

Homework 2

Bài 1. Xử lý ảnh “Mammogram.bin” – Nhị phân hóa và phát hiện biên xấp xỉ

I. Mục tiêu

- Chuyển ảnh xám **Mammogram.bin** (256×256 , 8-bit) thành ảnh nhị phân bằng phương pháp thresholding đơn giản.
- Áp dụng Approximate Contour Generation để tạo ảnh biên xấp xỉ của vùng mô.
- Đánh giá khả năng dùng chain code để biểu diễn biên chính.

II. Lý thuyết

1. Nhị phân hóa (Thresholding):

Ảnh xám $f(x,y)$ được chuyển thành ảnh nhị phân $g(x,y)$ theo ngưỡng

Mục đích: tách nền tối và vùng mô sáng trong ảnh chụp X-quang.

2. Phát hiện biên xấp xỉ (Approximate Contour):

Dựa trên phép co (Erosion) trong hình thái học.

Biên xấp xỉ được tính bằng hiệu giữa ảnh nhị phân ban đầu và ảnh sau khi co

Chain code:

Là phương pháp mã hóa biên dạng theo chuỗi hướng (0–7) trong lân cận 8 điểm.

Phù hợp để biểu diễn các đường biên liên tục và khép kín.

III. Các bước thực hiện

1. Đọc ảnh nhị phân thô `Mammogram.bin` bằng `numpy.fromfile()` và reshape thành 256×256 .
2. Chuẩn hóa dữ liệu về dạng 8-bit nếu cần.
3. Chọn ngưỡng $T=70$ ($T=70$ (xác định bằng quan sát histogram)).
4. Tạo ảnh nhị phân: giá trị $\geq 70 \rightarrow 255$ (vùng mô), ngược lại $\rightarrow 0$ (nền).
5. Thực hiện phép Erosion 3×3 bằng `cv2.erode()` để co vùng sáng.
6. Tính ảnh biên: `Contour = Binary - Eroded`.
7. Lưu và hiển thị kết quả:
 - o Ảnh gốc
 - o Ảnh nhị phân
 - o Ảnh biên xấp xỉ

IV. Kết luận

- Ảnh biên thu được thể hiện rõ ranh giới vùng mô.
- Có thể dùng chain code để biểu diễn biên vì đường bao là liên tục, khép kín và đơn giá trị.

```

homework_2_b.py x homework_2_3.py x homework_2_4.py x
1 # Obtain the image "johnny.bin" from the course web site. This image has 256x256
2 # pixels. Each pixel has 8 bits. Plot the histogram of the original image. Write a program
3 # to perform histogram equalization on this image. Show the equalized image and plot
4 # its histogram.
5
6 import cv2
7 import numpy as np
8 import matplotlib.pyplot as plt
9
10 # -----
11 # (1) Load the binary image
12 # -----
13 filename = "johnny.bin"
14 img = np.fromfile(filename, dtype=np.uint8).reshape((256, 256))
15
16 # -----
17 # Save original image to visualize
18 cv2.imwrite("JohnnyOriginal.png", img)
19
20 # -----
21 # (2) Plot original histogram
22 plt.figure(figsize=(6,4))
23 plt.hist(img.ravel(), bins=256, range=(0,255), color='gray')
24 plt.title("Original Histogram")
25 plt.xlabel("Pixel Value")
26 plt.ylabel("Frequency")
27
28 python homework_2_1.py
(.venv)
x Sun 14 Sep - 23:16 ~/workspace/ImageProcessing [?] certain [?]
python E python homework_2_4.py
Warning: Ignoring XDG_SESSION_TYPE=wayland on Gnome. Use QT_QPA_PLATFORM=wayland to run on Wayland anyway.
(.venv)
Sun 14 Sep - 23:16 ~/workspace/ImageProcessing [?] certain [?]
python E python homework_2_1.py
Warning: Ignoring XDG_SESSION_TYPE=wayland on Gnome. Use QT_QPA_PLATFORM=wayland to run on Wayland anyway.
(.venv)
Sun 14 Sep - 23:16 ~/workspace/ImageProcessing [?] certain [?]
python E python homework_2_1.py
Warning: Ignoring XDG_SESSION_TYPE=wayland on Gnome. Use QT_QPA_PLATFORM=wayland to run on Wayland anyway.
(.venv)
Sun 14 Sep - 23:16 ~/workspace/ImageProcessing [?] certain [?]
python E python homework_2_1.py
Warning: Ignoring XDG_SESSION_TYPE=wayland on Gnome. Use QT_QPA_PLATFORM=wayland to run on Wayland anyway.
(.venv)
TERMINAL [?] master [?] term://bin/zsh

```

Bài 2. Xử lý ảnh “lady.bin” – Kéo giãn tương phản toàn phần

I. Mục tiêu

- Thực hiện Full-Scale Contrast Stretching trên ảnh **lady.bin**.
- So sánh histogram trước và sau khi kéo giãn.

