Introduction au traitement d'images

CIEL – 2

K. Boudjelaba

Table des matières

Introduction

- Une image est un ensemble de points qui représentent une forme au sens large du terme (paysage, personne, objet, dessin . . .).
- Les points d'une image sont généralement appelés des pixels (du terme anglais "picture element"). Ce terme est réservé aux images numérisées.
  Dans le cas d'une image 3D, on les appelle voxels (volume element).
- Une image peut être représentée sous la forme d'une matrice de dimensions $N \times M$, où chaque élément serait un pixel. N et M définissent la taille de la matrice et donc la taille de l'image numérisée encodée par la matrice.
- Le traitement d'images va se résumer, pour l'essentiel, à du calcul matriciel et de la manipulation d'éléments d'une matrice.

Image numérique

Si x et y sont les coordonnées spatiales d'un point de l'image, $I(x, y)$ est une fonction de l'intensité lumineuse et de la couleur.

- $I(x, y)$ signal analogique continue
- Inexploitable par la machine $\Longrightarrow$ numérisation

Numérisation d'une image

Les images sont des signaux 2-D, l'échantillonnage se fait selon les dimensions spatiales x et y (et non pas selon le temps comme pour les signaux).
Le nombre de niveaux de quantification de la luminance est généralement de 256, chaque niveau étant alors codé sur 8 bits (code binaire naturel).

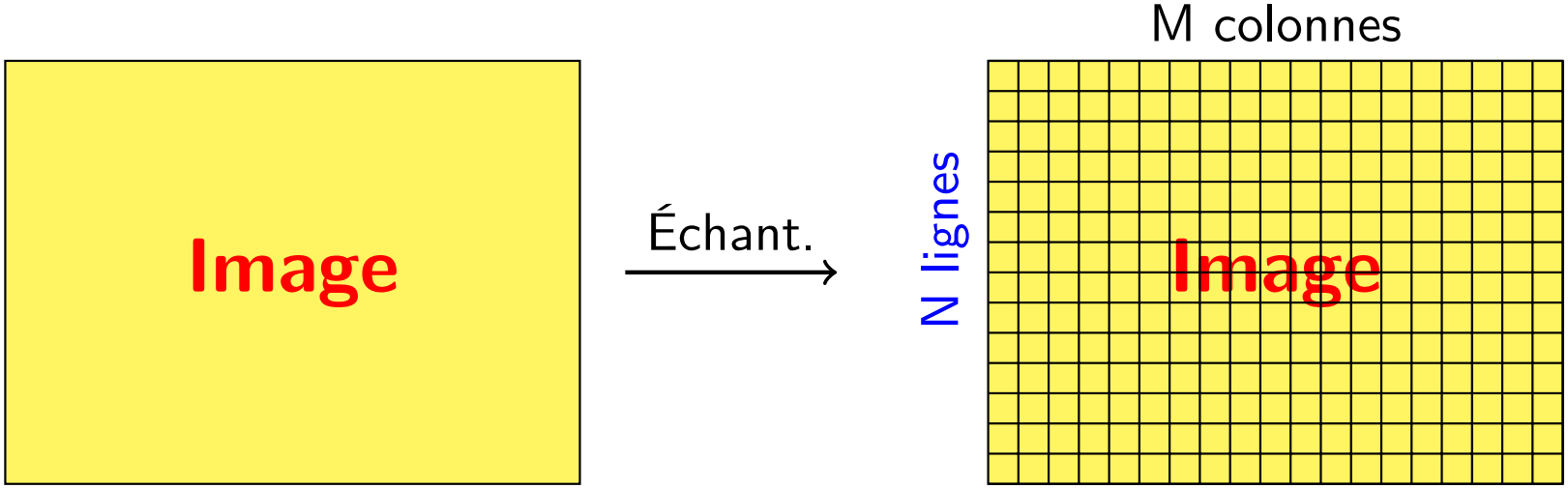


Image numérique - Pixel

Un pixel est un nombre, qui code son aspect dans l'image, sa couleur et parfois sa transparence. La nature informatique du nombre, short, integer, ou long, dépend donc du nombre de couleurs que l'on veut coder.

- Si on veut coder une image en noir et blanc, un seul bit est suffisant pour exprimer un pixel.
- Si on veut coder une image en 256 ($= 2^8$) couleurs, il faudra 8 bits et pour 64K couleurs il faudra 16 bits ($2^{16} = 65536$). Ainsi, une image de 64K couleurs sera codée par une matrice de pixels, chacun étant codé par un nombre de 16 bits (en Python un entier sera donc largement suffisant).

Pour résumer, dans une image, un pixel est caractérisé par sa position qui est indiquée par les indices ligne et colonne de la matrice et par sa valeur, qui code la couleur du pixel.

