Negatif.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

NEGATIF CITRA

Langkah-langkah transformasi negatif :

1. Baca gambar
2. Dapatkan tinggi dan lebar gambar
3. Setiap piksel berisi 3 saluran. Jadi, ambil nilai piksel dan kumpulkan 3 saluran dalam 3 variabel berbeda.
4. Negasikan nilai 3 piksel dari 255 dan simpan lagi dalam piksel yang digunakan sebelumnya.
5. Lakukan untuk semua nilai piksel yang ada dalam gambar.

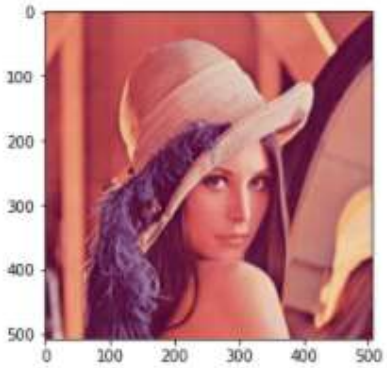
[11]
```
import cv2
import matplotlib.pyplot as plt
import numpy as np
from skimage import data, color
from skimage.io import imread, imshow

from google.colab import files
uploaded = files.upload()
```

Choose Files lenna.JPG

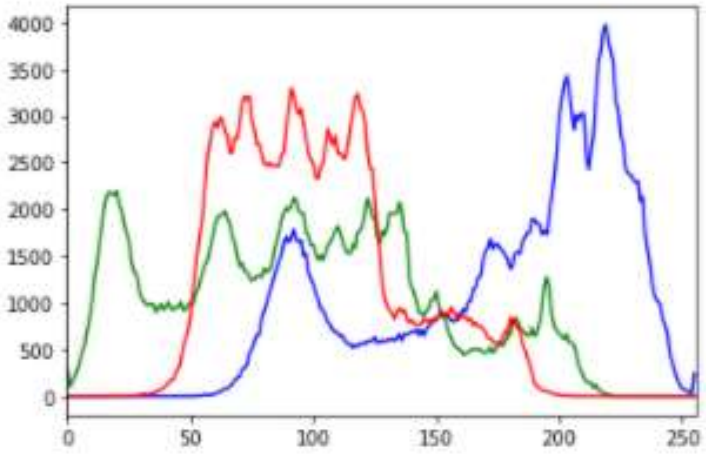
- **lenna.JPG**(image/jpeg) - 51779 bytes, last modified: 4/6/2021 - 100% done

```
Saving lenna.JPG to lenna (1).JPG
```

```
img_bgr = imread('lenna.JPG')

plt.imshow(img_bgr)
plt.show()

```

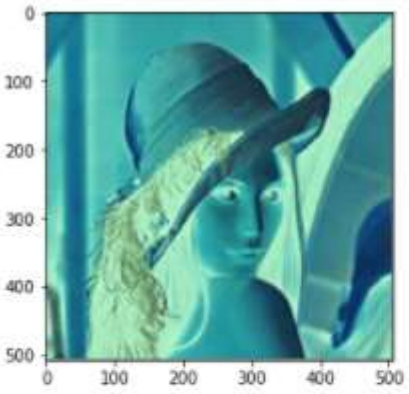


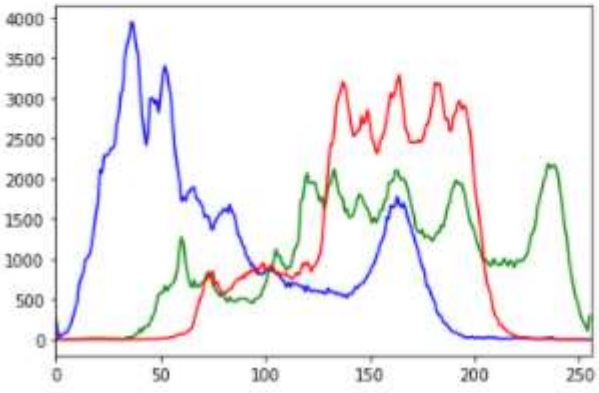
[23]
```
# Histogram plotting of the image
color = ('b', 'g', 'r')

for i, col in enumerate(color):

    histr = cv2.calcHist([img_bgr],
                         [i], None,
                         [256],
                         [0, 256])

    plt.plot(histr, color = col)

    # Limit X - axis to 256
    plt.xlim([0, 256])

plt.show()
```