II. Lý thuyết

1. Kéo giãn tương phản (Contrast Stretching):

Dùng để tăng độ tương phản bằng cách ánh xạ miền giá trị xám

$[r_{min}, r_{max}] \rightarrow [0, 255]$:

$$s = (r - r_{min})(r_{max} - r_{min}) \times 255 \\ s = \frac{(r - r_{min})(r_{max} - r_{min})}{(r_{max} - r_{min})} \times 255 \\ s = (r_{max} - r_{min}) \times 255$$

2. Histogram:

Biểu đồ phân bố giá trị xám của ảnh.

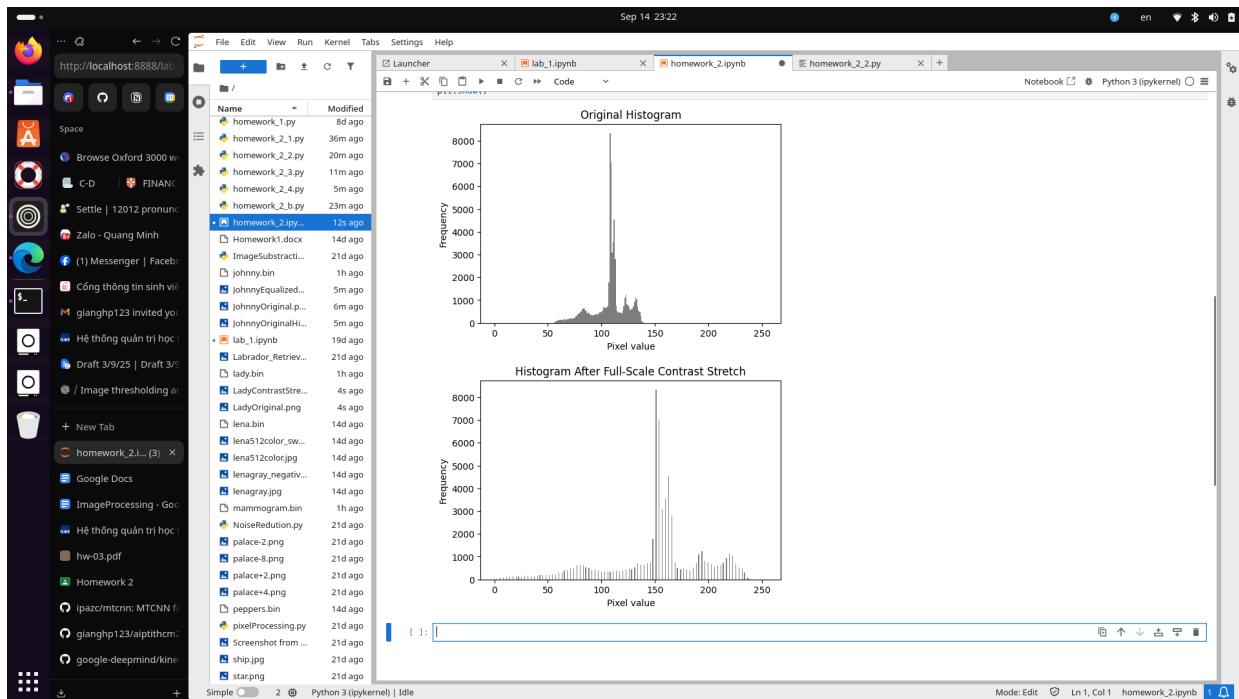
Sau khi kéo giãn, histogram được dàn đều hơn trên toàn miền 0–255.

III. Các bước thực hiện

1. Đọc ảnh `lady.bin` (256x256, 8-bit).
 2. Tính `rminr_{min}` và `rmaxr_{max}` của ảnh.
 3. Áp dụng công thức kéo giãn tương phản để tạo ảnh mới.
 4. Vẽ histogram trước và sau khi kéo giãn bằng `matplotlib`.
 5. Lưu ảnh kết quả và biểu đồ histogram.
-

IV. Kết luận

- Ảnh sau khi kéo giãn trở nên sáng hơn, các vùng tối/sáng phân biệt rõ.
- Histogram trải đều từ 0 đến 255, chứng tỏ ảnh đã đạt tương phản tối đa.



