Prácticas de Visión Artificial

OpenCV - Python

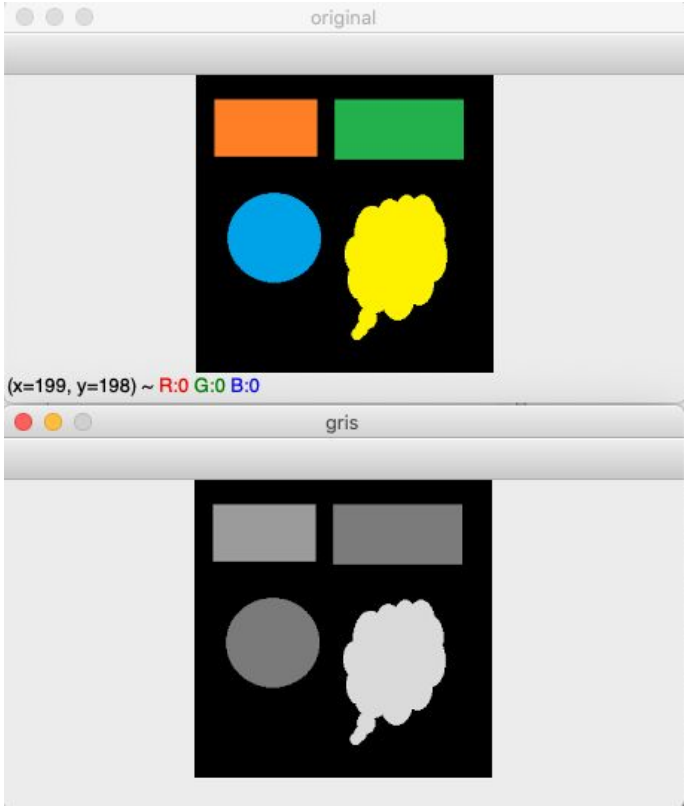
Dónde encontrar más información y ejemplos

- Tutoriales de OpenCV con Python:
  - https://docs.opencv.org/3.4/d6/d00/tutorial_py_root.html
- Tutoriales de Matplotlib:
  - https://matplotlib.org/2.0.2/index.html
- Tutoriales de Numpy:
  - https://docs.scipy.org/doc/numpy/user/quickstart.html

Ejercicio 1: abrir, mostrar, convertir y salvar imágenes

```
1  import cv2
2
3  img = cv2.imread('formas.png',cv2.IMREAD_COLOR)
4  cv2.imshow('original',img)
5
6  gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
7  cv2.imshow('gris',gray)
8
9  k = cv2.waitKey(0) & 0xFF
10 if k == 27:         # ESC para salir
11     cv2.destroyAllWindows()
12 elif k == ord('s'): # s para salvar en JPG y salir
13     cv2.imwrite('formas.jpg',img)
14     cv2.imwrite('formas_gris.jpg',gray)
15     cv2.destroyAllWindows()
16
```

Ejercicio 1: abrir, mostrar, convertir y salvar imágenes

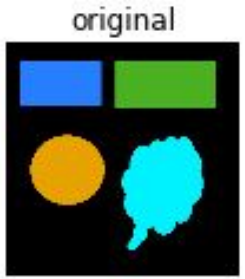


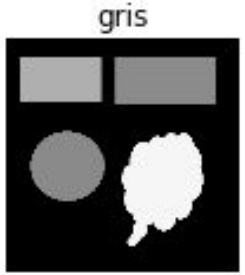
Ejercicio 2: emplear **matplotlib** para visualizar imágenes

```python
import cv2
from matplotlib import pyplot as plt

img = cv2.imread('formas.png',cv2.IMREAD_COLOR)

# quitar una vez comprobado el problema de formato BGR / RGB
cv2.imshow('original',img)

gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

plt.subplot(211), plt.title('original'), plt.axis("off")
plt.imshow(img)
# OpenCV usa BGR y matplotlib usa RGB
# plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))

plt.subplot(212), plt.title('gris'), plt.axis("off")
plt.imshow(gray, 'gray')

plt.show()
```

Ejercicio 2: emplear **matplotlib** para visualizar imágenes

```
In [1]: runfile('/Users/arroyo/Ej02.py', wdir='/Users/arroyo')
```



