

VNUHCM - UNIVERSITY OF SCIENCE
FACULTY OF INFORMATION TECHNOLOGY



Xử lý ảnh số và video số

BÀI TẬP LỚP NHÀ

Giảng viên:

Thầy Lý Quốc Ngọc

Thầy Nguyễn Mạnh Hùng

Sinh viên:

Thầy Phạm Thanh Tùng

Trần Hải Đức

23127173

Trần Hoàng Nam

23127232

Nguyễn Công Chiến

23127331

Mục lục

I. Chương 1: Giới thiệu chung.....	1
II. Chương 2: Khái niệm cơ bản về xử lý ảnh.....	8
III. Chương 3: Biến đổi màu.....	9
IV. Chương 4: Biến đổi hình học	11
V. Chương 5: Làm trơn ảnh	19
VI. Chương 6: Phát hiện biên cạnh	35
VII. Chương 7: Phép biến đổi Fourier	40

I. Chương 1: Giới thiệu chung

1. Tại sao cần thông minh hóa trong quy trình sản xuất và quản lý xã hội

- Tăng năng suất và hiệu quả sản xuất
 - AI và robot giúp tự động hoá các công đoạn lặp lại, giảm sai sót và thời gian chết.
 - Hệ thống thông minh tối ưu hoá lịch trình, nguyên vật liệu và năng lượng sử dụng.
- Giảm chi phí vận hành
 - Bằng cách dự đoán bảo trì (predictive maintenance), phát hiện lỗi sớm → giảm hỏng hóc máy móc.
 - Tiết kiệm nhân công và nguyên liệu nhờ phân tích dữ liệu chính xác.
- Nâng cao chất lượng sản phẩm
- Cảm biến và thị giác máy tính kiểm tra sản phẩm theo thời gian thực → phát hiện lỗi nhỏ mà con người khó thấy.
 - AI điều chỉnh tự động thông số dây chuyền để đảm bảo chất lượng ổn định.
- Thích ứng nhanh với thị trường
 - Phân tích dữ liệu nhu cầu thị trường giúp điều chỉnh kế hoạch sản xuất linh hoạt hơn.
 - Tăng khả năng “mass customization” (sản xuất cá nhân hoá với số lượng lớn).

Hai vấn đề cốt lõi cần “máy móc” để giải quyết đó là cải thiện quy trình sản xuất và quản lý xã hội → Tại sao cần thông minh hóa trong quy trình sản xuất và quản lý xã hội → Cần thiết trong việc thông minh hóa.

- Cần thông minh hóa trong quy mô diện rộng và lấy AI làm cốt lõi, ví dụ:
 - Các công nghệ của các nước phát triển cần thích nghi được với các nước kém phát triển hơn để buôn bán các “sản phẩm”
 - AI giúp việc xây dựng mô hình nhanh chóng hơn, nhưng đặc biệt hơn cả là thúc đẩy việc tạo sinh các dữ liệu mới

2. Thị giác ảnh hưởng như thế nào đến quy trình sản xuất và quản lý xã hội

- Đối với quy trình sản xuất thì thị giác hỗ trợ khâu thiết kế, kiểm tra sản phẩm, thu hoạch nông sản, logistic, ...
- Đối với quản lý xã hội có thể kể đến quản lý các phương tiện giao thông (ví dụ CCTV) hoặc eKYC, ...

3. So sánh thiết bị số thu nhận thị giác (camera) với mắt

Điểm chung: để quay chụp một vật thể thì cả mắt và thiết bị thị giác đều cần ánh sáng.

Điểm khác:

- Mắt nhìn một vật dưới dạng ảnh ảo (bị ngược) và não bộ xử lý để ta nhìn vật như ảnh thật, trong khi đó thiết bị thị giác chỉ cần “thuật toán chụp” mà không cần quan xử lý gì là đã có tấm ảnh cần thiết.
- Với mắt, hay con người luôn có giới hạn. Mắt không thể xử lý liên tục với một khối lượng dữ liệu lớn tốt bằng máy tính. Qua đó, làm bật lên 3 điểm cộng của thiết bị thu nhận thị giác là: lưu trữ giữ liệu, xử lý dữ liệu lớn, tính chính xác khi xử lý dữ liệu.
- Tuy nhiên, điểm hạn chế của “các bức ảnh” là về các vấn đề chiều sâu, khi nhìn trực tiếp ta sẽ cảm quan các khoảng cách rõ ràng hơn. Khi mắt không nhìn thấy, con người có xu hướng “nheo” mắt để xử lý hình ảnh tốt hơn, trong khi đó hình ảnh thiên về xử lý hình ảnh để phóng to, thu nhỏ phù hợp.

4. Định nghĩa Color, Texture, Shape.

Color: Mỗi màu có một bước sóng khác nhau, ánh sáng chiếu vào vật, vật phản xạ lại vào mắt với bước sóng ánh sáng màu đó thì ta cảm giác màu đó.

Texture grain (vân): Tập hợp các điểm màu giống hoặc gần giống nhau thành một dải màu tạo thành vân → thể hiện sự phân bố của một tập hợp điểm theo không gian và màu sắc.

Shape: Tập hợp các điểm mà tại đó có biến thiên lớn về màu sắc. Ví dụ: Hình tròn là tập hợp các điểm cách tâm I một đoạn là R . Tương tự với nhận biết biên của người, nón, vật thể,...

5. Đồ họa máy tính, Xử lý ảnh, Thị giác máy tính khác nhau như thế nào?

Môn	Mục đích	Input	Output
Computer Graphic	Mô phỏng thế giới thực và ảo	Tập ảnh 2d, 3d + Entity	2d, 3d curve, suspect, animation

Digital Image Processing	Tăng cường chất liệu ảnh → ứng dụng	Ảnh, video	Ảnh, video đã được tăng cường
Computer Vision	Tăng cường khả năng hiểu ảnh, hiểu semantic	Ảnh, video	Semantic

6. Ví dụ cho AR

Giải trí / Game

- Pokémon GO: người chơi nhìn qua camera và thấy Pokémon xuất hiện trong thế giới thật.
- Snapchat / Instagram filters: bộ lọc thêm tai mèo, hiệu ứng mặt, trang điểm ảo,...
- AR Racing: đua xe ảo trên bàn thật (như game Hot Wheels AR).

Thương mại điện tử

- IKEA Place: xem trước đồ nội thất 3D trong phòng qua điện thoại.
- L'Oréal AR Makeup / Sephora
- Virtual Artist: thử son, phấn, tóc màu ngay trên khuôn mặt thật.
- Nike Fit: đo chân và thử giày bằng AR.

Giáo dục & Đào tạo

- Human Anatomy AR: học giải phẫu bằng mô hình 3D trên cơ thể người thật.
- AR Chemistry: hiển thị cấu trúc phân tử 3D, phản ứng hóa học trực quan.
- History AR Tours: xem các công trình cổ được tái hiện ngay tại di tích.

Công nghiệp & Kiến trúc

- AR Blueprint: kỹ sư xem bản thiết kế 3D chồng lên công trình đang xây.
- AR Remote Assist (Microsoft HoloLens): chuyên gia hướng dẫn từ xa bằng mô hình ảo.

7. Visual data understanding: Detection - Recognition - Classification - Tracking - Retrieval - Counting - Generating (khác biệt so với còn lại) - Semantation. Ví dụ cụ thể trong tình huống cảnh giác nguy hiểm trong trường.

Tác vụ	Goal	Input	Output	Ví dụ (quan sát đánh bắt cá tôm)

Detection	Xác định được đối tượng, bao gồm bounding box và specific id của object đó	Ảnh, Video	Bounding box, specific id	Detect cá, tôm, cua
Recognition	Nhận diện cụ thể một đối tượng	Ảnh, Video	Biết được specific id cụ thể là gì	Recognition con tôm A, B, C
Classification	Phân ra thành các lớp cụ thể, gồm nhiều specific id → class id	Ảnh, Video	Lớp đối tượng	Phân loại cá, tôm
Tracking	Theo dõi một đối tượng qua từng frame, đóng bounding box cho vật đang theo dõi	Ảnh, Video	Bounding box, vị trí của object đang theo dõi qua từng frames	Theo dõi một con cá có id là xyz
Retrieval	Tìm những tấm ảnh trong kho, có độ tương đồng với ảnh query	Ảnh query, kho ảnh	Các bức ảnh có độ tương đồng với retrieval, các bức ảnh có mức độ và thứ tự tương đồng	Truy xuất các bức ảnh cá hay ăn ở khu vực M, N, O
Counting	Đếm số phần tử	Ảnh, video	Số các object cần đếm	Đếm số con cá đang bơi
Generating	Tạo mới chứ không phải lấy lại những gì đã có như các tác vụ khác → đột phá	Text, Ảnh, Video,...	Các thông tin tạo sinh theo nhu cầu	Tạo ảnh cá bơi trong tối (tái tạo ảnh không hoàn thiện)

Segmentation	Phân lớp từng điểm ảnh, các điểm ảnh cùng tính chất → 1 cụm	Ảnh, video	Các cụm điểm ảnh chung tính chất	Coi cá, tôm có bị bệnh không, nếu có sẽ bị nổi đốm.
--------------	---	------------	----------------------------------	---

8. Tại sao phải sử dụng lidar thay vì ảnh?

Lidar lưu thông tin dạng khác, không chỉ màu mà còn có các thông số khác (ví dụ: hệ số phản xạ, density, location). Với con người nhìn vào thì thấy lốm đốm, mờ nhòe nhưng với máy tính đây là tập dữ liệu hình học chính xác để đo khoảng cách giữa vật thể, phát hiện vật cản, phân loại vật thể dựa trên hình dạng (bounding box 3D).

9. So sánh trí tuệ tự nhiên và trí tuệ nhân tạo.

Điểm mạnh	Điểm yếu
NI: sáng tạo, cảm xúc, thích nghi linh hoạt.	Chậm, mệt mỏi, giới hạn sinh học.
AI: nhanh, chính xác, xử lý dữ liệu khổng lồ.	Thiếu cảm xúc, thiếu ý thức, lệ thuộc vào dữ liệu.

10. Giải thích câu “Intelligent is the ability to adapt the change.”

Để thích ứng với sự thay đổi cần kiến thức (knowledge) và vận nó vào thực tế (practice) → thông minh, nhưng khó.