Bài 3. Xử lý ảnh “actontBin.bin” – Nhận dạng ký tự “T” bằng Binary Template Matching

I. Mục tiêu

- Phát hiện các vị trí chứa ký tự “T” trong ảnh nhị phân.
 - Ứng dụng thuật toán Binary Template Matching (M2 Measure).
-

II. Cơ sở lý thuyết

1. Binary Template Matching:

Dò tìm mẫu TTT bằng cách trượt cửa sổ mẫu trên ảnh.

Mức độ tương đồng tại mỗi vị trí được đo bằng chỉ số M2:

$$M2 = 1 - \sum(I \oplus T) / NM2 = 1 - \frac{\sum(I \oplus T)}{NM2} = 1 - N \sum(I \oplus T)$$

Trong đó \oplus là phép XOR, NNN là tổng số điểm trong mẫu.

Khi $M2$ gần 1 → vùng ảnh trùng khớp với mẫu.

2. Ngưỡng (Thresholding):

Sau khi tính $M2$, ta ngưỡng ảnh để tạo ảnh nhị phân thể hiện vị trí có ký tự “T”.

III. Các bước thực hiện

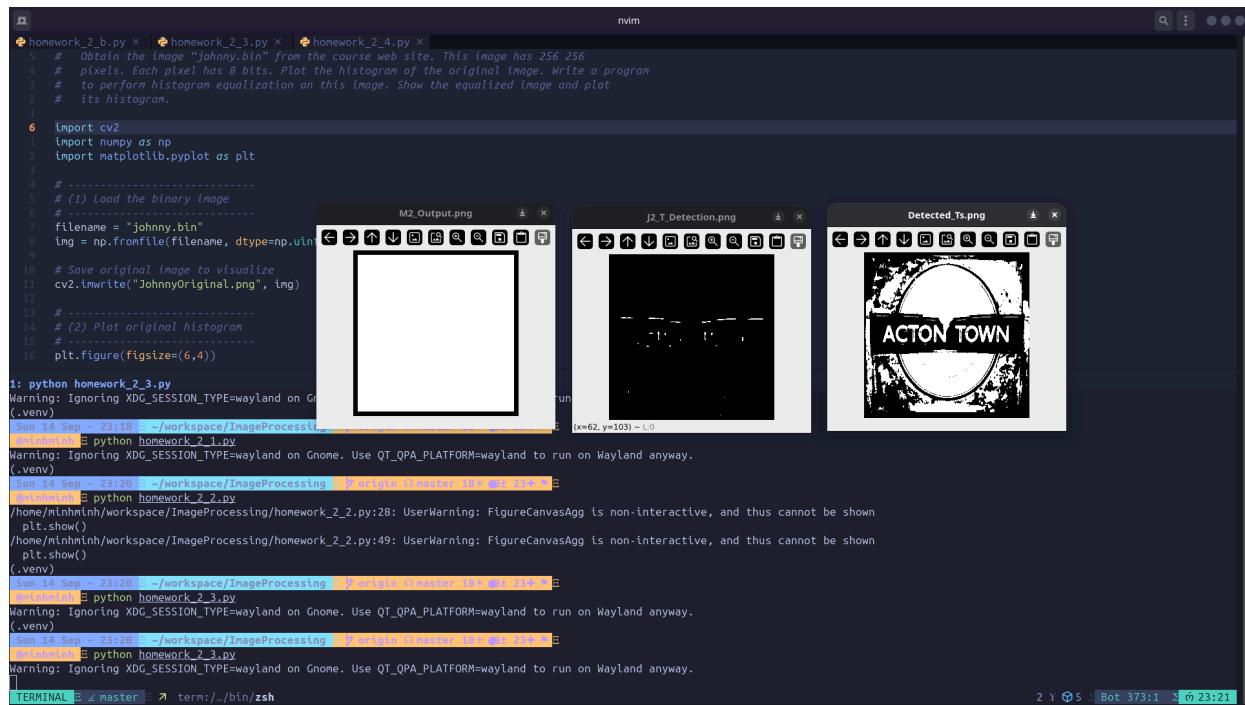
1. Đọc ảnh nhị phân `actontBin.bin` và chuẩn hóa về giá trị 0–1.
2. Tạo mẫu chữ “T” kích thước 15×15 gồm thanh ngang trên và thanh dọc giữa.
3. Trượt mẫu trên ảnh:
 - Tại mỗi vị trí, tính phép XOR giữa mẫu và vùng con.
 - Tính giá trị $M2 = 1 - (\text{tổng XOR} / N)$.
4. Gán kết quả $M2$ vào ma trận $J1$ tại vị trí trung tâm.
5. Dùng ngưỡng 0.8 để tạo ảnh nhị phân $J2$ (nơi có “T”).