Image numérique - Caractéristiques

- **Dimension :** Comme l'image est une matrice. Le nombre de lignes de cette matrice multiplié par le nombre de colonnes nous donne le nombre total de pixels dans cette image.
- **Résolution :** C'est la clarté (finesse de détails). La résolution est exprimée en nombre de pixels par unité de mesure (pouce ou centimètre). DPI : Dots per inch.
- **Luminance :** C'est le degré de luminosité des points de l'image.
- **Contraste :** Le contraste est défini en fonction des luminances de deux zones d'images.
- **Types d'images :** Binaire, en niveaux de gris, en couleur ...
- **Dimension des pixels :** Chaque pixel d'une image peut être scalaire ou vectoriel. Par exemple, les pixels d'une image 2D à niveaux de gris ne contiennent qu'une seule valeur (l'intensité du gris). Les pixels d'une photographie classique contiennent trois valeurs (correspondant aux proportions de rouge, de vert et de bleu).

Image numérique

Types d'images numériques

Image binaire : Matrice de 0 et de 1 (0 noir, 1 blanc)

Image en niveaux de gris : La couleur du pixel peut prendre des valeurs allant du noir au blanc en passant par un nombre fini de niveaux intermédiaires.
Matrice d'entiers compris entre 0 et L_{max}.
En général, $L_{max} = 2^n - 1$

Image en couleur : Représenter les couleurs à l'aide de leurs composantes primaires (modèle R.G.B. par exemple).
Combinaison de trois images (RGB) (Red, Green, Blue)

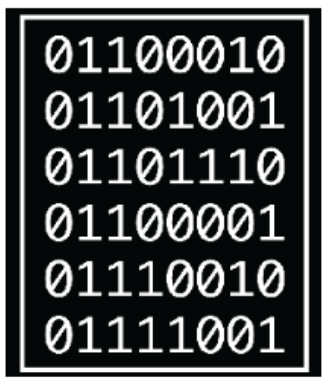


Image numérique

Types d'images numériques

Image binaire : Matrice de 0 et de 1 (0 noir, 1 blanc)

Image en niveaux de gris : La couleur du pixel peut prendre des valeurs allant du noir au blanc en passant par un nombre fini de niveaux intermédiaires.
Matrice d'entiers compris entre 0 et L_{max}.
En général, $L_{max} = 2^n - 1$

Image en couleur : Représenter les couleurs à l'aide de leurs composantes primaires (modèle R.G.B. par exemple).
Combinaison de trois images (RGB) (Red, Green, Blue)

Formats d'images (fichiers)

- **Bitmap (.bmp) :** Non compressé
- **GIF (.gif) :** Compressé. Conservation d'une qualité très correcte
- **TIFF (.tif) :** Compressé. Adapté aux impressions
- **JPEG (.jpg) :** Compressé. Mauvaise conservation de la qualité ...

Système de traitement d'images numériques

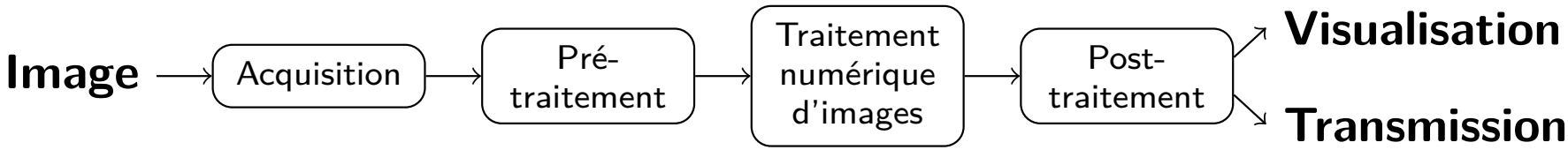


Figure 1: Une chaîne de traitement d'images

- **Pré-traitement :** Pour améliorer la qualité de l'image en vue de sa segmentation (modification d'histogramme, réduction du bruit par filtrage . . .)
  - Améliorer la qualité visuelle de l'image
  - Restaurer l'image en éliminant les défauts et les bruits et en renforçant l'information utile qu'elle contienne.
  - Compresser l'image afin de réduire l'espace nécessaire à son stockage et diminuer le temps de sa transmission
- On désigne par **traitement numérique d'images** l'ensemble des techniques permettant de modifier une image numérique afin d'améliorer ou d'en extraire des informations.

Transformations des niveaux d'intensité

Soit une image $f(x, y)$. En appliquant une transformation T sur les valeurs d'intensité de $f(x, y)$, on obtient une image améliorée $g(x, y)$.

$g(x, y) = T\left[f(x, y)\right]$

Une transformation permet de modifier la valeur de chaque pixel afin d'obtenir une nouvelle image de même taille mais ayant des propriétés plus intéressantes.

1. **Ponctuelle** (pixel à pixel) : La nouvelle valeur $g(x, y)$ est fonction de $f(x, y)$ seulement. Exemples : correction gamma, inversion, manipulations d'histogramme, ajustement luminosité/contraste, recadrage dynamique
2. **Locales :** de voisinage : $g(x, y)$ est obtenue à partir de l'ensemble des valeurs $f(V(x, y))$ dans un voisinage autour du pixel (x, y). Exemple : filtrage
3. **Globale :** La nouvelle valeur de chaque pixel $g(x, y)$ dépend de l'image initiale. Exemple : transformation dans l'espace de Fourier

Transformations des niveaux d'intensité

Inversion dynamique

On inverse les extrêmes noir et blanc. Parfois, on distingue mieux certains détails en blanc sur fond noir qu'en noir sur fond blanc :
$g(x, y) = g_{max} - f(x, y) = 255 - f(x, y)$.
$g_{max} = 255$ pour une image 8 bits.

