THỰC HÀNH VỀ XỬ LÝ HÌNH THÁI

1) Làm sạch văn bản quét (Opening để khử nhiễu muối tiêu)

- **Đề bài**: Cho ảnh tài liệu/bản in bị nhiễu muối tiêu, hãy loại bỏ nhiễu mà vẫn giữ nét chữ.

Mục tiêu: Nắm phép **mở** (erosion → dilation) để khử nhiễu hạt nhỏ.

Yêu cầu: So sánh kernel 3×3, 5×5; báo cáo PSNR/SSIM trước–sau (tùy chọn).

Hướng dẫn: Nhị phân hóa Otsu → MORPH_OPEN → đánh giá.

THỰC HÀNH VỀ XỬ LÝ HÌNH THÁI

```
import cv2, numpy as np, matplotlib.pyplot as plt

img = cv2.imread('docs/noisy_scan.png', 0)
_, bw = cv2.threshold(img, 0, 255, cv2.THRESH_BINARY + cv2.THRESH_OTSU)

for k in [3,5]:
    kernel = cv2.getStructuringElement(cv2.MORPH_RECT, (k,k))
    open_img = cv2.morphologyEx(bw, cv2.MORPH_OPEN, kernel)
    plt.figure();
    plt.subplot(1,3,1); plt.imshow(img, 'gray'); plt.title('Gốc')
    plt.subplot(1,3,2); plt.imshow(bw, 'gray'); plt.title('Nhị phân')
    plt.subplot(1,3,3); plt.imshow(open_img, 'gray'); plt.title(f'Opening
{k}x{k}')
plt.show()
```

THỰC HÀNH VỀ XỬ LÝ HÌNH THÁI

2) Lấp lỗ và nối nét (Closing để phục hồi vật thể)

- **Đề bài**: Ảnh linh kiện/biometric có lỗ nhỏ và khe hở; hãy lấp lỗ và nối nét.

Mục tiêu: Dùng **đóng** (dilation → erosion) để lấp lỗ nhỏ, hàn mép.

Yêu cầu: Chọn kernel phù hợp hình học (RECT/ELLIPSE/LINE).

Hướng dẫn: Nhị phân hóa → MORPH_CLOSE → so sánh diện tích vật thể.

THỰC HÀNH VỀ XỬ LÝ HÌNH THÁI

```
import cv2, numpy as np, matplotlib.pyplot as plt
img = cv2.imread('parts/gapped.png', 0)
_, bw = cv2.threshold(img, 0, 255, cv2.THRESH_BINARY+cv2.THRESH_OTSU)

kernel = cv2.getStructuringElement(cv2.MORPH_ELLIPSE, (7,7))
closed = cv2.morphologyEx(bw, cv2.MORPH_CLOSE, kernel)

plt.subplot(131); plt.imshow(img,'gray'); plt.title('Gốc')
plt.subplot(132); plt.imshow(bw,'gray'); plt.title('Nhị phân')
plt.subplot(133); plt.imshow(closed,'gray'); plt.title('Closing 7x7')
plt.show()
```

THỰC HÀNH VỀ XỬ LÝ HÌNH THÁI

3) Trích biên bằng đối ngẫu giãn–xói (Morphological Gradient)

Đề bài: Làm nổi đường biên của chi tiết công nghiệp hoặc tế bào.

Mục tiêu: Dùng **gradient hình thái** = dilation − erosion. Liên hệ câu hỏi “phát hiện đường biên? giải pháp” trong slide.

Yêu cầu: So với Canny; báo cáo sự khác biệt biên thô/mịn.

Hướng dẫn: MORPH_GRADIENT với kernel 3×3.

THỰC HÀNH VỀ XỬ LÝ HÌNH THÁI

```
import cv2, numpy as np, matplotlib.pyplot as plt
img = cv2.imread('objects/sample.png', 0)
_, bw = cv2.threshold(img,0,255,cv2.THRESH_BINARY+cv2.THRESH_OTSU)
kernel = cv2.getStructuringElement(cv2.MORPH_RECT,(3,3))
grad = cv2.morphologyEx(bw, cv2.MORPH_GRADIENT, kernel)
edges = cv2.Canny(img, 50, 150)
plt.figure(figsize=(10,3))
plt.subplot(131); plt.imshow(bw,'gray'); plt.title('Nhị phân')
plt.subplot(132); plt.imshow(grad,'gray'); plt.title('Morph. gradient')
plt.subplot(133); plt.imshow(edges,'gray'); plt.title('Canny')
plt.show()
```

THỰC HÀNH VỀ XỬ LÝ HÌNH THÁI

4) Đếm đồng xu/viên nén dính nhau (Opening + Distance Transform + Watershed)

Đề bài: Ảnh sản phẩm dạng “viên” chạm nhau; hãy tách và **đếm** số lượng.

