De la photographie numérique à la photographie computationnelle

CES8AH - École des Mines de Nancy

Séance 2

Frédéric Sur

L'objectif de cette séance est de travailler sur la notion de spectre d'une image.

1 Manipulations élémentaires

Chargez l'image `boat.tif` dans une variable et visualisez-la. Par exemple :

```
>> Image=imread('boat.tif');
>> figure, imshow(Image);
```

La fonction `imshow` est réservée à la visualisation de tableaux au format *entier non-signé* (`uint`). Un tableau de `double`s peut être visualisé par l'intermédiaire de `imagesc` comme suit :

```
>> Ima=double(Image);
>> figure, imagesc(Ima);
>> colormap(gray);
>> axis equal;
```

Que font les fonctions `fft2` et `fftshift` ? (utilisez la documentation)

Affichez le spectre d'amplitude de la transformée de Fourier de cette image. Il pourra être utile de le visualiser en échelle logarithmique.

```
>> fIma=fft2(Ima);
>> figure, imagesc(log(abs(fftshift(fIma)))), colorbar;
```

Vous pouvez utiliser `colormap(gray)` ou `colormap(1-gray)`.

Constatez la symétrie des modules et la décroissance des coefficients de Fourier.

Vérifiez la formule de Parseval.

Constatez la différence avec le spectre obtenu à partir d'une image dans laquelle les bords de l'image originale ont été assombris (mis à 0) pour assurer des transitions gauche/droite et haut/bas continues. On pourra assombrir les bords à l'aide de `H` défini par :

```
>> H=hann(size(Ima,1))*hann(size(Ima,2))';
>> figure, imagesc(H), colormap(gray), axis equal, colorbar
>> figure, imagesc(H.*Ima), colormap(gray), axis equal
```

Procédez aux mêmes expériences avec, par exemple, l'image `D3.tif` (ou les autres images de texture).

2 Hautes et basses fréquences

Mettez en évidence le rôle des hautes et basses fréquences dans une image. Pour ce faire, vous pourrez créer un masque permettant de mettre à 0 les hautes ou les basses fréquence, comme ceci :

```
>> halfwidth=50;
>> G=zeros(size(Ima));
>> G(1+size(Ima,1)/2+[-halfwidth:halfwidth],...
>>   1+size(Ima,2)/2+[-halfwidth:halfwidth])=1;
>> figure, imagesc(log(abs(G.*fftshift(fIma))));
>> colormap(gray), axis equal, colorbar
>> figure, imagesc(log(abs((1-G).*fftshift(fIma))));
>> colormap(gray), axis equal, colorbar
```

puis reconstruire les images correspondantes par FFT inverse.

Notez le phénomène de *ringing* (effet de Gibbs).

3 Synthèse de textures

La synthèse de textures est un des problèmes de l'infographie (CGI, *Computer-Generated Imagery*). Il s'agit d'un domaine de recherche actuellement très actif, comme vous pouvez le constater en lançant la requète "texture synthesis" dans Google Scholar (`scholar.google.com`).

Étude préliminaire : expérience d'Openheim et Lim [3] : prenez deux images naturelles de même taille, échangez leurs phases, et visualisez les images correspondant à la transformée de Fourier inverse. Qu'observez-vous ?

Certaines textures sont caractérisées par le *module* de leur transformée de Fourier (et non leur *phase*), ce qu'on va vérifier expérimentalement.

Générez une nouvelle image de texture en remplaçant la phase d'une texture « modèle » par une phase aléatoire répartie uniformément sur un intervalle de longueur 2π (la phase doit néanmoins correspondre à celle d'une image réelle...).

Que pensez-vous des textures synthétisées ? À votre avis, quelles sont les textures qui seront correctement caractérisées par le module de leur transformée de Fourier ?

Cette méthode a été proposée originellement dans [1], et est aussi discutée dans [2].

Pour ceux qui avancent vite : pour les images de texture en couleur, programmez une synthèse en couleur. Vous pourrez récupérer des images de [4].

Pour ceux qui avancent très vite (ou veulent approfondir) : Implantez la méthode de synthèse de textures de taille arbitraire proposée dans la Section V.C. de [2] (disponible sur la page web de [4]).

Références

[1] J.J van Wijk, *Spot noise texture synthesis for data visualization*, Proceedings of SIGGRAPH, 1991.

[2] B. Galerne, Y. Gousseau and J.-M. Morel, *Random Phase Textures : Theory and Synthesis*, IEEE Transactions on Image Processing, 2011. http://perso.telecom-paristech.fr/˜gousseau/random_phase.pdf

[3] A.V. Openheim and J.S. Lim, *The importance of phase in signals*, Proceedings of the IEEE, 1981.

[4] B. Galerne, Y. Gousseau and J.-M. Morel, *Micro-Texture Synthesis by Phase Randomization*, Image Processing On Line, 2011. http://www.ipol.im/pub/art/2011/ggm_rpn/