Transformation logarithmique

Cette transformation de l'intensité permet de dilater les intensités faibles et de compresser les niveaux d'intensités élevées.

$$g(x, y) = \frac{g_{max}}{\ln(1 + g_{max})} . \ln(1 + f(x, y))$$

Correction Gamma

Dilatations linéaires

...

Histogramme d'une image

L'histogramme des niveaux de gris (couleurs) d'une image est une fonction qui donne la fréquence d'apparition de chaque niveau de gris (couleur) dans l'image.
Convention : l'abscisse d'un histogramme représente les niveaux d'intensité allant du plus foncé à gauche au plus clair à droite.
L'histogramme d'une image est la fonction discrète h telle que :

$$\boxed{h(i) = n_i}$$

où n_i est le nombre de pixels de l'image ayant l'intensité i.

Propriétés

L'histogramme peut être interprété comme la densité de probabilité discrète des intensités si les effectifs sont normalisés par le nombre de pixels $N \times M$:

$$p(i) = \frac{n_i}{N \times M}$$

Histogramme d'une image

Exemple

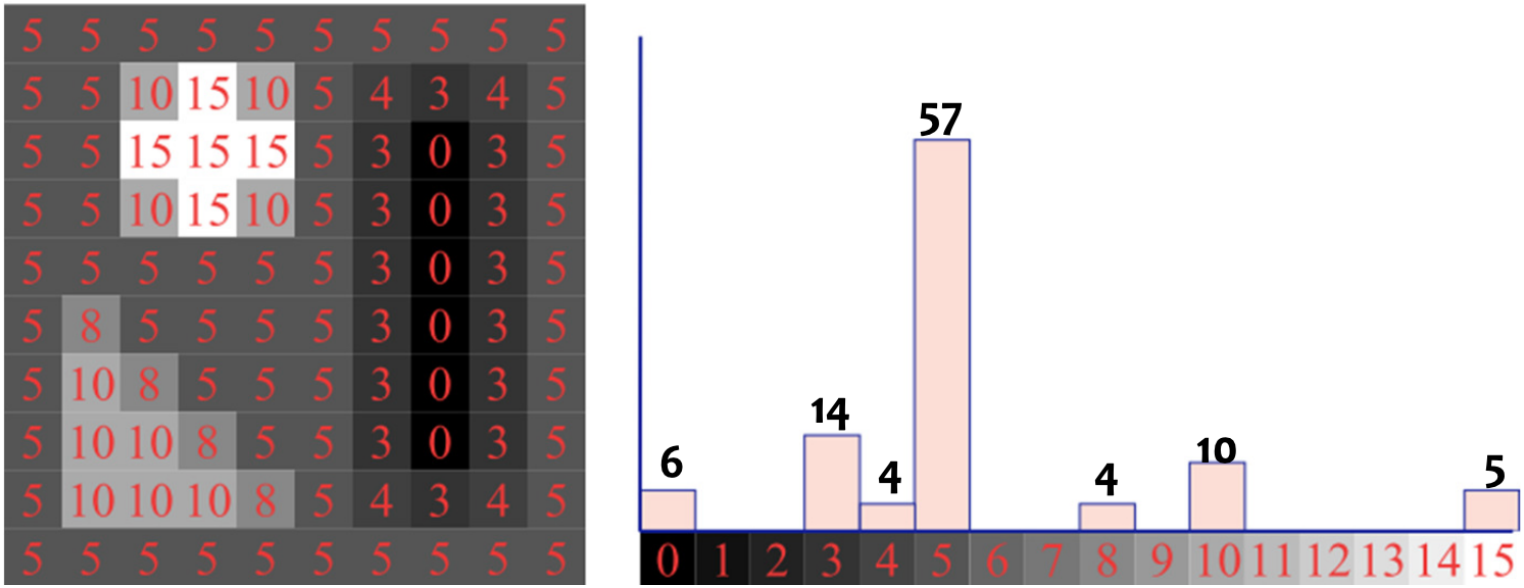
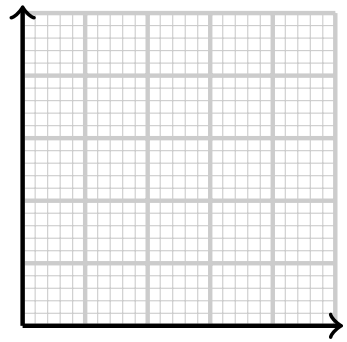


Figure 2: Image (à gauche) et son histogramme (à droite)

Exercice

Représenter l'histogramme de l'image ci-contre :

3	2	2	1
0	3	0	3
3	2	0	0
0	1	1	1



Seuillage

Dans certains cas, l'histogramme peut être un outil efficace pour segmenter une image en deux classes, c'est-à-dire pour distinguer les objets de l'image suivant leur luminosité. En effet, lorsque l'histogramme présente deux modes distincts, on peut définir un seuil S entre ces deux modes, et appliquer ensuite un seuillage sur les pixels de l'image :

1. si le pixel a une valeur inférieure au seuil, le pixel est dans la classe 0
2. si le pixel a une valeur supérieure au seuil, le pixel est dans la classe 1

Le seuillage permet d'obtenir une image binaire qui ne contient que deux valeurs.