Mục tiêu: Giải quyết “đếm số đồng xu khi chạm nhau” nêu trong slide.

Yêu cầu: Minh họa mask “sure foreground / sure background”.

Hướng dẫn: Otsu → Opening → Distance Transform → Watershed.

THỰC HÀNH VỀ XỬ LÝ HÌNH THÁI

```
import cv2, numpy as np

img = cv2.imread('coins/touching.jpg')
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
_, th = cv2.threshold(gray, 0, 255, cv2.THRESH_BINARY_INV+cv2.THRESH_OTSU)

kernel = np.ones((3,3), np.uint8)
opening = cv2.morphologyEx(th, cv2.MORPH_OPEN, kernel, iterations=2)
sure_bg = cv2.dilate(opening, kernel, iterations=3)
dist = cv2.distanceTransform(opening, cv2.DIST_L2, 5)
_, sure_fg = cv2.threshold(dist, 0.5*dist.max(), 255, 0)
sure_fg = np.uint8(sure_fg)
unknown = cv2.subtract(sure_bg, sure_fg)

_, markers = cv2.connectedComponents(sure_fg)
markers = markers + 1
markers[unknown==255] = 0
markers = cv2.watershed(img, markers)
count = len(np.unique(markers)) - 2  # trừ nền & biên

print("Số đối tượng:", count)
```

THỰC HÀNH VỀ XỬ LÝ HÌNH THÁI

5) Phân đoạn ký tự biển số/seri (Opening/Closing + CC)

Đề bài: Cho ảnh biển số hoặc tem seri in, hãy tách ký tự để OCR.

Mục tiêu: Dùng **mở/đóng** sạch nền, sau đó Connected Components.

Yêu cầu: Loại bỏ “dấu chấm/bavia” bằng opening; nối nét bằng closing.

Hướng dẫn: Thử RECT vs ELLIPSE kernels tùy chiều nét.

THỰC HÀNH VỀ XỬ LÝ HÌNH THÁI

```
import cv2, numpy as np, matplotlib.pyplot as plt
img = cv2.imread('plates/plate.jpg', 0)
_, bw = cv2.threshold(img, 0, 255, cv2.THRESH_BINARY_INV+cv2.THRESH_OTSU)
bw = cv2.morphologyEx(bw, cv2.MORPH_OPEN,
cv2.getStructuringElement(cv2.MORPH_RECT,(3,3)))
bw = cv2.morphologyEx(bw, cv2.MORPH_CLOSE,
cv2.getStructuringElement(cv2.MORPH_RECT,(5,5)))

n, labels = cv2.connectedComponents(bw)
print("Số thành phần (kể cả nền):", n)
```

THỰC HÀNH VỀ XỬ LÝ HÌNH THÁI

6) Đo đường kính hạt/lỗ trên bề mặt (morphology + đo đạc)

Đề bài: Ảnh bề mặt có lỗ/hạt; hãy phân cụm theo **kích thước** (nhỏ–vừa–lớn).

Mục tiêu: Gắn với bài tập "đếm hình tròn theo 3 nhóm kích thước".

Yêu cầu: Tính diện tích contour, chia ngưỡng 3 nhóm, đếm từng nhóm.

Hướng dẫn: Closing để làm tròn; findContours → diện tích.

THỰC HÀNH VỀ XỬ LÝ HÌNH THÁI

```
import cv2, numpy as np
img = cv2.imread('surface/holes.png', 0)
_, bw = cv2.threshold(img, 0, 255, cv2.THRESH_BINARY_INV+cv2.THRESH_OTSU)
bw = cv2.morphologyEx(bw, cv2.MORPH_CLOSE, np.ones((3,3),np.uint8),
iterations=2)
contours,_ = cv2.findContours(bw, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
areas = sorted([cv2.contourArea(c) for c in contours])
t1, t2 = np.percentile(areas,[33,66])
small = sum(a<=t1 for a in areas); mid = sum((a>t1)&(a<=t2) for a in areas); big
= sum(a>t2 for a in areas)
print("Nhỏ:", small, " Vừa:", mid, " Lớn:", big)
```

THỰC HÀNH VỀ XỬ LÝ HÌNH THÁI

7) Xóa pixel thừa ở cạnh (pruning bằng Hit-or-Miss)

Đề bài: Ảnh biên/skeleton còn gai (spurs), hãy xóa pixel thừa thành mép gọn.