→ Kim chỉ nam định nghĩa cho thông minh.

11. Giải thích câu của Tom Mitchell: 1 program được gọi là học từ Experience E từ Task T với thang đo Measure P nếu T thực thi với thang đo P cải thiện E.

"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E."

Một chương trình được xem là “học” nếu nhờ có kinh nghiệm (E), nó thực hiện tốt hơn nhiệm vụ (T) được đo bằng một thước đo hiệu suất (P). Nói cách khác: Càng có nhiều kinh nghiệm → chương trình càng làm tốt nhiệm vụ → theo thước đo đánh giá cụ thể.

12. So sánh sự khác nhau giữa lập trình bình thường và lập trình trí tuệ nhân tạo

Tiêu chí	Lập trình truyền thống	Lập trình trí tuệ nhân tạo
Cách hoạt động	Lập trình viên viết quy tắc, công thức và logic cụ thể để máy thực hiện.	Lập trình viên thiết kế mô hình, cho dữ liệu học, máy tự rút ra quy tắc.
Đầu vào → Đầu ra	Input + Rules → Output	Input + Output (data) → Model → Rules (learned)
Ví dụ minh họa	Viết chương trình tính thuế: thuế = thu nhập * 10%.	Dạy máy nhận dạng khuôn mặt bằng hàng nghìn ảnh mẫu.
Dữ liệu	Dữ liệu chỉ là đầu vào phụ, không ảnh hưởng đến logic.	Dữ liệu là yếu tố cốt lõi, quyết định độ chính xác của mô hình.
Kết quả	Cố định, dựa đúng theo quy tắc lập trình.	Thay đổi, phụ thuộc vào dữ liệu học và mô hình.
Độ phức tạp	Dễ dự đoán, kiểm thử đơn giản.	Khó dự đoán, cần đánh giá bằng thống kê & độ chính xác (accuracy, loss...).
Kiến thức cần thiết	Toán logic, thuật toán, cấu trúc dữ liệu.	Toán xác suất, đại số tuyến tính, thống kê, học máy (Machine Learning).
Thay đổi theo thời gian	Ít thay đổi trừ khi code lại.	Mô hình có thể tự cập nhật, học thêm từ dữ liệu mới.
Khả năng thích nghi	Kém linh hoạt — chỉ làm đúng điều đã lập trình.	Linh hoạt hơn — có thể tự học để cải thiện hiệu năng.
Công cụ phổ biến	C/C++, Java, Python cơ bản, SQL, .NET...	Python (TensorFlow, PyTorch, Scikit-learn), R, Jupyter, GPU...

13. Kiến trúc CNN (mạng tích chập) giải quyết Semantic Gap như thế nào?

CNN là một mô hình học sâu hoạt động dựa trên nguyên lý của mạng neural truyền thống nhưng điểm khác biệt chính là khả năng tự động trích xuất đặc trưng mà không cần sự can thiệp thủ công từ con người. Thông qua nhiều lớp xử lý, CNN có thể phát hiện từ các đặc trưng đơn giản như cạnh, góc, màu sắc cho đến những chi tiết phức tạp hơn như hình dạng, kết cấu và toàn bộ đối tượng trong ảnh. Mỗi tầng trong CNN tạo ra một biểu diễn trừu tượng hơn của hình ảnh, nên máy tính dần hiểu hình ảnh “giống con người hơn”.

14. Cho ví dụ từng cấp độ thông minh

Cấp độ	Ví dụ thực tế minh họa
Data (Dữ liệu)	Camera chụp được 10.000 ảnh; cảm biến ghi lại 30°C, 1012 hPa, 60% độ ẩm.
Information (Thông tin)	Trung bình nhiệt độ hôm nay cao hơn hôm qua 2°C; ảnh được gắn nhãn “mèo”, “chó”.
Knowledge (Tri thức)	Khi nhiệt độ tăng → năng suất lao động giảm; biết vì sao ảnh đó là “mèo”.
Intelligence (Trí tuệ)	Hệ thống điều hòa tự động bật khi nhiệt độ vượt 28°C; AI gợi ý hành động phù hợp.
Wisdom (Sự thông thái)	Nhà lãnh đạo hoặc hệ thống ra quyết định cân bằng giữa lợi ích – đạo đức – môi trường. → Có thêm tính thần trách nhiệm
Knowledge Ecology (Sinh thái tri thức) (tầng mở rộng trong hình)	Mạng lưới AI – con người cùng học (như hệ thống ChatGPT, Google, Wikipedia mở).

→ Kim chỉ nam định nghĩa output thông minh

II. Chương 2: Khái niệm cơ bản về xử lý ảnh

1. Kiểm tra sản phẩm thuộc tầng Output nào? Ứng dụng là gì? Để thực hiện ứng dụng đó cần tác vụ là gì?

STT	Ứng dụng (Ví dụ)	Thuộc tầng Output	Mục tiêu / Ứng dụng cụ thể	Tác vụ cần thực hiện
1	AICX – Quan sát mức độ hài lòng khi xem phim rạp	Tầng 2: xử lý tác vụ	Phân tích phản ứng người xem để đánh giá mức độ hài lòng theo thời gian thực	- Phát hiện khuôn mặt- Nhận diện cảm xúc- Theo dõi chuyển động mắt / nụ cười
2	Quan sát khách hàng – Vẽ đường cong cảm xúc	Tầng 2: xử lý tác vụ	Biểu diễn biến thiên cảm xúc khách hàng theo thời gian (phục vụ marketing / trải nghiệm)	- Nhận diện khuôn mặt- Phân loại cảm xúc- Ghi log thời gian – tạo biểu đồ cảm xúc
3	Ghi nhớ lịch sử mua sắm	Tầng 3: Tầng Hệ thống	Lưu trữ và gợi ý sản phẩm dựa trên lịch sử giao dịch	- Nhận diện khách hàng- Trích xuất ID cá nhân- Truy xuất & học thói quen mua
4	Dynamic Ads – Nhận diện người xem đưa quảng cáo tương ứng	Tầng 2: Xử lý tác vụ	Hiển thị quảng cáo phù hợp với độ tuổi, giới tính, cảm xúc hoặc danh tính	- Nhận diện khuôn mặt- Phân loại nhóm khách hàng- Ghép nội dung quảng cáo phù hợp

5	Bettle Bot – Robot đưa thức ăn, thuốc	Tầng 2: Xử lý tác vụ	Robot phục vụ tự động, nhận biết người nhận và định vị di chuyển	- Nhận dạng người- Định vị và di chuyển- Giao – nhận vật phẩm
6	Contrastless Access – Quản lý người ra vào cửa hàng	Tầng 3: Tầng hệ thống	Kiểm soát truy cập dựa trên nhận diện khuôn mặt (thay thẻ hoặc mật mã)	- Nhận diện khuôn mặt- So khớp dữ liệu người dùng- Mở khóa hoặc báo động
7	Nhận diện Admin trong công ty	Tầng 3: Tầng hệ thống	Xác thực quyền truy cập hoặc ghi nhận sự hiện diện	- Nhận diện danh tính- Xác thực quyền- Ghi log truy cập

2. Viết biểu diễn toán học của video.

$$V = \{f_i(x, y)\} \quad i = 1, n$$

V = danh sách các frame

III. Chương 3: Biến đổi màu

Câu 1: Chứng minh các biểu thức tương đương

Biểu thức: $|x| + |y| = 1$

- Chọn $A(x_0, y_0)$ thuộc (P) thì A thuộc một trong bốn đường: $x + y = 1$, $x - y = 1$, $y - x = 1$, $-x - y = 1$

- Rõ ràng $|x_0|, |y_0| \leq 1$

- Kết hợp hai điều trên \rightarrow thuộc hình thoi

Biểu thức: $x^2 + y^2 = 1$

\rightarrow Thuộc đường tròn

Biểu thức: $\max(|x|, |y|) = 1$

- Chọn $A(x_0, y_0)$ thuộc (P) thì A thuộc một trong bốn đường: $x = \pm 1$, $y = \pm 1$

- Rõ ràng $|x_0|, |y_0| \leq 1$

- Kết hợp hai điều trên \rightarrow thuộc hình vuông

Câu 2: Đặc trưng về vân:

Feature	Scale			ViewPoint	Light condition	Occlusion	Clutter Background	Storage
	Position	Size	Rotate					
Color	Không phụ thuộc vị trí, nhưng dễ nhiễu nền	Ảnh hưởng khi thay đổi kích thước vùng ảnh	Ảnh hưởng nhẹ khi xoay (vùng màu trộn)	Thay đổi mạnh khi góc nhìn đổi	Rất nhạy với ánh sáng, bóng đổ	Mất thông tin nếu bị che khuất	Rất nhạy với nền phức tạp	Dung lượng nhỏ, dễ lưu
Shape (HOG)	Ảnh hưởng khi đổi tượng dịch chuyển	Ảnh hưởng nếu không chuẩn hóa kích thước	Nhạy với xoay (cần xoay chuẩn hóa)	Méo dạng khi góc nhìn thay đổi	Ít bị ảnh hưởng bởi ánh sáng	Mất biên nếu bị che khuất	Nền lộn xộn gây biên giả	Vector đặc trưng lớn, tốn bộ nhớ
Haralick (vân)	Không phụ thuộc vị trí	Thay đổi kích thước ảnh làm thay đổi độ chi tiết vân	Một số đặc trưng vân nhạy cảm với hướng, nhưng có thể tính theo nhiều hướng	Góc nhìn khác nhau làm thay đổi cách vân xuất hiện	Ánh sáng thay đổi độ tương phản và độ rõ của vân	Che khuất làm mất một phần vùng vân	Nền phức tạp có thể trộn lẫn với vân	Chỉ lưu vector đặc trưng (13 giá trị Haralick)

Câu 3: Chứng minh bất đẳng thức:

$$d_{\infty}(x, y) < d_2(x, y) < d_1(x, y)$$

Trong không gian 2 chiều: $\max(|x_1|, |x_2|) < \sqrt{(x_1^2 + x_2^2)} < |x_1| + |x_2|$

Đặt $a = |x_1|$, $b = |x_2|$, $a, b \geq 0$. Ta cần chứng minh: $\max(a, b) < \sqrt{a^2 + b^2} < a + b$

Giả sử $a > b$:

$$- \max(a, b) = a$$

$$- a < \sqrt{a^2 + b^2} \Leftrightarrow a^2 < a^2 + b^2 \Leftrightarrow b^2 > 0 \text{ (luôn đúng)}$$

$$- \sqrt{a^2 + b^2} < a + b \Leftrightarrow a^2 + b^2 < a^2 + b^2 + 2ab \Leftrightarrow 2ab > 0 \text{ (luôn đúng)}$$

Tổng quát n chiều:

$$\max(|x_1|, \dots, |x_n|) < \sqrt{(x_1^2 + \dots + x_n^2)} < |x_1| + \dots + |x_n|$$

Câu 4: Định lượng màu

Ảnh thường có 16,7 triệu màu \rightarrow lãng phí. Quy định số màu: $N_h = 18$, $N_s = 10$, $N_v = 10 \rightarrow$

$$N_{\text{colors}} = 1800$$

Không sao vì nhiều màu giống nhau \rightarrow gom nhóm.

