Images avec Python : manipulations bovines

Le 18 mai 2015
http://blog.psi945.fr/ - stephane@gonnord.org

Buts du TP

- Comprendre la représentation « tableau de pixels » des images.
- Distinguer les trois niveaux d'abstractions des images : fichier, objet Python et matrice de pixels.
- Réaliser des traitements simples d'images.

Exercice 1 Exhumer les transparents « Images en Python » du machin dans lequel vous entassez ce qui concerne l'informatique.

Exercice 2 Créer (au bon endroit) un dossier associé à ce TP. Lancer Spyder, sauvegarder immédiatement le fichier édité au bon endroit sous un nom pertinent. Écrire une commande absurde, de type `print(6*7)` dans l'éditeur, sauvegarder. Avec F6, imposer les directives « exécuter dans un nouvel interpréteur dédié » et « interagir avec l'interpréteur Python après exécution ». Exécuter.

Exercice 3 Dans le dossier du TP du jour, créer un sous-dossier `images-entree` et placer dedans les fichiers suivants [1] :

`vache.jpg`, `goutte1.png`, `goutte2.png`

Créer également un sous-dossier `images-sortie` dans lequel vous placerez les images créées en cours de ce TP.

Figure 1 – Mafalda va nous accompagner pendant tout ce TP

1 Premières manipulations

Exercice 4 Taper les lignes suivantes dans le script Python [2] ; sauver, exécuter, comprendre !

```
import Image.PIL as im
import numpy as np
entree = 'images_entree\\'
sortie = 'images_sortie\\'

meuh = im.open( entree + 'vache.jpg' )
meuh.save( sortie + 'vache.bmp' )

meuhBW = meuh.convert('L')
meuhBW.save( sortie + 'vacheBW.jpg' )

(L, H) = meuh.size
```

1. extraits de l'archive `materiel-tp-images.zip` fournie dans le répertoire de travail de la classe.
2. Si `Image.PIL` ne passe pas, on pourra essayer `PIL`, `PIL.Image`, `Image`, `PIL.image`...

Observer le poids (la taille !) des différents fichiers sur le disque.

Exercice 5 En s'inspirant du cours :
- placer un carré blanc dans la zone `[ 100:200 , 0:50 ]` de Mafalda (noir et blanc) ;
- placer un carré blanc à la place de la cloche de Mafalda (N&B) ;
- placer un carré violet à la place de la cloche de Mafalda (RGB).

Toutes les images seront bien entendu sauvées dans le répertoire de sortie.

2 Transformations ponctuelles

Dans cette partie, on applique à l'image des transformations ponctuelles : la nouvelle valeur du pixel $M_{i,j}$ devient $f(M_{i,j})$, avec f une application de $[\![0, 255]\!]$ dans lui-même.

Exercice 6 Définir en Python la fonction $f_1 : x \mapsto \begin{cases} 0 & \text{si } x \leqslant 64 \\ 128 + 2(x - 128) & \text{si } 64 < x < 192 \\ 255 & \text{si } x \geqslant 192 \end{cases}$

Vérifier sa bonne définition en demandant quelques valeurs (dans l'interpréteur) puis en traçant le graphe :

```
import matplotlib.pyplot as pypl

les_x = range(256)
les_y = map(f1, les_x)

pypl.plot(les_x, les_y)
pypl.savefig('f1.pdf')
```

Exercice 7 Appliquer f à chaque pixel de `vacheBW.jpg` comme vu dans le cours :
- d'une part via une double boucle ;
- d'autre part par application directe de f_1 vectorisée sur la matrice des pixels.

Comparer les temps de calculs.
Appliquer f_1 aux pixels (RGB) de `vache.jpg`

Exercice 8 Définir maintenant l'application $f_2 : x \mapsto 128 + (x - 128)/2$ (avec une division euclidienne, pour rester chez les entiers). Appliquer cette fonction à Mafalda.

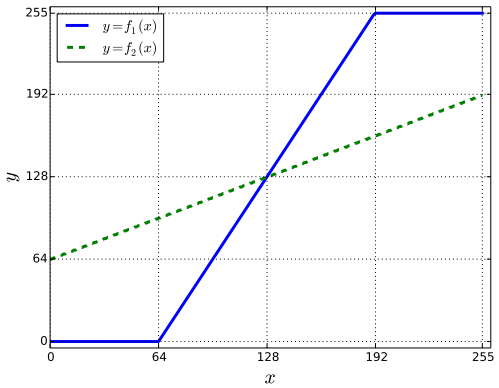

Figure 2 – Augmentation ou atténuation de contraste

Exercice 9 Définir en Python les application $f_3 : x \mapsto 255 + (x - 255)/2$ et $f_4 : x \mapsto x/2$. Les appliquer à Mafalda.

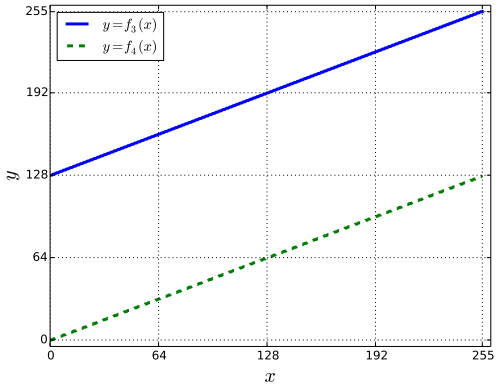

FIGURE 3 – Augmentation ou atténuation de la luminosité

3 Transformations locales

Dans cette partie, on applique à l'image des transformations locales : la nouvelle valeur du pixel $M_{i,j}$ ne dépend plus seulement de la valeur initiale de $M_{i,j}$, mais des valeurs de $M_{i,j}$ et de ses voisins plus ou moins proches.

