Chapitre 7: Traitement d'images

I- Qu'est-ce qu'une image ? 27ANJ7

Une photo est associée à tableau à 2 dimensions (image en niveau noir et blanc) ou 3 dimensions (image en couleurs avec une composante pour le rouge, le vert et le bleu) dont les valeurs positives sont souvent converties sur 8 bits : *uint*8.

Les valeurs rencontrées sont donc comprises entre 0 (noir) et 255 (intensité maximale). La manipulation des tableaux nécessite une réflexion afin d'éviter les problèmes d'overflow suivant car les valeurs sont 256 périodiques.

-256	...	....	-1	0	1...	....	...	255	256	...	...	511		
0			255	0				255	0			255		

II- Listes et tableaux numpy 6KBQEM

a) Comparaisons

Un tableau numpy ne suit pas les mêmes règles de calcul qu'une liste :

	Liste	Tableau numpy

Création	#creation d'une liste
	L1=[0,1,2,3]
	#creation d'un tableau en liste
	L2=T1.tolist()#
	
	#creation d'une liste de liste
	L2=[[0,1,2,3],[4,5,6,7]]
		import numpy as np
		#creation d'un tableau d'une ligne (équivalent à un vecteur 1D)
		T1=np.array([0,1,2,3])
		#Conversion d'une liste en tableau :
		T2=np.array(L1)
		On peut utiliser un 2e argument optionnel pour fixer ou convertir le type de données (par défaut int64)
		T1=np.array(T1,dtype ="uint8")
		#creation manuelle d'un tableau 2D :
		T3=np.array([[0,1,2,3],
		[4,5,6,7]])

Effet d'une multiplication	L3=2*L1 #L3 est une concaténation de L1 avec L1	T4=2*T1 #Chaque élément de T1 est multiplié par 2 (on parle de vectorisation)

b) Opérations « de bases » sur les tableaux numpy

Lignes de commandes	Interpréteur
"""deux fonctions intéressantes pour créer des tableaux 1D en ligne""" T5=np.linspace(0,4,5)#(départ,fin inclue, nbre de points) T6=np.arange(0,5,1)#(départ,fin exclue, incrément)	[0. 1. 2. 3. 4.] [0 1 2 3 4]
"""initialisation : création d'un tableau de zéros""" T7=np.zeros((2,3))#((nbre de lignes, nbre de colonnes))	[[0. 0. 0.] [0. 0. 0.]]
"""somme de tous les éléments d'un tableau""" s=np.sum(T1)	6

"""obtenir les dimensions d'un tableau"""
lignes,colonnes=np.shape((T7)) #renvoie un tuple(nbre lignes, nbre colonnes)
lignes= np.shape((T7))[0]
	(2,3)
	2

```
"""accès à une valeur ou création d'un sous tableau"""
T8=np.array([[0,1,2,3],[4,5,6,7],[8,9,10,11]])
print(T8[1])#renvoie la ligne d'index 1 (2e ligne)
print(T8[-1])#renvoie la dernière ligne
print(T8[1,2])#renvoie le motif en 2e ligne et 3e colonne
print(T8[1,:])#renvoie la ligne d'index 1
print(T8[1,:-1])#renvoie la ligne d'index 1 sans le dernier élément
print(T8[:,1])#renvoi tout le colonne d'index 1
print(T8[1:3,1:3])#[index 1e ligne:index de la dernière ligne exclue,index 1e colonne:index de la dernière colonne exclue]
print(T8[-1::-1,-1::-1])#début,fin, pas =-1 pour fixer le sens inverse

print(T8[[True,False,False],:])#que le ligne
```

```
[4 5 6 7]
[8 9 10 11]
6

[4 5 6 7]
[4 5 6]

[1 5 9]
[[ 5  6]
 [ 9 10]]

[[11 10  9  8]
 [ 7  6  5  4]
 [ 3  2  1  0]]
[[0 1 2 3]]
```

c) Obtenir le négatif d'une photo avec une liste et avec un tableau

```
from PIL import Image
import numpy as np
import time
im=Image.open("tours.JPG")#ouverture du tableau
tab=np.array(im,dtype="uint8")#conversion
l,c,p=np.shape(tab)#dimension l : lignes, colonnes, p=1 ou 3
```

liste	Tableau

liste=tab.tolist()#conversion du tableau en liste
t0=time.time()#initialisation de la mesure du temps
for i in range(l):
for j in range(c):
for k in range(p):
liste[i][j][k]=255-liste[i][j][k]#calcul du négatif
print(time.time()-t0)#temps d'exécution
tab_new=np.array(liste,dtype="uint8")
photo new=Image.fromarray(tab new)
photo_new.show()
	tab_new=np.copy(tab)
	t0=time.time()
	tab_new=255-tab
	print(time.time()-t0)
	photo_new=Image.fromarray(tab_new)
	photo_new.show()

La vectorisation des tableaux numpy permet un gain de temps non négligeable dans la manipulation de photos

Commenté [AM4]: Autrement appelé découpage ou slicing

Commenté [AM5]: On pourra noter une différence dans les notations des tranches :
Liste[sélections lignes][sélection de la 1e sélection]
Tab[sélections lignes][sélection de la 1e sélection]
ou Tab[index_lignes][index_colonne]
Exemple :
L=[[0,1,2,3],
[4,5,6,7]]
tab=np.array(L)
print(tab[:][-1])#[4 5 6 7]
print(tab[:,-1])#[3 7]

Commenté [AM1]: Exemples de dépassement :
255 + 1 = 0
0 – 1 = 255

Commenté [AM2]: Les séquences ou structures de données rencontrées depuis le 1e semestre sont :
-Les chaînes et les tuples (qui sont immuables : les valeurs stockées dans ces séquences ne peuvent être modifiées par affectation après création- pour les chaînes une modification par la méthode replace est possible mais conduit à la création d'une nouvelle chaîne avec une autre adresse mémoire)
-Les listes (muables, on dit aussi variables)

Commenté [AM3]: Valeur max de 2^63-1 soit 10^19