Thuật toán định lượng màu:

For each pixel of the image:

- Lấy màu RGB, chuyển sang HSV
- Tính chỉ số h, s, v và quy đổi sang index
- Cập nhật histogram $H[\text{index}] += 1$

IV. Chương 4: Biến đổi hình học

Câu 1: Vì sao $a \cdot f(x, y)$ lại giúp ta sửa đổi độ tương phản.

$$g(x_1, y_1) = a \cdot f(x_1, y_1)$$

$$g(x_2, y_2) = a \cdot f(x_2, y_2)$$

$$\Rightarrow |g(x_1, y_1) - g(x_2, y_2)| = |a \cdot f(x_1, y_1) - a \cdot f(x_2, y_2)| = |a| \Delta f$$

$$|a| \leq 1 \Rightarrow \Delta g \leq \Delta f$$

$$|a| > 1 \Rightarrow \Delta g > \Delta f$$

Câu 2: Có 1000 bức ảnh, tạo 5000 bức ảnh dựa trên phép biến đổi linear mapping. (tạo 4000 g())

Ta có $g = af + b$

Thay đổi b từ $[0, 255] \rightarrow$ ta thu được 256 ảnh

Thay đổi a từ $[0, 20] \rightarrow$ ta thu được 10 ảnh

Thay đổi cặp (a, b) \rightarrow ta thu được $256 \cdot 10 = 2560$ ảnh \rightarrow thỏa mãn

Câu 3: Tại sao biểu đồ $g(x, y) = c \cdot \log(f(x, y))$ thì $\text{delta}(f1) = \text{delta}(f2)$ mà $\text{delta}(g1)$ lớn hơn $\text{delta}(g2)$?

Chọn $f1, f2, f3, f4$ sao cho $\text{delta}(f1) = \text{delta}(f2)$, $f1 + h = f2$, $f3 + h = f4$, $f2 < f3$

Ta cần cm $\text{delta}(g1) > \text{delta}(g2)$

Ta xét $\text{delta}(g1) < \text{delta}(g2)$

$$\Leftrightarrow c \log f2 - c \log f1 < c \log f4 - c \log f3$$

$$\Leftrightarrow \log(f2/f1) < \log(f4/f3) \text{ (vì là hàm tăng nên } c > 0, \text{ chia không đổi dấu)}$$

$$\Leftrightarrow f2/f1 < f4/f3 \text{ (vì } f2 > f1 \text{ và } f4 > f3, \text{ nên bên trong log } > 1 \rightarrow \text{log đồng biến, bỏ log không đổi dấu)}$$

$$\Leftrightarrow (f1 + h)/f1 < (f3 + h)/f3$$

$$\Leftrightarrow h/f1 < h/f3$$

$$\Leftrightarrow f1 > f3 \text{ (vô lý)}$$

Vậy $\text{delta}(g1) < \text{delta}(g2)$ là điều sai

$$\rightarrow \text{delta}(g1) > \text{delta}(g2)$$

Tại sao biểu đồ $g(x, y) = e^{f(x, y)}$ thì $\text{delta}(f1) = \text{delta}(f2)$ mà $\text{delta}(g1)$ bé hơn $\text{delta}(g2)$?

Chọn $f1, f2, f3, f4$ sao cho $\text{delta}(f1) = \text{delta}(f2)$, $f1 + h = f2$, $f3 + h = f4$, $f2 < f3$

Ta cần cm $\text{delta}(g1) < \text{delta}(g2)$

Ta xét $\text{delta}(g1) > \text{delta}(g2)$

$$\Leftrightarrow e^{f2} - e^{f1} > e^{f4} - e^{f3}$$

$$\Leftrightarrow e^{(f1+h)} - e^{f1} > e^{(f3+h)} - e^{f3}$$

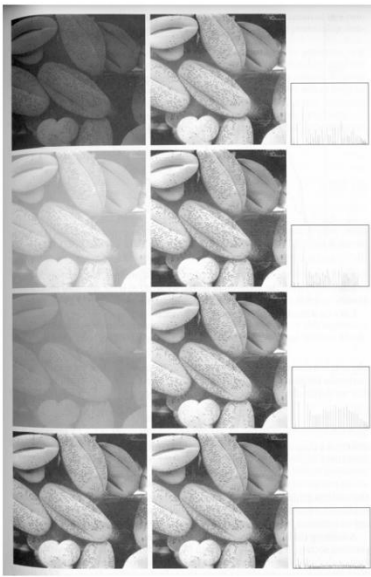
$$\Leftrightarrow e^{f1}(e^h - 1) > e^{f3}(e^h - 1)$$

$$\Leftrightarrow f1 > f3 \text{ (vô lý)}$$

Vậy $\text{delta}(g1) > \text{delta}(g2)$ là điều sai

$$\rightarrow \text{delta}(g1) < \text{delta}(g2)$$

Câu 4: Vì sao cột bên trái không ra cột bên phải bằng phép biến đổi linear & non-linear.



Lý do KHÔNG thể dùng Linear & Non-linear mapping:

1. Bản chất của Linear & Non-linear mapping:

- Các phép biến đổi này chỉ ánh xạ từng giá trị pixel **độc lập**: $g(x,y) = T[f(x,y)]$
- Công thức:
 - Linear: $g = a \times f + b$
 - Non-linear: $g = c \times \log(f)$ hoặc $g = e^{(\alpha f)}$

2. Vấn đề với ảnh cột trái:

- Ảnh có độ tương phản **cực bộ** rất thấp (các vùng hoa quả gần như đồng màu)
- Histogram của ảnh gốc tập trung ở một vùng hẹp \rightarrow thiếu thông tin chi tiết
- **Thông tin vân, texture đã bị mất** trong quá trình chụp/lưu trữ

3. Tại sao không thể khôi phục:

- Linear/Non-linear mapping chỉ **dãn/nén histogram** theo công thức toán học
- Không thể **tạo ra thông tin mới** không có trong ảnh gốc
- Nếu 2 pixels có cùng giá trị trong ảnh gốc \rightarrow chúng vẫn có cùng giá trị sau biến đổi
- **Chi tiết texture đã mất không thể phục hồi** bằng ánh xạ đơn giản

Câu 5: Thực hiện phép cân bằng lược đồ xám:

2	2	2	2	2	2	2	5
2	6	6	6	6	5	5	5
1	7	7	6	6	5	5	5
1	7	7	1	1	8	6	4
2	9	8	8	8	8	6	4
2	9	10	10	11	12	12	3
2	9	9	10	10	14	13	3
2	2	2	2	2	3	3	3

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
H	0	4	16	5	2	7	8	4	5	4	4	1	2	1	1
T	0	4	20	25	27	34	42	46	51	55	59	60	62	63	64
TG	0	1	4	5	6	7	9	10	11	12	13	13	14	14	14

4	4	4	4	4	4	4	7
4	9	9	9	9	9	7	7
1	10	10	9	9	7	7	7
1	10	10	1	1	11	9	6
4	12	11	11	11	11	9	6
4	12	13	13	13	14	14	5
4	12	13	13	13	14	14	5
4	4	4	4	4	5	5	5

Câu 6: So sánh các toán tử với nhau

Nhóm toán tử	Nguyên lý	Phương pháp	Độ phức tạp thuật toán	Khuyết điểm	Ưu điểm	Ứng dụng
Tuyến tính	Dựa trên phép biến đổi tuyến tính $f \rightarrow g$. Giá trị màu phụ thuộc vào chính điểm đó mà không phụ thuộc vào các điểm lân cận.	- Brightness modification: $g(x,y) = f(x,y) + b$ - Contrast modification: $g(x,y) = a.f(x,y)$ - Brightness + Contrast : $g(x,y) = a.f(x,y) + b$	- $O(n)$: Duyệt qua từng điểm ảnh pixel. - Phép tính cơ bản: +, x	Chưa sử dụng giá trị độ sáng trong lân cận vùng đang xét.	Dễ cài đặt, nhanh, ổn định, thích hợp thao tác điều chỉnh cơ bản, thay đổi đồng nhất, đồng đều. $\Delta f1 = \Delta f2 \Rightarrow \Delta g1 = \Delta g2$	Tích hợp với các thao tác xử lý ảnh cơ bản, điều chỉnh, cân bằng cho tổng thể. Ứng dụng trong học sâu.
Phi tuyến	Dựa trên phép biến đổi phi tuyến logarit hoặc hàm mũ. Giá trị màu của hàm $f(x,y)$ chỉ phụ thuộc vào điểm x,y mà không phụ thuộc vào các điểm lân cận.	Logarit: $g(x,y) = c.log(1 + f(x,y))$ Exponential: $g(x,y) = e^{f(x,y)}$	- $O(n)$: Duyệt qua từng điểm ảnh - Phép tính phức tạp hơn: mũ, logarit	Chưa sử dụng giá trị độ sáng trong lân cận vùng đang xét.	Tăng độ tương phản vùng tối / sáng khác nhau thay vì bằng nhau giống phương pháp tuyến tính, thể hiện sự linh hoạt của phương pháp này.	Tiền xử lý ảnh, cải thiện hiển thị ảnh. Vd: ảnh đêm, ảnh vệ tinh,...