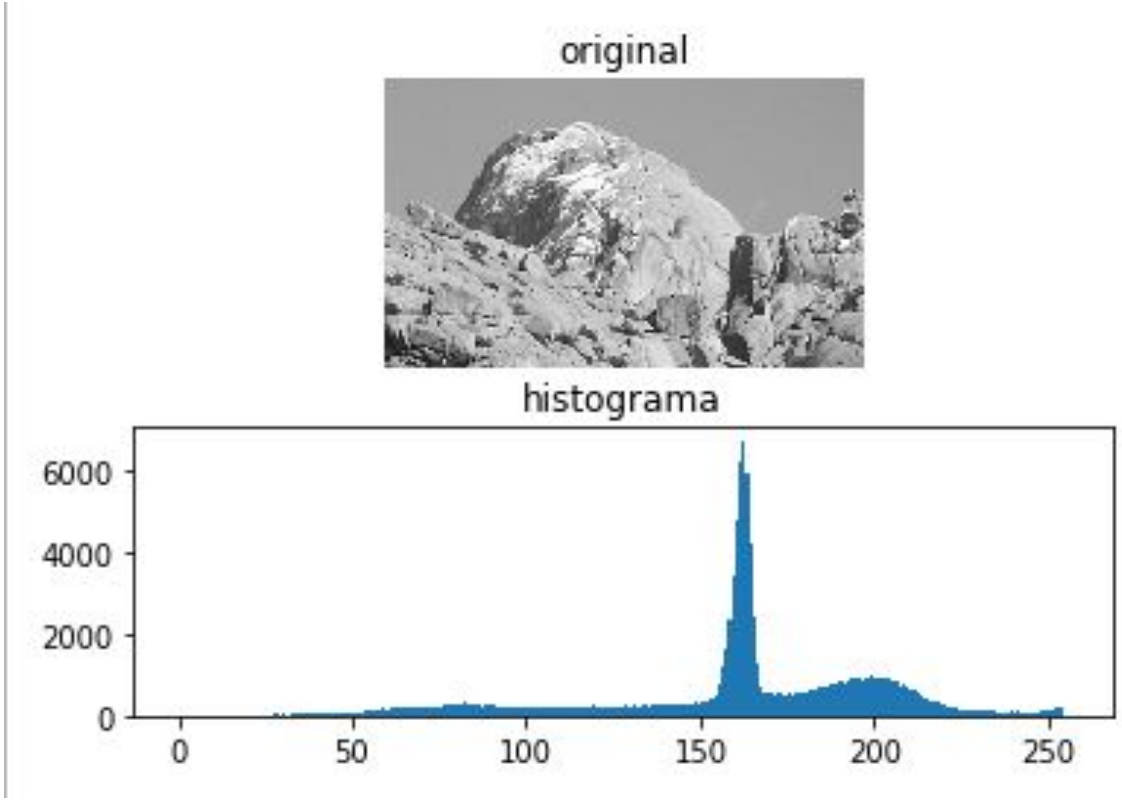



```
In [2]:
```

Ejercicio 3: histograma de una imagen

```
1  import cv2
2  from matplotlib import pyplot as plt
3
4  img = cv2.imread('yelmo.png', cv2.IMREAD_GRAYSCALE)
5
6  hist = cv2.calcHist([img],[0],None,[256],[0,256])
7
8  plt.subplot(211), plt.title('original'), plt.axis("off")
9  plt.imshow(img, 'gray')
10
11 plt.subplot(212), plt.title('histograma')
12 plt.hist(img.ravel(),256,[0,256])
13
14 plt.show()
15
```