Opérations arithmétiques

Des opérations mathématiques simples peuvent être effectuées sur les éléments de deux images.

Addition d'images

L'addition de deux images f et g de même taille est une nouvelle image h de même taille dont chaque pixel correspond à la somme des valeurs des pixels des images originales :

$$\forall n, m \quad h(n, m) = f(n, m) + g(n, m).$$

Soustraction d'images

La soustraction de deux images est utilisée par exemple pour détecter des changements

$$\forall n, m \quad h(n, m) = f(n, m) - g(n, m).$$

$$\text{Ou} \quad \forall n, m \quad h(n, m) = |f(n, m) - g(n, m)|.$$

Division d'images

La division de deux images permet notamment de corriger une illumination non homogène (suppression de l'ombre . . .)

$$\forall n, m \quad h(n, m) = \frac{f(n, m)}{g(n, m)}.$$

Analyse d'images

L'analyse d'images recouvre un très grand nombre d'applications potentielles, c'est sans doute le domaine pour lequel les exemples sont les plus variés (analyse médicale : classification des chromosomes ; lecture automatique : authentification de signatures ; la télédétection : segmentation et classification de zones géographiques . . .)

Détection d'objets et extraction (segmentation) : Objets d'interêt

Reconnaissance de formes : Classification et identification d'objets

Interprétation et analyse de la scène : La scène pourra, entre autres, être décrite de façon quantitative et/ou qualitative

Transformée de Fourier

La transformée de Fourier bidimensionnelle est la version étendue de la transformée de Fourier classique aux images.

On rappelle que la transformée de Fourier permet de décomposer un signal en une somme de sinusoïdes, permettant ainsi de mettre en évidence les fréquences contenues dans un signal.

La transformée de Fourier discrète (TFD) d'une image f de taille $N \times M$ est une image F de taille $N \times M$ à valeurs complexes :

$$F(u,v) = \sum_{x=0}^{N-1} \sum_{y=0}^{M-1} f(n,m) e^{-\mathrm{j}2\pi \left(\frac{un}{N} + \frac{vm}{M} \right)}$$

Comme la TF est généralement complexe, elle ne peut être représentée directement : on représente séparément son module et sa phase.

Dans la TF d'une image, les fréquences basses se situent au centre de l'image, les fréquences hautes sur le pourtour.

Transformée de Fourier

La transformée de Fourier discrète inverse permet de calculer l'image originale à partir d'une transformée de Fourier :

$$f(n,m) = \frac{1}{NM} \sum_{u=0}^{N-1} \sum_{v=0}^{M-1} F(u,v) e^{+\mathrm{j}2\pi \left(\frac{un}{N} + \frac{vm}{M}\right)}$$

Propriétés

La TFD est périodique de périodes N et M :

$$F(u,v) = F(u + k_n N, v) = F(u, v + k_m M) = F(u + k_n N, v + k_m M)$$

où $k_n,\ k_m \in \mathbb{Z}$

Traitement d'images - Filtrage d'images

Le filtrage est une opération qui consiste à appliquer une transformation à l'image ou à une partie de l'image.
Il existe deux types de filtrage : le filtrage spatial et le filtrage fréquentiel.
On peut assimiler un filtre à un opérateur qui réaliserait une opération sur un pixel en fonction de la valeur de ce pixel et des pixels de son voisinage.

Filtrage par convolution

En filtrage d'images, le plus classique est d'appliquer sur un pixel un opérateur qui dépend du voisinage de ce pixel. La valeur du pixel sera modifiée en fonction de la valeur de ses voisins immédiats. Il s'agit de filtrage par convolution, car il utilise le produit de convolution.
Cette technique de filtrage consiste à considérer notre image comme une matrice de pixels que l'on va convoluer avec une autre matrice, plus petite, qui est le masque de convolution. C'est cette matrice qui va décider de la nature du filtre.
En signal, cette matrice correspond à la réponse impulsionnelle du filtre.

$$g(x,y) = (f \otimes h)(x,y) = \sum_n \sum_m f(x-n, y-m)h(n,m)$$

Filtrage par convolution (suite)

h est une image qui porte plusieurs noms, suivant le contexte : filtre (filter), masque (mask), noyau (kernel), fenêtre (window), motif (pattern) ou fonction d'étalement (point spread function : PSF).

Problèmes aux bords

La formule du produit de convolution n'est pas définie sur les bords de l'image : ainsi, le calcul de $g(1,1)$ nécessite de connaître, par exemple, la valeur de $f(0,0)$ qui n'existe pas.
Il existe donc plusieurs manières de fixer les valeurs des pixels situés en dehors de l'image (complétion avec des zéros, périodisation, reproduire le bord, miroir).
En tous les cas, il n'existe pas de manière parfaite pour fixer les valeurs des pixels situés en dehors de l'image : toutes introduisent des erreurs.
Aussi, le mieux est de s'arranger pour que les objets d'intérêt soient loin du bord.

