

Homework 1

1 — Làm việc với file .bin (256 × 256, 8 bits/pixel)

Mục tiêu

- (a) Đọc và hiển thị hai ảnh `lena.bin` và `peppers.bin`.
 - (b) Tạo ảnh J có nửa trái từ Lena và nửa phải từ Peppers.
 - (c) Tạo ảnh K bằng cách hoán đổi nửa trái/phải của J.
 - (d) Nộp mã và printouts (ảnh gốc, J, K).
- File `.bin` chứa giá trị gray-level liên tiếp, 1 byte/pixel. Đọc toàn bộ file vào 1D numpy array rồi reshape thành (256,256).
 - Kiểm tra kích thước file: phải bằng $256 \times 256 = 65536$ bytes.

Các bước chi tiết

1. Đọc file .bin

- Dùng `np.fromfile(path, dtype=np.uint8)` để đọc tất cả byte.
- Kiểm tra `arr.size == 256*256` — nếu khác, báo lỗi.
- Chuyển `arr.reshape((256,256))`.

2. Tạo ảnh J

- Khởi tạo `J = np.zeros_like(image1)` (cùng kích thước, dtype).
- Gán `J[:, :128] = lena[:, :128]` (cột 0..127)
- Gán `J[:, 128:] = peppers[:, 128:]` (cột 128..255)
- Lưu ý: indexing của NumPy 0-based, cột 128 là cột thứ 129 theo đề.

3. Tạo ảnh K (swap halves)

- Khởi tạo `K = np.zeros_like(J)`
- `K[:, :128] = J[:, 128:]`
- `K[:, 128:] = J[:, :128]`

4. Hiển thị ảnh

- Dùng `cv2.imshow(window_name, image)` cho mỗi ảnh.

- `cv2.waitKey(0)` để đợi phím, `cv2.destroyAllWindows()` để đóng cửa sổ.
- **ĐỂ in vào báo cáo:** lưu ảnh bằng `cv2.imwrite("lena.png", lena)` v.v. (PNG giữ chất lượng, tốt hơn JPEG cho ảnh đồ họa).

5. Ghi chú về lưu file

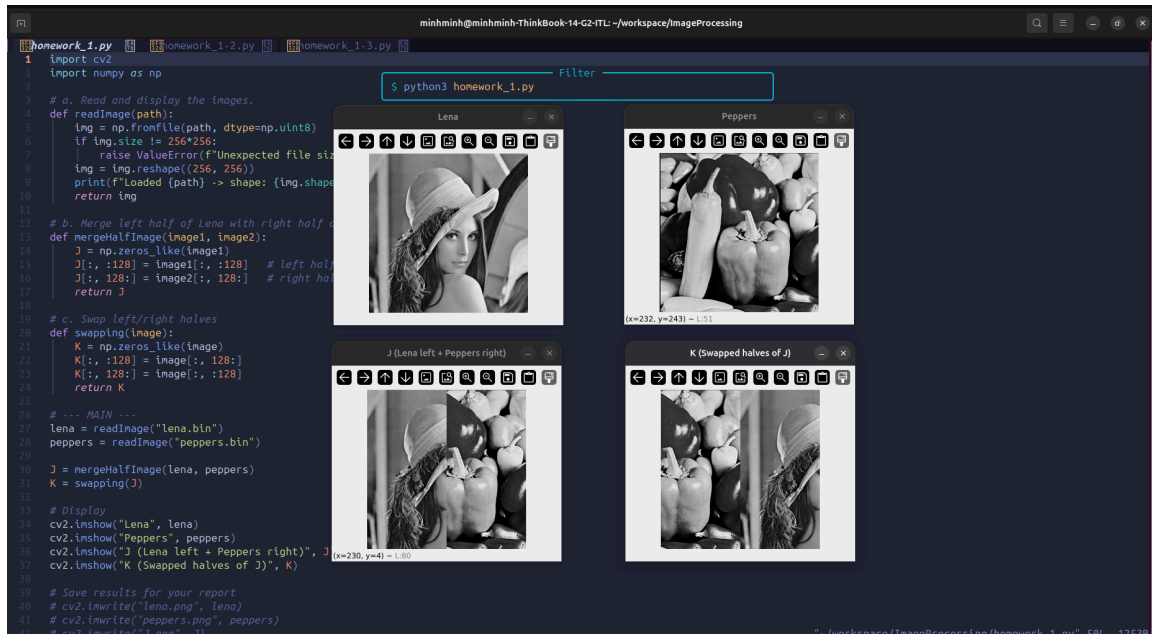
- `np.fromfile` đọc theo byte — không cần lo về endianness vì mỗi pixel 1 byte.
- Nếu file `.bin` có header, phương pháp này sai — kiểm tra kích thước.

Giải thích đoạn mã chính (ý chính)

- `readImage(path)`:
 - đọc file, kiểm tra kích thước, reshape.
- `mergeHalfImage(image1, image2)`:
 - tạo ảnh mới J bằng nửa trái `image1` và nửa phải `image2`.
- `swapping(image)`:
 - hoán đổi hai nửa của ảnh.