Phân bố xác suất	Dựa trên phân bố xác suất (PDF) của mức xám trong ảnh $H[p]$	Histogram Equalization: $T[p] = T[p-1] + H[p]$ $T[p] = \text{round}((nG - 1 / N.M).T[p])$	- $O(n)$ - Cần thống kê, lược đồ tích lũy	- Phức tạp, có thể ảnh hưởng nhiều đến Histogram không phù hợp. - Histogram đã tốt thì không cần cải thiện.	- Kiểm soát dựa trên toàn bộ điểm ảnh của ảnh. - Cân bằng, chuẩn hóa phân bố mức xám toàn thể ảnh	- Chuẩn hóa màu, mức xám. Ứng dụng tốt trong học sâu. - Tăng độ hiệu quả và chính xác khi áp dụng DeepLearning.
------------------	--	--	--	--	--	--

Câu 7: Tìm giải pháp cải tiến cho các lược đồ màu.

Giải pháp cải tiến cho các lược đồ màu (Color Space)

1. Vấn đề của các lược đồ màu truyền thống

RGB:

- Phụ thuộc thiết bị (device-dependent)
- 3 kênh tương quan cao, dư thừa thông tin
- Nhạy cảm với thay đổi ánh sáng
- Không phản ánh cách con người nhìn màu

HSV/HSI:

- Hue không ổn định khi saturation thấp
- Không đồng đều về mặt tri giác (perceptually non-uniform)

2. Các giải pháp cải tiến

A. Sử dụng không gian màu Lab (CIELAB)

Ưu điểm:

- Perceptually uniform: Khoảng cách Euclidean trong Lab tương ứng với độ khác biệt màu mà mắt người cảm nhận
- Device-independent: Không phụ thuộc thiết bị
- Tách biệt độ sáng (L) và màu sắc (a, b)
- Ít nhạy cảm với thay đổi ánh sáng

Ứng dụng:

L: 0-100 (đen đến trắng)

a: xanh lá (-) đến đỏ (+)

b: xanh dương (-) đến vàng (+)

B. Không gian màu YCbCr/YUV

Ưu điểm:

- Tách biệt luminance (Y) và chrominance (Cb, Cr)
- Giảm dư thừa thông tin
- Phù hợp với nén ảnh/video (JPEG, MPEG)
- Mắt người ít nhạy với chrominance → có thể subsample

C. Normalized RGB (rgb)

Công thức:

$$r = R/(R + G + B)$$

$$g = G/(R + G + B)$$

$$b = B/(R + G + B)$$

Ưu điểm:

- Giảm ảnh hưởng của cường độ ánh sáng
- Chỉ cần 2 thành phần (r, g) vì r+g+b=1
- Tốt cho nhận dạng đối tượng có màu sắc đặc trưng

D. Opponent Color Space

Nguyên lý:

- Dựa trên lý thuyết màu đối lập của thị giác người
- 3 kênh: Red-Green, Blue-Yellow, White-Black

Công thức:

$$O1 = (R - G)/\sqrt{2}$$

$$O2 = (R + G - 2B)/\sqrt{6}$$

$$O3 = (R + G + B)/\sqrt{3}$$

Ưu điểm:

- Giảm tương quan giữa các kênh
- Mô phỏng cách xử lý màu của não người

3. Kỹ thuật cải tiến bổ sung

A. Color Constancy Algorithms

- Retinex algorithm: Loại bỏ ảnh hưởng ánh sáng, giữ màu cố hữu
- Gray World assumption: Giả định trung bình ảnh là xám
- White Patch: Pixel sáng nhất là trắng

B. Adaptive Color Space Selection

- Chọn không gian màu phù hợp với từng ứng dụng:
 - Nhận dạng da người: YCbCr, HSV
 - Phân đoạn ảnh: Lab
 - Tracking: Normalized RGB, HSV
 - Y tế: Grayscale, Lab

C. Multi-Color Space Features

- Kết hợp đặc trưng từ nhiều không gian màu
- Ví dụ: RGB + HSV + Lab \rightarrow vector đặc trưng phong phú hơn

D. Illumination Invariant Color Spaces

c1c2c3 color space:

$$c1 = \arctan(R/\max(G, B))$$

$$c2 = \arctan(G/\max(R, B))$$

$$c3 = \arctan(B/\max(R, G))$$

Ưu điểm:

- Bất biến với thay đổi cường độ và màu ánh sáng
- Tốt cho nhận dạng trong điều kiện ánh sáng thay đổi

4. Deep Learning Approaches

Learned Color Spaces

- CNN tự động học biểu diễn màu tối ưu cho từng task
- Không cần thiết kế thủ công
- Ví dụ: Color constancy networks, automatic white balance

V. Chương 5: Làm trơn ảnh

Câu 1: Hạn chế của phương pháp nội suy bằng cách làm tròn giá trị màu gần nhất

- Không có màu mới mà lấy lại những màu cũ
- Bị trùng màu, không tự nhiên

Câu 2: Diễn giải phép nội suy song tuyến tính

Mong muốn sau phép ánh xạ thì cả 4 điểm lân cận đều ảnh hưởng vào (x, y)

Gần điểm nào hơn thì điểm nguyên của đỉnh đó ảnh hưởng nhiều hơn

Ví dụ: nếu a càng lớn thì 2 điểm bên phải ảnh hưởng nhiều hơn, b càng lớn thì bên trên ảnh hưởng nhiều hơn

→ nếu tất cả các điểm lọt vô, với mỗi điểm có a, b khác nhau → g khác nhau → tự nhiên hơn (không chỉ cố định những màu cũ)

Câu 3: So sánh phép biến đổi hình học với bên đồ họa máy tính

Ví dụ đường cong bezier (chỉ tác động các điểm cấu thành thôi, không tác động toàn bộ các điểm ảnh)

Phép biến đổi hình học trong Xử lý ảnh

Đặc điểm:

- Tác động lên **toàn bộ pixels** của ảnh raster (bitmap)
- Công thức: $(x', y') = T(x, y)$ cho **MỌI pixel**
- Sau biến đổi cần **brightness interpolation** (nearest-neighbor, bilinear)
- Xử lý **dữ liệu rời rạc** (discrete grid)

Quy trình:

1. Transform coordinates: $(x', y') = T(x, y)$
2. Inverse mapping: $(x, y) = T^{-1}(x', y')$
3. Interpolation: $f(x', y')$ từ $f(x, y)$

Phép biến đổi trong Đồ họa máy tính

Đặc điểm:

- Tác động lên **các điểm điều khiển** (control points) hoặc vertices
- Ví dụ: Đường cong Bézier chỉ cần 4 điểm điều khiển để tạo đường cong mượt
- Xử lý **dữ liệu vector** (liên tục)
- Không cần interpolation brightness

Câu 4: Thực hiện phép quay ảnh 30, 45, 60 độ

Công thức quay (Rotation)

$$x' = x \cdot \cos(\theta) - y \cdot \sin(\theta)$$

$$y' = x \cdot \sin(\theta) + y \cdot \cos(\theta)$$

Mã trận quay:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

Các giá trị cụ thể:

Quay 30°:

$$\cos(30^\circ) = \sqrt{3}/2 \approx 0.866$$

$$\sin(30^\circ) = 1/2 = 0.5$$

$$x' = 0.866x - 0.5y$$

$$y' = 0.5x + 0.866y$$

Quay 45°:

$$\cos(45^\circ) = \sin(45^\circ) = \sqrt{2}/2 \approx 0.707$$

$$x' = 0.707x - 0.707y = 0.707(x - y)$$

$$y' = 0.707x + 0.707y = 0.707(x + y)$$

Quay 60°:

$$\cos(60^\circ) = 1/2 = 0.5$$

$$\sin(60^\circ) = \sqrt{3}/2 \approx 0.866$$

$$x' = 0.5x - 0.866y$$

$$y' = 0.866x + 0.5y$$

Câu 5: So sánh toán tử scale ảnh và toán tử image-supper resolution

Scale ảnh (Image Scaling) - Truyền thống

Nguyên lý:

- Thay đổi kích thước bằng interpolation đơn giản
- Công thức: $(x', y') = (sx \cdot x, sy \cdot y)$

Phương pháp interpolation:

1. Nearest-neighbor: $f(x',y') = f(\text{round}(x), \text{round}(y))$
2. Bilinear: Nội suy tuyến tính từ 4 pixels lân cận
3. Bicubic: Nội suy từ 16 pixels (4×4)

Đặc điểm:

- Nhanh, đơn giản
- Không tạo chi tiết mới, chỉ "kéo dẫn" thông tin có sẵn
- Upscale nhiều \rightarrow mờ, răng cưa
- Không phục hồi được chi tiết đã mất

Image Super-Resolution (SR)

Nguyên lý:

- Sử dụng học máy/deep learning để "đoán" và tạo chi tiết mới
- Học từ cặp ảnh (low-res, high-res)
- Tái tạo thông tin dựa trên prior knowledge

Các phương pháp:

1. SRCNN (Super-Resolution CNN)
2. ESRGAN (Enhanced SRGAN)
3. Real-ESRGAN
4. Diffusion models

Quy trình:

Low-res image \rightarrow CNN/Transformer \rightarrow High-res image

\uparrow

Learned features từ dataset

Đặc điểm:

- Tạo được chi tiết mới hợp lý
- Chất lượng cao, sắc nét
- Có thể upscale $4\times$, $8\times$ mà vẫn rõ
- Chậm hơn nhiều
- Cần GPU mạnh
- Có thể "hallucinate" (tạo chi tiết sai)

Câu 6: Khử nhiễu làm sao? Nguyên lý là gì?