Filtre passe-haut :

Un filtre passe-haut est un filtre qui accentue les fréquences hautes par rapport aux fréquences basses. Dans une image, cela se traduit par l'accentuation des détails et des contours.

Filtre passe-bas :

Un filtre passe-bas est un filtre qui accentue les fréquences basses par rapport aux fréquences hautes. Dans une image, cela se traduit par l'adoucissement des détails et la réduction du bruit.

Traitement d'images - Filtrage d'images

Filtre médian : Lissage en préservant les contours

Filtre moyenneur (passe-bas) : Remplace chaque pixel par la moyenne des pixels adjacents et du pixel central

Filtre directionnel : Rehausse les caractéristiques suivant une direction donnée

Filtre Laplacien (passe-haut) : Met en valeur les détails qui ont une variation rapide de luminosité. Détecteur de contour, reconnaissance de formes

Traitement d'images - Détection de contours

La reconnaissance de formes dans une image est une composante importante de l'analyse d'images. Elle se décompose en plusieurs étapes qui consistent à extraire les contours des objets dans l'image afin de les reconnaitre ou d'en détecter le mouvement. La première de ces étapes est la mise en évidence des contours des objets dans l'image.
Un contour définit la limite d'un objet dans une image. Cette limite est caractérisée par un changement dans l'image : un changement de couleur ou de contraste. Ce changement se traduit dans la valeur des pixels qui sont localisés de part et d'autre de la limite.
On utilise habituellement des outils comme le $\mathcal{G}$radient et le $\mathcal{L}$aplacien.

Traitement d'images - Détection de contours

Soit un pixel $p(i, j)$ dans une image couleur.

- Ce pixel est-il semblable, de même couleur, que ses voisins ?
- Si non, quelle est la différence de couleur entre lui et ses voisins ?
- Est-elle grande, ce qui signifierait qu'il est situé à la limite d'un objet ?
- Que signifie une "grande" ou une "petite" différence ?
- Comment la mesurer pratiquement ?

Ce problème a fait l'objet de très nombreuses recherches. Il existe des algorithmes très efficaces et compliqués pour résoudre ce problème, surtout s'agissant d'images en couleurs. Mais il existe aussi des moyens simples, pas très performants mais utiles pour comprendre.

Traitement d'images sous Python

Taille d'une image

```
import imageio
import matplotlib.pyplot as plt
%matplotlib inline

if __name__ == '__main__':
    pic = imageio.imread('lenna.jpg')
    plt.figure(figsize = (15,15))
    plt.imshow(pic)

print('Type de l\'image : ' , type(pic))
print()
print('Taille de l\'image : {}'.format(pic.shape))
print('Hauteur de l\'image : {}'.format(pic.shape[0]))
print('Largeur de l\'image : {}'.format(pic.shape[1]))
print('Dimension de l\'image : {}'.format(pic.ndim))
print("")
print('Taille globale de l\'image : {}'.format(pic.size),"=
    225x225x3")
print('Valeur RGB max dans cette image : {}'.format(pic.max
    ()))
print('Valeur RGB min dans cette image {}'.format(pic.min())
    )
```

Traitement d'images sous Python

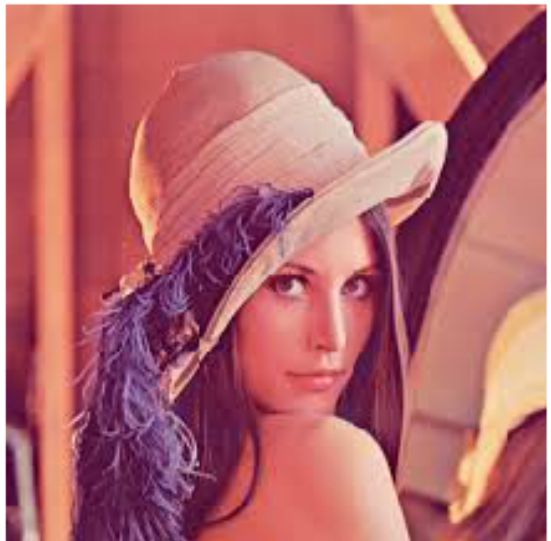
Dans une image couleur au format RGB, chaque pixel est codé par un 3-tuple (R,G,B) qui indique le poids, compris entre 0 et 255, de chaque canal de couleur : Red, Green et Blue.

