ENS Cachan
Département de mathématiques
61, avenue du Président Wilson
94235 Cachan cedex

Méthodes stochastiques pour l'analyse d'images (M2 MVA, 2016-2017)

TP : Méthodes par patchs pour la synthèse de textures

Bruno Galerne

Le but de cet TP est de tester expérimentalement l'algorithme d'Efros-Leung grâce à sa démo IPOL ainsi que d'implémenter sous Matlab et de tester l'algorithme d'*image quilting*.

1 Mise en route

Depuis un terminal (placez vous dans le dossier de votre choix) tapez les commandes suivantes pour télécharger les fichiers nécessaires au TP :

```
wget http://w3.mi.parisdescartes.fr/~bgalerne/mva/tp_efros.zip
unzip tp_efros.zip
cd tp_efros
```

Le dossier `tp_efros` contient plusieurs textures et quelques fonctions Matlab.

► **Question 1 :** Vérifiez l'amélioration des résultats de l'algorithme d'Efros-Leung lorsque l'on augmente la taille du voisinage (par exemple avec l'image `matrix.jpg`).

2 *Image quilting* pour la synthèse de texture

Vous pouvez compléter le fichier `mon_script` pour tester les fonctions que vous implémenterez.

Le but de cette partie est d'implémenter les fonctions pour reproduire les étapes de la Figure 1.

Un patch dans une image sera codé par les coordonnées de son coin supérieur gauche ainsi que par sa largeur. Ainsi, en Matlab, pour une image RGB `U` le patch de coordonnées `i,j` et de largeur `w` est la sous-image

$$\mathtt{U}(\mathtt{i}:(\mathtt{i}+\mathtt{w}-1),\mathtt{j}:(\mathtt{j}+\mathtt{w}-1),:)$$

les derniers "`:`" étant ajouter pour obtenir les 3 canaux couleurs.

Le dossier `tp6_efros` contient une fonction

```
NV = replace_patch(V, iv, jv, U, iu, ju, w, mask)
```

qui permet de remplacer le patch de coordonnées `iv,jv` d'une image cible `V` par le patch de coordonnées `iu,ju` d'une image source `U` selon un masque binaire (on change le pixel (i,j) si `mask(i,j)`=1 et on garde l'ancienne valeur sinon).

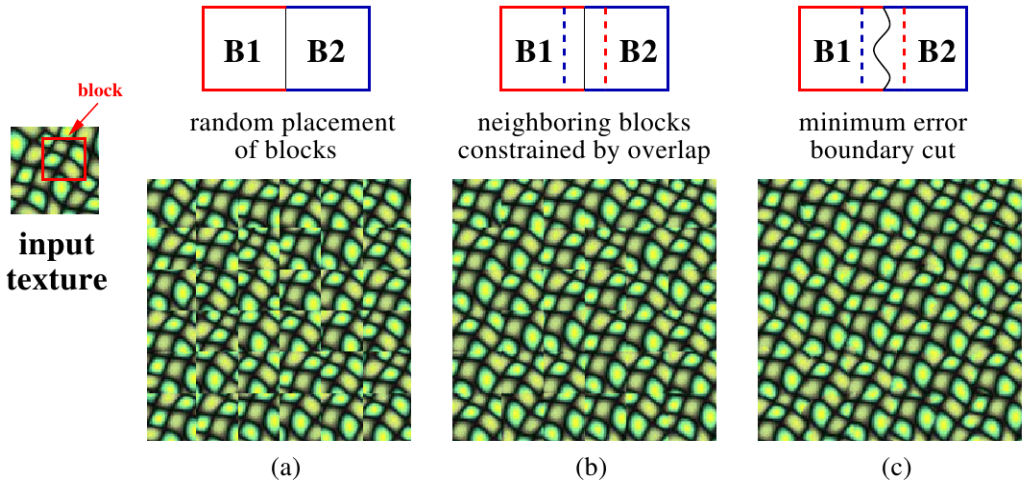

Figure 2: Quilting texture. Square blocks from the input texture are patched together to synthesize a new texture sample: (a) blocks are chosen randomly (similar to [21, 18]), (b) the blocks overlap and each new block is chosen so as to "agree" with its neighbors in the region of overlap, (c) to reduce blockiness the boundary between blocks is computed as a minimum cost path through the error surface at the overlap.