Nguyên lý cơ bản của khử nhiễu:

Giả thiết nền tảng:

- Signal (tín hiệu thật): Có tính liên tục, tương quan không gian cao
- Noise (nhiều): Ngẫu nhiên, không tương quan với lân cận

Mô hình nhiễu:

$$g(x, y) = f(x, y) + n(x, y)$$

Trong đó:

- $g(x, y)$: Ảnh quan sát được (có nhiễu)
- $f(x, y)$: Ảnh gốc (signal)
- $n(x, y)$: Nhiễu

Các nguyên lý khử nhiễu:

1. Nguyên lý Averaging (Trung bình hóa)

Ý tưởng:

- Nhiễu là ngẫu nhiên $\rightarrow E[n(x, y)] = 0$
- Lấy trung bình nhiều pixels \rightarrow nhiễu triệt tiêu, signal được giữ lại

Toán học:

$$\widehat{f(x, y)} = (1/|\Omega|) \times \sum g(i, j)$$
$$(i, j) \in \Omega$$

Với Ω là vùng lân cận của (x, y)

Ưu điểm: Đơn giản, giảm nhiễu Gaussian hiệu quả

Nhược điểm: Làm mờ cạnh và chi tiết

2. Nguyên lý Median (Trung vị)

Ý tưởng:

- Nhiễu impulse (Salt & Pepper) là outlier

- Median không bị ảnh hưởng bởi outliers

Toán học:

$$\widehat{f(x,y)} = \text{mediang}(i,j) | (i,j) \in \Omega$$

Đặc tính:

- Loại bỏ impulse noise cực tốt
- Bảo toàn cạnh (edge-preserving)
- Không làm mờ như mean filter

3. Nguyên lý Weighted Average (Trung bình có trọng số)

Ý tưởng:

- Pixels gần \rightarrow ảnh hưởng nhiều (trọng số cao)
- Pixels xa \rightarrow ảnh hưởng ít (trọng số thấp)

Gaussian Filter:

$$\widehat{f(x,y)} = \sum h(i,j) \times g(x+i, y+j)$$

$$(i,j) \in \Omega$$

$$\text{Với: } h(i,j) = (1/(2\pi\sigma^2)) \times \exp(-(i^2 + j^2)/(2\sigma^2))$$

Ưu điểm: Làm mượt tự nhiên hơn averaging đơn thuần

4. Nguyên lý Edge-Preserving

Ý tưởng:

- Làm mượt trong vùng đồng nhất
- Giữ nguyên ở vùng có cạnh

Bilateral Filter:

$$\widehat{f(x,y)} = (1/W) \times \sum w(i,j) \times g(i,j)$$

$$w(i,j) = \exp(-d^2/(2\sigma_s^2)) \times \exp(-|g(x,y) - g(i,j)|^2/(2\sigma_r^2))$$

\uparrow

\uparrow

spatial weight

range weight

- Spatial: Gần về vị trí
- Range: Gần về giá trị intensity

5. Nguyên lý Frequency Domain

Ý tưởng:

- Nhiều tập trung ở tần số cao
- Signal (cấu trúc) ở tần số thấp

Low-pass Filter:

$$F(\widehat{u, v}) = H(u, v) \times G(u, v)$$

$$f(\widehat{x, y}) = IFFT[F(\widehat{u, v})]$$

$$H(u, v) = 0 \text{ với } (u^2 + v^2) > \text{cutoff frequency}$$

Câu 7: Điểm cần làm mượt nằm ở đâu?

Phân loại vùng trong ảnh:

1. Vùng CẦN làm mượt (Smooth regions)

A. Vùng đồng nhất (Flat regions):

Đặc trưng:

- Variance thấp: $\text{Var}\{g(i, j)\} < \text{threshold}$
- Gradient nhỏ: $|\nabla g| \approx 0$
- Không có cấu trúc quan trọng

Ví dụ: Bầu trời, tường trơn, nền đơn sắc

Lý do: Làm mượt MẠNH ở đây không mất thông tin

B. Vùng có nhiễu cao:

Phát hiện:

- Local variance cao bất thường
- Pixels outliers: $|g(x, y) - \text{mean}(\Omega)| > k \times \sigma$
- Salt & Pepper noise points

Lý do: Nhiễu cần được loại bỏ

2. Vùng không nên làm mờ (Feature regions)

A. Cạnh (Edges):

Phát hiện:

- Gradient lớn: $|\nabla g| > \text{threshold}$
- Canny, Sobel detector

Lý do:

- Chứa biên của đối tượng
- Làm mờ \rightarrow ảnh mờ, mất thông tin phân đoạn

B. Góc (Corners):

Phát hiện:

- Harris corner detector
- λ_1, λ_2 (eigenvalues) đều lớn

Lý do:

- Đặc trưng quan trọng cho matching
- Mốc điểm (landmarks)

C. Texture (Kết cấu):

Phát hiện:

- High-frequency components
- Variance cao nhưng có cấu trúc

Lý do:

- Thông tin bề mặt vật liệu
- Đặc trưng nhận dạng

Câu 8: Các ma trận 5×5 , 7×7 , 11×11 của các phương pháp làm trơn ảnh

1. Averaging Filter (Box Filter)

Nguyên lý: Tất cả pixels có trọng số bằng nhau

5×5 Averaging:

$$\begin{array}{ccccc}
 1 & 1 & 1 & 1 & 1 \\
 1 & 1 & 1 & 1 & 1 \\
 h = 1/25 \times & 1 & 1 & 1 & 1 & 1 \\
 & 1 & 1 & 1 & 1 & 1 \\
 & 1 & 1 & 1 & 1 & 1
 \end{array}$$

Tổng = 25, mỗi phần tử = 1/25

Công thức tổng quát:

$h(i,j) = 1/N^2$ với kernel size $N \times N$

7×7 Averaging:

$$\begin{array}{ccccccc}
 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
 h = 1/49 \times & 1 & 1 & 1 & 1 & 1 & 1 \\
 & 1 & 1 & 1 & 1 & 1 & 1 \\
 & 1 & 1 & 1 & 1 & 1 & 1 \\
 & 1 & 1 & 1 & 1 & 1 & 1
 \end{array}$$

Mỗi phần tử = 1/49

11×11 Averaging:

$h = 1/121 \times$ [ma trận 11×11 toàn số 1]

Mỗi phần tử = 1/121

Đặc điểm:

- Kernel càng lớn → làm mượt càng mạnh
- Làm mờ đều tất cả, không bảo toàn cạnh

2. Gaussian Filter**Công thức:**

$$h(i,j) = (1/(2\pi\sigma^2)) \times \exp(-(i^2+j^2)/(2\sigma^2))$$

Sau đó normalize: $h(i,j) = h(i,j) / \sum h(i,j)$

5×5 Gaussian ($\sigma = 1.0$):

Tính toán:

Tâm (0,0): $h(0,0) = \exp(0) = 1$

Cạnh (0,1): $h(0,1) = \exp(-1/(2 \times 1^2)) = \exp(-0.5) \approx 0.607$

Góc (1,1): $h(1,1) = \exp(-2/(2 \times 1^2)) = \exp(-1) \approx 0.368$

Mã trận (trước normalize):

0.135 0.368 0.607 0.368 0.135

0.368 1.000 1.649 1.000 0.368

$h \approx$ 0.607 1.649 2.718 1.649 0.607

0.368 1.000 1.649 1.000 0.368

0.135 0.368 0.607 0.368 0.135

Mã trận chuẩn hóa (approximation phổ biến):

1 4 7 4 1

4 16 26 16 4

$h = 1/273 \times$ 7 26 41 26 7

4 16 26 16 4

1 4 7 4 1

7×7 Gaussian ($\sigma = 1.5$):**Đặc điểm:**

- Phân bố rộng hơn 5×5
- Tâm vẫn có trọng số cao nhất
- Giảm dần theo hàm Gaussian

Cấu trúc (symbolic):

s s m l m s s

s m l L l m s

m l L M L l m

$h \approx$ l L M MAX M L l

m l L M L l m

s m l L l m s

s s m l m s s

s: small, m: medium, l: large, L: larger, M: much larger, MAX: maximum

Giá trị gần đúng (1/1000):

$$\begin{array}{ccccccc} 0 & 1 & 2 & 3 & 2 & 1 & 0 \\ 1 & 4 & 8 & 12 & 8 & 4 & 1 \\ 2 & 8 & 24 & 36 & 24 & 8 & 2 \\ h \approx 1/1000 \times & 3 & 12 & 36 & 52 & 36 & 12 & 3 \\ & 2 & 8 & 24 & 36 & 24 & 8 & 2 \\ & 1 & 4 & 8 & 12 & 8 & 4 & 1 \\ & 0 & 1 & 2 & 3 & 2 & 1 & 0 \end{array}$$

11×11 Gaussian ($\sigma = 2.0$):

Đặc điểm:

- Kernel rất lớn \rightarrow smooth mạnh
- σ lớn \rightarrow phân bố rộng, trọng số phân tán hơn

Cấu trúc:

Tâm (5,5): Trọng số cao nhất

Giảm dần theo công thức Gaussian

Corners (0,0), (0,10), (10,0), (10,10): Trọng số rất nhỏ

Tính toán minh họa:

Distance từ tâm (5,5):

- (5,5): $d=0 \rightarrow h = \exp(0) = 1.000$ (max)
- (5,6): $d=1 \rightarrow h = \exp(-1/(2 \times 4)) = 0.882$
- (6,6): $d=\sqrt{2} \rightarrow h = \exp(-2/(2 \times 4)) = 0.779$
- (5,10): $d=5 \rightarrow h = \exp(-25/(2 \times 4)) = 0.043$
- (10,10): $d=5\sqrt{2} \rightarrow h = \exp(-50/(2 \times 4)) \approx 0.002$

Kết quả: Ma trận 11×11 với tâm có giá trị cao, giảm dần ra ngoài

Câu 9: So sánh mức độ làm mờ biên cạnh của 2 giải pháp Gauss và trung bình cộng

Xét 1 điểm trên biên

Khi dùng Avg: tất cả các điểm lân cận ảnh hưởng trực tiếp đến biên

Khi dùng Gauss: Chỉ những điểm gần mới tương tác với biên, những pixel xa ít đóng góp → đỡ làm mờ biên hơn

Câu 10: Làm sao để biến mức độ nhiễu nhiễu → sạch trơn

Ảnh nhiễu nhiễu $g(x,y)$



[Stage 1: Analysis]

→ Phát hiện loại nhiễu

→ Ước lượng noise level σ



[Stage 2: Impulse Removal]

→ Median Filter 3×3 hoặc 5×5

→ Output: $g_1(x,y)$



[Stage 3: Gaussian Denoising]

→ Gaussian Filter (σ phù hợp)

→ Output: $g_2(x,y)$



[Stage 4: Edge Preservation]

→ Bilateral Filter

→ Output: $g_3(x,y)$



[Stage 5: Adaptive Refinement]

→ Adaptive smoothing theo local variance

→ Output: $g_4(x,y)$



[Stage 6: High-Quality (optional)]

→ Non-Local Means

→ Output: $g_5(x,y)$



[Stage 7: Post-process]

→ Mild sharpening (nếu cần)

→ Output: $g_final(x,y)$



Ảnh sạch trơn

Câu 11: Cho công thức sau trong Median Filter :

$$\sum_{(i,j) \in O} |f(x+i, y+j) - \text{med}| = \min_{(i',j') \in O} \sum_{(i,j) \in O} |f(x+i, y+j) - f(x+i', y+j')|$$

Chứng minh tổng độ dị biệt là bé nhất với trường hợp tổng quát, trong hợp chẵn thì sao?