Importation d'une image :

Librairie PIL

```
import sys
from PIL import Image
# ouverture du fichier image
ImageFile = 'Lenna.jpg'
try:
    img = Image.open(ImageFile)
except IOError:
    print('Erreur sur ouverture du
    fichier' + ImageFile)
    sys.exit(1)
# caractéristiques de l'image
print(img.format,img.size,img.mode)
# affichage de l'image
img.show()
# fermeture du fichier image
img.close()
```

JPEG (225, 225) RGB



Traitement d'images sous Python

Librairie MatPlotLib et ImageIO

```
from matplotlib.pyplot import imread
im1 = imread('Lenna.jpg')
print(im1.shape)
plt.imshow(im1)
plt.axis('off')
plt.show()
```

(225, 225, 3)

```
import imageio
im = imageio.imread('Lenna.jpg')
im.shape
```

Traitement d'images sous Python

Manipulation d'images

```
import sys ; from PIL import Image
ImageFile = 'Lenna.jpg'
img = Image.open(ImageFile)
print(img.format,img.size, img.mode)
img.show() ; colonne,ligne = img.size
imgF = Image.new(img.mode,img.size)
imgS = Image.new(img.mode,img.size)
for i in range(ligne):
    for j in range(colonne):
        pixel = img.getpixel((j,i)) # récupération du pixel
        # Complement à Max pour chaque pixel: effet négatif
        p = (255 - pixel[0], 255 - pixel[1], 255 - pixel[2])
        # composition de la nouvelle image
        imgF.putpixel((j,i), p)
for i in range(ligne):
    for j in range(colonne):
        pixel = img.getpixel((j,i))
        imgS.putpixel((colonne-j-1,i), pixel)
imgR = img.rotate(90)
imgF.show() ; imgR.show() ; imgS.show()
img.close() ; imgF.close()
imgS.close() ; imgR.close()
```

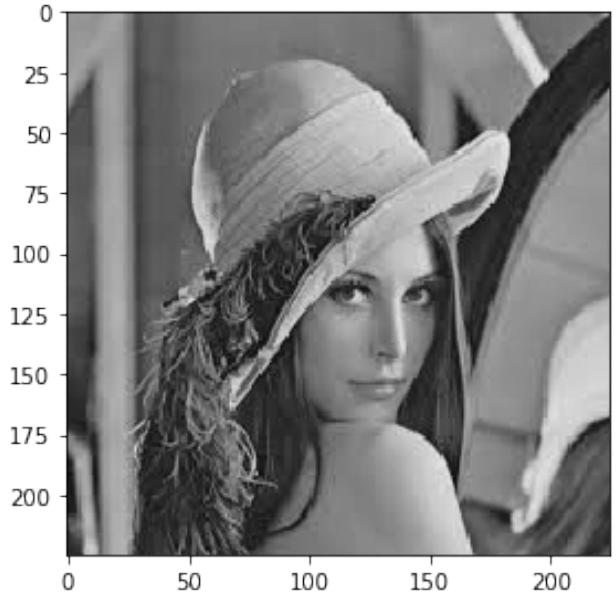
Rehaussement d'images

```
from PIL import Image, ImageFilter
#Lire l'image
im = Image.open( 'Lenna.jpg' )
#Affiche l'image
im.show()
from PIL import ImageEnhance
enh = ImageEnhance.Contrast(im)
enh.enhance(1.8).show("30% more
    contrast")
im.close()
```

Traitement d'images sous Python

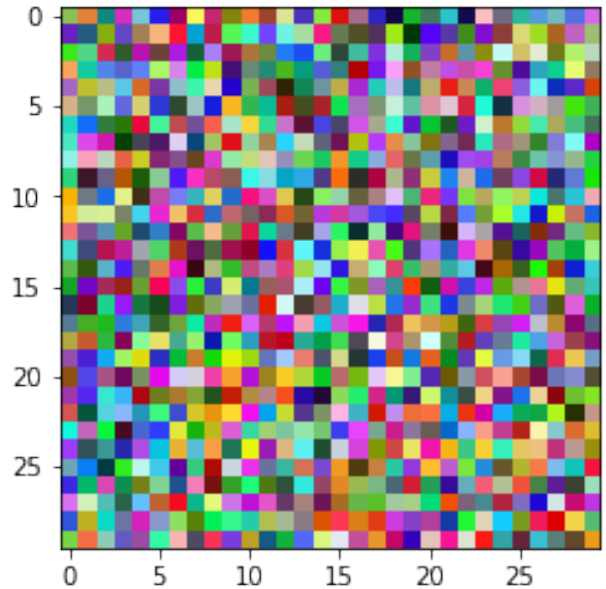
Image en niveaux de gris

```
from skimage import io
lena_gray = io.imread('Lenna.jpg',
    as_gray=True)
lena_gray.shape
io.imshow(lena_gray)
```



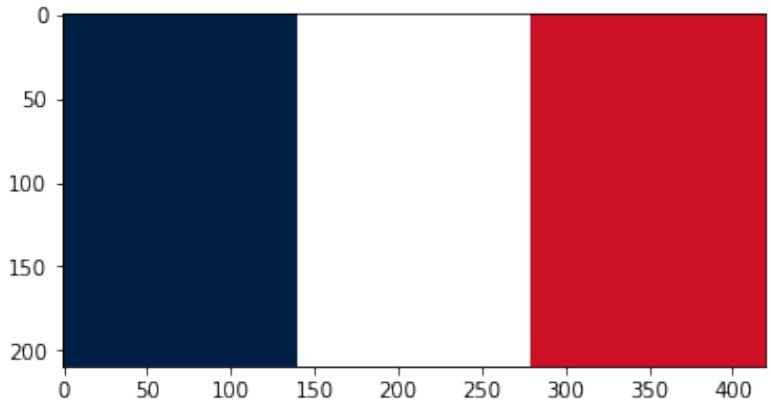
Générer une image de couleurs aléatoires

```
import matplotlib.pyplot as plt
%matplotlib inline
randim_image = np.random.rand
    (30,30,3)
plt.imshow(randim_image,
    interpolation="nearest")
```



