Librairies Python

Yann Strozecki
yann.strozecki@uvsq.fr

Université de Versailles St-Quentin-en-Yvelines

Année universitaire 2022-2023

Charger et manipuler une image avec Pillow

Pour lire et manipuler les images en Python, nous allons utiliser la librairie **Pillow**. Voir la documentation sur `https://pillow.readthedocs.io/en/stable/index.html`. Pour apprendre, le plus simple est de faire le tutoriel.

Ouvrir une image :

```
from PIL import Image
im = Image.open("hopper.ppm")}
```

Afficher ses informations :

```
print(im.format, im.size, im.mode)
PPM (512, 512) RGB
```

Afficher l'image :

```
im.show()
```

Quelques exemples de méthodes

La librairie Pillow, en plus de permettre de charger et de convertir de nombreux formats d'images.

Redimensionner l'image :

```
out = im.resize((128, 128))
```

Faire une rotation de l'image :

```
out = im.rotate(45)
```

Passer en niveau de gris :

```
im = im.convert("L")
```

Appliquer une fonction à chaque pixel de l'image :

```
out = im.point(lambda i: i * 1.2)
```

Numpy

Pour tout type de calcul scientifique et en particulier manipuler des tableaux à une ou plusieurs dimension, la librairie de référence est NumPY. La documentation : `https://numpy.org/doc/stable/` et un tutoriel `https://numpy.org/doc/stable/user/absolute_beginners.html`.

Création d'un tableau à une dimension :

```
a = np.array([1, 2, 3, 4, 5, 6])
```

Création d'un tableau à deux dimensions :

```
a = np.array([[1, 2, 3, 4], [5, 6, 7, 8], [9, 10, 11, 12]])
```

Tableaux spéciaux :

```
a = np.zeros(2)
a = np.arange(4)
```

Accéder aux informations d'un tableau

Obtenir les informations d'un tableau. Nombre de dimension, nombre d'éléments, dimensions :

```
array_example.ndim
array_example.size
array_example.shape
```

Accéder à un élément :

```
array_example.ndim[3,5]
```

Extraire un sous-tableau(slicing) :

```
a[:8]
array_example[3:4,5:7]
array_example[3,:]
```

Manipulation de tableau

Créer une matrice à partir d'une image :

```
toLoad= Image.open(filename)
np.asarray(toLoad)
```

Ajouter deux tableaux de taille égale :

```
for i in range(len(a)):
  c[i] = a[i] + b[i]
```

Marche aussi sur les tableaux multidimensionnels de mêmes dimensions.

```
c = a + b
```

Multiplier par un scalaire ou une matrice :

```
5 * b
np.matmul(a, b)
```

Méthodes des tableaux existante

Changer les dimensions d'un tableau :

```
b = a.reshape(3, 2)
```

Trier, sommer les éléments d'un tableau :

```
np.sort(a)
a.max()
a.sum()
```

Sélectionner certains éléments d'un tableau :

```
a[(a > 2) & (a < 11)]
```

Image bitmap

Il existe de nombreuses façons de représenter une image. Nous utiliserons la plus simple, l'image bitmap. Il s'agit de stocker l'image comme un tableau de pixels RGB.

Il s'agit d'un tableau M à deux dimensions (largeur$\times$hauteur) où $M[i, j]$ est un triplet de trois entiers dans $[0, 255]$ qui représente l'intensité des canaux rouge, vert et bleu (RGB).
Le pixel $M[0, 0]$ est le pixel en haut à gauche de l'image. L'image se lit ligne par ligne du haut vers le bas et chaque ligne se lit de la gauche vers la droite.

Image bitmap

Il existe de nombreuses façons de représenter une image. Nous utiliserons la plus simple, l'image bitmap. Il s'agit de stocker l'image comme un tableau de pixels RGB.

Il s'agit d'un tableau M à deux dimensions (largeur$\times$hauteur) où $M[i,j]$ est un triplet de trois entiers dans $[0,255]$ qui représente l'intensité des canaux rouge, vert et bleu (RGB).
Le pixel $M[0,0]$ est le pixel en haut à gauche de l'image. L'image se lit ligne par ligne du haut vers le bas et chaque ligne se lit de la gauche vers la droite.

Dessiner l'image suivante :
$[[(0,0,255),(255,0,0)],[(0,0,0),(255,255,255)]]$

Image en Numpy

Pour stocker une image RGB, on utilise une matrice à trois dimensions en Numpy. Par exemple $M[0,0,0]$ me donne l'intensité de rouge du pixel en haut à gauche de l'image. $M[0,0,:]$ me donne le triplet de valeurs RGB pour ce pixel.

Exemple de programme pour échanger le canal vert avec le bleu :

```
for i in range(M.shape[0]):
    for j in range(M.shape[1]):
        (M[i,j,1],M[i,j,2])=(M[i,j,2],M[i,j,1])
```

Version utilisant les méthodes de Numpy.

```
Temp = M[:,:,1].copy()
M[:,:,1] = M[:,:,2]
M[:,:,2] = Temp
```

Créer une interface utilisateur avec Tkinter

Tkinter est une interface pour Tk, une librairie de création d'interface graphique.
Le cours avec exemple est disponible sur `https://github.com/uvsq-info/l1-python/blob/master/cours/interface_graphique/07_gui.ipynb` et voici un site de référence : `http://tkinter.fdex.eu/`.
Tkinter permet de créer des fenêtres avec des widgets :

- bouton
- label (afficher du texte)
- canvas (afficher des images et des formes)
- champs de saisie

La librairie permet aussi de réagir aux entrées utilisateur (déplacement de souris, clic de souris, entrée clavier . . .).