

CARRERA: COMPUTACIÓN**ASIGNATURA:** SISTEMAS EMBEBIDOS**NRo. PRÁCTICA:** #1**TÍTULO PRÁCTICA:** Procesamiento de imágenes Numpy**OBJETIVOS:**

- Aprender a visualizar, generar, cargar y modificar imágenes en blanco y negro o color a través de las operaciones de matrices de la librería Numpy, y las facilidades de las librerías PyPlot y skimage.

ACTIVIDADES POR DESARROLLAR (Anotar las actividades que deberá seguir el estudiante para el cumplimiento de la práctica)

1. Ejercicio #1: Modificar la variable imagen_gradiente para que cada fila tenga una intensidad creciente, La fila 0 debe tener intensidad 0; La fila 1 debe tener intensidad 0.1, La fila 1 debe tener intensidad 0.2 , ... , La fila 9 debe tener intensidad 0.9, La fila 10 debe tener intensidad 1.

```
1 #Ejercicio para repasar (NO SE HARA EN EL TALLER)
2 # Modificar la variable imagen_gradiente para que cada fila tenga una int
3 #ensidad creciente
4 #Resolver Problema 1
5 # La fila 0 debe tener intensidad 0
6 # La fila 1 debe tener intensidad 0.1
7 # La fila 2 debe tener intensidad 0.2
8 # ...
9 # La fila 9 debe tener intensidad 0.9
10 # La fila 10 debe tener intensidad 1
11 A = np.ones((10,10))
12 B = np.linspace(0,1,10)
13 C = B*A
14 #IMPLEMENTAR
15 plt.imshow(C.transpose())
16
```

2. Ejercicio #2: Mensaje de error generado y corregirlo /data/dev/dpt/.env/lib/python3.5/site-packages/skimage/io/_io.py:132: Use rWarning: lena_generada.png is a low contrast image warn('%s is a low contrast image' % fname)
/data/dev/dpt/.env/lib/python3.5/site-packages/skimage/util/dtype.py:122: UserWarning: Possible precision loss when converting from float64 to uint 16 .format(dtypeobj_in, dtypeobj_out))

```
2 from skimage import img_as_ubyte
3 lena_gris = (lena_rgb[:, :, 0]+lena_rgb[:, :, 1]+lena_rgb[:, :, 2])/3
4 plt.imshow(lena_gris)
5 io.imsave("lena_generada.png",img_as_ubyte(lena_gris))
6
```

3. Ejercicio #3: Entre la imagen azul y verde hacer una suma de matrices para generar una imagen x

```
1 #Entre la imagen azul y verde hacer una suma de matrices para generar una imagen x
2 lena_blue=np.copy(lena_rgb)
3 lena_blue[:, :, 0]=0
4 lena_blue[:, :, 1]=0
5 plt.title("Lena_ canal azul")
6 plt.imshow(lena_blue)
7 plt.figure()
8 lena_green=np.copy(lena_rgb)
9 lena_green[:, :, 0]=0
10 lena_green[:, :, 2]=0
11 plt.title("Lena_ canal verde")
12 plt.imshow(lena_green)
13 plt.figure()
14 suma_blue_green = lena_blue+lena_green
15 plt.imshow(suma_blue_green)
16 plt.title("Lena_ canal Azul y Verde")
17 plt.figure()
```

4. Ejercicio #4: Convertir la imagen original en blanco y negro

```
1 #Convertir la imagen original en blanco y negro
2 from PIL import Image
3 image_file = Image.open("lena.png") # open colour image
4 image_file = image_file.convert('1') # convert image to black and white
5 image_file.save('lena_BN.png')
```


```
1 from PIL import Image
2 img = Image.open('lena.png')
3 thresh = 200
4 fn = lambda x : 255 if x > thresh else 0
5 r = img.convert('L').point(fn, mode='1')
6 r.save('lena_bn2.png')
```

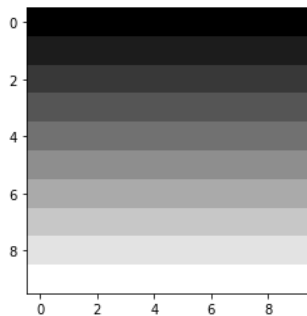
5. Ejercicio #5: Segmentación basada en histograma (sin información espacial)

```
1 from scipy import ndimage
2 import matplotlib.pyplot as plt
3 f = 10 #valor n
4 c = 256 #valor rgb
5 lena_img = np.copy(lena_rgb) #se toma la foto de lena.png
6 points = c*np.random.random((2,f**2))
7 lena_img[(points[0]).astype(np.int),(points[1].astype(np.int))] = 1
8 lena_img = ndimage.gaussian_filter(lena_img, sigma=c/(4.*f)) #metodo gaussiano
9 m = (lena_img > lena_img.mean()).astype(np.float)
10 m += 0.1 * lena_img
11 lena_histogram = m + 0.2*np.random.randn(*m.shape)
12 h, bedges = np.histogram(lena_img, bins=60)
13 bcenter = 0.5*(bedges[:-1]+bedges[1:])
14 bin_lena = lena_histogram > 0.5
15 plt.plot(bcenter, h, lw=2)
16 plt.axvline(0.5, color='r', ls='--', lw=2) #linea punteada del histograma
17 plt.text(0.57, 0.8, 'histogram', fontsize=20, transform = plt.gca().transAxes) #histogrma
18 plt.show()
19 plt.imshow(lena_histogram)
```