FIGURE 1 – Synthèse de texture par *image quilting*

2.1 Juxtaposition aléatoire de patchs

► **Question 2 :** Ecrire une fonction

```
V = paste_random_patch(U, w, m, n)
```

qui crée une image `V` de taille $mw \times nw$ en juxtaposant des patchs de `U` tirés uniformément (image du type (a) dans la Figure 1). La fonction fera appel à la fonction `replace_patch`, il suffira donc de tirer les coordonnées des patchs.

2.2 Distance entre patchs

On suppose que les images n'ont qu'un seul canal (pour les images RGB il faut sommer sur les trois canaux pour avoir les distances totales) et l'on considère ici un système de coordonnées commençant en $(0, 0)$ (compatible avec la transformée de Fourier discrète).

Etant donnés un patch P de taille $w \times w$, un masque de poids Q de même taille et une image d'entrée U de taille $M \times N$, nous devons déterminer la distance au carré entre P et tous les patchs de U, soit,

$$D(x, y) = \sum_{i,j=0}^{w-1} Q(i, j)(P(i, j) - U(x + i, y + j))^2,$$

pour chaque coordonnée (x, y) "admissible". On note P_{ext} et Q_{ext} les extensions de P et Q en des images de taille $M \times N$ en ajoutant des 0. On note également $\Gamma(V, W)$ la corrélation croisée entre deux images, à savoir

$$\Gamma(V, W)(x, y) = \sum_{s=0}^{M-1} \sum_{t=0}^{N-1} V(s, t)W(x + s, y + t) = \tilde{V} * W(-x, -y),$$

où $\tilde{V}(x, y) = V(-x, -y)$ est l'image symétrisée de V.

► **Question 3 :** Démontrez que pour toute coordonnée (x, y) "admissible",

$$D(x,y) = \Gamma(Q_{\text{ext}}, U^2)(x,y) + \sum_{i,j=0}^{w-1} Q(i,j)(P(i,j))^2 - 2\Gamma(Q_{\text{ext}}P_{\text{ext}}, U)(x,y).$$

En déduire que D se calcule en faisant quatre appels à la FFT.

L'identité de la question précédente est utilisée par la fonction

```
D = squared_distance_to_patches(P,U,mask)
```

2.3 Synthèse par contrainte de superposition

► **Question 4 :** Lire la fonction

```
[iu,ju,dmin] = draw_patch_good_overlap(V, iv, jv, w, U, mask, eps)
```

► **Question 5 :** En faisant appel à la fonction `draw_patch_good_overlap`, compléter le code de la fonction

```
V = paste_patch_overlap(U, w, m, n)
```

dont le but est de reproduire des images du type (b) sur la Figure 1 en juxtaposant $m \times n$ patchs se superposant sur 25% de leur taille, en parcourant l'image de gauche à droite puis de haut en bas. On fera attention aux différentes zones de superposition (verticale, horizontale, ou en forme de L). On prendra $\varepsilon = 0.1$ pour la tolérance $D(x,y) \leqslant (1+\varepsilon)\min D$.

2.4 Coupure minimale

► **Question 6 :** Lire et tester sur des couples de patchs la fonction

```
mask = mincutmask_L(V, iv, jv, U, iu, ju, w, bw)
```

qui calcule la coupure minimale pour des superpositions en forme de L.

► **Question 7 :** Implémenter et tester une fonction

```
mask = mincutmask_left(V, iv, jv, U, iu, ju, w, bw)
```

qui calcule la coupure minimale pour une zone de superposition verticale à gauche du nouveau patch de sommet `U(iu,ju)`.

Cette fonction est utilisée pour la première ligne de la texture pour l'algorithme de quilting. Cette nouvelle fonction est également appelée par `mincutmask_top` qui gère le cas de la première colonne.

2.5 Quilting

Grâce à la fonction `mincutmask_left` vous pouvez maintenant appeler la fonction

```
function V = quilting(U, w, M, N)
```

► **Question 8 :** Lire et tester cette fonction sur des textures. Observez que la qualité de la synthèse se dégrade souvent lorsque la hauteur et où la longueur de la texture est élevée par rapport à celle du bloc.