Kiểm tra & lưu kết quả

- Lưu cả 4 ảnh: `lena.png`, `peppers.png`, `J.png`, `K.png`.



2 — Ảnh grayscale lenagray.jpg (phóng ảnh âm bản)

Mục tiêu

- Đọc ảnh grayscale (J1) và tạo ảnh âm bản $J2 = 255 - J1$.
- Hiển thị và lưu kết quả.
- Với ảnh 8-bit grayscale, đảo sáng $= 255 - \text{pixel}$.
- Đảm bảo ảnh đọc bằng flag `cv2.IMREAD_GRAYSCALE`.

Các bước chi tiết

1. Đọc ảnh:

- `J1 = cv2.imread("lenagray.jpg", cv2.IMREAD_GRAYSCALE)`
- Kiểm tra `if J1 is None: raise FileNotFoundError(...)`

2. Tạo negative:

- `J2 = 255 - J1` (NumPy broadcast, dtype uint8 -> kết quả vẫn uint8)

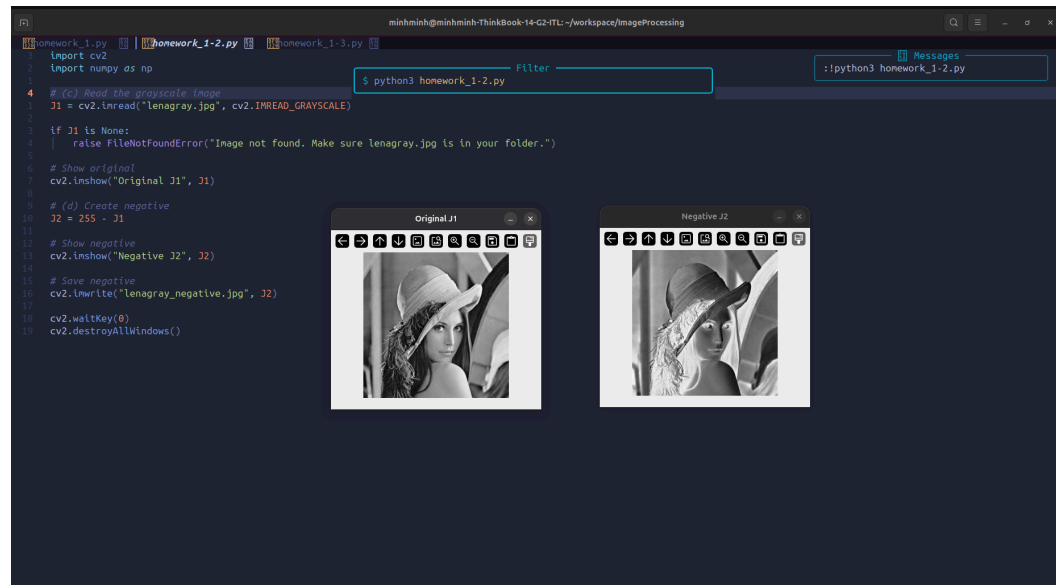
3. Hiển thị & lưu:

- `cv2.imshow("Original J1", J1)`
- `cv2.imshow("Negative J2", J2)`
- `cv2.imwrite("lenagray_negative.jpg", J2)` (hoặc png nếu muốn không nén)

4. Ghi chú:

- Nếu J1 không phải uint8, convert `J1 = J1.astype(np.uint8)` trước khi tính toán.

- 255 - J1 giữ nguyên dtype là uint8 (trừ khi J1 là khác).



3 — Ảnh màu lena512color.jpg (hoán đổi kênh màu)

Mục tiêu

- Đọc ảnh màu 512×512, tạo ảnh J2 bằng cách:
 - Red(J2) = Blue(J1)
 - Green(J2) = Red(J1)
 - Blue(J2) = Green(J1)

Các bước chi tiết

1. **Đọc ảnh:**
 - J1 = cv2.imread("lena512color.jpg", cv2.IMREAD_COLOR)
 - Kiểm tra if J1 is None: raise FileNotFoundError(...)
2. **Khởi tạo J2:**
 - J2 = np.zeros_like(J1) (giữ shape và dtype)
3. **Gán kênh:**
 - Theo đề: Red(J2)=Blue(J1) → trong OpenCV: J2[:, :, 2] = J1[:, :, 0]
 - Green(J2)=Red(J1) → J2[:, :, 1] = J1[:, :, 2]
 - Blue(J2)=Green(J1) → J2[:, :, 0] = J1[:, :, 1]
4. **Hiển thị & lưu:**
 - cv2.imshow("Original J1", J1)

- `cv2.imshow("Modified J2 (Swapped Bands)", J2)`
- `cv2.imwrite("lena512color_swapped.jpg", J2)` (lưu JPEG hoặc PNG)

5. Ghi chú:

- Nếu muốn xuất file mà giữ màu chuẩn khi mở bằng các công cụ khác (Pillow, trình duyệt), dùng PNG.
- Kiểm tra kích thước `J1.shape == (512, 512, 3)`.