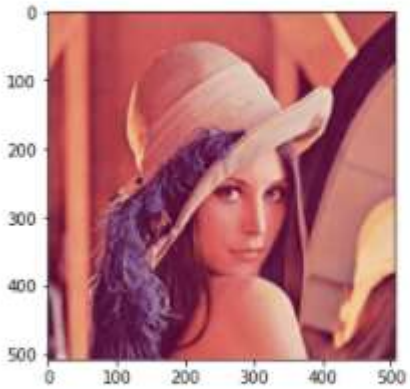
[19]
```
# get height and width of the image
height, width, _ = img_bgr.shape

for i in range(0, height - 1):
    for j in range(0, width - 1):

        # Get the pixel value
        pixel = img_bgr[i, j]

        # Negate each channel by
        # subtracting it from 255

        # 1st index contains red pixel
        pixel[0] = 255 - pixel[0]

        # 2nd index contains green pixel
        pixel[1] = 255 - pixel[1]

        # 3rd index contains blue pixel
        pixel[2] = 255 - pixel[2]

        # Store new values in the pixel
        img_bgr[i, j] = pixel
```

[20]
```
# Display the negative transformed image
plt.imshow(img_bgr)
plt.show()

# Histogram plotting of the
# negative transformed image
color = ('b', 'g', 'r')

for i, col in enumerate(color):

    histr = cv2.calcHist([img_bgr],
                         [i], None,
                         [256],
                         [0, 256])

    plt.plot(histr, color = col)
    plt.xlim([0, 256])

plt.show()
```

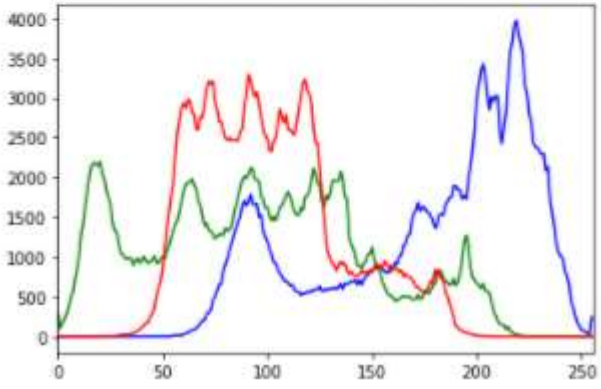




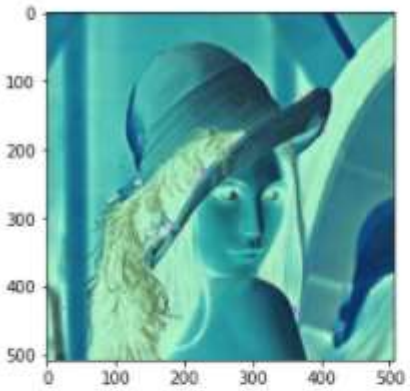
[26]
```
# Metode Negatif ke-2
img_bgr = imread('lenna.JPG')
plt.imshow(img_bgr)
plt.show()


```

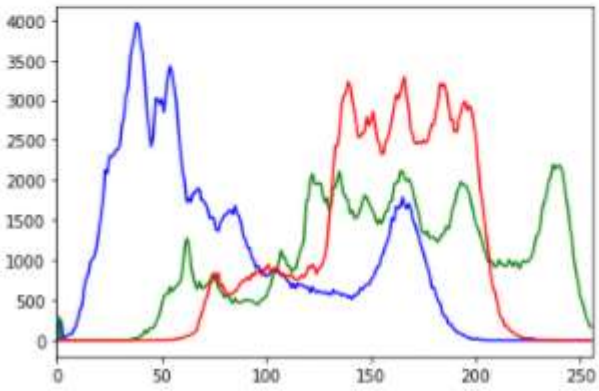



```
# Histogram plotting of original image
color = ('b', 'g', 'r')

for i, col in enumerate(color):

    histr = cv2.calcHist([img_bgr],
                         [i], None,
                         [256],
                         [0, 256])

    plt.plot(histr, color = col)

    # Limit X - axis to 256
    plt.xlim([0, 256])

plt.show()