6. Hiển thị và lưu các ảnh:

- **Ảnh M2 (mức tương đồng)**
- **Ảnh J2 (kết quả nhị phân)**
- **Ảnh đánh dấu các chữ “T” phát hiện được**

IV. Kết luận

- Phương pháp template matching cho kết quả chính xác khi kích thước mẫu phù hợp.
- M2 cao (≈ 1) tại vị trí có chữ “T”, thấp hơn tại vùng khác.
- Có thể mở rộng để phát hiện các ký tự khác bằng cách thiết kế template tương ứng.



The screenshot shows a terminal window with the following content:

```
# homework_2_b.py X homework_2_3.py X homework_2_4.py
# Obtain the image "johnny.bin" from the course web site. This image has 256 256 pixels. Each pixel has 8 bits. Plot the histogram of the original image. Write a program to perform histogram equalization on this image. Show the equalized image and plot its histogram.
# (1) Load the binary image
filename = "johnny.bin"
img = np.fromfile(filename, dtype=np.uint8)
# Save original image to visualize
cv2.imwrite("JohnnyOriginal.png", img)
# (2) Plot original histogram
plt.figure(figsize=(6,4))
1: python homework_2_3.py
Warning: Ignoring XDG_SESSION_TYPE=wayland on Gnome. Use QT_QPA_PLATFORM=wayland to run on Wayland anyway.
(.venv)
Sun 14 Sep - 23:18 ~-/workspace/ImageProcessing E python homework_2_1.py
Warning: Ignoring XDG_SESSION_TYPE=wayland on Gnome. Use QT_QPA_PLATFORM=wayland to run on Wayland anyway.
(.venv)
Sun 14 Sep - 23:20 ~-/workspace/ImageProcessing E ./original_Disaster_10k@1:234 E
(.venv)
/home/minhminh/workspace/ImageProcessing/homework_2_2.py:28: UserWarning: FigureCanvasAgg is non-interactive, and thus cannot be shown
plt.show()
/home/minhminh/workspace/ImageProcessing/homework_2_2.py:49: UserWarning: FigureCanvasAgg is non-interactive, and thus cannot be shown
plt.show()
(.venv)
Sun 14 Sep - 23:20 ~-/workspace/ImageProcessing E ./original_Disaster_10k@1:234 E
(.venv)
/home/minhminh/workspace/ImageProcessing/homework_2_3.py:28: UserWarning: FigureCanvasAgg is non-interactive, and thus cannot be shown
plt.show()
/home/minhminh/workspace/ImageProcessing/homework_2_3.py:49: UserWarning: FigureCanvasAgg is non-interactive, and thus cannot be shown
plt.show()
(.venv)
Sun 14 Sep - 23:20 ~-/workspace/ImageProcessing E ./original_Disaster_10k@1:234 E
(.venv)
/home/minhminh/workspace/ImageProcessing/homework_2_4.py:28: UserWarning: FigureCanvasAgg is non-interactive, and thus cannot be shown
plt.show()
/home/minhminh/workspace/ImageProcessing/homework_2_4.py:49: UserWarning: FigureCanvasAgg is non-interactive, and thus cannot be shown
plt.show()
(.venv)
TERMINAL E master :2 term:./bin/zsh
```

Three windows are displayed side-by-side:

- M2_Output.png: A grayscale image of a person's face.
- J2_T_Detection.png: A binary (black and white) image showing detected T-shaped regions.
- Detected_Ts.png: A grayscale image of a circular logo with the text "ACTON TOWN" and several T-shaped markers overlaid.

Bài 4. Xử lý ảnh “johnny.bin” – Cân bằng Histogram

I. Mục tiêu

- Áp dụng Histogram Equalization cho ảnh `johnny.bin` để cải thiện độ tương phản.
 - So sánh histogram trước và sau khi cân bằng.
-

II. Cơ sở lý thuyết

1. Histogram Equalization:

Là kỹ thuật biến đổi giá trị xám để phân bố histogram đồng đều hơn, giúp ảnh có độ tương phản cao hơn.

Công thức biến đổi:

$s=T(r)=(L-1)\sum_{j=0}^r p_r(j)$ $T(r) = (L-1) \sum_{j=0}^r p_r(j)$ $s=T(r)=(L-1)\sum_{j=0}^r p_r(j)$ với $p_r(j)$ là xác suất xuất hiện mức xám r , $L=256$.