RESULTADO(S) OBTENIDO(S):

- Ejercicio #1

 <matplotlib.image.AxesImage at 0x7f27dc2c7080>

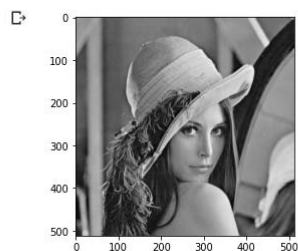


- Ejercicio #2

Subir Actualizar Activar Drive

..
sample_data
lena.png
lena_BN.png
lena_bn2.png
lena_generada.png
lena_generada2.png
lena_gray.png

```
[ ] 1 #Ejercicio: Convertir la imagen de lena
2 from skimage import img_as_ubyte
3 lena_gris = (lena_rgb[:, :, 0] + lena_rgb[:, :, 1] + lena_rgb[:, :, 2]) / 3
4 plt.imshow(lena_gris)
5 io.imsave("lena_generada.png", img_as_ubyte(lena_gris))
6
```

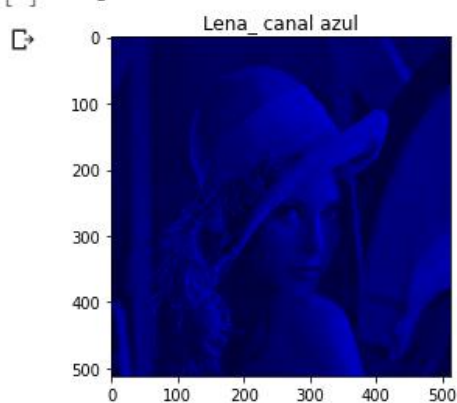


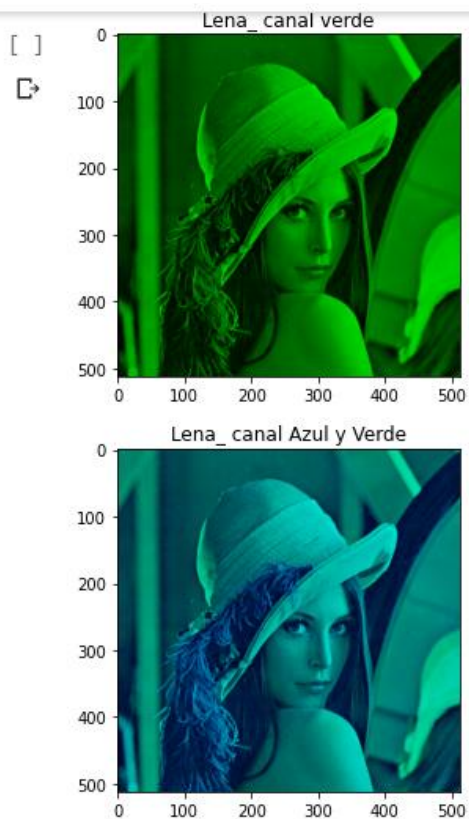
lena_generada.png X



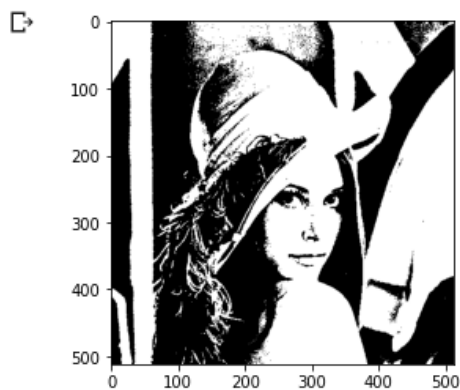
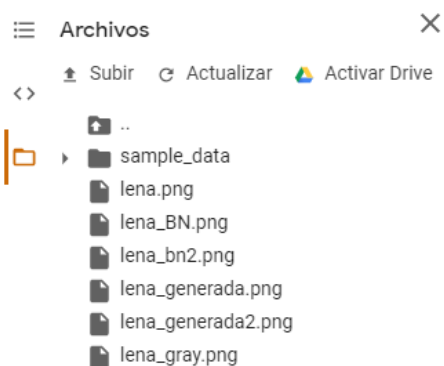
- Ejercicio #3