Khi n chẵn:

- Có 2 giá trị median: $I_{n/2}$ và $I_{n/2+1}$
- Bất kì giá trị nào trong đoạn $[I_{n/2}, I_{n/2+1}]$ đều cho cùng tổng độ lệch nhỏ nhất

Trong thực tế:

Các cách chọn phổ biến:

Lower median: $\text{med} = I_{n/2}$

Upper median: $\text{med} = I_{n/2+1}$

Average: $\text{med} = (I_{n/2} + I_{n/2+1})/2$, thông dụng nhất

Chứng minh tính chất bất biến:

Cho $m \in [I_{n/2}, I_{n/2+1}]$:

$$S_m = \sum_{i=1}^{\frac{n}{2}} |I_i - m| + \sum_{i=\frac{n}{2}+1}^n |I_i - m|$$

Vì $I_{n/2} \leq m \leq I_{n/2+1}$:

$$S_m = \sum_{i=1}^{\frac{n}{2}} |m - I_i| + \sum_{i=\frac{n}{2}+1}^n |m - I_i|$$

Đạo hàm:

$$dS/dm = n/2 - n/2 = 0 \quad \forall m \in [I_{n/2}, I_{n/2+1}]$$

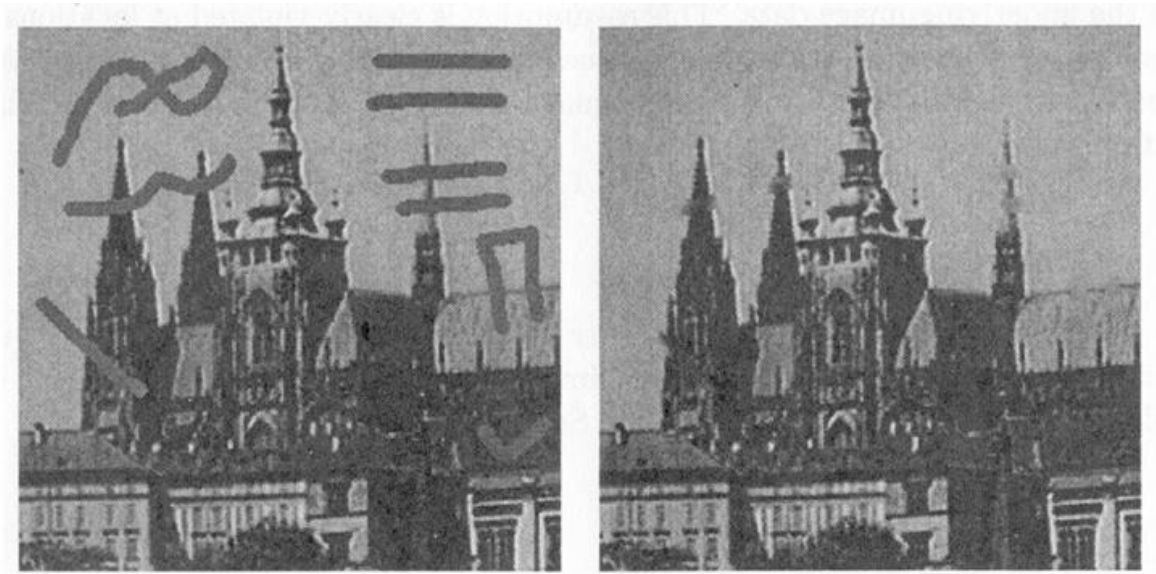
→ Tất cả các điểm trong đoạn này đều là cực tiểu

Câu 12: So với phương pháp trung bình cộng, thì phương pháp median có cái hay là gì?

Nếu thay đổi 1 giá trị màu thì trung bình cộng sẽ thay đổi → không ổn định (khi có sự xáo trộn ở 2 cực → thay đổi)

Nếu thay đổi 1 giá trị màu thì trung vị ít bị thay đổi → bền vững (khi có sự thay đổi ở 2 cực ổn định)

Câu 13: Dùng phương pháp Averaging, Gaussian, Median để lọc ảnh bên trái dưới đây:



Phân tích ảnh bên trái:

- Ảnh có nhiều nạng (nhiều hạt trắng/đen)
- Độ tương phản thấp, mờ nhạt
- Có cấu trúc (tòa lâu đài) cần bảo toàn
- Nhiều có vẻ là Salt & Pepper + Gaussian noise

1. Averaging Filter

Cách hoạt động:

$$g(x, y) = (1/N^2) \times \sum f(x + i, y + j)$$

$$(i, j) \in \Omega$$

Kernel 5×5:

$$h = \frac{1}{25} \times \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Kết quả trên ảnh này:

Ưu điểm:

- Giảm được Gaussian noise (nhiều phân bố đều)
- Làm mượt toàn bộ ảnh
- Tính toán nhanh

Nhược điểm:

- Salt & Pepper noise chỉ giảm nhẹ, không loại bỏ hoàn toàn
 - Các điểm nhiễu trắng/đen vẫn "lan tỏa" ra vùng xung quanh
 - Ví dụ: 1 pixel = 255 trong vùng 5×5 → trung bình vẫn bị kéo lên
- Làm mờ mạnh các cạnh (outline của lâu đài)
- Mất chi tiết kiến trúc

2. Gaussian Filter

Cách hoạt động:

$$g(x, y) = \sum h(i, j) \times f(x + i, y + j)$$

$$h(i, j) = \exp(-(i^2 + j^2)/(2\sigma^2))$$

Kernel 5 × 5 ($\sigma = 1.0$):

$$h = \frac{1}{273} \times \begin{bmatrix} 1 & 4 & 7 & 4 & 1 \\ 4 & 16 & 26 & 16 & 4 \\ 7 & 26 & 41 & 26 & 7 \\ 4 & 16 & 26 & 16 & 4 \\ 1 & 4 & 7 & 4 & 1 \end{bmatrix}$$

Kết quả trên ảnh này:

Ưu điểm:

- Làm mượt tự nhiên hơn Averaging
- Giảm Gaussian noise hiệu quả (vì cùng domain)
- Pixels gần tâm được ưu tiên → giữ được cấu trúc tốt hơn

Nhược điểm:

- Vẫn không tốt với Salt & Pepper noise
 - Điểm nhiễu trắng/đen có ảnh hưởng lớn do có trọng số
- Làm mờ cạnh (ít hơn Averaging nhưng vẫn đáng kể)
- Cần chọn σ phù hợp, nếu σ lớn → mờ quá

3. Median Filter

Cách hoạt động:

$$g(x, y) = \text{median}\{f(x + i, y + j) \mid (i, j) \in \Omega\}$$

Ví dụ 3×3:

Pixels: [10, 12, 255, 11, 13, 0, 14, 15, 12]

$\uparrow \quad \uparrow$
 nhiễu trắng nhiễu đen

Sắp xếp: [0, 10, 11, 12, 12, 13, 14, 15, 255]

Median = 12 ← Loại bỏ được cả 2 outliers!

Kết quả trên ảnh này:

Ưu điểm:

- LOẠI BỎ HOÀN TOÀN Salt & Pepper noise
 - Các điểm nhiễu trắng (255) và đen (0) là outliers
 - Median không bị ảnh hưởng bởi outliers
- BẢO TỒN CẠNH tốt
 - Không làm mờ biên của lâu đài
 - Edge-preserving property
- Giữ được chi tiết kiến trúc
- Không tạo "ghosting" như averaging

Nhược điểm:

- Chậm hơn Averaging/Gaussian (cần sort)
- Với Gaussian noise thuần túy thì kém hơn Gaussian filter
- Kernel quá lớn có thể làm "boxy" artifacts

Câu 14: Phân tích công thức toán tử sai số bình phương trung bình tối thiểu

1. Công thức cơ bản

$$\hat{f}(x,y) = \mu_{\text{local}} + k \times [g(x,y) - \mu_{\text{local}}]$$

Với hệ số Wiener:

$$k = \sigma^2_f / (\sigma^2_f + \sigma^2_n) = (\sigma^2_{\text{local}} - \sigma^2_n) / \sigma^2_{\text{local}}$$

2. Ý nghĩa các thành phần

$$\hat{f}(x,y) = \mu_{\text{local}} + k \times [g(x,y) - \mu_{\text{local}}]$$

↑ ↑ ↑

Điểm neo Độ tin cậy Độ lệch

- μ_{local} : Mean của vùng lân cận (giá trị an toàn)
- k : Hệ số adaptive ($0 \leq k \leq 1$)
- $[g - \mu_{\text{local}}]$: Độ lệch của pixel so với mean

3. Thuật toán

FOR mỗi pixel (x,y):

- Tính local statistics (window 5×5 hoặc 7×7)

$$\mu_{\text{local}} = \text{mean}(\text{vùng lân cận})$$

$$\sigma_{\text{local}}^2 = \text{variance}(\text{vùng lân cận})$$

- Ước lượng signal variance

$$\sigma_f^2 = \max(0, \sigma_{\text{local}}^2 - \sigma_n^2)$$

- Tính hệ số

$$k = \sigma_f^2 / \sigma_{\text{local}}^2$$

Ước lượng:

$$\widehat{f(x, y)} = \mu_{\text{local}} + k \times [g(x, y) - \mu_{\text{local}}]$$

Câu 15: Bảng so sánh 3 phương pháp làm trơn ảnh:

Tên	Nguyên lý	Phương pháp	Khả năng khử nhiễu	Khả năng làm mờ biên cạnh	Độ phức tạp
Averaging	Min < Avg < Max Thay thế điểm ảnh bằng giá trị trung bình của vùng lân cận, kéo avg xa min, max	$F * h$ Nhân chập ảnh F với mặt nạ h có các hệ số bằng nhau	Tốt	Cao	$O(n^2)$
Gaussian	Lấy những điểm ảnh hưởng nhiều thay thế	$F * h$ $h(i, j) = \frac{1}{2\pi\sigma^2} e^{-\frac{i^2+j^2}{2\sigma^2}}$	Tốt hơn	Ít hơn	$O(n^2)$
Median	Thay thế điểm ảnh bằng giá trị nằm ở giữa (trung vị) sau khi đã sắp xếp dãy điểm ảnh lân cận.	$g(x, y) = \text{med}\{f(x + i, y + j), (i, j) \in O\}$	Có khả năng khử nhiễu muối tiêu	Thấp	$O(n^2)$ Lâu hơn, vì phải sắp xếp

VI. Chương 6: Phát hiện biên cạnh

Câu 1: Tại sao phải dùng 3 cặp (A0, A2), (A7, A3), (A6, A4) để phát hiện biên cạnh

A ₀	A ₁	A ₂
A ₇	F(j,k)	A ₃
A ₆	A ₅	A ₄

Trong sobel hay prewitt, khi đạo hàm theo x (sai phân trung tâm) ta có:

$$\frac{\partial f}{\partial x} \approx f(x+1, y) - f(x-1, y)$$

Hay có thể viết cách khác

$$\text{Sobel: } Wx = \begin{matrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{matrix}$$

Khi nhân tích chập với ảnh f ta có: $x = (A_0 + A_7 + A_6) - (A_2 + A_3 + A_4)$

Hay $x = (A_0 - A_2) + (A_7 - A_3) + (A_6 - A_4)$

Khi lấy x vượt ngưỡng threshold thì ta xem F(j, k) là biên, và x được tính bằng hiệu các cặp pixel 2 phía đối diện (hay lân cận của F) → dựa vào tính chất của biên để tìm biên: điểm biên là điểm có màu sắc thay đổi với các điểm lân cận

$$\text{Prewitt: } Wx = \begin{matrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{matrix}$$

Khi nhân tích chập với ảnh f ta có: $x = (A_0 + 2A_7 + A_6) - (A_2 + 2A_3 + A_4)$

Hoàn toàn tương tự với Prewitt, ta cũng so sánh x với threshold và xác định biên, điểm khác duy nhất là prewitt tăng sự quan trọng ở cặp pixel ở trục Ox hơn, tùy trường hợp mà Prewitt hy vọng làm vậy sẽ tốt hơn

Câu 2: Chứng minh công thức

$$\begin{aligned} (\nabla^2 f)(x, y) &= \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \\ &\approx f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1) - 4f(x, y) \end{aligned}$$

Đạo hàm bậc nhất:

$$\frac{\partial f}{\partial x}(x, y) = \lim_{h \rightarrow 0} \frac{f(x + h, y) - f(x, y)}{h}$$

Chọn $h = 1$:

$$\frac{\partial f}{\partial x}(x, y) \approx f(x + 1, y) - f(x, y)$$

Đạo hàm bậc hai:

$$\frac{\partial^2 f}{\partial x^2} = \frac{\partial}{\partial x} \left(\frac{\partial f}{\partial x} \right)$$

Dùng sai phân trung tâm:

$$\frac{\partial^2 f}{\partial x^2}(x, y) \approx \frac{f(x + 1, y) - 2f(x, y) + f(x - 1, y)}{h^2}$$

Với $h = 1$:

$$\frac{\partial^2 f}{\partial x^2}(x, y) \approx f(x + 1, y) - 2f(x, y) + f(x - 1, y)$$

Tương tự với y

$$\frac{\partial^2 f}{\partial y^2}(x, y) \approx f(x, y + 1) - 2f(x, y) + f(x, y - 1)$$

Định nghĩa Laplacian

$$\nabla^2 f(x, y) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

Thay các xấp xỉ rồi rạc vào:

$$\begin{aligned} \nabla^2 f(x, y) &\approx [f(x + 1, y) - 2f(x, y) + f(x - 1, y)] \\ &\quad + [f(x, y + 1) - 2f(x, y) + f(x, y - 1)] \end{aligned}$$

Gom các hạng tử

$$\nabla^2 f(x, y) \approx f(x + 1, y) + f(x - 1, y) + f(x, y + 1) + f(x, y - 1) - 4f(x, y)$$

Câu 3: Chứng minh toán tử Laplace bất biến với phép quay

Phát biểu lại bài toán như sau:

Cho hàm trơn $f(x, y)$.

Xét phép quay hệ trục một góc θ :

$$\begin{cases} x = x' \cos \theta - y' \sin \theta \\ y = x' \sin \theta + y' \cos \theta \end{cases}$$

Cần chứng minh:

$$\nabla^2 f(x, y) = \nabla'^2 f(x', y')$$

tức là toán tử Laplace không đổi dưới phép quay.

Chứng minh như sau:

Từ phép quay:

$$\begin{aligned} \frac{\partial}{\partial x} &= \frac{\partial x'}{\partial x} \frac{\partial}{\partial x'} + \frac{\partial y'}{\partial x} \frac{\partial}{\partial y'} \\ &= \cos \theta \frac{\partial}{\partial x'} - \sin \theta \frac{\partial}{\partial y'} \\ \frac{\partial}{\partial y} &= \sin \theta \frac{\partial}{\partial x'} + \cos \theta \frac{\partial}{\partial y'} \end{aligned}$$

Bình phương theo x :

$$\begin{aligned} \frac{\partial^2}{\partial x^2} &= (\cos \theta \frac{\partial}{\partial x'} - \sin \theta \frac{\partial}{\partial y'})^2 \\ &= \cos^2 \theta \frac{\partial^2}{\partial x'^2} - 2 \sin \theta \cos \theta \frac{\partial^2}{\partial x' \partial y'} + \sin^2 \theta \frac{\partial^2}{\partial y'^2} \end{aligned}$$

Bình phương theo y :

$$\begin{aligned} \frac{\partial^2}{\partial y^2} &= (\sin \theta \frac{\partial}{\partial x'} + \cos \theta \frac{\partial}{\partial y'})^2 \\ &= \sin^2 \theta \frac{\partial^2}{\partial x'^2} + 2 \sin \theta \cos \theta \frac{\partial^2}{\partial x' \partial y'} + \cos^2 \theta \frac{\partial^2}{\partial y'^2} \end{aligned}$$

Cộng hai biểu thức:

$$\begin{aligned} \nabla^2 &= \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} \\ &= (\cos^2 \theta + \sin^2 \theta) \frac{\partial^2}{\partial x'^2} + (\cos^2 \theta + \sin^2 \theta) \frac{\partial^2}{\partial y'^2} \end{aligned}$$

Do:

$$\cos^2 \theta + \sin^2 \theta = 1$$

suy ra:

$$\nabla^2 = \frac{\partial^2}{\partial x'^2} + \frac{\partial^2}{\partial y'^2} = \nabla'^2$$

Từ đó suy ra điều phải chứng minh.

Câu 4: Chứng minh công thức

$$\nabla^2 f \approx f * \text{Laplace}$$

$$\text{Laplace} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

$$\text{Laplace} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Ta có :

$$\nabla^2 f(x, y) \approx f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1) - 4f(x, y)$$

Biểu thức trên có thể viết dưới dạng phép chập giữa ảnh f và mặt nạ Laplacian:

$$\nabla^2 f \approx f * \text{Laplacian}$$

với kernel Laplacian 4 lân cận:

$$\text{Laplace} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Laplacian rời rạc 8 lân cận (8-neighbor)

Trong trường hợp cần xấp xỉ Laplacian có tính đẳng hướng tốt hơn (gần với Laplacian liên tục), ta mở rộng xét cả các điểm chéo.

Khi đó, Laplacian rời rạc được xấp xỉ bởi:

$$\begin{aligned} \nabla^2 f(x, y) \approx & f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1) \\ & + f(x+1, y+1) + f(x-1, y-1) + f(x+1, y-1) + f(x-1, y+1) \\ & - 8f(x, y) \end{aligned}$$

Biểu diễn dưới dạng phép chập:

$$\text{Laplace} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Nhận xét

- Kernel Laplacian 4 lân cận xuất phát trực tiếp từ sai phân trung tâm của đạo hàm bậc hai.
- Kernel Laplacian 8 lân cận cải thiện tính đẳng hướng, gần hơn với Laplacian trong không gian liên tục.
- Trong xử lý ảnh, phép chập với các kernel trên được dùng để phát hiện biên và tăng cường chi tiết.

Câu 5: Cho thấy filter ở mức 5x5 7x7 11x11 của laplace of gauss

Laplacian of Gaussian là toán tử kết hợp giữa:

- Làm trơn Gaussian (giảm nhiễu)
- Toán tử Laplace (phát hiện biên)

Được định nghĩa trong không gian liên tục bởi:

$$\text{LoG}(x, y) = \nabla^2 (G(x, y) * f(x, y)) = (\nabla^2 G(x, y)) * f(x, y)$$

Hàm Gaussian 2D

$$G(x, y) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right)$$

Công thức Laplacian của Gaussian

Áp dụng toán tử Laplace cho $G(x, y)$:

$$\nabla^2 G(x, y) = \frac{x^2 + y^2 - 2\sigma^2}{\sigma^4} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right) \quad (1)$$

Kernel LoG 5x5

Một kernel LoG 5x5 chuẩn hóa:

$$\begin{bmatrix} 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & -2 & -1 & 0 \\ -1 & -2 & 16 & -2 & -1 \\ 0 & -1 & -2 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 \end{bmatrix}$$

Kernel LoG 7x7 và Kernel LoG 11x11 có thể suy ra từ (1) giống 5x5

Câu 7: Làm bảng thống kê các phương pháp edge detection

Tên phương pháp	Nguyên lý	Phương pháp	Độ phức tạp	Khả năng phát hiện biên cạnh	Khả năng chống chọi nhiễu	Ưu điểm	Khuyết điểm
Gradient (Roberts, Prewitt, Sobel)	Biên là nơi độ lớn gradient đạt cực đại	Tính đạo hàm bậc nhất theo (x, y), sau đó lấy độ lớn gradient	Thấp	Tốt với biên rõ, đơn giản	Thấp – nhạy với nhiễu	Dễ cài đặt, tính toán nhanh	Biên dày, dễ nhiễu, phụ thuộc hướng
Laplace	Biên tại điểm zero-crossing của Laplacian	Tính Laplacian bậc hai, tìm điểm đổi dấu	Thấp	Phát hiện tốt vị trí biên (zero-crossing)	Kém (rất nhạy với nhiễu)	Xác định biên chính xác về vị trí	Không cho hướng biên, rất nhạy nhiễu
LoG (Laplacian of Gaussian)	Làm trơn bằng Gaussian rồi áp dụng Laplacian	Gaussian smoothing + Laplacian + zero-crossing	Trung bình	Tốt, ổn định hơn Laplace	Trung bình	Giảm nhiễu tốt hơn Laplace, đẳng hướng	Kernel lớn, tốn tính toán
Canny	Biên là cực đại cục bộ của gradient sau khi làm trơn	Gaussian → Gradient → Non-maximum suppression → Hysteresis threshold	Cao	Rất tốt (mỏng, chính xác)	Rất tốt	Chuẩn vàng trong edge detection, ít biên giả	Phức tạp, nhiều tham số

VII. Chương 7: Phép biến đổi Fourier

Câu 1: Làm sao để chọn giá trị m hợp lý, sao để biết Error tối ưu?