Générer le drapeau tricolore

```
import numpy as np
import matplotlib.image as mimg
import matplotlib.pyplot as plt
# codage des couleurs
rouge = [0.807, 0.066, 0.149]
blanc = [1, 1, 1]
bleu =  [0, 0.129, 0.278]
h = 210  # Hauteur
la = 2*h # Largeur
# Image de départ, totalement noire
image = np.zeros((h, la, 3))
# Remplissage
image[:, 0:la//3 ] = bleu
image[:, la//3:2*la//3 ] = blanc
image[:, 2*la//3:la ] = rouge
plt.imshow(image)
plt.show()
```



Traitement d'images sous Python

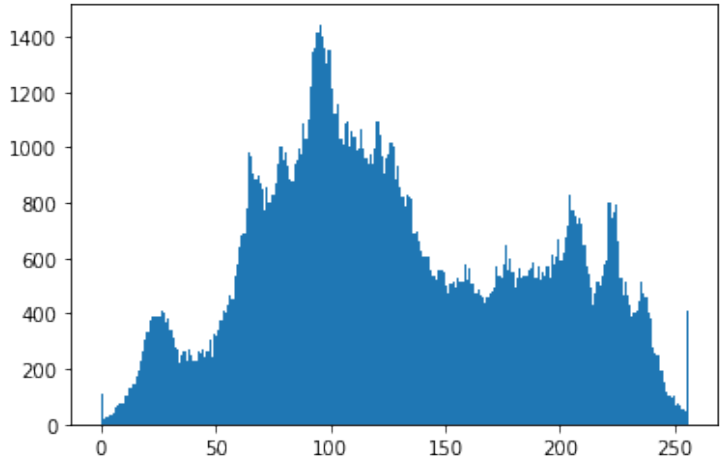
Extraire les différents plans couleurs

```
from matplotlib.pyplot import imread
im2 = imread('Lenna.jpg')
print(im1.shape)
imr=im2[:,:,0];      # Plan Rouge
imv=im2[:,:,1];      # Plan Vert
imb=im2[:,:,2];      # Plan Bleu
plt.figure(figsize=(8,6))
plt.subplot(221) ; plt.imshow(im2)
plt.title('Image originale')
plt.axis('off')
plt.subplot(222) ; plt.imshow(imr)
plt.title('Image Plan Rouge')
plt.axis('off')
plt.subplot(223) ; plt.imshow(imv)
plt.title('Image Plan Vert')
plt.axis('off')
plt.subplot(224) ; plt.imshow(imb)
plt.title('Image Plan Bleu')
plt.axis('off')
plt.tight_layout()
plt.show()
```

Traitement d'images sous Python

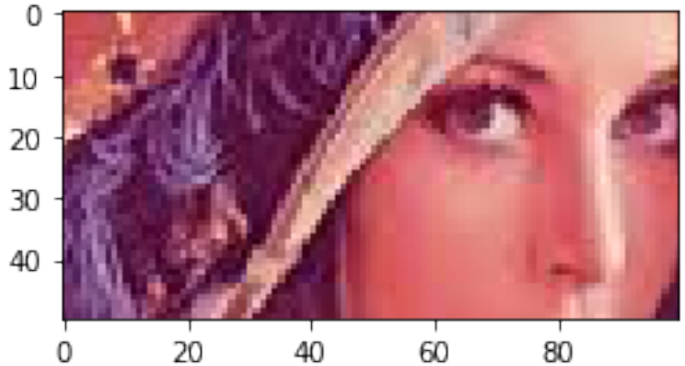
Histogramme d'une image

```
import numpy as np
from matplotlib import pyplot as plt
from skimage import io
img = io.imread('Lenna.jpg')
plt.hist(img.ravel(),256,[0,256])
plt.show()
```