```



[29]
```python
# Negatif the original image (Rumus Negatif)
img_neg = 1 - img_bgr

plt.imshow(img_neg)
plt.show()

# Histogram plotting of
# negative transformed image
color = ('b', 'g', 'r')

for i, col in enumerate(color):

    histr = cv2.calcHist([img_neg],
                         [i], None,
                         [256],
                         [0, 256])

    plt.plot(histr, color = col)
    plt.xlim([0, 256])

plt.show()
```





Log_Gamma.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
[2]  1 import cv2
     2 import matplotlib.pyplot as plt
     3 import numpy as np
     4 from skimage import data, color
     5 from skimage.io import imread, imshow
     6
     7 from google.colab import files
     8 uploaded = files.upload()
```

Choose Files lenna.JPG

- **lenna.JPG**(image/jpeg) - 51779 bytes, last modified: 4/6/2021 - 100% done

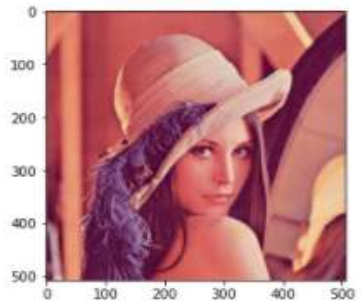
```
Saving lenna.JPG to lenna.JPG
```

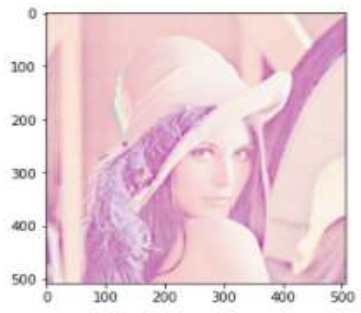
Implementasi Rumus Logaritmik (transformasi log dan inverse-log)

```
[3]  1 # Read an image
     2 image = imread('lenna.JPG')
     3
     4 # Apply log transformation method (Rumus di sini)
     5 c = 255 / np.log(1 + np.max(image))
     6 log_image = c * (np.log(image + 1))
     7
     8 # Specify the data type so that
     9 # float value will be converted to int
    10 log_image = np.array(log_image, dtype = np.uint8)
    11
    12 # Display both images
    13 plt.imshow(image)
    14 plt.show()
    15 plt.imshow(log_image)
    16 plt.show()
```

OUTPUT :

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:6: RuntimeWarning: divide by zero encountered in log
```

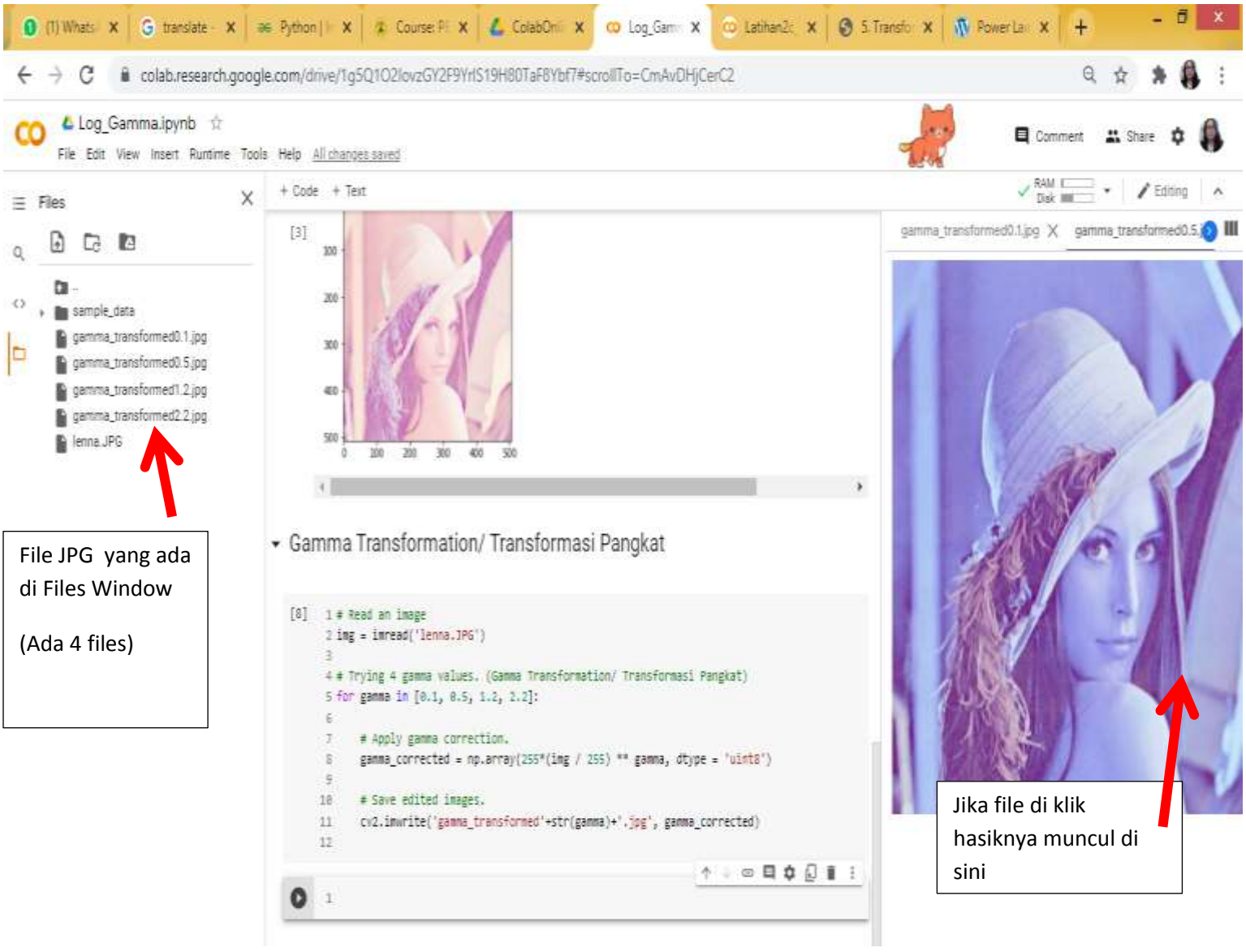




Gamma Transformation/ Transformasi Pangkat

```
[8]  1 # Read an image
     2 img = imread('lenna.JPG')
     3
     4 # Trying 4 gamma values. (Gamma Transformation/ Transformasi Pangkat)
     5 for gamma in [0.1, 0.5, 1.2, 2.2]:
     6
     7     # Apply gamma correction.
     8     gamma_corrected = np.array(255*(img / 255) ** gamma, dtype = 'uint8')
     9
    10     # Save edited images.
    11     cv2.imwrite('gamma_transformed'+str(gamma)+'.jpg', gamma_corrected)
    12
```

Output:



Equalisasi_NIM.ipynb ☆

File Edit View Insert Runtime Tools Help

+ Code + Text

HISTOGRAM DAN HISTOGRAM EQUALISASI

```
# read as grayscale
figsize = (10,10)
I = cv2.imread("lenna.JPG",0)

plt.figure(figsize=figsize)
plt.imshow(I, cmap='gray', vmin=0, vmax=255)
plt.title("Original image")

```

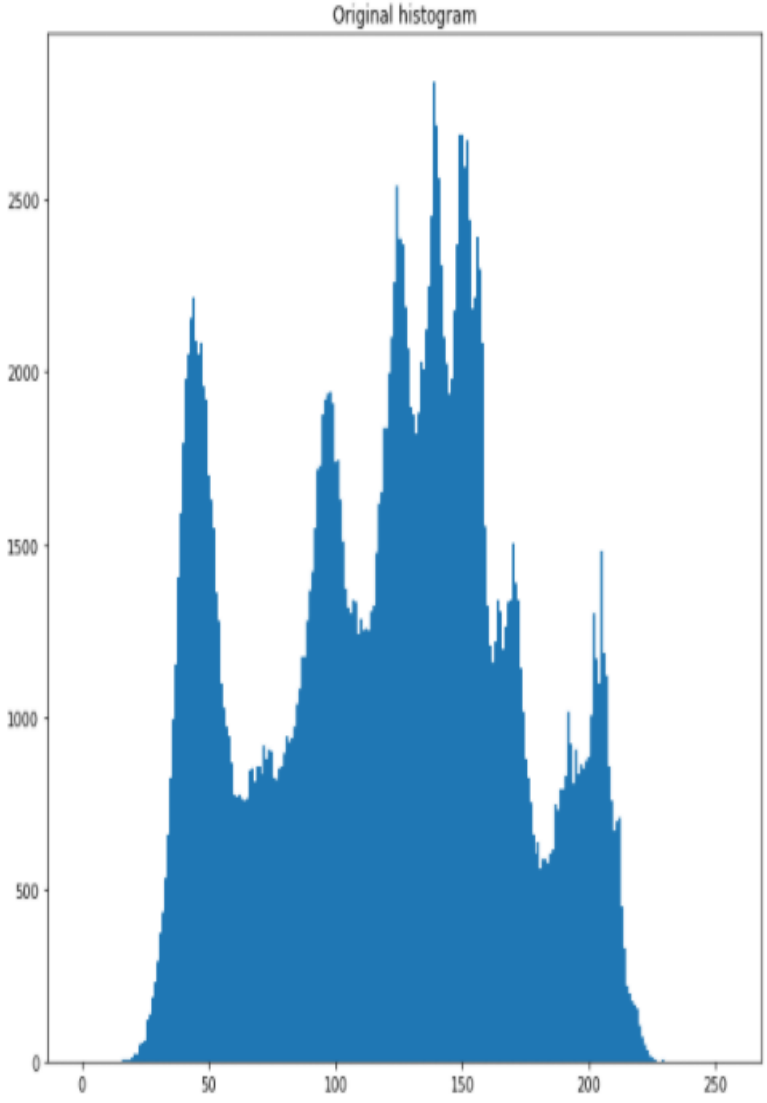
```
Text(0.5, 1.0, 'Original image')
```

Menghitung dan menampilkan histogram asli

```
[ ]  1 bins_edges_min_max = [0,256]
     2 num_bins=256
     3 bin_count,bins_edges = np.histogram(I,num_bins,bins_edges_min_max)
     4 bins_start = bins_edges[:-1]
     5
```

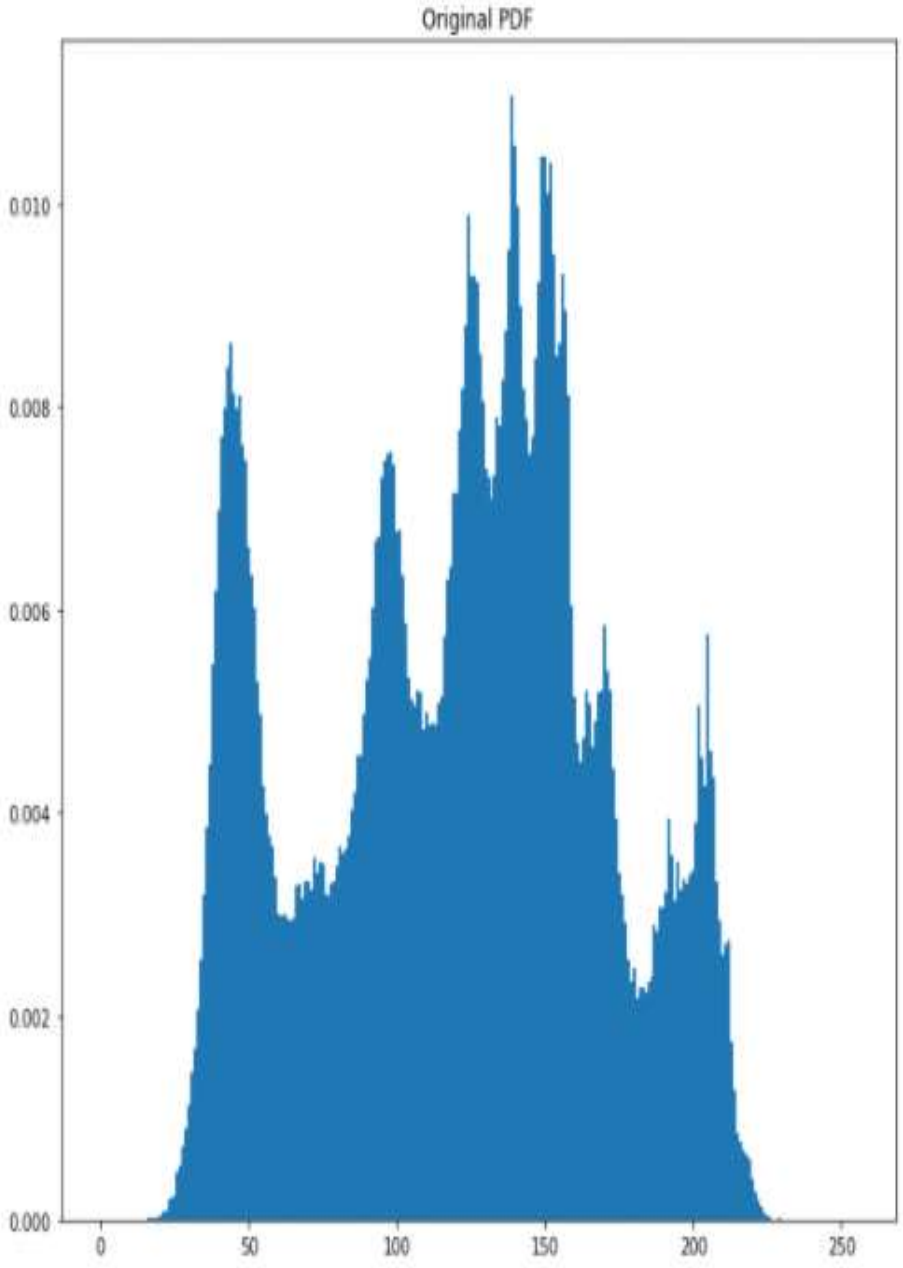
```
[ ]  1 def draw_hist(x_axis,input):
     2     fig,ax = plt.subplots(figsize=figsize)
     3     # kenapa tidak menggunakan plt.hist? karena kita ingin memplot juga beberapa turunan dari hist ini, jadi ini lebih mudah
     4     plt.bar(x_axis, input, width=input.shape[0]/(x_axis[-1]-x_axis[0]+1))
     5     return fig,ax
     6
     7 draw_hist(bins_start,bin_count)
     8 plt.title("Original histogram")
     9
```

```
Text(0.5, 1.0, 'Original histogram')
```



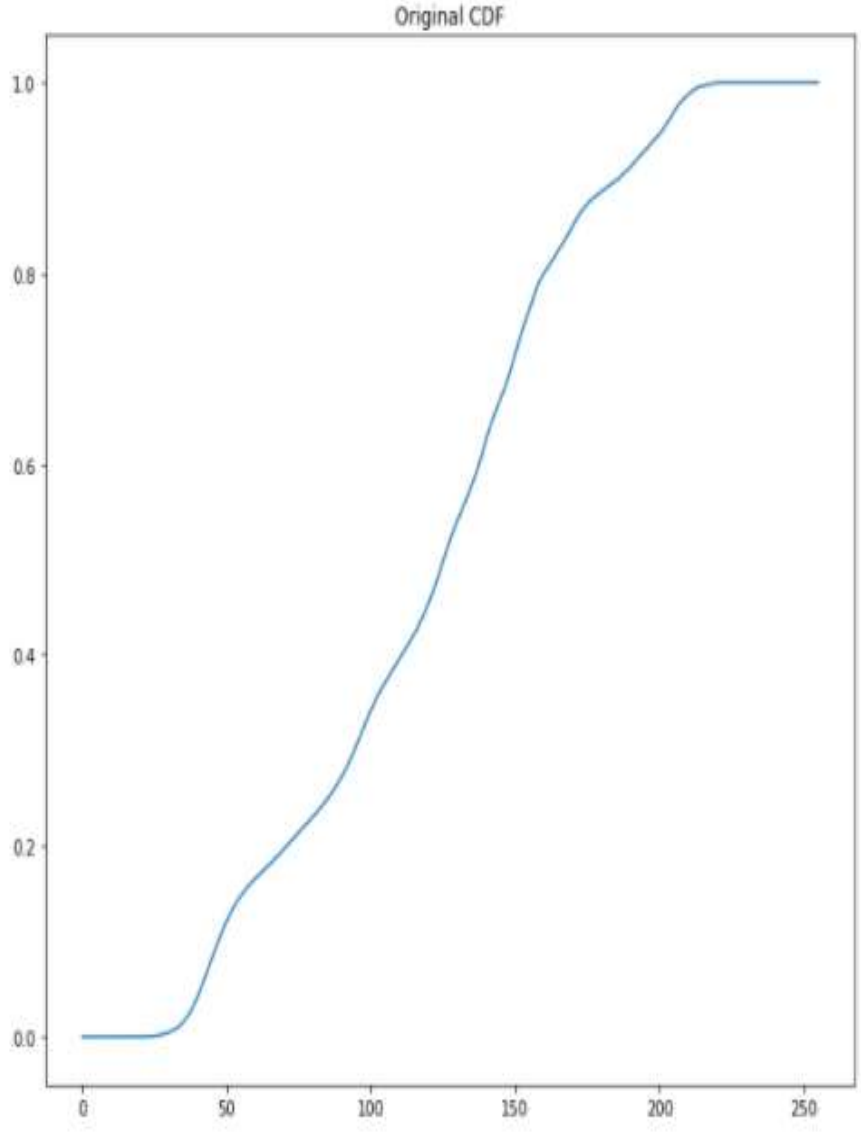
```
[ ]  1 # Normalisasi histogram untuk membuat PDF (Probability Distribution Function) terpisah
     2 pdf = bin_count/np.sum(bin_count)
     3
     4 draw_hist(bins_start,pdf)
     5 plt.title("Original PDF")
```

```
Text(0.5, 1.0, 'Original PDF')
```