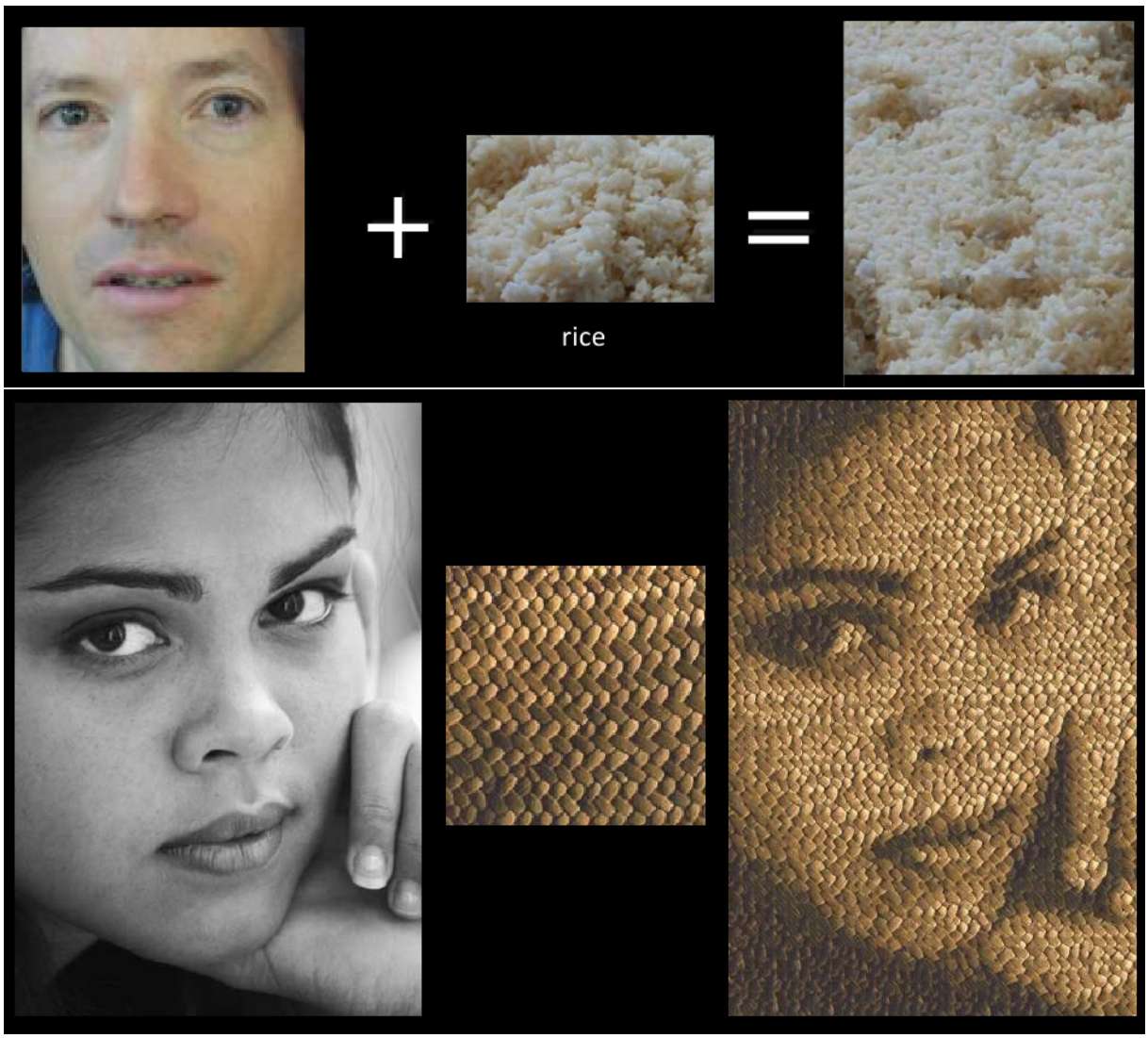

FIGURE 2 – Deux exemples de transfert de texture

2.6 (Facultatif) Transfert de texture

► **Question 9 :** Adapter l'algorithme pour guider le choix des patchs afin de guider la synthèse par une image comme à la Figure 2. On pourra effectuer des comparaisons entre des versions noir et blanc des images dont les histogrammes ont été égalisés.

3 Algorithme d'Efros-Leung

Pour tester l'algorithme d'Efros-Leung nous allons utiliser la démo en ligne :

Exemplar-based Texture Synthesis : the Efros-Leung Algorithm
de Cecilia Aguerrebere, Yann Gousseau et Guillaume Tartavel :
`http://demo.ipol.im/demo/59/`

► **Question 10 :** Vérifiez l'amélioration des résultats de l'algorithme d'Efros-Leung lorsque l'on augmente la taille du voisinage (par exemple avec l'image `matrix.jpg`).