1. Cơ sở để chọn và xác định cực tiểu mất mát (error tối ưu):

- Xác định Cực tiểu Mất mát (Error tối ưu):

- Trong Karhunen Loève (KLT), tổng sai số bình phương trung bình (Mean Squared Error) khi xấp xỉ vector đầu vào bằng sử dụng thành phần là tổng của các giá trị riêng m tương ứng với các thành phần không được sử dụng trong phép xấp xỉ.

- Cách chọn hợp lý:

- Để giảm thiểu sai số (Minimize error), ta phải chọn vector riêng tương ứng với m giá trị riêng lớn nhất. Bằng cách này, các thành phần bị bỏ qua ($i=m$ đến $N-1$) sẽ là những thành phần có giá trị riêng nhỏ nhất, tức là chúng chứa ít năng lượng hoặc thông tin nhất.

2. Điểm dừng hợp lý:

- KLT có đặc tính tập trung năng lượng vào các thành phần có giá trị riêng lớn. Các giá trị riêng thường được sắp xếp theo thứ tự giảm dần.
- Việc dừng ở một đủ bé sao cho các còn lại rất nhỏ là một chiến lược hiệu quả. Khi giảm rất nhanh, điều này ngụ ý rằng hầu hết thông tin quan trọng đã được giữ lại trong thành phần đầu tiên, và tổng các giá trị nhỏ còn lại sẽ đảm bảo sai số tái tạo đạt mức tối thiểu chấp nhận được.

Câu 2: Trình bày giải thuật Fourier?

1. Biến đổi Fourier Rời rạc (Discrete Fourier Transform - DFT)

Đây là thuật toán dùng để chuyển đổi ảnh từ miền không gian (spatial domain) sang miền tần số (frequency domain). Với một ảnh đầu vào $f(x, y)$ có kích thước $M \times N$, biến đổi Fourier rời rạc 2 chiều $F(u, v)$ được tính theo công thức:

$$F(u, v) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) e^{-i2\pi(\frac{ux}{M} + \frac{vy}{N})}$$

Trong đó:

- x, y : Tọa độ trong miền không gian ($x = 0, \dots, M-1, y = 0, \dots, N-1$).
- u, v : Tọa độ trong miền tần số ($u = 0, \dots, M-1, v = 0, \dots, N-1$).
- MN : Tổng số điểm ảnh.

2. Biến đổi Fourier Rời rạc Ngược (Inverse DFT - IDFT)

Đây là thuật toán tái tạo lại ảnh gốc từ miền tần số. Công thức biến đổi ngược để tìm lại $f(x, y)$ từ $F(u, v)$ như sau:

$$f(x, y) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u, v) e^{i2\pi(\frac{ux}{M} + \frac{vy}{N})}$$

3. Các Thành phần của Phổ Fourier

Hàm $F(u, v)$ là một số phức, có dạng tổng quát:

$$F(u, v) = R(u, v) + iI(u, v)$$

(với R là phần thực và I là phần ảo).

Các đại lượng quan trọng bao gồm:

- **Phổ Biên độ (Magnitude/Spectrum):** Thể hiện mức năng lượng tại mỗi tần số, thường được dùng để hiển thị trực quan phổ Fourier:

$$|F(u, v)| = \sqrt{R(u, v)^2 + I(u, v)^2}$$

- **Phổ Pha (Phase Angle):** Thể hiện thông tin về cấu trúc và vị trí các cạnh trong ảnh:

$$\phi(u, v) = \arctan \left[\frac{I(u, v)}{R(u, v)} \right]$$

- **Thành phần DC (Direct Current):** Tại gốc tọa độ (0,0), giá trị $F(0,0)$ tỉ lệ với độ sáng trung bình của toàn bộ ảnh.

4. Ứng dụng trong Lọc Ảnh (Frequency Domain Filtering)

Trong miền tần số, phép nhân chập (convolution) phức tạp ở miền không gian trở thành phép nhân điểm đơn giản.

Công thức lọc cơ bản:

$$G(u, v) = H(u, v) \times F(u, v)$$

Trong đó:

- $F(u, v)$: DFT của ảnh đầu vào.
- $H(u, v)$: Hàm truyền của bộ lọc (Filter Function) - quyết định giữ lại hay loại bỏ tần số nào.
- $G(u, v)$: DFT của ảnh kết quả sau khi lọc.

Quy trình lọc sử dụng Fourier Transform (4 bước):

1. Tiền xử lý: Lấy ảnh đầu vào $f(x, y)$.
2. Biến đổi thuận: Thực hiện DFT (thường dùng thuật toán FFT) để thu được $F(u, v)$.
3. Lọc: Nhân phổ ảnh với hàm bộ lọc: $G(u, v) = H(u, v)F(u, v)$.
4. Biến đổi ngược: Thực hiện IDFT lên $G(u, v)$ và lấy phần thực để thu được ảnh kết quả $g(x, y)$ ở miền không gian.

Câu 3: Trong Gaussian Lowpass Filter, khi D_0 càng lớn thì ảnh như thế nào?

Gaussian Lowpass Filter (Bộ lọc thông thấp Gaussian) được sử dụng để làm mịn (smoothing) ảnh trong miền tần số. Hàm bộ lọc Gaussian, $H(u, v)$, được định nghĩa như sau:

$$H(u, v) = e^{-\frac{D^2(u, v)}{2\sigma^2}}$$

Trong đó:

- $D(u, v)$ là khoảng cách từ điểm (u, v) đến tâm phổ.
- σ là độ lệch chuẩn. Trong ngữ cảnh lọc, thường thay thế bằng D_0 (với $D_0^2 = 2\sigma^2$) để $H(u, v) = 0.607$ tại vị trí cắt D_0 .

Tác động của D_0 (bán kính cutoff):

D_0 kiểm soát phạm vi của các tần số thấp được giữ lại:

- Nguyên lý: Bộ lọc thông thấp cho phép các tần số thấp tương ứng với vùng nền, thay đổi chậm đi qua và làm suy giảm các tần số cao tương ứng với chi tiết, cạnh sắc nét.
- Khi D_0 càng lớn: Dải tần số thấp mà bộ lọc cho phép đi qua càng rộng.
- Kết quả: Nếu D_0 rất lớn, bộ lọc sẽ giữ lại hầu hết các tần số bao gồm cả tần số cao. Điều này có nghĩa là bộ lọc làm mịn ít hơn. Ảnh đầu ra sẽ ít bị nhòe, giữ lại được nhiều chi tiết hơn và gần giống với ảnh gốc.
- Ngược lại: Khi D_0 nhỏ, bộ lọc chặn hầu hết tần số cao, dẫn đến ảnh bị nhòe mạnh.

Câu 4: Tìm những filter $H(u, v)$?

Trong xử lý ảnh, hàm bộ lọc $H(u, v)$ được sử dụng trong miền tần số để sửa đổi phổ Fourier $F(u, v)$. Với $D(u, v)$ là khoảng cách từ điểm (u, v) đến tâm phổ có hai nhóm chính:

1. Smoothing Frequency Domain Filters: Các bộ lọc này loại bỏ hoặc làm suy giảm tần số cao để làm mịn ảnh.

1.1 Ideal Lowpass Filters (ILPF)

Cắt đứt hoàn toàn các tần số cao hơn tần số cắt D_0 .

$$H(u, v) = \begin{cases} 1 & \text{nếu } D(u, v) \leq D_0 \\ 0 & \text{nếu } D(u, v) > D_0 \end{cases}$$

1.2 Butterworth Lowpass Filters (BLPF)

Chuyển đổi mượt hơn giữa tần số thấp và cao, ít gây hiện tượng gợn sóng.

$$H(u, v) = \frac{1}{1 + [D(u, v)/D_0]^{2n}}$$

1.3 Gaussian Lowpass Filters (GLPF)

Bộ lọc thông cao dạng Gaussian, chuyển tiếp mượt mà nhất, không gây hiện tượng ringing trong miền không gian.

$$H(u, v) = e^{-\frac{D^2(u, v)}{2\sigma^2}}$$

2. Sharpening Frequency Domain Filters

Các bộ lọc này làm suy giảm tần số thấp và làm nổi bật tần số cao.

2.1 Ideal Lowpass Filters (IHPF)

Cắt đứt hoàn toàn các tần số thấp hơn tần số cắt D_0 .

$$H(u, v) = \begin{cases} 0 & \text{nếu } D(u, v) \leq D_0 \\ 1 & \text{nếu } D(u, v) > D_0 \end{cases}$$

2.2 Butterworth Highpass Filters (BHPF)

Bộ lọc thông cao có sự chuyển đổi mượt mà.

$$H(u, v) = \frac{1}{1 + [D_0/D(u, v)]^{2n}}$$

2.3 Gaussian Highpass Filters (GHPF)

Bộ lọc thông cao dạng Gaussian, đảm bảo kết quả sắc nét mượt mà.

$$H(u, v) = 1 - e^{-\frac{D^2(u, v)}{2\sigma^2}}$$

2.4 Laplacian in Frequency Domain

Dùng toán tử Laplacian trong miền tần số để làm sắc nét ảnh.

$$H(u, v) = -(u^2 + v^2)$$