La retouche d'images par programmation python

Les drapeaux … d'après le travail de M. CANTUS

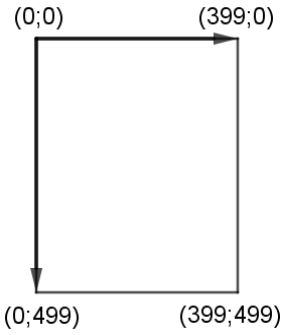
1. Introduction

Nous avons vu en cours qu'une image est constituée de pixels et que la couleur d'un pixel peut être définie par le triplet (R,V,B).

> **Exemples** : le triplet (255, 0, 0) signifie que le pixel est rouge (le rouge est à fond, bleu et vert ne sont pas utilisés) ; (0, 0, 0) signifie que le pixel est noir (absence de couleurs) ; (255, 255, 0) signifie que le pixel est jaune (synthèse additive du rouge et du vert).

Les positions des pixels par contre sont décrites grâce à un repère qui par convention :

- a l'origine du repère situé en haut à gauche de l'image ;
- a l'axe des abscisses qui correspond au bord supérieur de l'image et pointe vers la droite, et celui des ordonnées correspond au bord gauche de l'image et pointe vers le bas ;
- commence à compter à 0.



Ainsi dans l'exemple ci-contre, les abscisses correspondent à 400 valeurs numérotées de 0 à 399 et les ordonnées à 500 valeurs numérotées de 0 à 499.
Le pixel situé à l'intersection de la ligne `l` et de la colonne `c` de l'image a donc pour coordonnées `(c,l)`.

2. Lecture de la couleur d'un pixel

2.1. Programme de base

A l'aide du logiciel Edupython, ouvrir le fichier **`analyse.py`**.

Voici le début de code d'une fonction qui renverra le triplet des valeurs (R,V,B) décrivant la couleur du pixel de coordonnées `coord=(c,l)` (situé à l'intersection de la ligne `l` et de la colonne `c` de l'image.) A partir de la ligne 6, l'image est mémorisée dans la variable `image`.

```
1 """ Ne pas modifier ces premières lignes !"""
2 # Importation des librairies utilisées
3 from PIL import Image
4 # Déclaration de l'image à utiliser, on la mémorie dans la variable image
5 nom="statue.jpg"
6 image=Image.open(nom)
7
8 def lecture_pixel(coord):
9     """ Partie modifiable ci_dessous"""
```

2.2. Entraînement

1. Compléter le code de la façon ci-contre. En ligne 9, on lit le triplet décrivant la couleur (R,V,B) du pixel à l'aide de la méthode `.getpixel`

```
    """ Partie modifiable ci_dessous"""
    couleur=image.getpixel(coord)
    return couleur
```

2. Tester votre programme en appelant la fonction via la console pour le pixel de coordonnées (0,0). Vous devriez obtenir le triplet (142, 184, 200).Attention à ne pas oublier de parenthèse, et à utiliser une virgule pour séparer les coordonnées.

```
>>> lecture_pixel((0,0))
(142, 184, 200)
```

3. Déterminer les triplets (R,V,B) des pixels de coordonnées (399,0), (399,499) et (100,212).

3. Les drapeaux

3.1. Programme de base

Ouvrir le fichier **`creation.py`** et sauvegarder le script sous le nom **`DrapeauCyan.py`**.

Nous allons créer une image totalement cyan. Quand on déclare une nouvelle image, on commence par préciser son mode (ici RGB) et sa définition sous forme d'un couple (largeur, hauteur).
A la fin, on la sauvegarde en lui donnant un nom et en précisant on format (ici c'est png) et on la visualise automatiquement.

```
1  # Importation des librairies utilisées
2  """ Partie à ne pas modifier !"""
3  from PIL import Image
4  nouvelleImage=Image.new("RGB",(90,60))
5  """ Partie à modifier : description des pixels"""
6
7
8  """ Partie à ne pas modifier : sauvegarde puis affichage de l'image."""
9  nouvelleImage.save("cyan.png")
10 nouvelleImage.show()
```

Par défaut, un pixel non décrit est considéré comme noir. En lançant le script, nous créons et visualisons un drapeau noir. Essayer en cliquant sur la flèche verte !

Il reste à colorer chaque pixel, en précisant sa position (ses coordonnées) et la couleur choisie (triplet de nombres RGB). On utilise pour cela la méthode **`.putpixel()`** qui fonctionne ainsi **:**

`.putpixel(couple de coordonnées , triplet couleur)`

Par exemple, définir en cyan le pixel de coordonnées (0,0) dans l'image appelée `nouvelleImage` se code ainsi :

`nouvelleImage.putpixel((0,0),(0,255,255))`

3.2. Le drapeau cyan

1. Compléter le code pour que la première ligne soit de couleur cyan.
2. Compléter le code pour que les 2 premières lignes soient de couleur cyan.
3. Compléter le code pour que les 3 premières lignes soient de couleur cyan.
4. Terminer le code.

Attention, petite subtilité avec Python. L'instruction « **`for l in range(0,60) :`** » signifie qu'on prend $0 \leq l < 60$. On compte donc bien 60 valeurs entières pour la variable `l`, ni plus, ni moins. Ici 60 est le premier entier non pris en compte.

3.3. Le drapeau allemand

1. Ouvrir le fichier **`creation.py`** et sauvegarder le script sous le nom **`drapeauAll.py`** puis modifier la ligne 9 en remplaçant **`cyan.png`** par **`allemand.png`.**
2. Coder une image représentant le drapeau allemand.

3.4. Le drapeau français

1. Ouvrir le fichier **`creation.py`** et sauvegarder le script sous le nom **`drapeauFr.py`** puis modifier la ligne 9 en remplaçant **`cyan.png`** par **`france.png`.**
2. Coder une image représentant le drapeau français.

3.5. Le défi : le drapeau suédois

Coder une image représentant le drapeau suédois. Toutes les répétitions seront codées de préférence à l'aide des boucles.