2. Lợi ích:

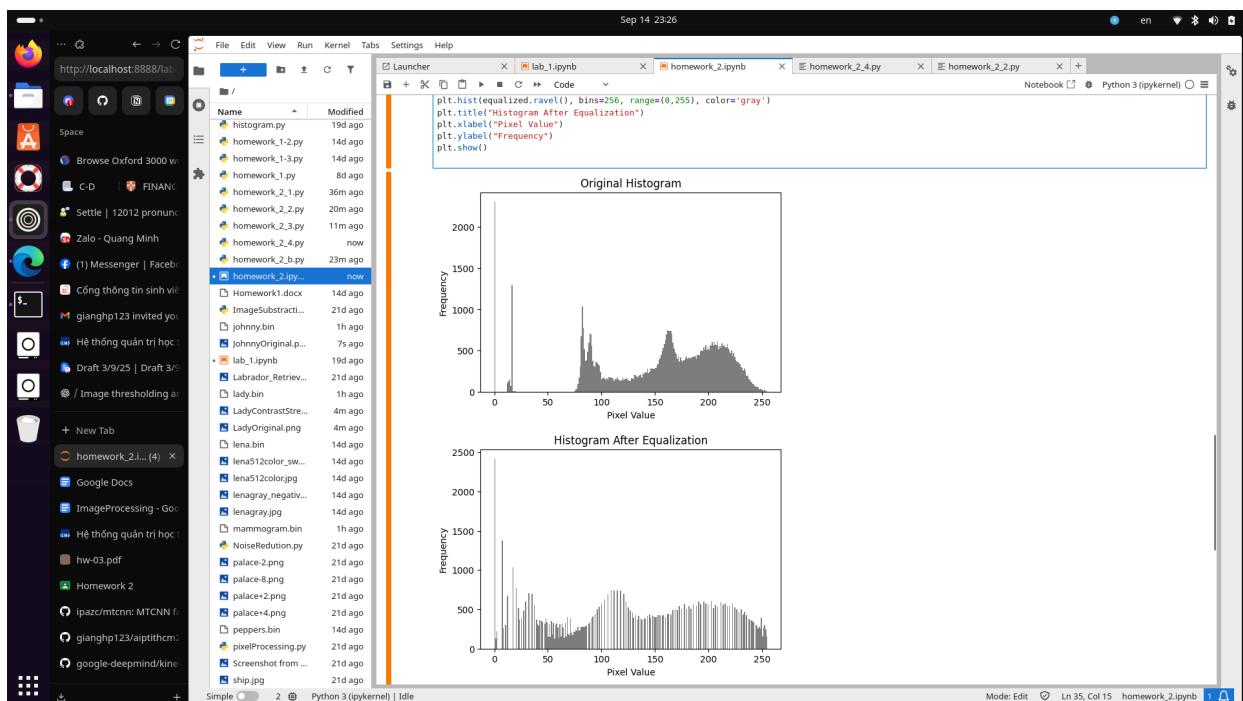
- Tăng cường chi tiết ở vùng tối/sáng.
 - Cân bằng phân bố mức xám trên toàn ảnh.
-

III. Các bước thực hiện

1. Đọc ảnh `johnny.bin` (256×256 , 8-bit).
2. Vẽ histogram ban đầu bằng `matplotlib`.
3. Dùng hàm `cv2.equalizeHist()` để cân bằng histogram.
4. Hiển thị ảnh đã cân bằng và vẽ lại histogram sau khi xử lý.
5. Lưu các ảnh kết quả:
 - `JohnnyOriginal.png`

- **JohnnyEqualized.png**
 - **JohnnyOriginalHistogram.png**
 - **JohnnyEqualizedHistogram.png**

A screenshot of a Jupyter Notebook interface. The left sidebar shows a file tree with various Python files and image files like 'lena.png' and 'lena.bin'. The main area has tabs for 'Launcher', 'lab_1.ipynb', 'homework_2.ipynb', and 'homework_2_4.py'. The 'homework_2.ipynb' tab is active, displaying Python code for histogram equalization. The code uses cv2 to load 'johnny.jpg', equalize its histogram, and then save the result as 'johnnyEqualized.png'. It also plots histograms before and after equalization. A preview window shows the original grayscale image of a man's face and the resulting equalized version. A warning message at the bottom indicates that XDG_SESSION_TYPE=wayland is being ignored.



IV. Kết luận

- Histogram sau khi cân bằng phân bố đều hơn, độ tương phản ảnh cải thiện rõ rệt.
- Phù hợp cho các ảnh có dài cường độ hẹp hoặc bị tối/sáng cục bộ.