Extraction d'une partie de l'image

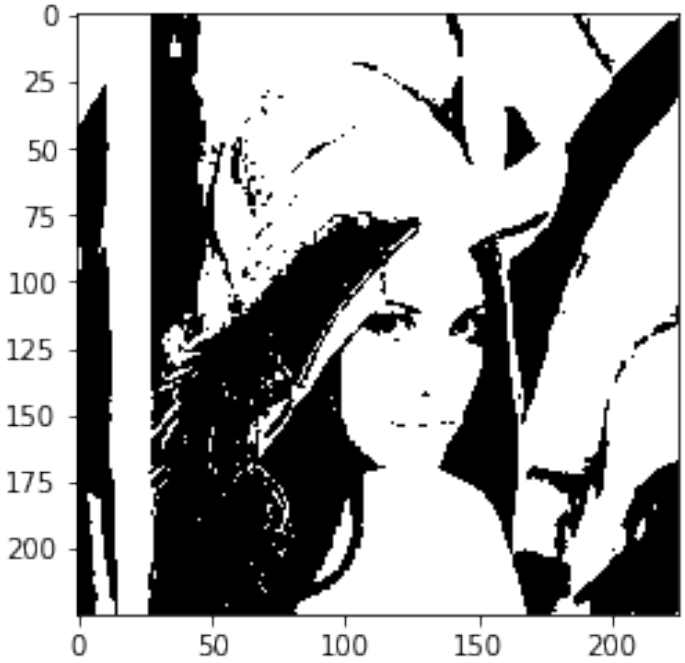
```
import imageio
import numpy
from matplotlib.pyplot import *
img = imread('Lenna.jpg')
partie = img[100:150,50:150]
figure(figsize=(4,4))
imshow(partie,cmap='gray')
```



Traitement d'images sous Python

Seuillage d'une image

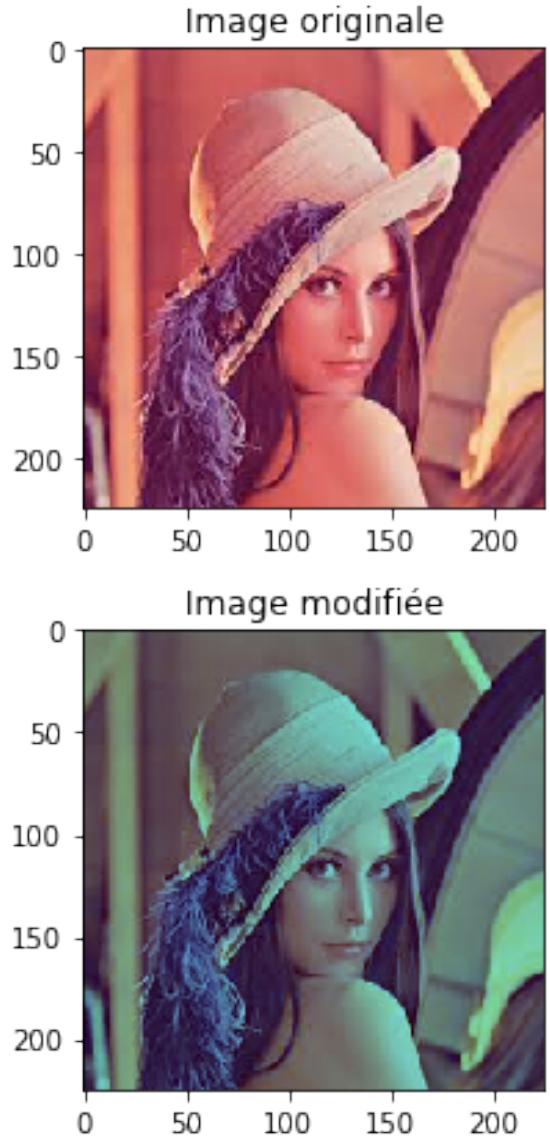
```
import imageio
import numpy
from matplotlib.pyplot import *
img = imread('Lenna.jpg')
def seuillage(image,seuil):
    resultat = image.copy()
    s = image.shape
    for j in range(s[0]):
        for i in range(s[1]):
            if image[j,i] > seuil:
                resultat[j,i] = 255
            else:
                resultat[j,i] = 0
    return resultat

im4 = seuillage(img[:,:,0],180)
figure(figsize=(4,4))
imshow(im4,cmap='gray',vmin=0,vmax
    =256)
```



Traitement d'images sous Python

Modification d'une image

```
import imageio
import numpy
from matplotlib.pyplot import *
img = imread('Lenna.jpg')
s = img.shape
couleur = numpy.zeros((s[0],s[1],3),
    dtype=numpy.int)
couleur[:,:,0] = img[:,:,0]*0.5
couleur[:,:,1] = img[:,:,1]
couleur[:,:,2] = img[:,:,2]
figure(figsize=(6,6))
subplot(211)
imshow(img)
title('Image originale')
subplot(212)
imshow(couleur)
title('Image modifiée')
tight_layout()
show()
```



Traitement d'images sous Python

Gradient d'une image

```
import matplotlib as mpl
import matplotlib.pyplot as plt
import numpy ; import scipy
import scipy.signal ; import scipy.misc
lm = scipy.misc.ascent()
# définition des filtres
fx = numpy.array([[-1, 0, 1]])
fy = numpy.transpose(fx)
# convolutions
lm_convol_x = scipy.signal.convolve2d(
    lm, fx, mode = 'same')
lm_convol_y = scipy.signal.convolve2d(
    lm, fy, mode = 'same')
lm_gradient = numpy.sqrt(lm_convol_x**2
     + lm_convol_y**2)
fig = plt.figure()
ax1 = fig.add_subplot(2, 1, 1)
ax1.imshow(lm, cmap = mpl.cm.gray)
ax2 = fig.add_subplot(2, 1, 2)
ax2.imshow(lm_gradient, cmap = mpl.cm.
    gray)
plt.show()
```