Ejercicio 3: histograma de una imagen

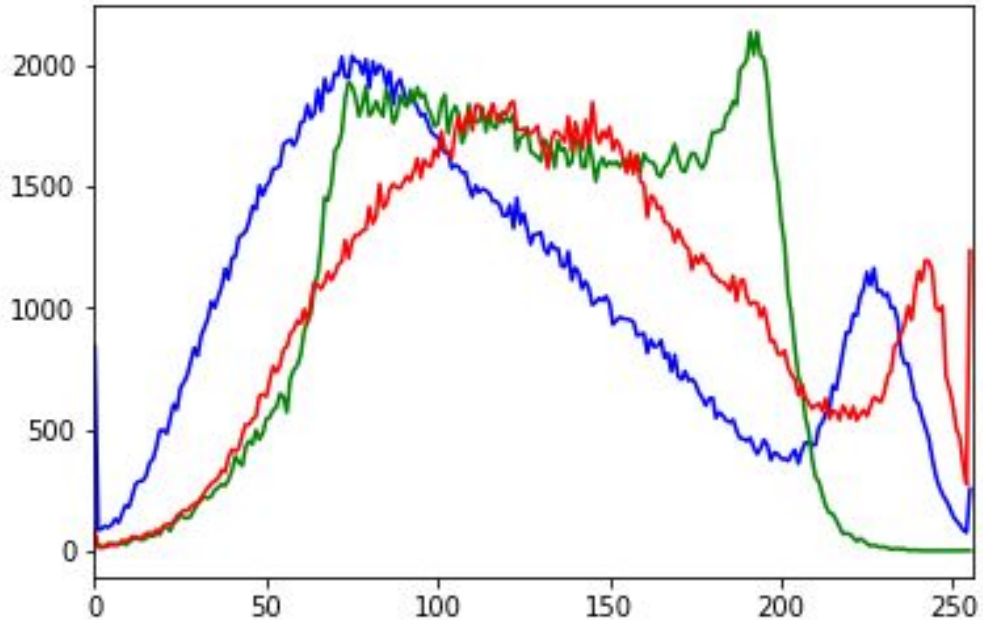


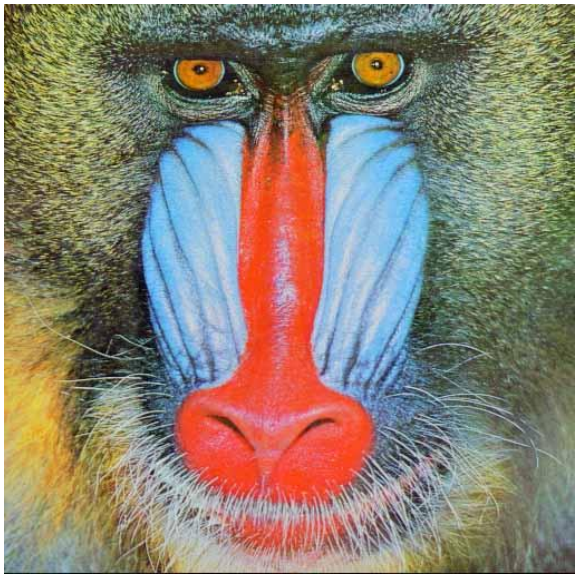
Ejercicio 4: histograma de una imagen en color

```
1 import cv2
2 from matplotlib import pyplot as plt
3
4 img = cv2.imread('baboon.jpg',cv2.IMREAD_COLOR)
5
6 color = ('b','g','r')
7
8 for i,col in enumerate(color):
9     histr = cv2.calcHist([img],[i],None,[256],[0,256])
10    plt.plot(histr,color = col)
11    plt.xlim([0,256])
12 plt.show()
13
```

Ejercicio 4: histograma de una imagen en color

```
In [1]: runfile('/Users/arroyo/Ej09.py', wdir='/Users/arroyo')
```





Ejercicio 5: ecualización del histograma de una imagen

```
1  import cv2
2  import numpy as np
3
4  img = cv2.imread('reloj.png',0)
5
6  # ecualizamos la imagen
7  equ = cv2.equalizeHist(img)
8
9  # concatenamos la imagen y su histograma
10 res = np.hstack((img,equ))
11
12 cv2.imshow('img',res)
13 cv2.waitKey()
14 cv2.destroyAllWindows()
15
16
```

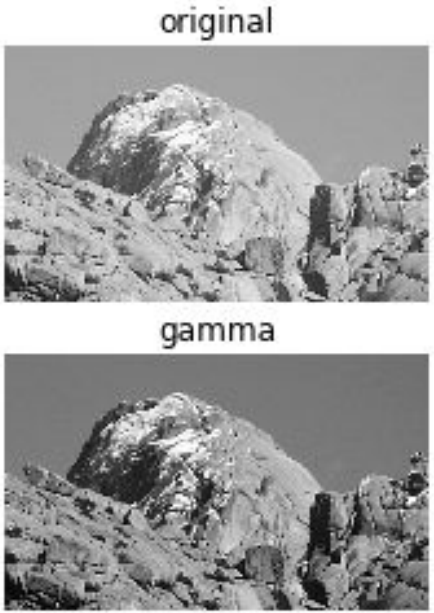
Ejercicio 5: ecualización del histograma de una imagen

Ejercicio 6: aplicar LUT (corrección gamma)

```python
import cv2
import numpy as np
from matplotlib import pyplot as plt

img = cv2.imread('yelmo.png', cv2.IMREAD_GRAYSCALE)

gamma = 0.6
invGamma = 1.0 / gamma

table = np.array([((i / 255.0) ** invGamma) * 255 for i in np.arange(0, 256)]).astype("uint8")

res = cv2.LUT(img, table)

plt.subplot(211), plt.title('original'), plt.axis("off")
plt.imshow(img, 'gray')

plt.subplot(212), plt.title('gamma'), plt.axis("off")
plt.imshow(res, 'gray')

plt.show()

```

Ejercicio 6: aplicar LUT (corrección gamma)

```
In [2]: runfile('/Users/arroyo/Ej10.py', wdir='/Users/arroyo')
```



Ejercicio 7: detectar caras en imágenes

```python
import cv2

face_cascade = cv2.CascadeClassifier('./opencv/data/haarcascades/haarcascade_frontalface_default.xml')
eye_cascade = cv2.CascadeClassifier('./opencv/data/haarcascades/haarcascade_eye.xml')

img = cv2.imread('24Lena.bmp')
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
faces = face_cascade.detectMultiScale(gray, 1.3, 5)
for (x,y,w,h) in faces:
    img = cv2.rectangle(img,(x,y),(x+w,y+h),(255,0,0),2)
    roi_gray = gray[y:y+h, x:x+w]
    roi_color = img[y:y+h, x:x+w]
    eyes = eye_cascade.detectMultiScale(roi_gray)
    for (ex,ey,ew,eh) in eyes:
        cv2.rectangle(roi_color,(ex,ey),(ex+ew,ey+eh),(0,255,0),2)

cv2.imshow('img',img)
cv2.waitKey()
cv2.destroyAllWindows()

```

Ejercicio 7: detectar caras en imágenes