (par exemple l'image originale et l'image transformée) et `Y1` et `Y2` leurs spectres respectifs (`Y1 = fft2(X1)` et `Y2 = fft2(X2)`).

1. Créer sous MATLAB un script vide nommé `TP2Partie5.m`
2. Ouvrir l'image `cameraman.tif`, la stocker dans une matrice `Z1` et la renormaliser.
3. Créer une image `Z2` correspondant à l'image `Z1` redimensionnée de taille 128×128. Créer ensuite une image `Z3` correspondant à l'image `Z2` redimensionnée de taille 256×256. Comparer les spectres de `Z1` et `Z3` et commenter.

Pour redimensionner une image en une image de taille $M \times N$, on peut utiliser

```
% X : Image originelle
Y = imresize(X,[M N]);
% Y : Image redimensionnee de taille M x N
```

4. Créer une image `Z4` correspondant à l'image `Z1` tournée d'un angle de 90 degrés. Comparer les spectres de `Z1` et `Z4` et commenter.

Pour appliquer une rotation d'angle θ dans le sens trigonométrique sur une image, on peut utiliser

```
% X : Image originelle
Y = imrotate(X,theta);
% Y : Image obtenue apres rotation
```

5. Créer une image `Z5` correspondant à l'image `Z1` translatée avec $T_x = 30$ pixels (selon l'axe x) et $T_y = 40$ pixels (selon l'axe y). Comparer les spectres de `Z1` et `Z5` et commenter.

Pour appliquer une translation de vecteur $[T_x, T_y]$ sur une image, on peut utiliser

```
% X : Image originelle
Y = circshift(X,[Tx Ty]);
% Y : Image obtenue apres translation
```