- Pour le floutage, on remplace $M_{i,j}$ par la moyenne de ses 9 voisins... ou par celle de ces 25 voisins. On pourra définir une constante d (égale à 3 en première approximation) et faire la moyenne des $(2d+1)^2$ voisins. En fonction du résultat obtenu, on prendra ensuite d plus grand ou plus petit.
- Pour détecter les bords, on définit $M_{i,j}$ comme étant égal à 0 (pixel noir) si la variation en (i,j), égale à
$$V_{i,j} = (M_{i+1,j} - M_{i-1,j})^2 + (M_{i,j+1} - M_{i,j-1})^2$$
dépasse un certain seuil s... et 255 (blanc) si $V_{i,j} \leqslant s$. On pourra prendre par exemple $s = 10^3$.

EXERCICE 10 Écrire une fonction `floutage` prenant en entrée une matrice (`array`) M et renvoyant la matrice F telle que :

$$\forall (i,j) \in [\![d, H-d-1]\!] \times [\![d, L-d+1]\!], \qquad F_{i,j} = \frac{1}{(2d+1)^2} \sum_{a=i-d}^{i+d} \sum_{b=j-d}^{j+d} M_{i,j}$$

On aura préalablement défini la matrice F comme une copie de M, et déterminé les dimensions via :

```
H, L = m.shape
Floue = np.copy(M)
```

Pour faire la moyenne d'un bloc autour de (i,j), on pourra utiliser le slicing :

```
np.sum( m[i-d:i+d+1 , j-d:j+d+1] )
```

EXERCICE 11 Écrire une fonction `flouter` prenant en entrée deux noms de fichiers (avec le chemin), telle que `flouter(source, but)` floute l'image du fichier `source` et sauvegarde le résultat dans le fichier `but`
Faire quelques essais, et ajuster la valeur de d.

```
flouter(sortie + 'vacheBW.jpg', sortie + 'vache-floue.jpg')
flouter(sortie + 'vache-floue.jpg', sortie + 'vache-floue2.jpg')
```

EXERCICE 12 Écrire une fonction `bordure` prenant en entrée une matrice M et renvoyant la matrice B définie comme dans le préambule de cette partie (constituée de 0 et de 255, donc). Écrire également une fonction `bords` sur le même modèle que `flouter` (exercice précédent).
On supposera ici que M est une matrice de pixels noirs et blancs.

Exercice 13 Essais bovins :

- Détecter les bords de Mafalda.
- Déterminer les bords de Mafalda contrastée.
- Observer le résultat obtenu si on définit la variation comme

$$V_{i,j} = (M_{i+d,j} - M_{i-d,j})^2 + (M_{i,j+d} - M_{i,j-d})^2$$

  avec $d \geqslant 1$ une distance à ajuster.
- Comment modifier (de façon la plus économique possible !) la fonction `bords` pour qu'elle puisse s'appliquer à des images RGB ?

Exercice 14 Vous pouvez jouer avec les deux images de gouttes fournies : contrastage, floutage, détection des bords...

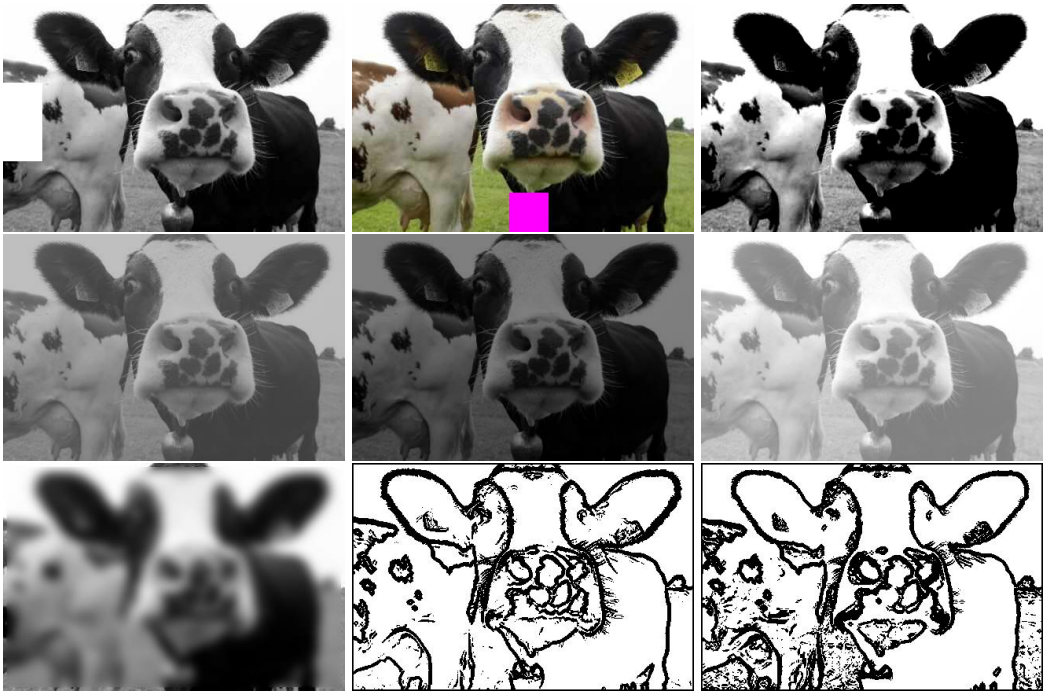

Figure 4 – 9 shades of Mafalda