[] <Figure size 432x288 with 0 Axes>

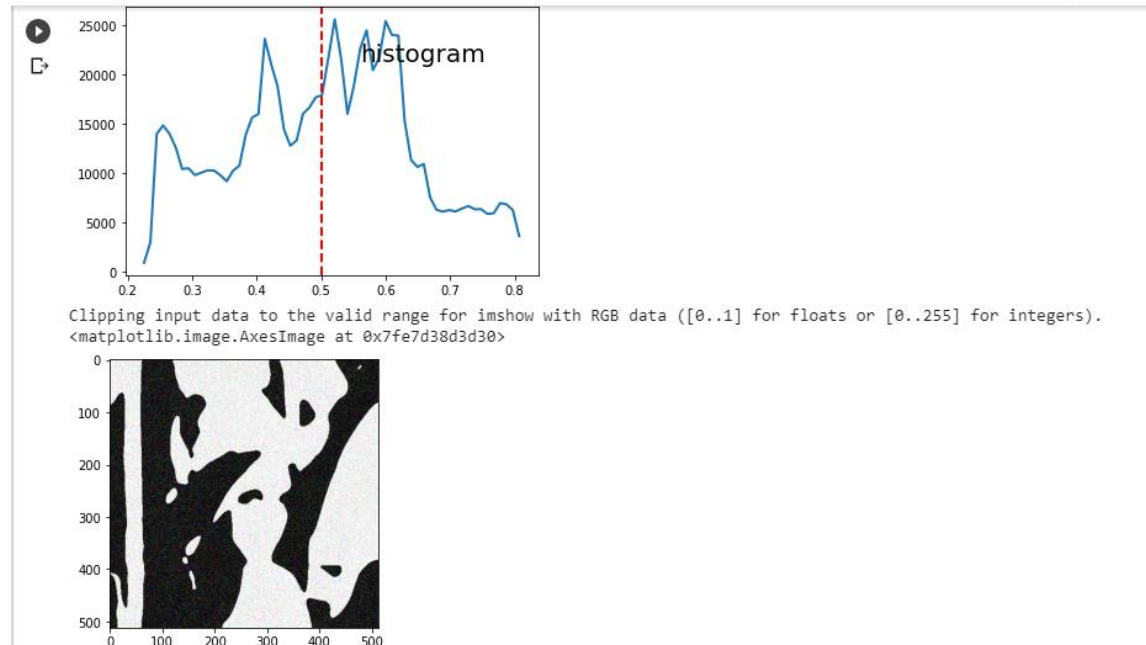




- Ejercicio #4



• Ejercicio #5



CONCLUSIONES:

- En el primer ejercicio el método **transpose()** permite que el grafico que antes era un degradado de intensidad de izquierda a derecha, ahora este de arriba hacia abajo por lo que tengo la habilidad de utilizar este nuevo método aprendido..
- En el segundo ejercicio la forma en que se resuelve el error es importando una funcionalidad de la librería de *skimage* en este caso **img_as_ubyte** o **img_as_uint** dependiendo del tipo de dato de la imagen, por lo que tengo la habilidad de diferenciar tipos de datos que poseen las imágenes.
- En el tercer ejercicio la suma de ambos rangos de colores de la imagen RGB tomando la imagen original en este caso "**lena.png**" nos da como resultado una combinación como de un color turquesa, por lo que tengo la habilidad de separar la imagen en un color específico para luego combinarlos.
- En el cuarto ejercicio podemos utilizar de la librería *PIL* la funcionalidad de **Image** para obtener una imagen en blanco y negro de la imagen original en RGB, tengo la habilidad de usar dos formas diferentes de conversión de RGB a blanco y negro de una imagen.
- En el quinto ejercicio usamos la librería *scipy* la funcionalidad **ndimage** para poder realizar el histograma se toma los valores rgb y un valor n como referencia, además de poner como referencia la imagen original "lena.png" y poder utilizar el método gaussiano para un cálculo interno, por lo que tengo la habilidad de aplicar algebra en la conversión de imagen rgb a un histograma.

RECOMENDACIONES:

- En el primer ejercicio se recomienda ingresar al siguiente link <https://www.it-swarm.dev/es/python/como-permuta-el-metodo-transpose-de-numpy-los-ejes-de-una-matriz/1054112945/> para una mayor profundización del métodos transpose().
- En el segundo ejercicio se recomienda ingresar al siguiente link para conocer los tipos de datos en las imágenes http://tonysyu.github.io/scikit-image/user_guide/data_types.html y evitar errores o warnings en los datos que contiene una imagen.
- En el tercer ejercicio se recomienda separar cada cuadro de color en varios bloques de código y finalmente hacer la suma para obtener un resultado de un color diferente al original, por lo que podre profundizar más al respecto para crear posiblemente mi propia paleta de colores en para una imagen.

-
- En el cuarto ejercicio se recomienda pasar la imagen rgb a escala gris y correspondiente a blanco y negro, también se puede utilizar otras librerías como **matplotlib.pyplot**, por lo que debo buscar más información de las librerías a utilizar.
 - En el quinto ejercicio se recomienda ingresar al siguiente link https://claudiovz.github.io/scipy-lecture-notes-ES/advanced/image_processing/index.html y https://claudiovz.github.io/scipy-lecture-notes-ES/advanced/image_processing/auto_examples/plot_histo_segmentation.html#example-plot-histo-segmentation-py para conocer la Librería y cálculos a realizar para obtener el histograma sin información espacial, ya que se necesita interpretar un poco más los cálculos realizados en el código.
-

Nombre de estudiante: Andrea Valentina Cárdenas García

Firma de estudiante: