Bài giảng môn học:

Thị giác máy tính (7080518)

CHƯƠNG 2:
XỬ LÝ VÀ NÂNG CAO CHẤT LƯỢNG ẢNH
(Phần 2)

Đặng Văn Nam

dangvannam@humg.edu.vn

Nội dung chương 2 – phần 2

II/ Cải thiện chất lượng ảnh

1. Cơ bản về cải thiện chất lượng ảnh
2. Các phép toán trên điểm ảnh
    1. Đảo ảnh
    2. Biến đổi gama
    3. Chuyển đổi ảnh nhị phân
3. Cân bằng sáng histogram

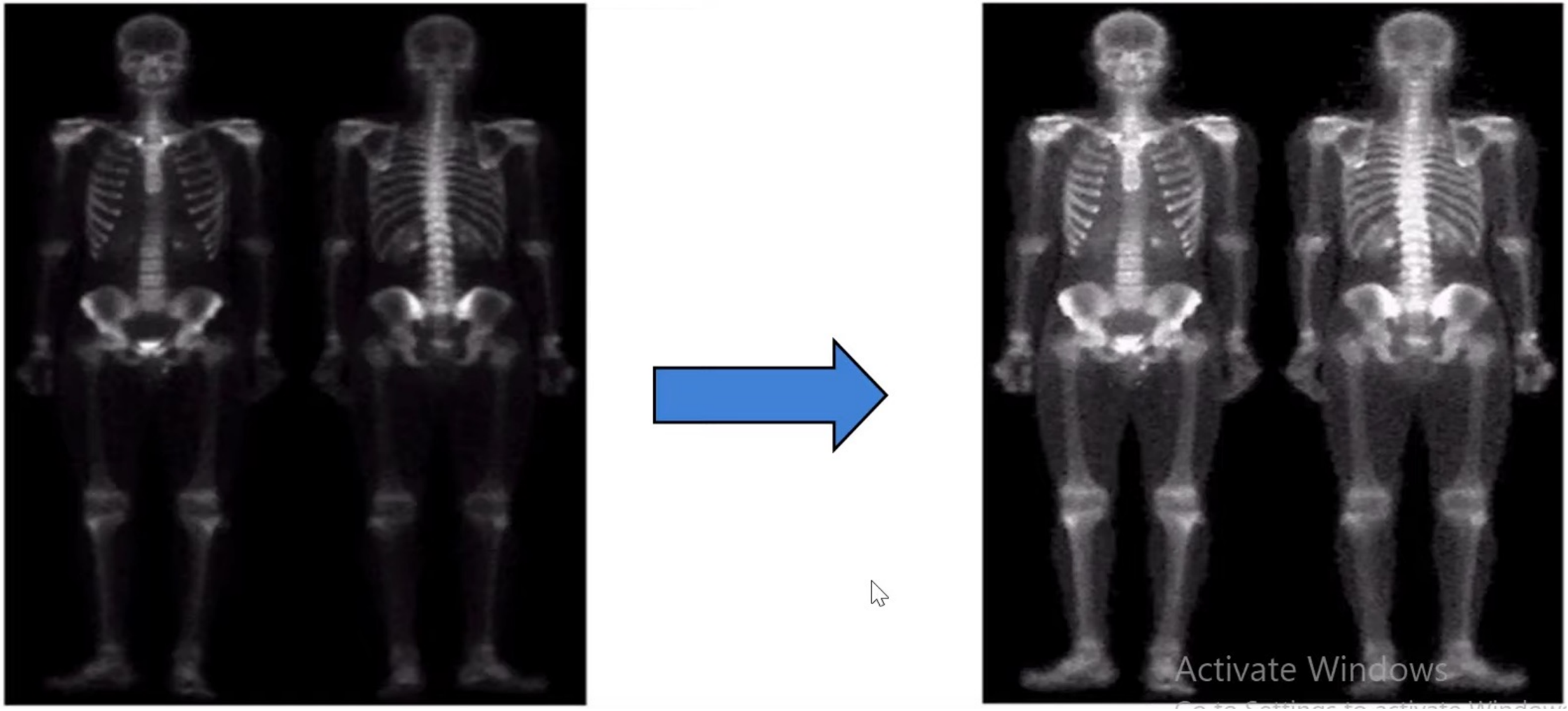
2.1. Cải thiện chất lượng ảnh

2.1. Cải thiện chất lượng ảnh là gì

Cân bằng sáng

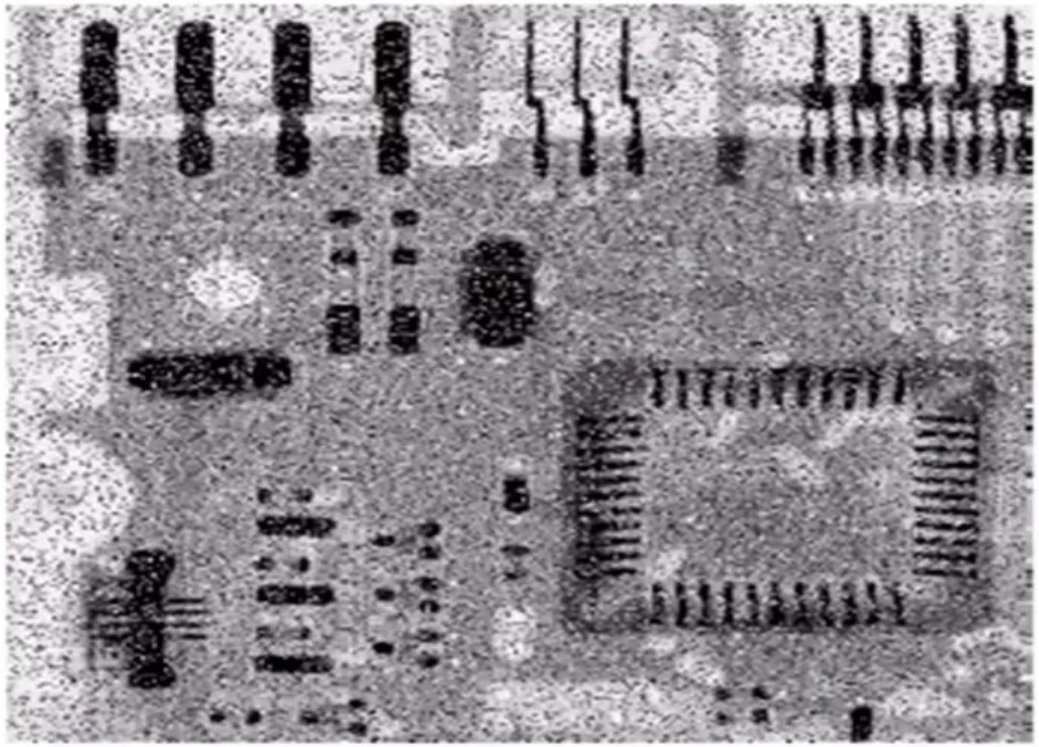
2.1. Cải thiện chất lượng ảnh là gì

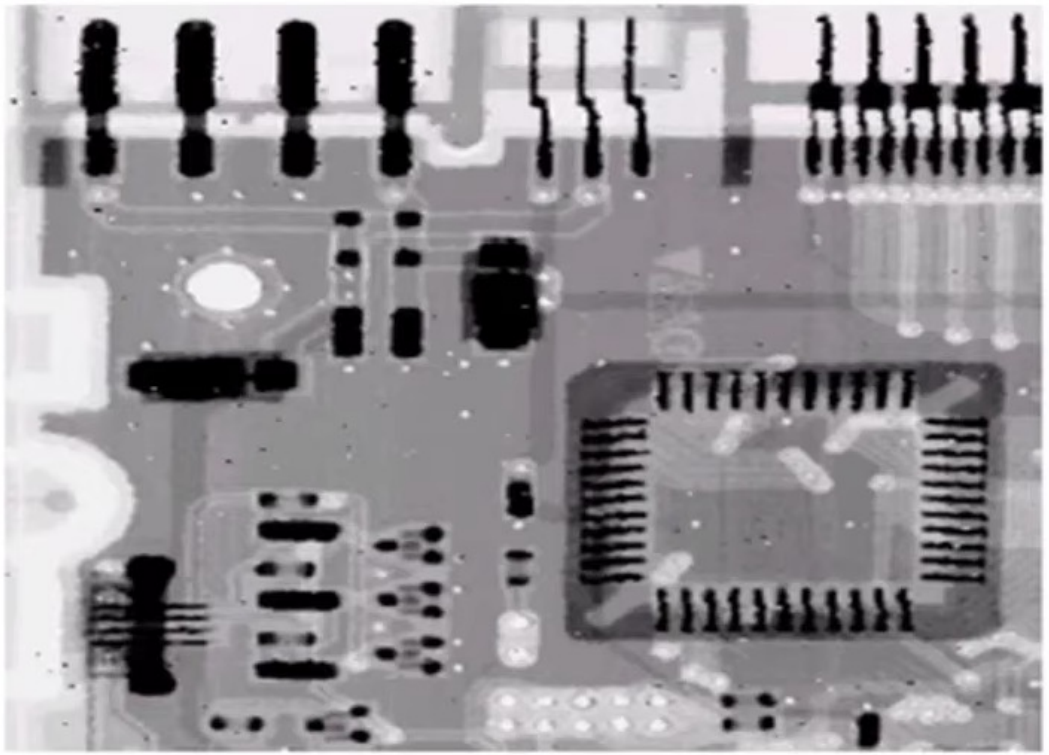
Cải thiện độ tương phản



2.1. Cải thiện chất lượng ảnh là gì

Cải thiện độ mịn/trơn của của ảnh





2.1. Cải thiện chất lượng ảnh là gì

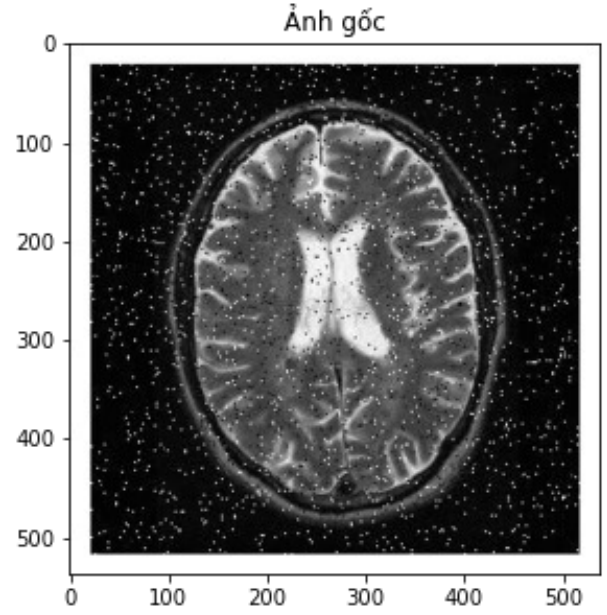
Cải thiện độ sắc nét của ảnh

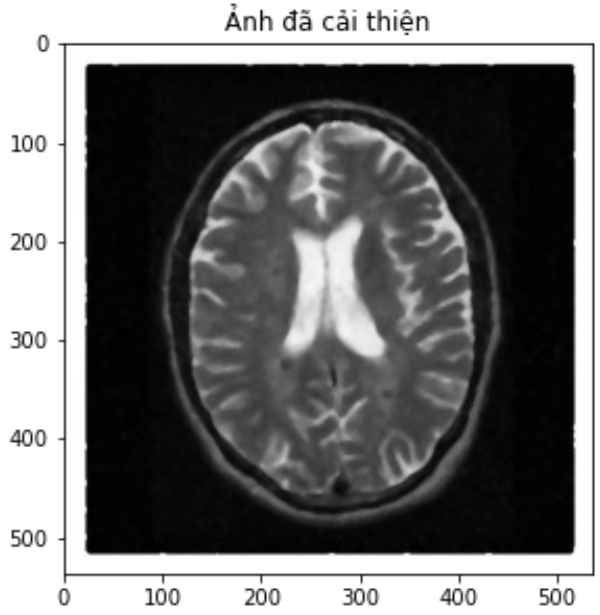
Cải thiện ảnh là gì?

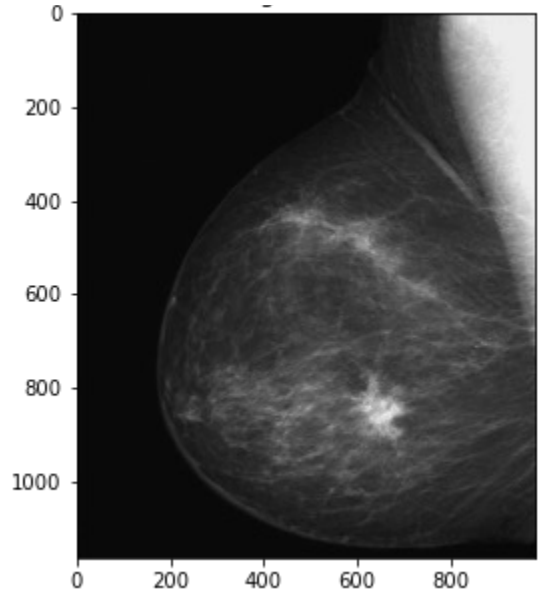
Cải thiện ảnh là quá trình làm cho ảnh trở nên hữu ích hơn. Là quá trình xử lý một hình ảnh để cho kết quả phù hợp hơn với kế quả ban đầu.

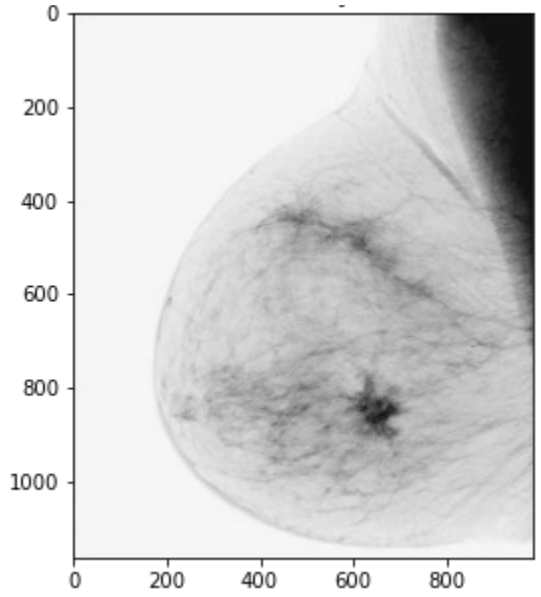
→Làm nổi các chi tiết cần quan tâm trong ảnh.

→Làm cho ảnh trở nên trực quan, hấp dẫn hơn.









Phân loại các kiểu cải thiện ảnh:

Các kỹ thuật cải thiện ảnh được phân thành 2 nhóm:

- **Các kỹ thuật theo miền không gian:** thao tác trực tiếp lên các pixel điểm ảnh.
  - Các phép toán trên điểm ảnh
  - Các bộ lọc
  - Kỹ thuật Histogram
- **Các kỹ thuật theo miền tần số:** Ảnh được xem như tín hiệu 2 chiều
  - Tác động lên tần số để cải thiện chất lượng ảnh
  - Biến đổi Fourrier, biến đổi sóng …

2.2. Các phép toán trên điểm ảnh

Các phép toán trên điểm ảnh:

Thao tác xử lý điểm ảnh có dạng:

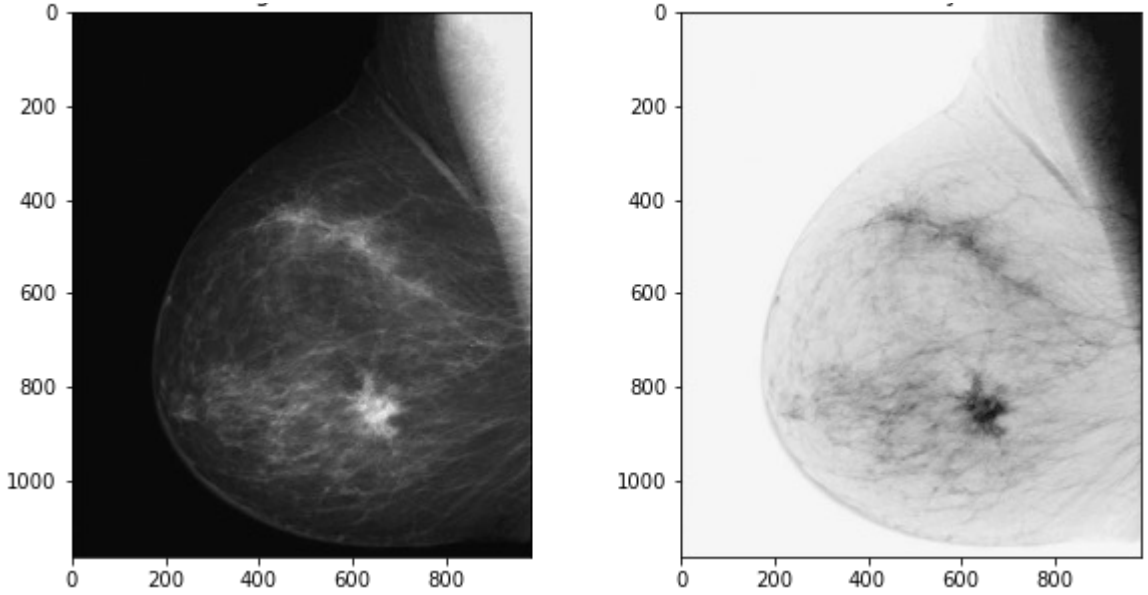
$$s = T(r)$$

Trong đó:

- s: là điểm ảnh sau khi xử lý
- r: là điểm ảnh ban đầu
- T: là phép toán xử lý điểm ảnh:
  - *Phép đảo ảnh*
  - *Phép biến đổi Logarit*
  - *Phép biến đổi Gamma*
  - *Cắt ngưỡng (chuyển đổi sang ảnh nhị phân)*

Các phép toán trên điểm ảnh:

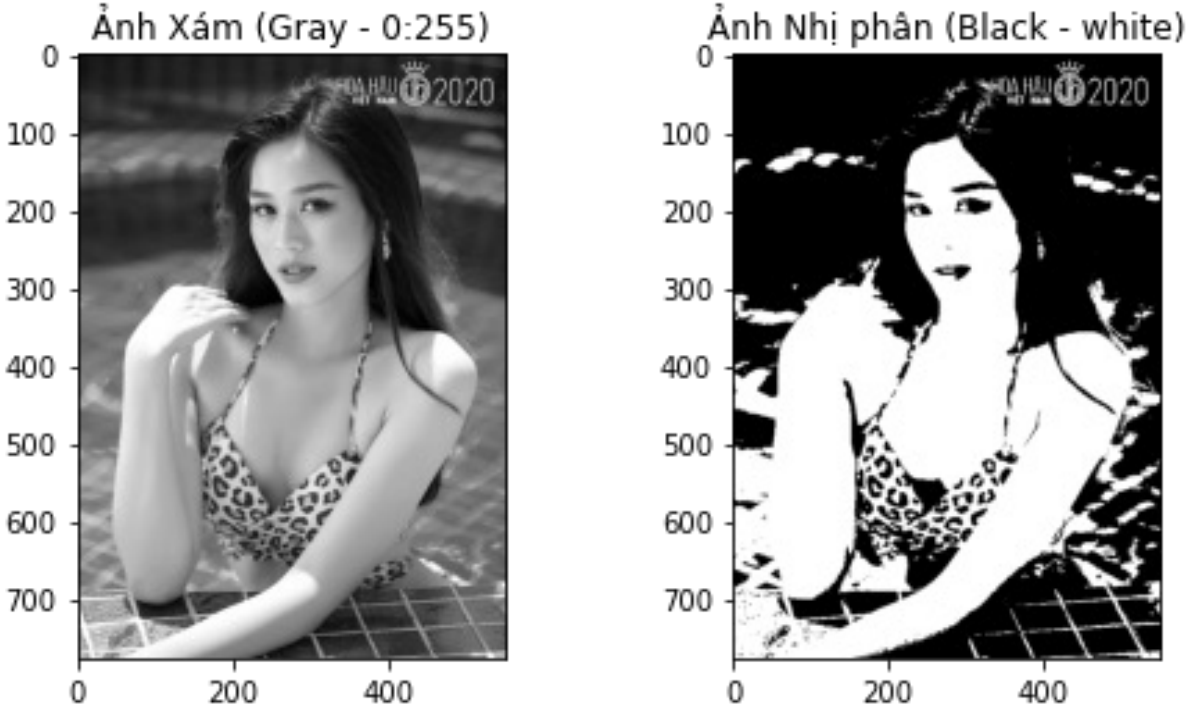
Đảo ảnh: s = max - r



Chuyển đổi ảnh xám → đen trắng:

s = 0 if (r<128)

s = 255 if (r>=128)



2.2.1. Đảo ảnh

Đảo ảnh

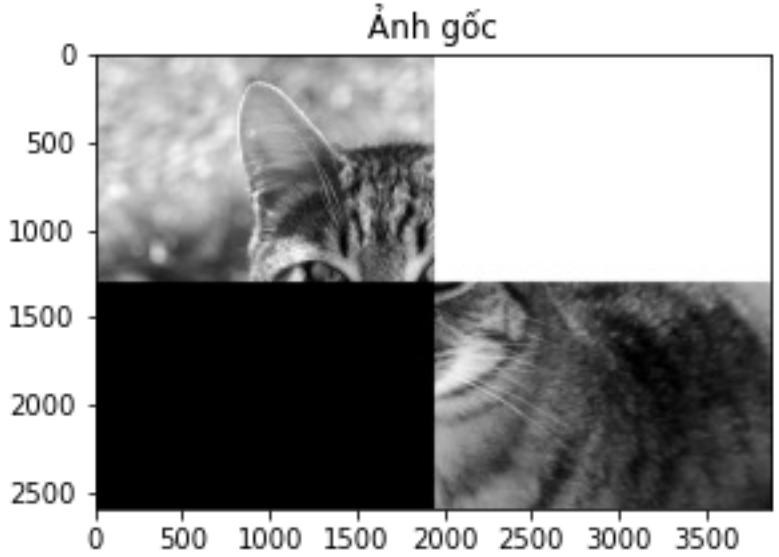
Thực hiện trên các ảnh xám. Giá trị mức xám tại các pixel điểm ảnh nằm trong khoảng [0,N]. Phép toán đảo ảnh có dạng: s = N – r

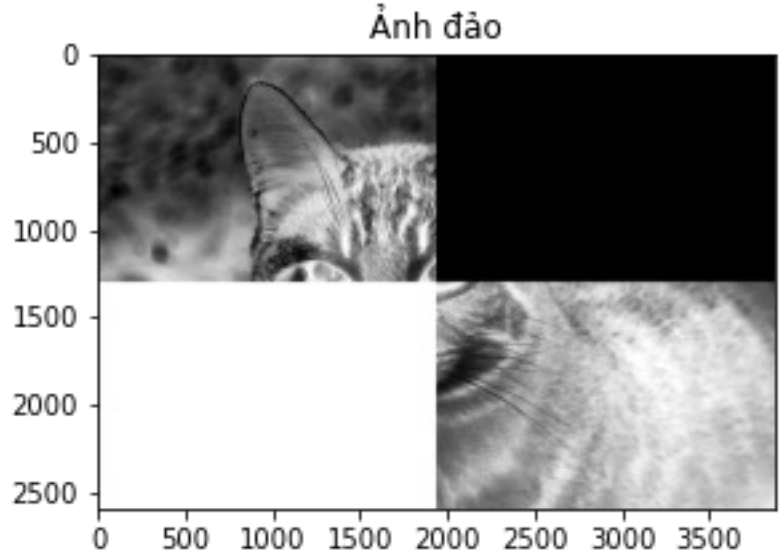
Trong đó:

- s: điểm ảnh đã xử lý
- r: điểm ảnh ban đầu
- N: giá trị xám cực đại của ảnh

12	135	0	112
248	135	175	210
234	22	45	255
11	0	27	4

243	120	255	143
7	120	80	45
21	233	210	0
244	255	228	251



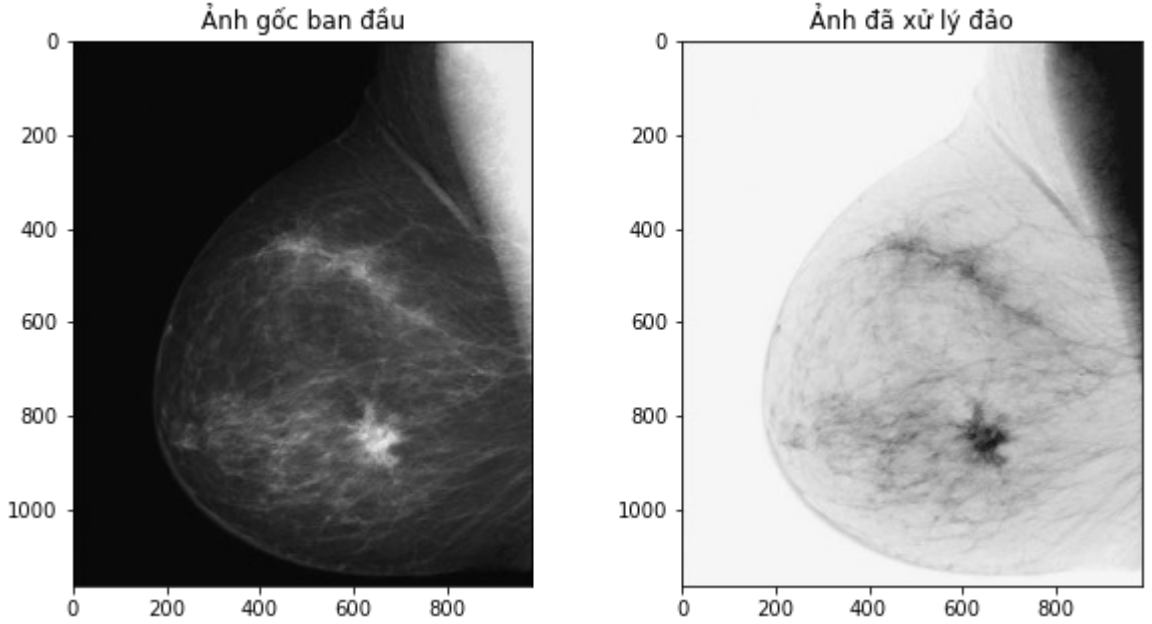


Đảo ảnh

Xử lý trong Python:

```python
#Hàm đảo ảnh:
def dao_anh(img):
    return img.max()-img
```

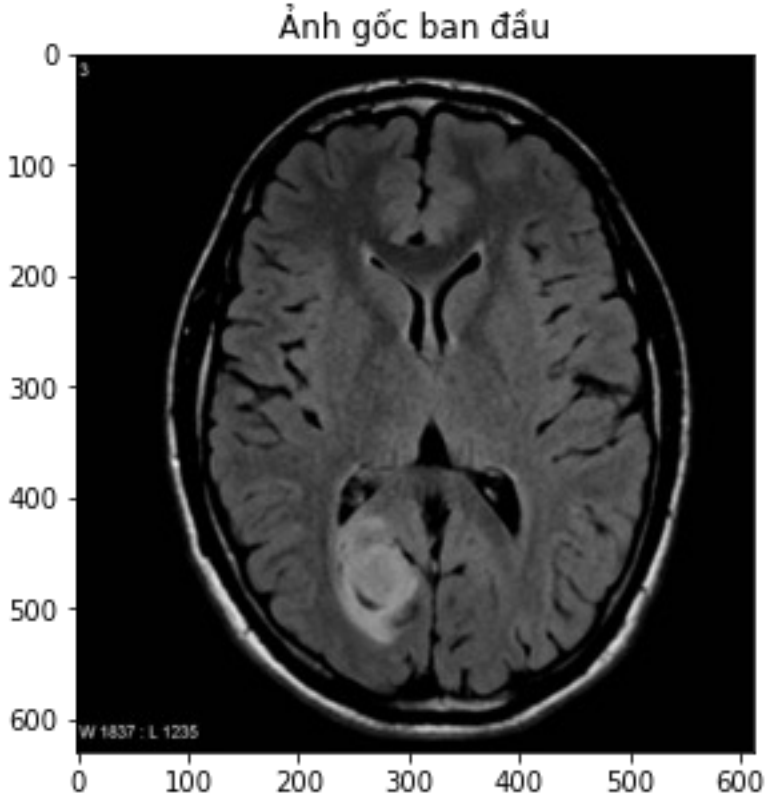
```python
#Đọc ảnh gốc:
img = cv2.imread('images/pic_healthcare.tif',0)
#Thực hiện đảo ảnh:
img_dao = dao_anh(img)

#Hiển thị kết quả:
plt.figure(figsize=(10,5))
plt.subplot(1, 2, 1)
plt.imshow(img, cmap='gray')
plt.title('Ảnh gốc ban đầu')

plt.subplot(1, 2, 2)
plt.imshow(img_dao, cmap='gray')
plt.title('Ảnh đã xử lý đảo')
plt.show()
```

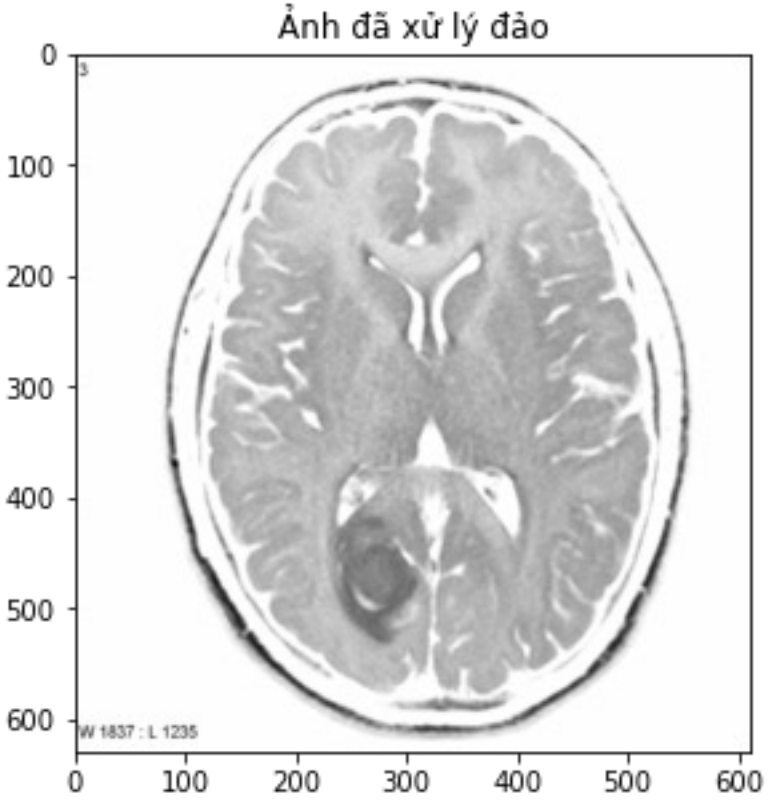


Thực hành số 2.4

Thực hành 2.4

Yêu cầu 1: Sinh viên đọc ảnh images/Thuchanh2_4.jpeg và hiển thị ảnh
Yêu cầu 2: Thực hiện đảo ảnh và hiển thị kết quả. Lưu file ảnh đã đảo vào thư mục Saves/MSV_daoanh.jpg





2.2.2. Biến đổi ảnh bằng hàm mũ

Biến đổi ảnh bằng hàm mũ (gamma)

Phép toán biến đổi ảnh bằng hàm mũ có dạng: $s = c * r^a$

Trong đó:

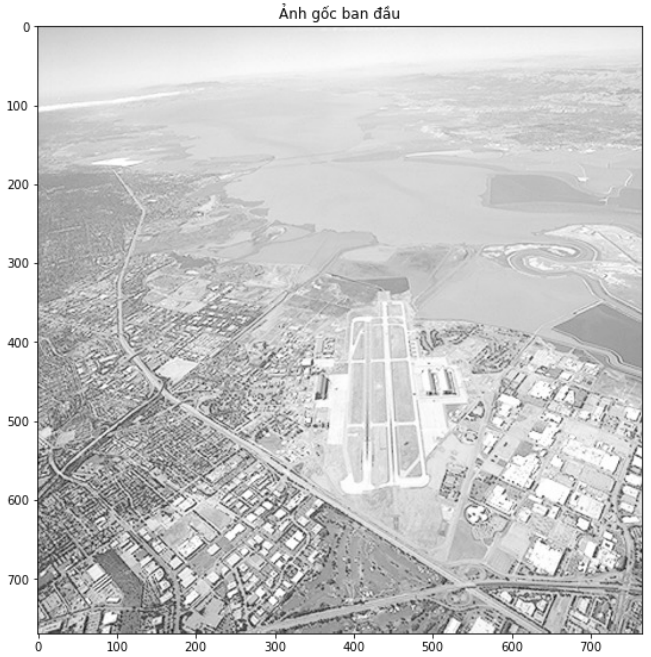
- s: điểm ảnh đã xử lý
- r: điểm ảnh ban đầu
- c,a: các hằng số

Ví dụ: c = 1; a = 2.5

12	135	0	112
248	135	175	210
234	22	45	255
11	0	27	4

→

498	211755	0	132753
968565	211755	405130	639069
837605	2270	13584	1038365
401	0	3787	32



Biến đổi ảnh bằng hàm mũ (gamma)

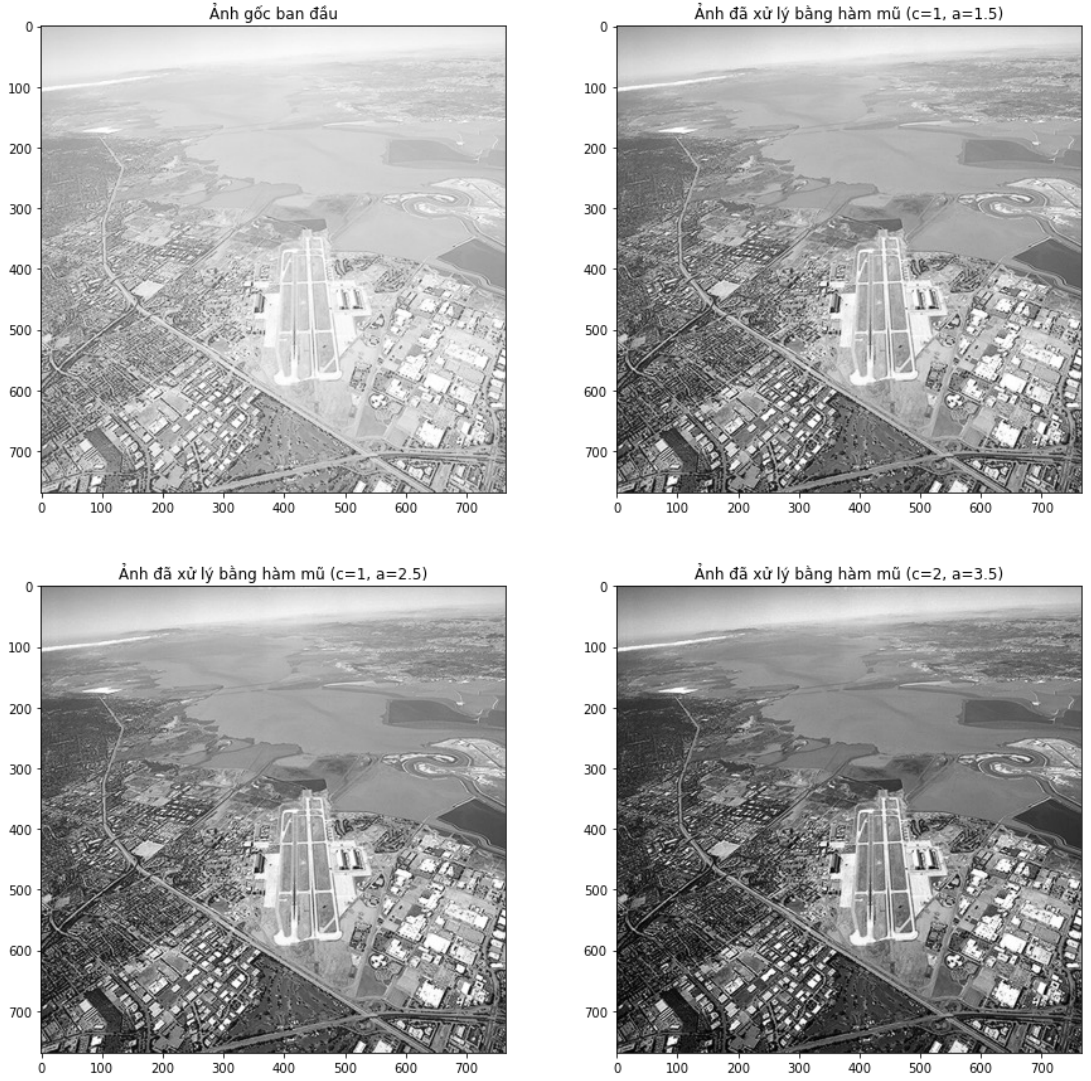
Xử lý trong Python:

```python
#Viết hàm xử lý ảnh với hàm mũ:
def pic_gamma(img,c,a):
    #Biến đổi hàm mũ:
    img1 = c*pow(img,a)
    #Chuyển các phần tử về số nguyên:
    img1 = img1.astype(np.int32)
    return img1
```

```python
#Biến đổi ảnh với hàm mũ sử dụng các tham số khác nhau:
pic1 = pic_gamma(img,1,1.5)
pic2 = pic_gamma(img,1,2.5)
pic3 = pic_gamma(img,2,3.5)

#Hiển thị kết quả:
plt.figure(figsize=(15,15))
plt.subplot(2, 2, 1)
plt.imshow(img, cmap='gray')
plt.title('Ảnh gốc ban đầu')

plt.subplot(2, 2, 2)
plt.imshow(pic1, cmap='gray')
plt.title('Ảnh đã xử lý bằng hàm mũ (c=1, a=1.5)')

plt.subplot(2, 2, 3)
plt.imshow(pic2, cmap='gray')
plt.title('Ảnh đã xử lý bằng hàm mũ (c=1, a=2.5)')

plt.subplot(2, 2, 4)
plt.imshow(pic3, cmap='gray')
plt.title('Ảnh đã xử lý bằng hàm mũ (c=2, a=3.5)')
plt.show()
```

Biến đổi ảnh bằng hàm mũ (gamma)

Kết quả:

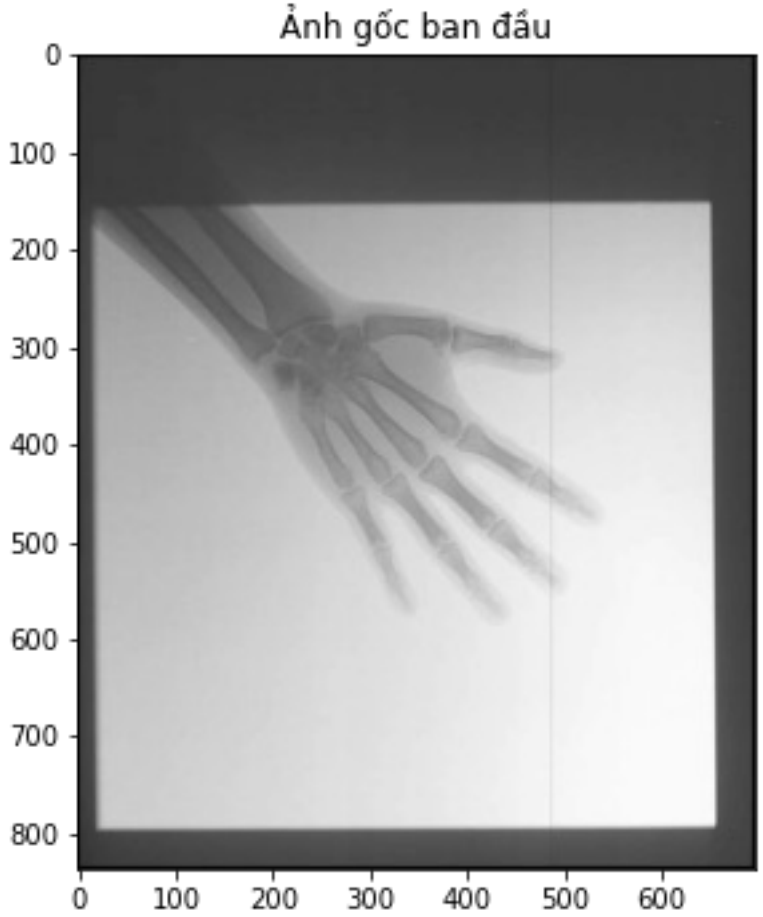


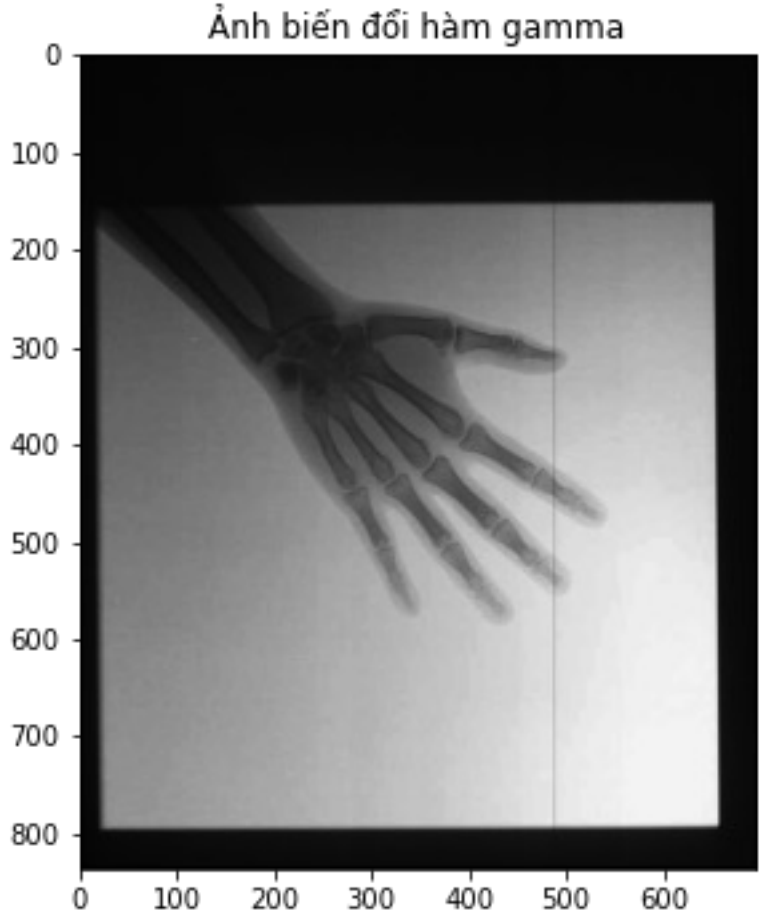
Thực hành số 2.5

Thực hành 2.5

Yêu cầu :

1. Sinh viên đọc ảnh images/Thuchanh2_5.jpg ở dạng xám.
2. Thiết lập tham số c, a phù hợp để tăng cường chất lượng ảnh và hiển thị kết quả (slide). Lưu file ảnh đã đổi vào thư mục Saves/MSV_gamma.jpg





2.3. Cắt ngưỡng

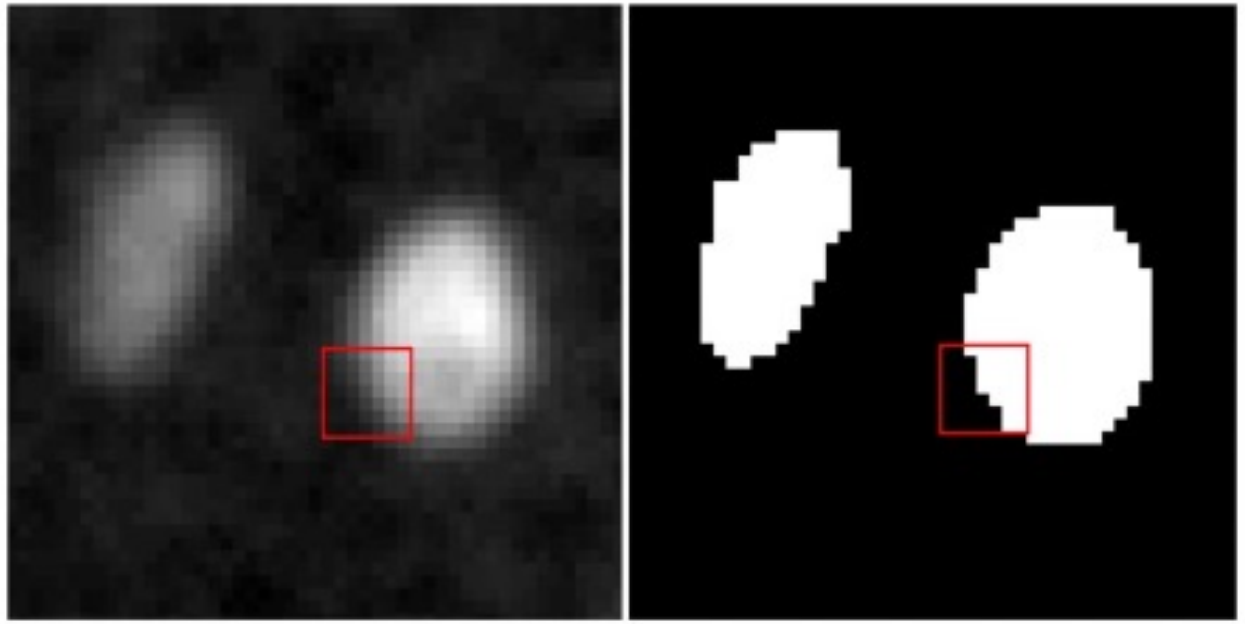
Cắt ngưỡng ảnh

Sử dụng một ngưỡng (threshold) T để xử lý. Nếu:

- **r<T thì s=0**
- **r>=T thì s=N (hoặc s = 1)**

Trong đó:

- s: điểm ảnh đã xử lý
- r: điểm ảnh ban đầu



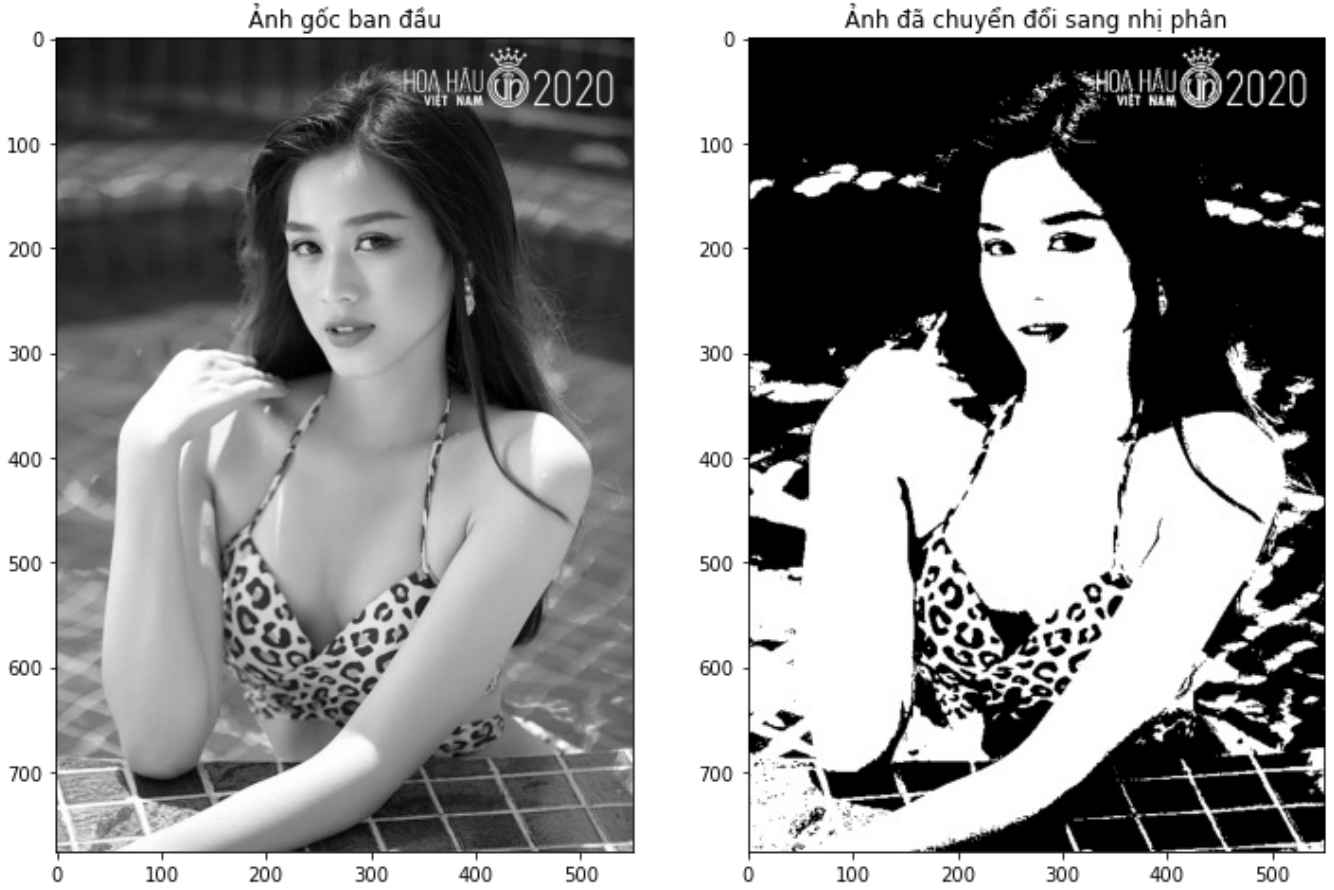
61	66	75	86	96
59	64	73	83	92
59	64	73	83	92
56	60	68	79	88
54	58	64	72	81

0	0	1	1	1
0	0	1	1	1
0	0	1	1	1
0	0	0	1	1
0	0	0	0	1

Cắt ngưỡng ảnh

Xử lý với Numpy:

```
#Đọc ảnh xám:
img = cv2.imread('images/pic_gray.png',0)

#Cắt ngưỡng ảnh xám:
T = 127#Thiết lập ngưỡng

#Thực hiện biến đổi, nếu <T = 0, >T=255
img_t = np.where(img<T,0,255)

#Hiển thị kết quả:
plt.figure(figsize=(12,10))
plt.subplot(1, 2, 1)
plt.imshow(img, cmap='gray')
plt.title('Ảnh gốc ban đầu')

plt.subplot(1, 2, 2)
plt.imshow(img_t, cmap='gray')
plt.title('Ảnh đã chuyển đổi sang nhị phân')
plt.show()
```



Cắt ngưỡng ảnh

Ngoài ra, Thư viện OpenCV cũng cung cấp một phương thức để thực hiện cắt ngưỡng đơn giản:

ret,thresh1 = cv2.threshold(img, T, max, type)

Trong đó:

- img: ảnh gốc
- T: giá trị ngưỡng
- max: giá trị đc gán nếu > ngưỡng T
- type: Loại xử lý cắt ngưỡng

```
#Cắt ngưỡng ảnh xám:
T = 127#Thiết lập ngưỡng

#Thực hiện biến đổi, nếu <T = 0, >T=255
img_t = np.where(img<T,0,255)
```

```
#Sử dụng phương thức threshold:
ret,pic = cv2.threshold(img,127,255,cv2.THRESH_BINARY)
```

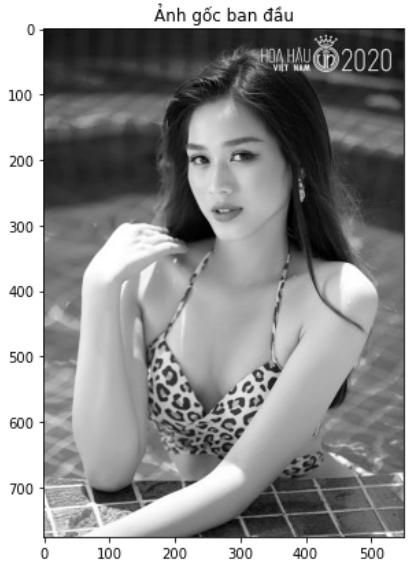
Cắt ngưỡng ảnh

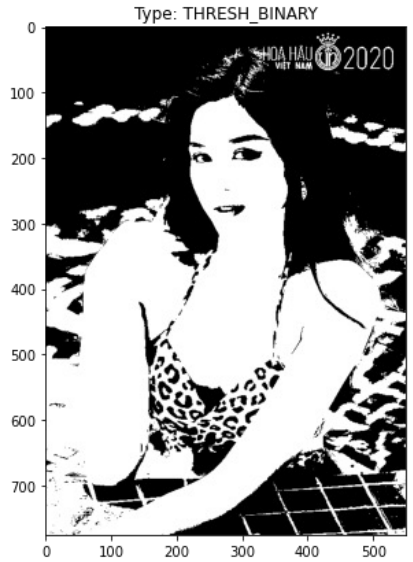
Thuật toán sample thresholding của OpenCV còn có 1 tham số quan trọng là loại ngưỡng (type).OpenCV hỗ trợ 8 loại là: THRESH_BINARY, THRESH_BINARY_INV, THRESH_TRUNC, THRESH_TOZERO, THRESH_TOZERO_INV, THRESH_MASK, THRESH_OTSU, THRESH_TRIANGLE. Ý nghĩa của từng loại như sau:

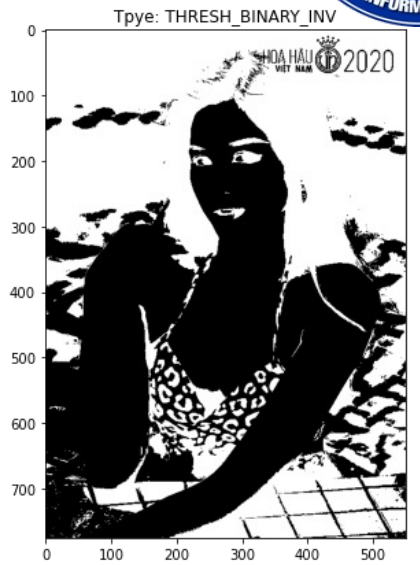
- **THRESH_BINARY**: Có thể dịch là ngưỡng nhị phân. Ý nghĩa y hệt những gì đề cập ở trên.
- **THRESH_BINARY_INV**: Ngưỡng nhị phân đảo ngược. Có thể hiểu là nó sẽ đảo ngược lại kết quả của THRESH_BINARY.
- **THRESH_TRUNC**: Những giá trị điểm ảnh bé hơn ngưỡng sẽ giữ nguyên giá trị, những điểm ảnh lớn hơn hoặc bằng ngưỡng sẽ được gán lại là maxvalue.
- **THRESH_TOZERO**: Những điểm ảnh bé hơn ngưỡng sẽ bị gán thành 0, những điểm còn lại giữ nguyên.
- **THRESH_TOZERO_INV**: Những điểm ảnh nhỏ hơn giá trị ngưỡng sẽ được giữ nguyên, những điểm ảnh còn lại sẽ bị gán thành 0.

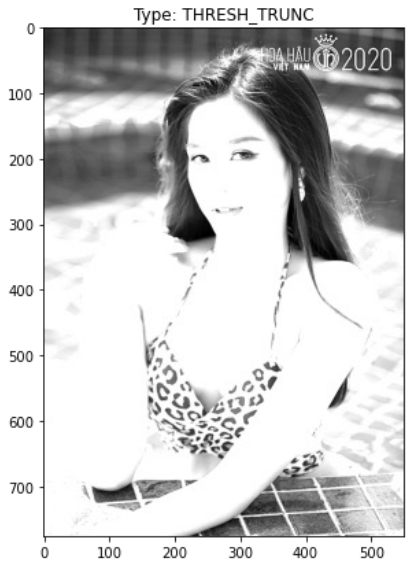
Cắt ngưỡng ảnh

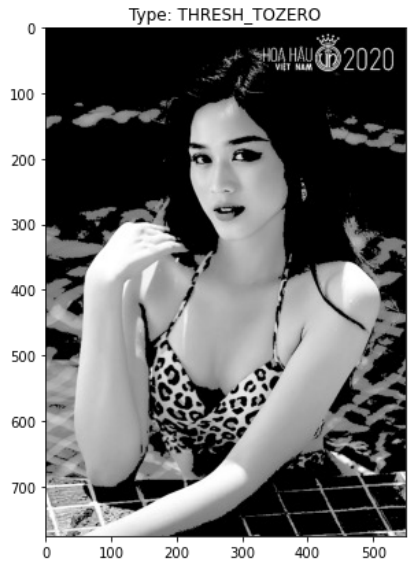
Sử dụng phương thức cv2.threshold với các loại khác nhau:

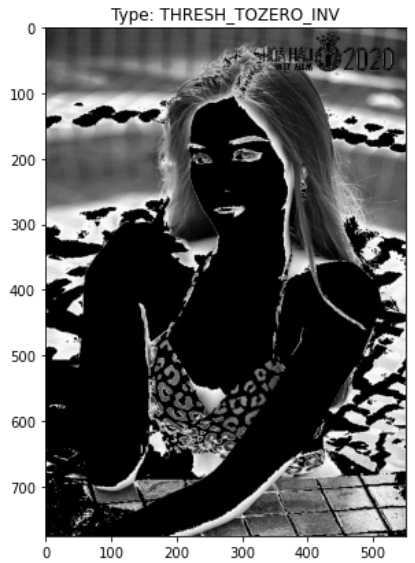










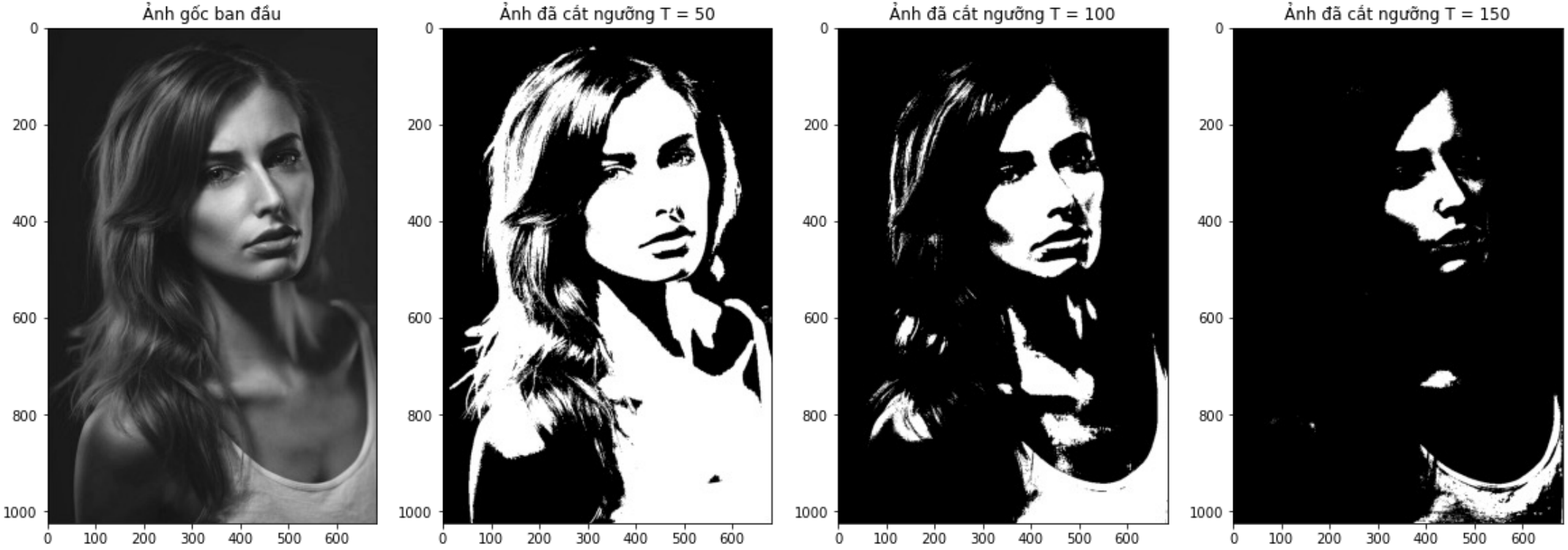


Thực hành số 2.6

Thực hành 2.6

Yêu cầu :

1. Sinh viên đọc ảnh images/Thuchanh2_6.jpg ở dạng xám
2. Thực hiện cắt ngưỡng ảnh với (Type = cv2.THRESH_BINARY) lần lượt với T = 50, 100, 150. Hiển thị kết quả như slide minh hoạ. Lưu ảnh vào thư mục Saves

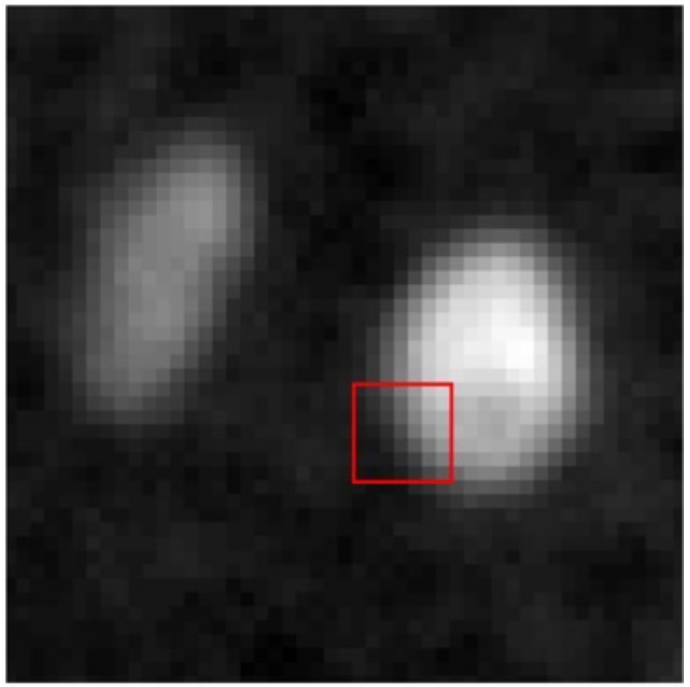


Thực hành 2.6

3. Thiết lập T = 75. Thực hiện cắt ngưỡng với Type = cv2.THRESH_TRUNC, cv2.THRESH_BINARY_INV, cv2.THRESH_TOZERO. Hiển thị kết quả như slide minh hoạ. Lưu ảnh vào thư mục Saves

3. Cân bằng sáng Histogram

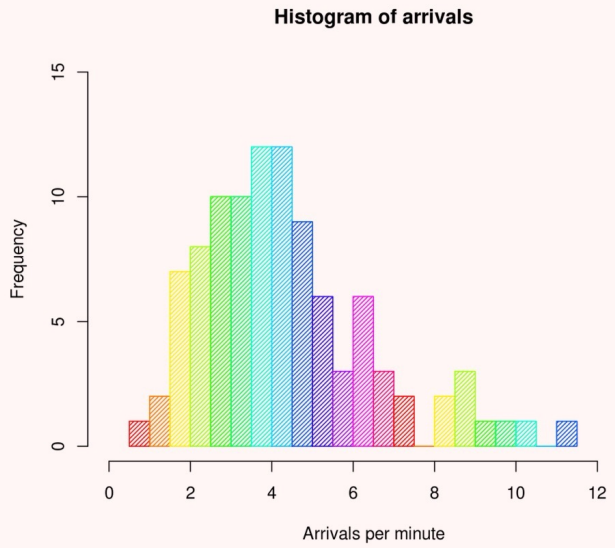
Histogram của ảnh là gì?



rk	54	56	58	59	60	61	64	66	68	72	73	75	79	81	83	86	88	92	96
h	1	1	1	2	1	1	3	1	1	1	2	1	1	1	2	1	1	2	1
p	0.04	0.04	0.04	0.08	0.04	0.04	0.12	0.04	0.04	0.04	0.08	0.04	0.04	0.04	0.08	0.04	0.04	0.08	0.04

61	66	75	86	96
59	64	73	83	92
59	64	73	83	92
56	60	68	79	88
54	58	64	72	81

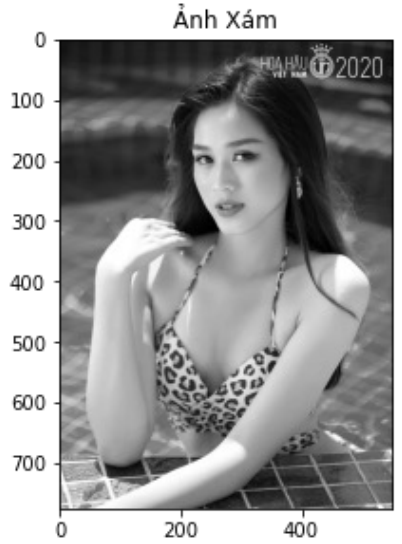
HISTOGRAM

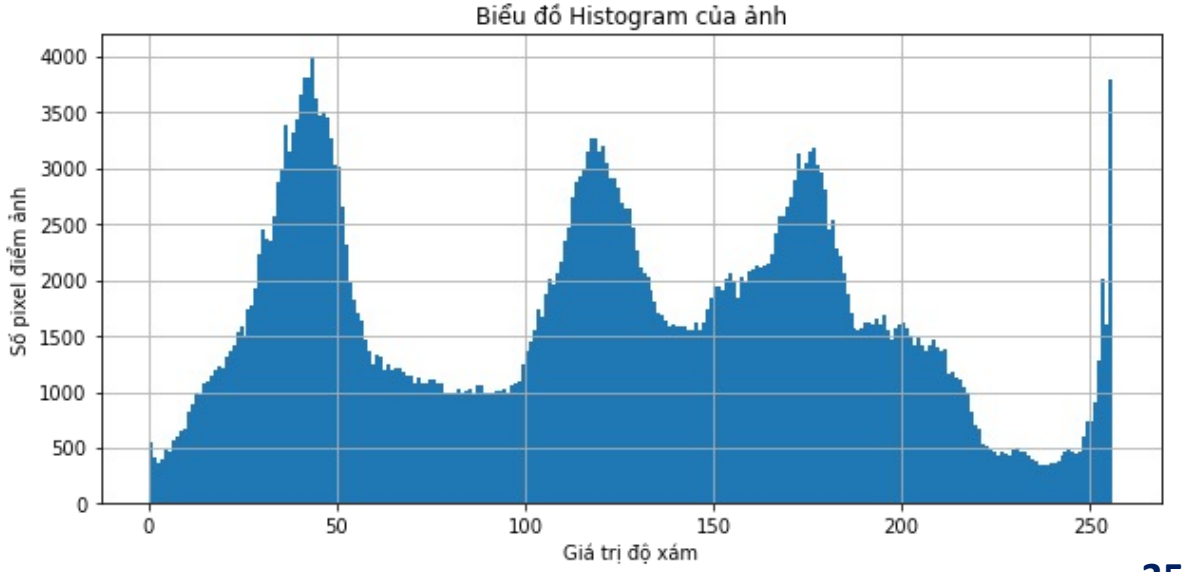


Biểu đồ tần suất

Histogram của ảnh là gì?

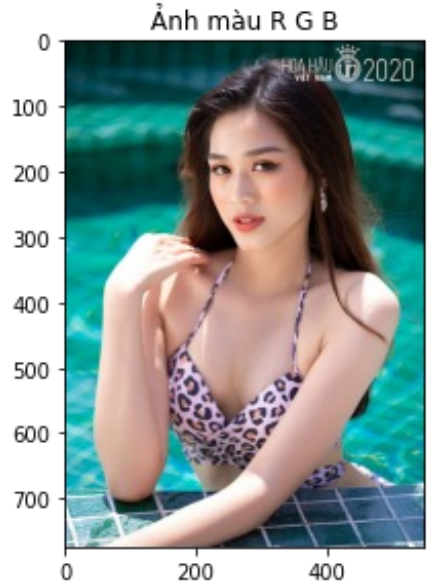
- Biểu đồ Histogram của ảnh là một dạng biểu đồ biểu diễn sự phân bố của số lượng điểm ảnh tương ứng với mức độ sáng tối của bức ảnh.
- Trong đó, trục dọc (y) biểu diễn số lượng điểm ảnh, các đỉnh càng cao thì càng có nhiều điểm ảnh ở khu vực đó và độ chi tiết càng nhiều.
- Trục ngang (x) tính từ trái qua phải với mốc giá trị từ 0 đến 255 biểu diễn độ sáng của mỗi khu vực ảnh. Gốc giá trị 0 được coi là tối nhất tựa như màu đen tuyền trong khi càng dịch sang phải giá trị này càng tăng, ngọn sáng nhất của ánh sáng ở giá trị 255.

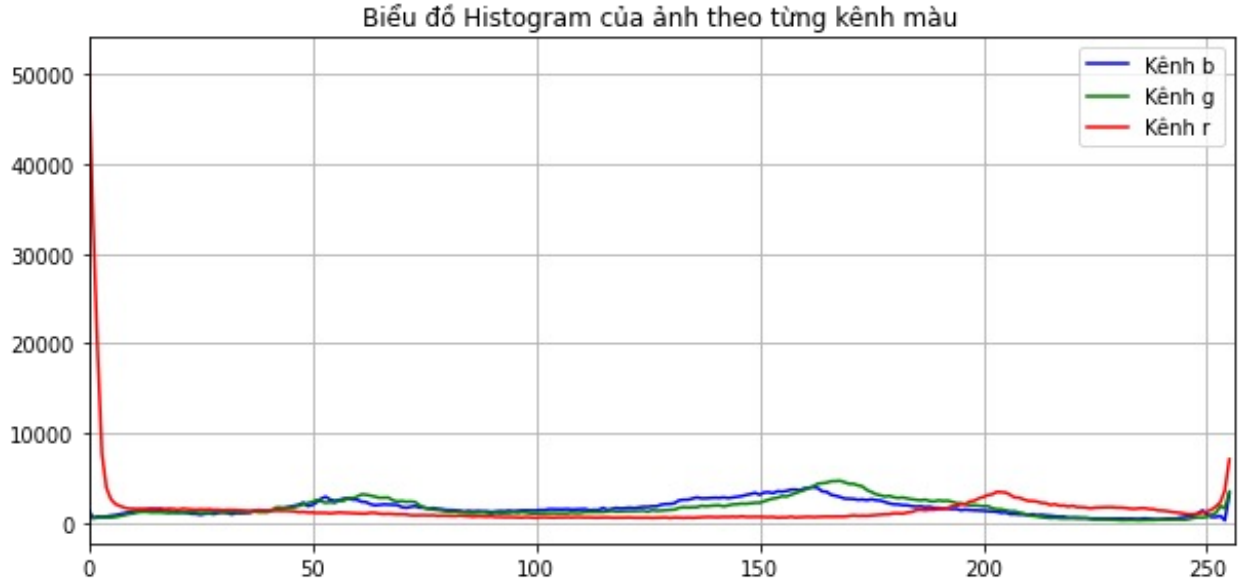




Histogram của ảnh là gì?

Với ảnh màu, Histogram sẽ biểu diễn cho từng kênh màu



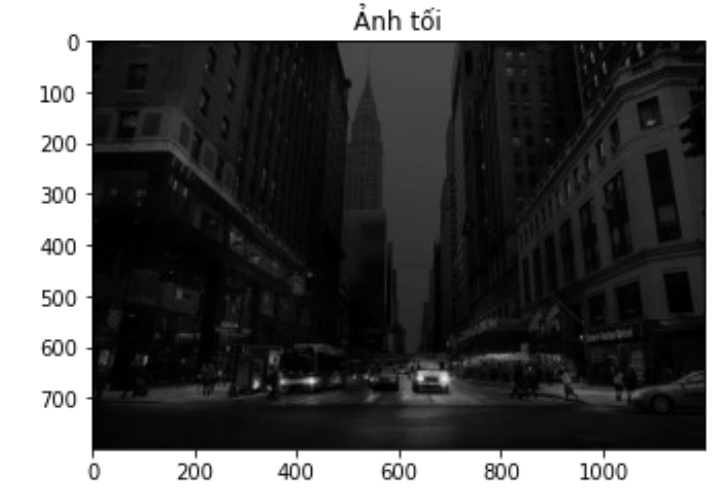


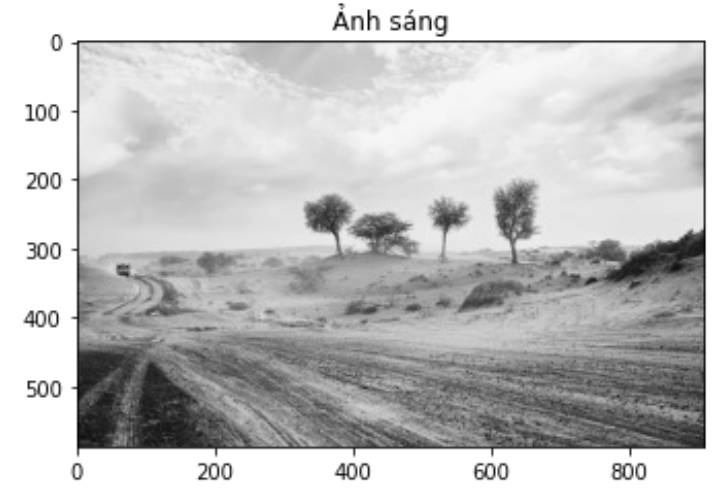
Histogram của ảnh là gì?

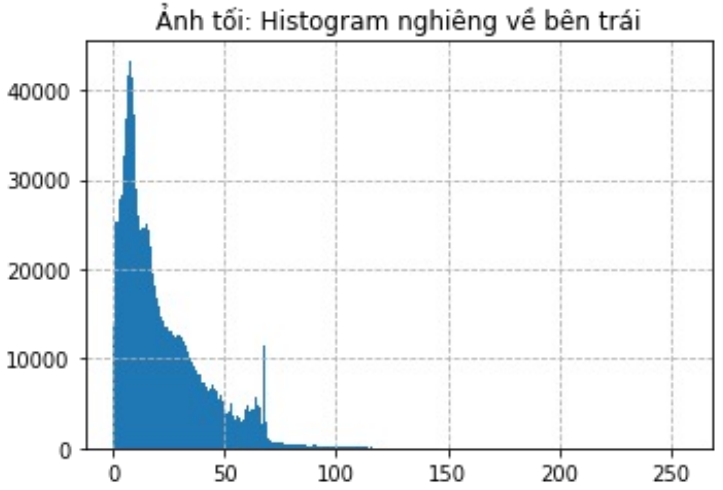
- Thông qua Histogram của ảnh có thể biết được ảnh sáng hay tối.

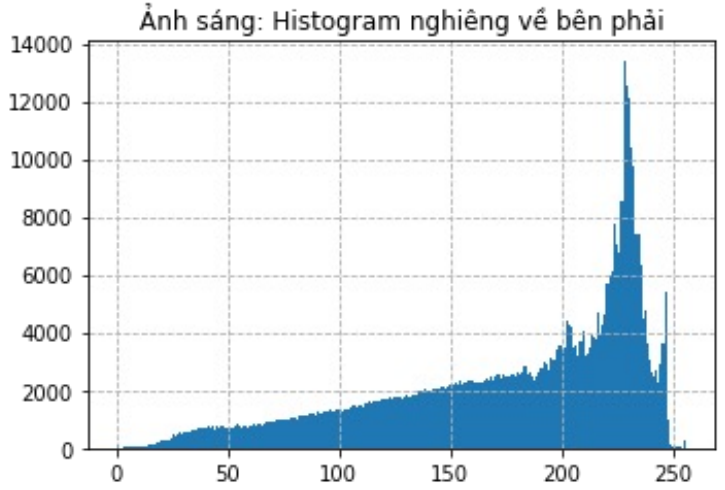
  - **Ảnh tối:** Histogram nghiêng về bên trái (Các pixel có giá trị thấp)

  - **Ảnh sáng:** Histogram nghiêng về bên phải (Các pixel có giá trị cao)

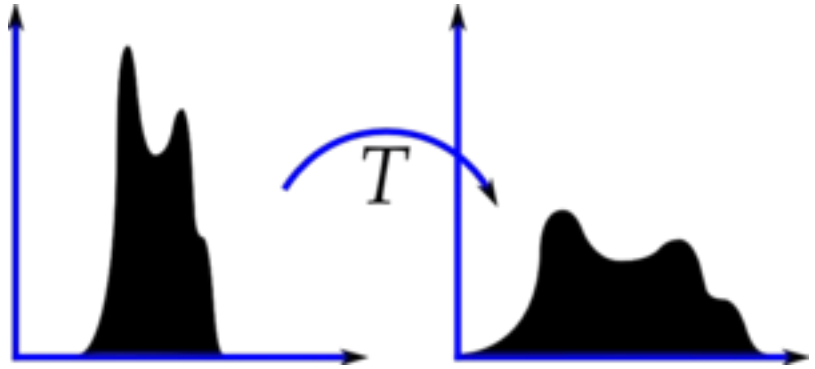




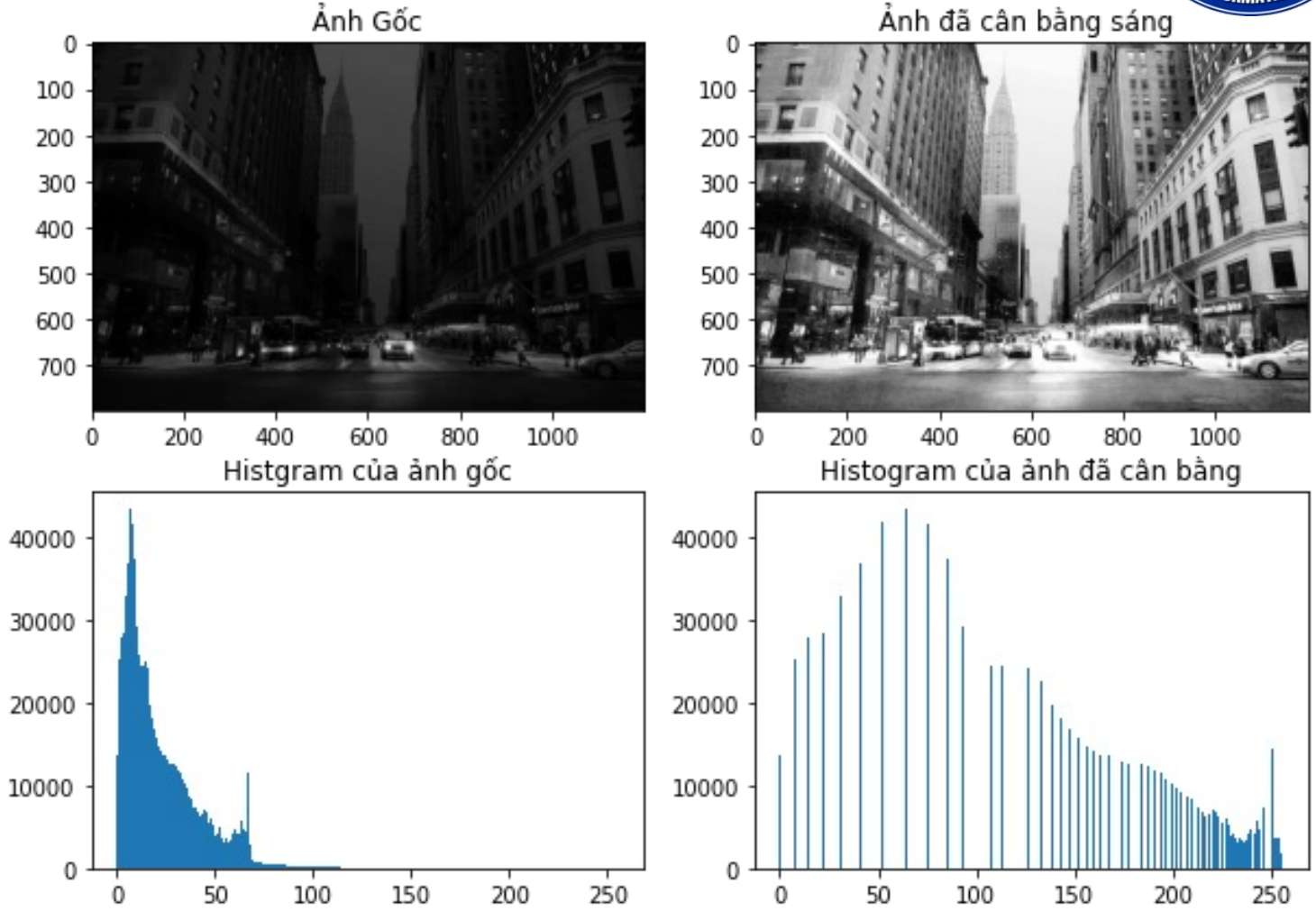




Cân bang sáng Histogram với OpenCV



- OpenCV hỗ trợ hàm cv2.equalizeHist() để cân bằng sáng.
- Trên ảnh xám, phương thức cv2.equalizeHist() thực hiện cân bằng trên toàn bộ ảnh.

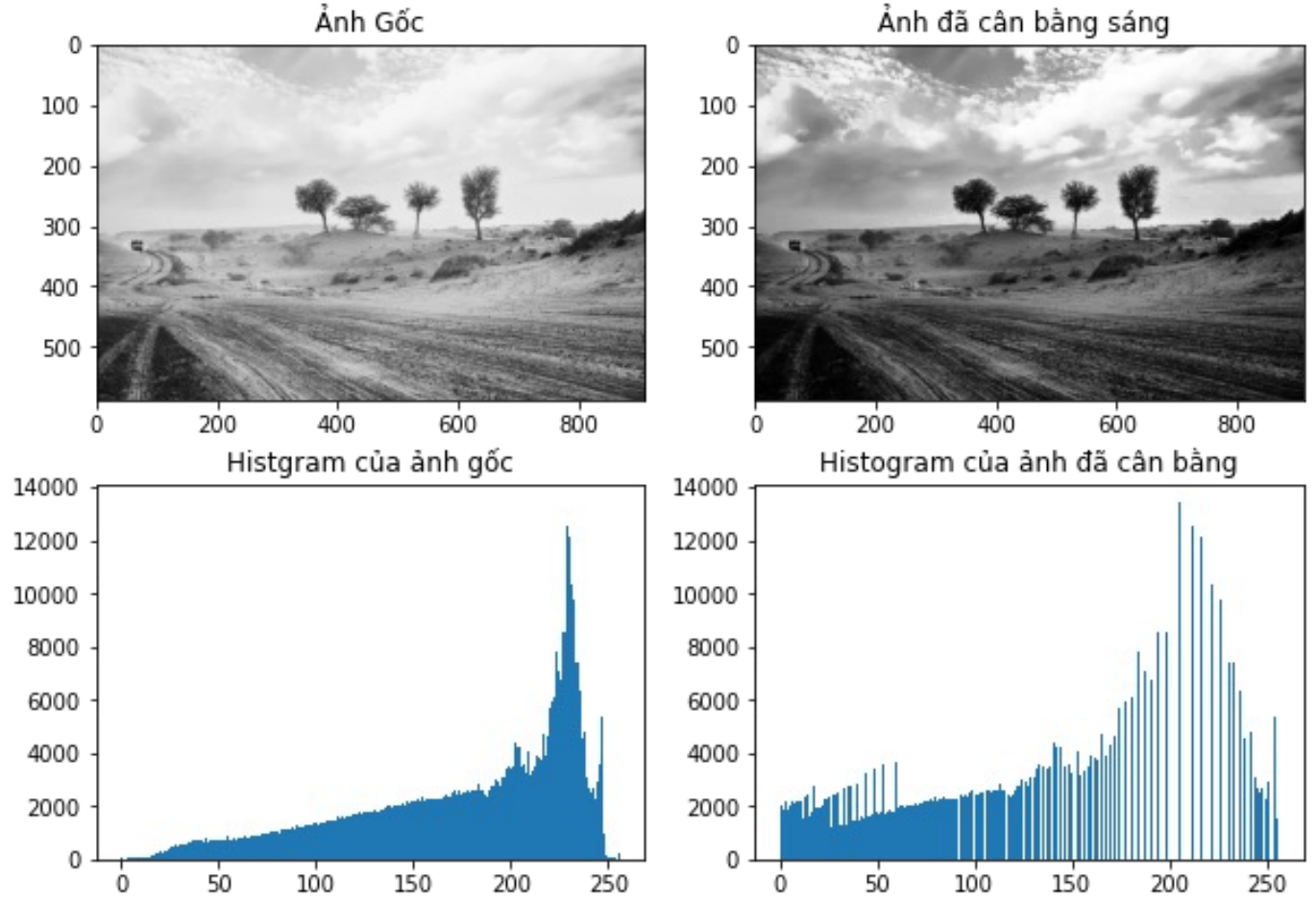


Kết quả thực hiện cân bằng sáng cho ảnh tối

Cân bang sáng Histogram với OpenCV

```python
#Đọc ảnh pic2:
img2 = cv2.imread('images/pic2.jpg', 0)

#Thực hiện cân bằng sáng với phương thức equalizeHist():
img_equ2 = cv2.equalizeHist(img2)

#Hiển thị ảnh và Histogram tương ứng.
#Ảnh 1
plt.figure(figsize=(10,7))
plt.subplot(2, 2, 1)
plt.imshow(img2, cmap='gray')
plt.title('Ảnh Gốc')

plt.subplot(2, 2, 3)
plt.hist(img2.ravel(),256,[0,256])
plt.title('Histgram của ảnh gốc')

#Ảnh 2:
plt.subplot(2, 2, 2)
plt.imshow(img_equ2, cmap='gray')
plt.title('Ảnh đã cân bằng sáng ')

plt.subplot(2, 2, 4)
plt.hist(img_equ2.ravel(),256,[0,256])
plt.title('Histogram của ảnh đã cân bằng')
plt.show()
```

Kết quả thực hiện cân bằng sáng cho ảnh sáng

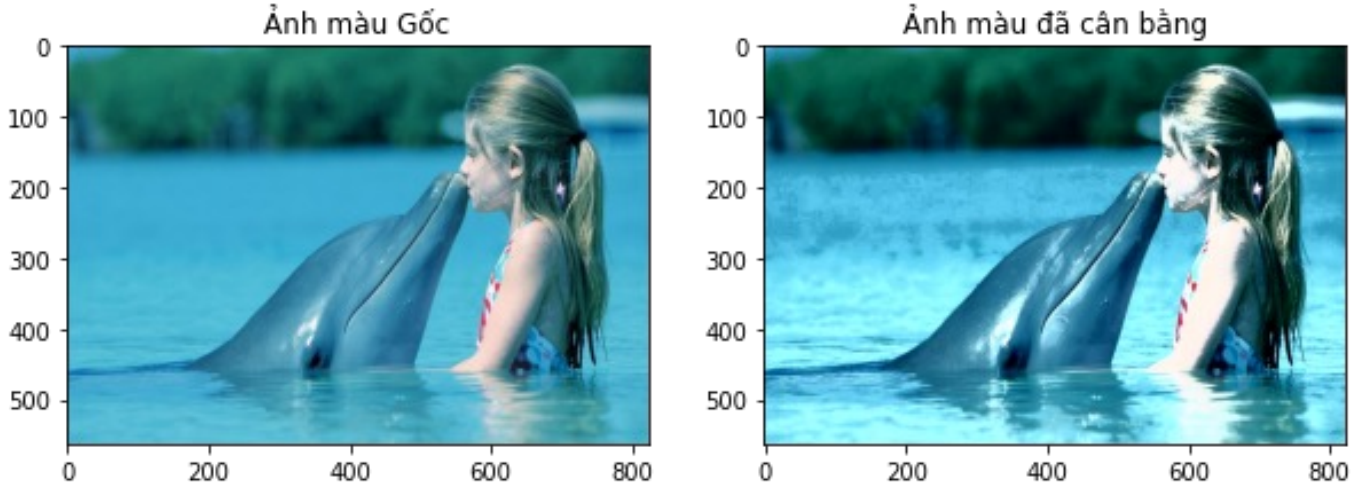


Cân bang sáng Histogram với OpenCV

Để cân bằng sáng trên ảnh màu:

- Convert sang không gian YUV, trong đó Y là độ sáng (luma hay brightness), U và V là kênh màu.
- Áp dụng equalizeHist trên kênh Y
- Convert trở về RGB (hoặc BGR tuỳ ảnh gốc)

```
#Đọc ảnh màu:
img_bgr = cv2.imread('images/fish.jpg')

#Chuyển đổi từ hệ màu BGR sang YUV:
img_yuv = cv2.cvtColor(img_bgr,cv2.COLOR_BGR2YUV)
#Cân bằng sáng trên kênh Y:
img_yuv[:,:,0] = cv2.equalizeHist(img_yuv[:,:,0])

#chuyển đổi ngược ảnh đã cân bằng từ hệ YUV  về BGR:
img_bgr2 = cv2.cvtColor(img_yuv,cv2.COLOR_YUV2BGR)

#Hiển thị ảnh gốc và ảnh đã cân bằng sáng:
plt.figure(figsize=(10,5))
plt.subplot(1, 2, 1)
plt.imshow(cv2.cvtColor(img_bgr,cv2.COLOR_BGR2RGB))
plt.title('Ảnh màu Gốc')

plt.subplot(1, 2, 2)
plt.imshow(cv2.cvtColor(img_bgr2,cv2.COLOR_BGR2RGB))
plt.title('Ảnh màu đã cân bằng')

plt.show()
```



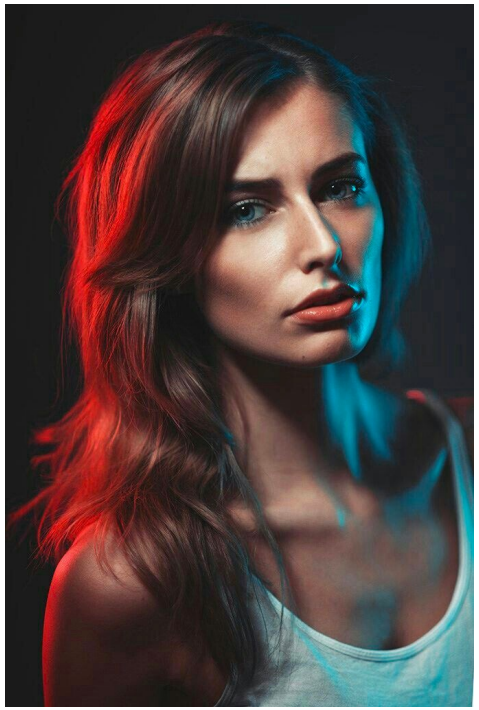
Không cân bằng Histogram cho ảnh nhị phân (đen trắng)

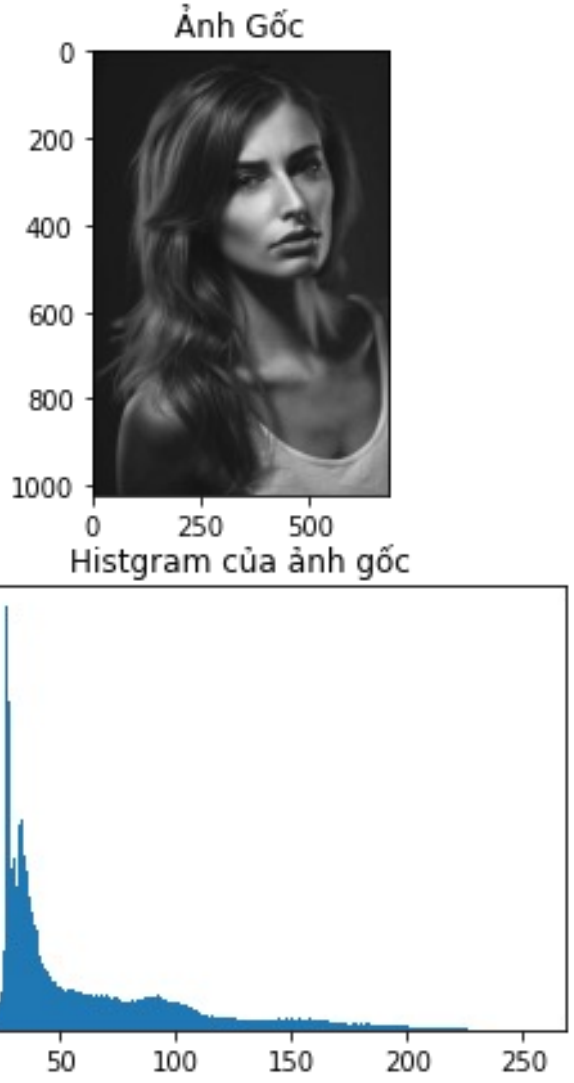
Thực hành số 2.7

Thực hành 2.7

Yêu cầu :

1. Sinh viên đọc ảnh images/Thuchanh2_6.jpg ở dạng ảnh xám. Hiển thị ảnh xám và Histogram của ảnh --> Dựa vào Histogram của ảnh cho biết ảnh này là sáng hay tối?

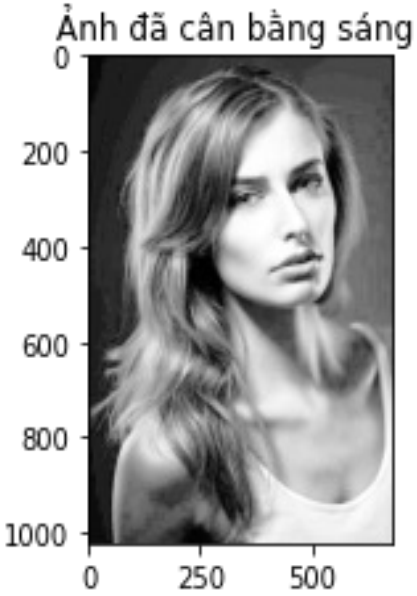


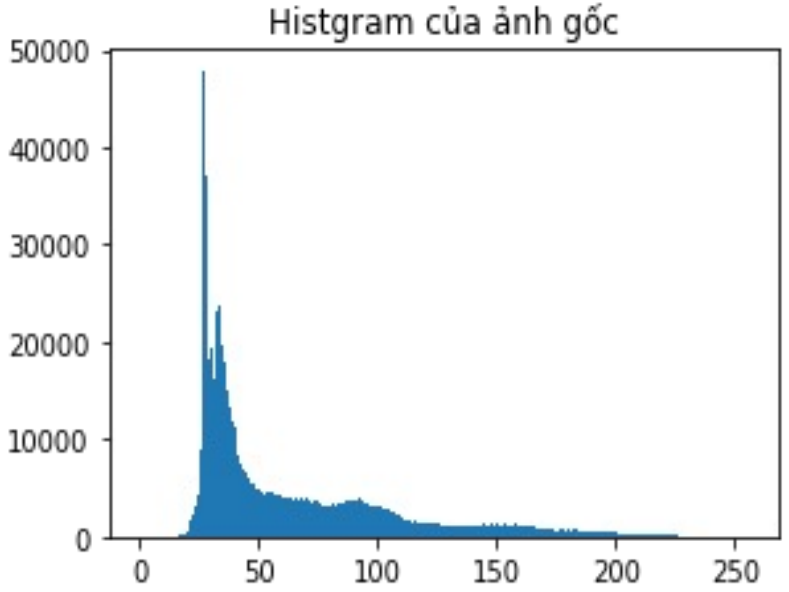


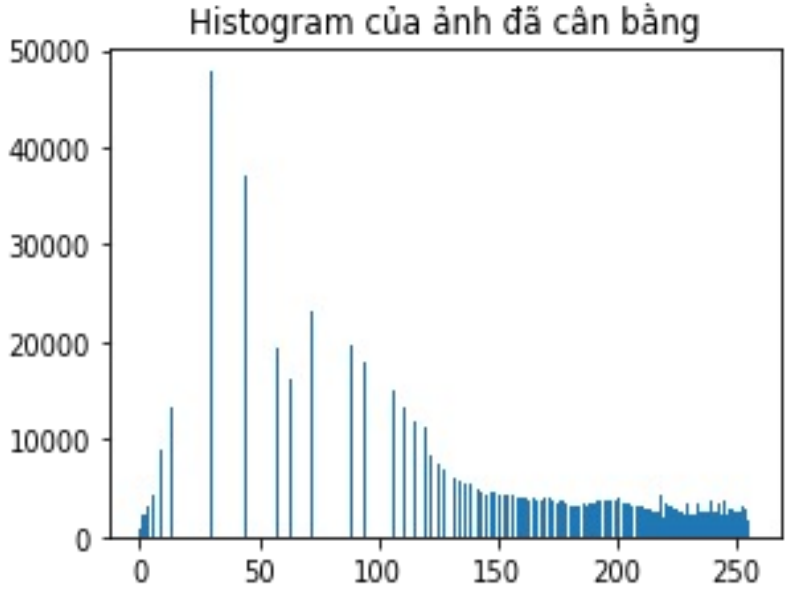
Thực hành 2.7

Yêu cầu :

2. Thực hiện cân bằng sáng trên ảnh Xám, hiển thị ảnh gốc và ảnh sau cần bằng cùng với Histogram tương ứng của mỗi ảnh.





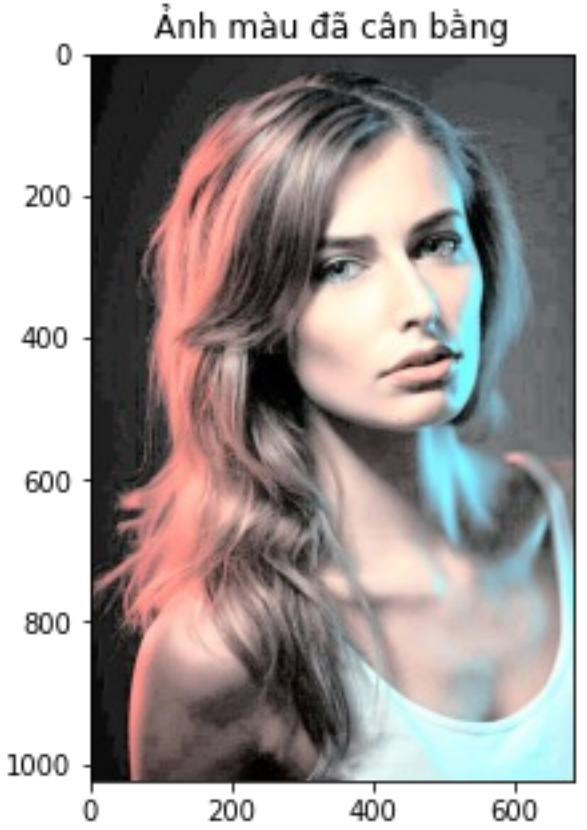


Thực hành 2.7

Yêu cầu :

3. Sinh viên đọc ảnh images/Thuchanh2_6.jpg ở dạng ảnh màu. Thực hiện cân bằng sáng trên ảnh màu, hiển thị ảnh gốc và ảnh sau cần bằng ở dạng ảnh màu.

4. Lưu ảnh xám và ảnh màu đã cân bằng vào thư mục images/Saves.




QUESTIONS
Q&A
ANSWERS