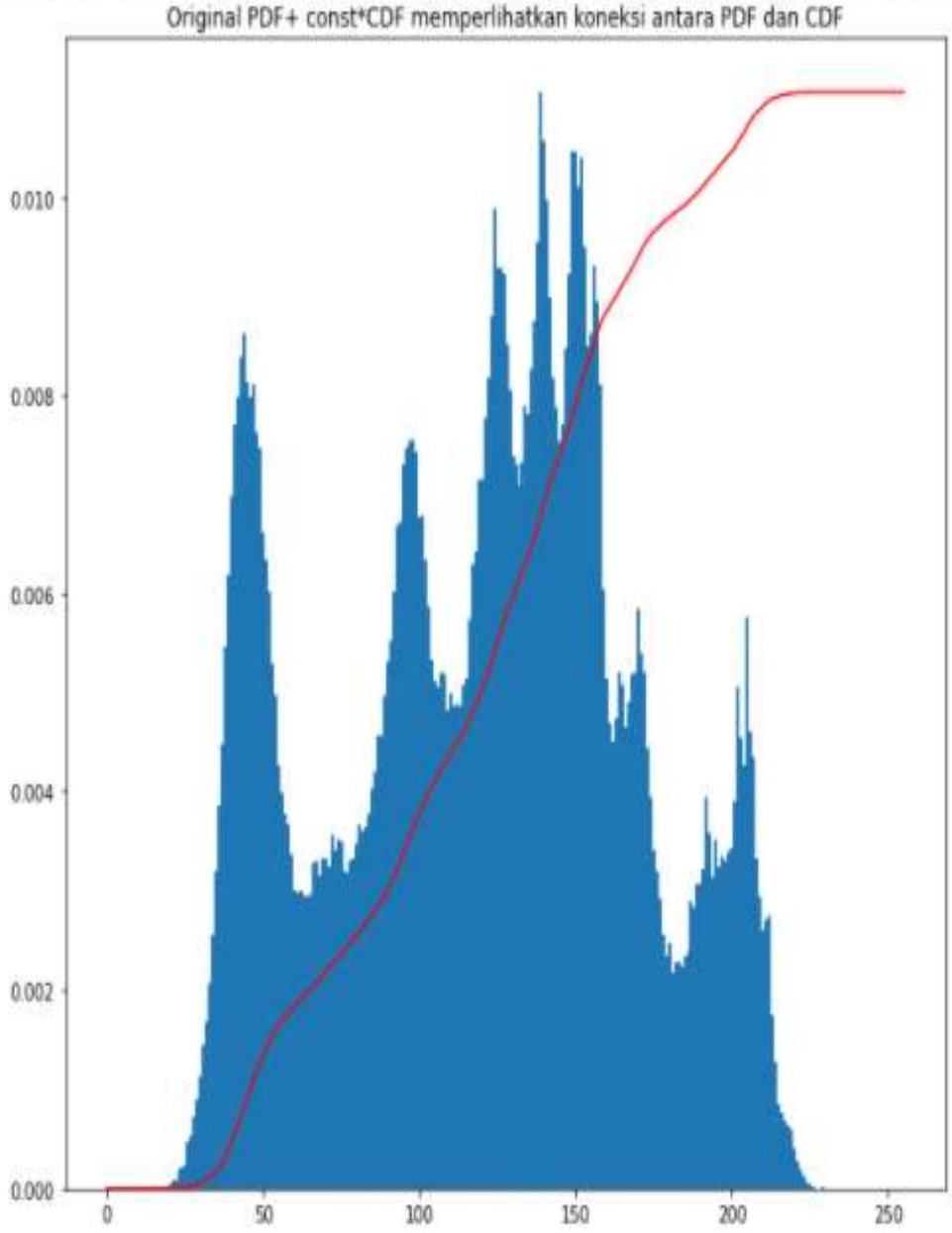
```
[ ]  1 ## Dapatkan CDF (cumulative distribution function) dengan menghitung jumlah kumulatif data pdf
     2 cdf = np.cumsum(pdf)
     3
     4 plt.figure(figsize=figsize)
     5 plt.plot(cdf)
     6 plt.title("Original CDF")
```

```
Text(0.5, 1.0, 'Original CDF')
```



```
[ ]  1 fig,ax = draw_hist(bins_start,pdf)
     2 ax.plot(cdf*np.max(pdf),'r')
     3 plt.title("Original PDF+ const*CDF memperlihatkan koneksi antara PDF dan CDF")
```

```
Text(0.5, 1.0, 'Original PDF+ const*CDF memperlihatkan koneksi antara PDF dan CDF')
```



```
[ ]  1 # UnNormalisasi CDF menjadi fungsi pemerataan
     2
     3 f_eq = np.round(cdf*255).astype(int)
     4
     5 f_eq
```

```
array([  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
         0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
         0,   0,   1,   1,   1,   2,   2,   3,   3,   4,   5,   6,   8,
         9,  11,  13,  15,  17,  19,  21,  23,  25,  27,  29,  31,  33,
        34,  35,  37,  38,  39,  40,  41,  42,  42,  43,  44,  45,  45,
        46,  47,  48,  49,  49,  50,  51,  52,  53,  54,  55,  55,  56,
        57,  58,  59,  60,  61,  62,  63,  64,  65,  66,  67,  68,  70,
        71,  73,  74,  76,  78,  80,  82,  84,  86,  87,  89,  91,  92,
        93,  95,  96,  97,  99, 100, 101, 102, 104, 105, 106, 108, 109,
       111, 112, 114, 116, 118, 120, 122, 125, 127, 129, 132, 134, 136,
       138, 140, 142, 143, 145, 147, 150, 152, 154, 157, 160, 162, 165,
       167, 169, 171, 173, 175, 177, 180, 182, 185, 188, 190, 192, 194,
       197, 199, 201, 203, 204, 205, 206, 208, 209, 210, 211, 213, 214,
       215, 217, 218, 219, 221, 222, 222, 223, 224, 225, 225, 226, 226,
       227, 228, 228, 229, 229, 230, 231, 232, 232, 233, 234, 235, 236,
       237, 238, 239, 239, 240, 241, 242, 243, 245, 246, 247, 248, 249,
       250, 251, 252, 252, 253, 253, 254, 254, 254, 254, 255, 255, 255,
       255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
       255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
       255, 255, 255, 255, 255, 255, 255, 255, 255])
```

```
     1 # Gunakan fungsio Equalisasi / fungsi pemerataan untuk mendapatkan gambar yang disamakan
     2
     3 I_eq = f_eq[I]
     4
     5 plt.figure(figsize=figsize)
     6 plt.imshow(I_eq, cmap='gray', vmin=0, vmax=255)
     7 plt.title("equalisasi citra")
     8
```

```
Text(0.5, 1.0, 'equalisasi citra')
```

```
[ ]  1 bin_count,bins_edges = np.histogram(I_eq,num_bins,bins_edges_min_max)
     2 bins_start = bins_edges[:-1]
     3
     4 draw_hist(bins_start,bin_count)
     5 plt.title("Histogram Equalisasi ")
     6
```

```
Text(0.5, 1.0, 'Histogram Equalisasi ')
```