

# Manipulation et affichage d'images

# Librairie PIL (Pillow)

```
from PIL import Image
```

```
im=Image.open("mon_image.jpg")  
im.show()
```



# Librairie PIL (Pillow)

Pour obtenir des informations sur le format

```
from PIL import Image

im=Image.open("mon_image.jpg")
print(im)
print(im.format,im.mode,im.size,im.palette)
print(im.info)
im.save("sortie.png", "PNG")
```

↑  
enregistrement d'une nouvelle image  
sur le disque

# Lecture informations couleurs

```
import numpy as np

im=Image.open("mon_image.jpg")
v=np.array(im)
print(v.shape)
print(v)
```

Nx Ny 3



# Modification d'une image

```
from PIL import Image
import numpy as np

im=Image.open("mon_image.jpg")
v=np.array(im) ← données des pixels
v/=4.0 ← modification
im=Image.fromarray(v) ← création d'une
im.show() image modifiée
```

# Modification d'une image

```
from PIL import Image
import numpy as np

im=Image.open("mon_image.jpg")
v=np.array(im)

r=v[:, :, 0]
g=v[:, :, 1]
b=v[:, :, 2]
g/=1.3
b/=1.5

v2=np.zeros((150, 200, 3), dtype='uint8')
v2[:, :, 0]=r
v2[:, :, 1]=g
v2[:, :, 2]=b

im=Image.fromarray(v2)
im.show()
```

*Canaux R,G,B*

*Reconstruction de l'image*



# Contours d'une image

```
from PIL import Image
import numpy as np

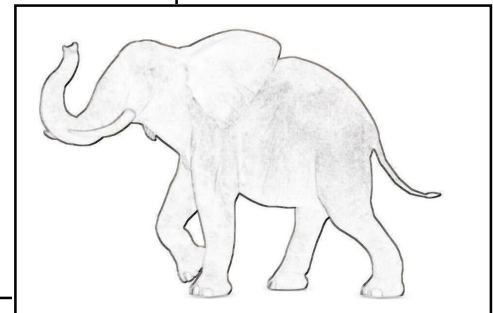
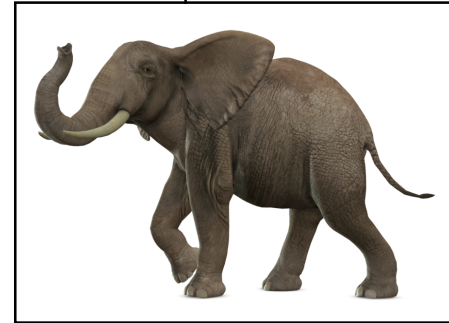
im=Image.open("elephant.png")
v=np.array(im,dtype='int')

dx=np.zeros_like(v,dtype='int')
dy=np.zeros_like(v,dtype='int')
Nx,Ny,dim=v.shape

for kx in range(Nx-1):
    for ky in range(Ny-1):
        dx[kx,ky,:]=abs(v[kx+1,ky,:]-v[kx,ky,:])
        dy[kx,ky,:]=abs(v[kx,ky+1,:]-v[kx,ky,:])

dx=np.array(dx,dtype='uint8')
dy=np.array(dy,dtype='uint8')

Image.fromarray(dx).show()
Image.fromarray(dy).show()
Image.fromarray(dx+dy).show()
```



# Contours d'une image

```
from PIL import Image
import numpy as np

im=Image.open("elephant.png")
v=np.array(im,dtype='int')

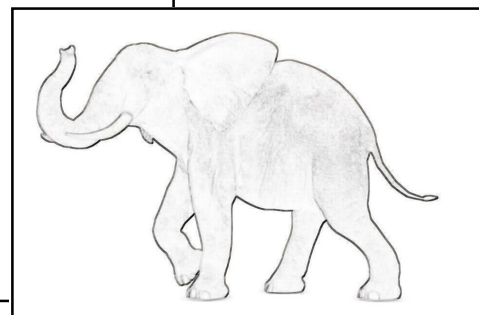
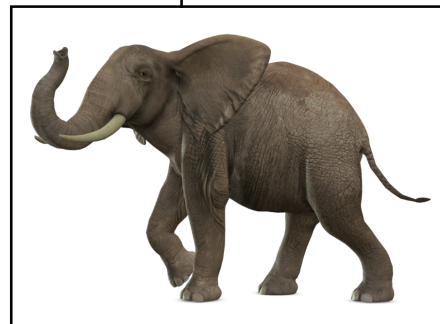
dx=np.zeros_like(v,dtype='int')
dy=np.zeros_like(v,dtype='int')
Nx,Ny,dim=v.shape
```

*version vectorielle*

```
dx[1:,:,:]=abs(v[1:,:,:]-v[:-1,:,:])
dy[:,1:,:]=abs(v[:,1:,:]-v[:,::-1,:])
```

```
dx=np.array(dx,dtype='uint8')
dy=np.array(dy,dtype='uint8')
```

```
Image.fromarray(dx).show()
Image.fromarray(dy).show()
Image.fromarray(dx+dy).show()
```





# Application: conversion

Convertir une image couleur en noir et blanc.

# Application: segmentation

Afficher en rouge clair tous pixel de niveau de gris supérieur à 150

# Application: histogramme

Construire l'histogramme des couleurs rouges vertes et bleue d'une image

