TP 16 informatique

BCPST 1 2019-2020

Points abordés

— Tableaux.
— Bibliothèques, manipulation d'images.

---

1 Codage des images

Une image peut être modéliser par une matrice de taille $n \times p$, où chaque case de la matrice contient une donnée codant la couleur ou le niveau de gris. Par exemple, dans le cas des images en noir et blanc, la case contient un entier entre 0 et 255, 0 correspondant au noir et 255 au blanc. Pour les images en couleur, la case contient un triplet (R,V,B) où R, V, B sont des entiers compris entre 0 et 255, le R correspondant à l'intensité du rouge, le V à l'intensité du vert et le B à l'intensité du bleu.

Ayant codé une image par une matrice, toute traitement de l'image se traduit alors par une fonction qui modifie la matrice.

2 Bibliothèques utilisées et fonctions utiles

Pour pouvoir manipuler les images, nous aurons besoin des bibliothèques suivantes :

1. `numpy`
2. `matplotlib`
3. `imageio`

Voici un prototype de code permettant de créer une image modifiée à partir d'une image initiale pixels par pixels :

```
import imageio                      # bibliothèques nécessaires
import matplotlib as mpl
import matplotlib.pyplot as plt
import numpy

def fonctionExemple(image) :
    nblignes=len(image)
    nbcolonnes=len(image[0])
    imageModifiee=numpy.zeros((nblignes,nbcolonnes),numpy.uint8)
    for i in range(nblignes) :
        for j in range(nbcolonnes) :
            imageModifiee[i][j]= "une manière de modifier"
    return imageModifiee
```

La ligne `imageModifiee=...` permet de construire un tableau de zéro que l'on modifie par la suite pour obtenir notre nouvelle image.

Si on veut transformer une image au format png ou jpg en tableau, on écrit dans le terminal :

```
imageEnTableau=imageio.imread("nomFichier")
```

le tableau `imageEnTableau` contient alors une représentation de "nomFichier" en tableau. Attention à l'emplacement de votre fichier.

Si on veut sauvegarder un tableau sous forme de fichier image, la commande est la suivante :

```
imageio.imsave("nomFichier",imageEnTableau)
```

où `imageEnTableau` est le tableau qui code une image, et `nomFichier` est le nom de sauvegarde de votre image. Attention à l'emplacement dans lequel vous sauvegardez votre image.

Si on veut afficher l'image correspondant au tableau `Image`, on saisit :

```
plt.imshow(Image,vmin=0,vmax=255)
```

Pour l'afficher en gris, on saisit `plt.imshow(Image,cmap=''gray'',vmin=0,vmax=255)`

3 Exercices

Dans les exercices qui suivent, on ne s'intéresse qu'aux images en noir et blanc. Télécharger les images et enregistrez-les sur votre PC à l'emplacement de votre choix.

Exercice 1. On rappelle que le négatif d'une image correspond à inverser les niveaux de gris. Écrire une fonction `negatif(image)` qui remplace la valeur i d'un pixel par $255 - i$. Appliquer cette fonction à l'une des images fournies.

Exercice 2. Écrire une fonction `seuillage(n,image)` qui remplace toutes les valeurs i du tableau image par 255 si $i \geq n$ et 0 sinon. Essayer cette fonction sur l'une des images fournies avec différents seuils, par exemple 25, 50, 100.

Exercice 3. Pour assombrir une image, on peut appliquer la transformation suivante :

$$t \mapsto \sqrt{t}.$$

Écrire une fonction permettant d'assombrir une image en utilisant cette fonction. On pourra tester la fonction sur une des images fournies.

Exercice 4. Écrire une fonction `Flou(image)` qui remplace la valeur i de chaque pixel par la moyenne des valeurs qui l'entoure. Pour simplifier, on ne changera pas la valeur du bord. On pourra tester la fonction sur une des images fournies.

Exercice 5. Un contour correspond à une variation importante entre un pixel positionné en (i, j) et ces 8 voisins. On pose :

$$\begin{aligned} V &= 8Im[i][j] - Im[i-1][j-1] - Im[i-1][j] - Im[i-1][j+1] \\ &\quad -Im[i][j-1] - Im[i][j+1] - Im[i+1][j-1] - Im[i+1][j] - Im[i+1][j-1] \end{aligned}$$

où Im est un tableau représentant une image en noir et blanc.

1. Que dire d'un pixel dont la quantité $|V|$ est petite ? grande ?
2. En déduire un algorithme déterminant le contour d'une image.
3. Écrire une fonction `contour(image,seuil)` qui construit une image ne gardant que les contours dans l'image initial. On pourra tester la fonction avec l'image "Hoche" ou "chocolat".

Exercice 6. On veut superposer deux images. Écrire une fonction qui permet de fusionner deux images de la manière suivante : si en position (i, j) la valeur du pixel est respectivement k et k' pour les deux images, la valeur du pixel de l'image superposée en position (i, j) sera $\lfloor \frac{k+k'}{2} \rfloor$. Généraliser en considérant $\alpha \in [0, 1]$ et tel que la valeur de sortie est $\lfloor \alpha k + (1 - \alpha)k' \rfloor$.

Exercice 7. $\star$ Écrire une fonction qui permet de crypter une image à l'aide de la procédure suivante : on dispose d'une image `S` que l'on veut crypter, et d'une image `M` qui sert de clé. L'image cryptée `C` est alors obtenue en effectuant l'opération suivante : pour chaque pixel (i, j), on a $C[i][j] = 17 * M[i][j] + 7 * S[i][j]$ modulo 256. On supposera que l'image à crypter est de même taille que l'image clé. Pour déchiffrer une image C, il suffit de considérer l'image D obtenue en effectuant la transformation $D[i][j] = 241 * (C[i][j] - 7 * S[i][j])$ modulo 256 pour chaque pixel (i, j). Sachant que l'image "mystere" a été obtenue à l'aide de la procédure présentée avec comme image clé "symbole", retrouver l'image qui a été chiffrée.