Mục tiêu: Thực hiện yêu cầu “xóa các pixel thừa ở cạnh… nêu rõ SE & bước”.

Yêu cầu: Dùng cv2.MORPH_HITMISS với bộ SE xoay 8 hướng.

Hướng dẫn: Chuẩn hóa nhị phân {0,1}; lặp hit-or-miss và trừ khỏi ảnh.

THỰC HÀNH VỀ XỬ LÝ HÌNH THÁI

```
import cv2, numpy as np

img = cv2.imread('edges/jagged.png', 0)
_, bw = cv2.threshold(img,0,1,cv2.THRESH_BINARY+cv2.THRESH_OTSU)

SEs = []
base = np.array([[0,0,0],
                 [-1,1,-1],
                 [1,1,1]], dtype=np.int8)  # -1=don't care
for k in range(4):
    SEs.append(np.rot90(base, k))
    SEs.append(np.rot90(np.fliplr(base), k))

changed = True
while changed:
    before = bw.copy()
    for se in SEs:
        hm = cv2.morphologyEx(bw.astype(np.uint8), cv2.MORPH_HITMISS, se)
        bw = np.where(hm==1, 0, bw)
    changed = np.any(before!=bw)

cv2.imwrite('edges/pruned.png', (bw*255).astype('uint8'))
```

THỰC HÀNH VỀ XỬ LÝ HÌNH THÁI

8) Tách tiền cảnh trong ảnh công nghiệp (Foreground = obj − erosion)

Đề bài: Từ ảnh băng chuyền có vật thể trên nền, tách **vùng tiền cảnh** ổn định.

Mục tiêu: Dùng xói mòn để xấp xỉ "core" của đối tượng, rồi lấy hiệu để ra biên/tiền cảnh (liên hệ tách biên A − (A ⊖ B) trong bài tập 7).

Yêu cầu: Trả 2 lớp: **core** và **rim**.

Hướng dẫn: core = erode(A,B); rim = A - core.

THỰC HÀNH VỀ XỬ LÝ HÌNH THÁI

```
import cv2, numpy as np
img = cv2.imread('conveyor/items.png', 0)
_, A = cv2.threshold(img, 0, 255, cv2.THRESH_BINARY+cv2.THRESH_OTSU)
B = cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(5,5))
core = cv2.erode(A, B)
rim = cv2.subtract(A, core)
cv2.imwrite('conveyor/core.png', core)
cv2.imwrite('conveyor/rim.png', rim)
```

THỰC HÀNH VỀ XỬ LÝ HÌNH THÁI

9) Khử nền không đồng đều (Grayscale Top-hat / Black-hat)

Đề bài: Ảnh tài liệu/PCB có chiếu sáng không đều; hãy loại bỏ nền sáng/tối.

Mục tiêu: Áp dụng **morphology mức xám**: top-hat (χ = Img − opening) và black-hat (β = closing − Img).

Yêu cầu: So sánh histogram trước–sau; thử kernel 15×15.

Hướng dẫn: Dùng MORPH_TOPHAT và MORPH_BLACKHAT.

THỰC HÀNH VỀ XỬ LÝ HÌNH THÁI

```
import cv2, numpy as np, matplotlib.pyplot as plt
img = cv2.imread('docs/uneven.jpg', 0)
kernel = cv2.getStructuringElement(cv2.MORPH_RECT,(15,15))
tophat = cv2.morphologyEx(img, cv2.MORPH_TOPHAT, kernel)
blackhat = cv2.morphologyEx(img, cv2.MORPH_BLACKHAT, kernel)
corrected = cv2.normalize(img + tophat - blackhat, None, 0, 255, cv2.NORM_MINMAX)
plt.figure(figsize=(10,3))
plt.subplot(131); plt.imshow(img,'gray'); plt.title('Gốc')
plt.subplot(132); plt.imshow(tophat,'gray'); plt.title('Top-hat')
plt.subplot(133); plt.imshow(corrected,'gray'); plt.title('Điều chỉnh')
plt.show()
```