INDUSTRIAL AI RESEARCH CENTER

산업컴퓨터비전실제

Homework #1

충북대학교 산업인공지능학과 사수진

Dept. of Industrial AI

차례

1. 히스토그램 평탄화

INDUSTRIAL AI RESEARCH CENTER

소스코드

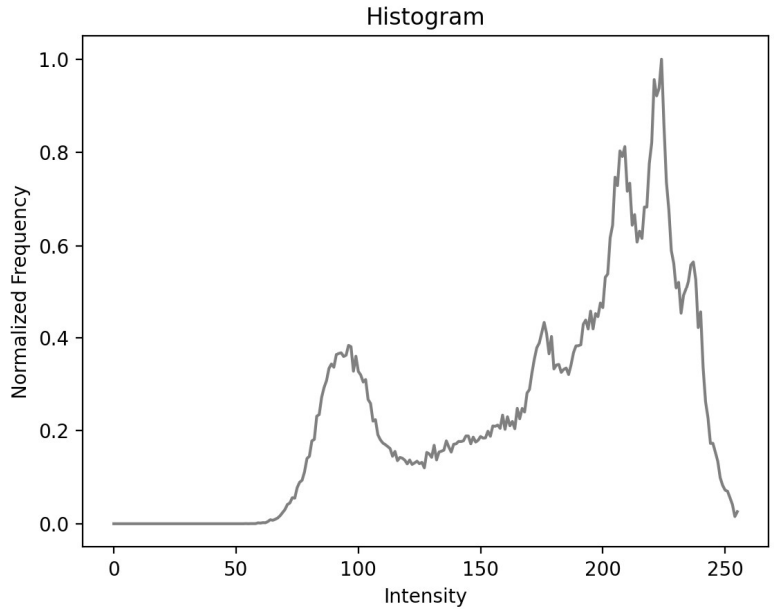
```python
import cv2
import numpy as np
import matplotlib.pyplot as plt

# 히스토그램 평탄화 함수
# 1개의 사용 위치
def histogram_equalization(image):
    # Check if image is color (3 channels) or grayscale
    if len(image.shape) == 3 and image.shape[2] == 3:
        # Convert image to grayscale
        gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    else:
        gray_image = image  # Assuming input is already grayscale

    # Equalize histogram
    equalized_image = cv2.equalizeHist(gray_image)

    # Calculate histogram
    hist = cv2.calcHist([gray_image], [0], None, [256], [0, 256])

    # Normalize histogram
    hist_normalized = hist.ravel() / hist.max()

    return equalized_image, hist_normalized

# HSV 컬러 스페이스에서 V 채널에 대한 히스토그램 평탄화 함수
# 1개의 사용 위치
def hsv_value_equalization(image):
    hsv_image = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
    h, s, v = cv2.split(hsv_image)
    equalized_v = cv2.equalizeHist(v)
    equalized_hsv_image = cv2.merge([h, s, equalized_v])
    equalized_color_image = cv2.cvtColor(equalized_hsv_image, cv2.COLOR_HSV2BGR)
    hist = cv2.calcHist([v], [0], None, [256], [0, 256])
    hist_normalized = hist.ravel() / hist.max()
    return equalized_color_image, hist_normalized

# 이미지 불러오기
input_image_path = '../data/lena.png'
input_image = cv2.imread(input_image_path)

# 이미지의 채널 수가 3개인지 확인
if input_image is None or input_image.shape[2] != 3:
    print("Error: Unable to load image or input image does not have 3 channels (BGR).")
    exit()

# 사용자로부터 채널을 선택하도록 입력
channel = input("Enter the channel to perform histogram equalization (R/G/B): ").upper()

# 사용자 입력에 따라 히스토그램 평활화를 수행함
if channel in ['R', 'G', 'B']:
    channel_index = {'R': 2, 'G': 1, 'B': 0}[channel]
    equalized_image, hist_normalized = histogram_equalization(input_image[:, :, channel_index])
    # Display histogram
    plt.plot(hist_normalized, color='gray')
    plt.xlabel('Intensity')
    plt.ylabel('Normalized Frequency')
    plt.title('Histogram')
    plt.show()
    # Display histogram equalized image
    cv2.imshow("Histogram Equalized Image", equalized_image)
    cv2.waitKey(0)
    cv2.destroyAllWindows()
elif channel == 'V':
    hsv_equalized_image, hist_normalized_hsv = hsv_value_equalization(input_image)
    # Display histogram
    plt.plot(hist_normalized_hsv, color='gray')
    plt.xlabel('Intensity')
    plt.ylabel('Normalized Frequency')
    plt.title('HSV Value Histogram')
    plt.show()
    # Display HSV value equalized image
    cv2.imshow("HSV Value Equalized Image", hsv_equalized_image)
    cv2.waitKey(0)
    cv2.destroyAllWindows()
else:
    print("Invalid channel selection.")
```

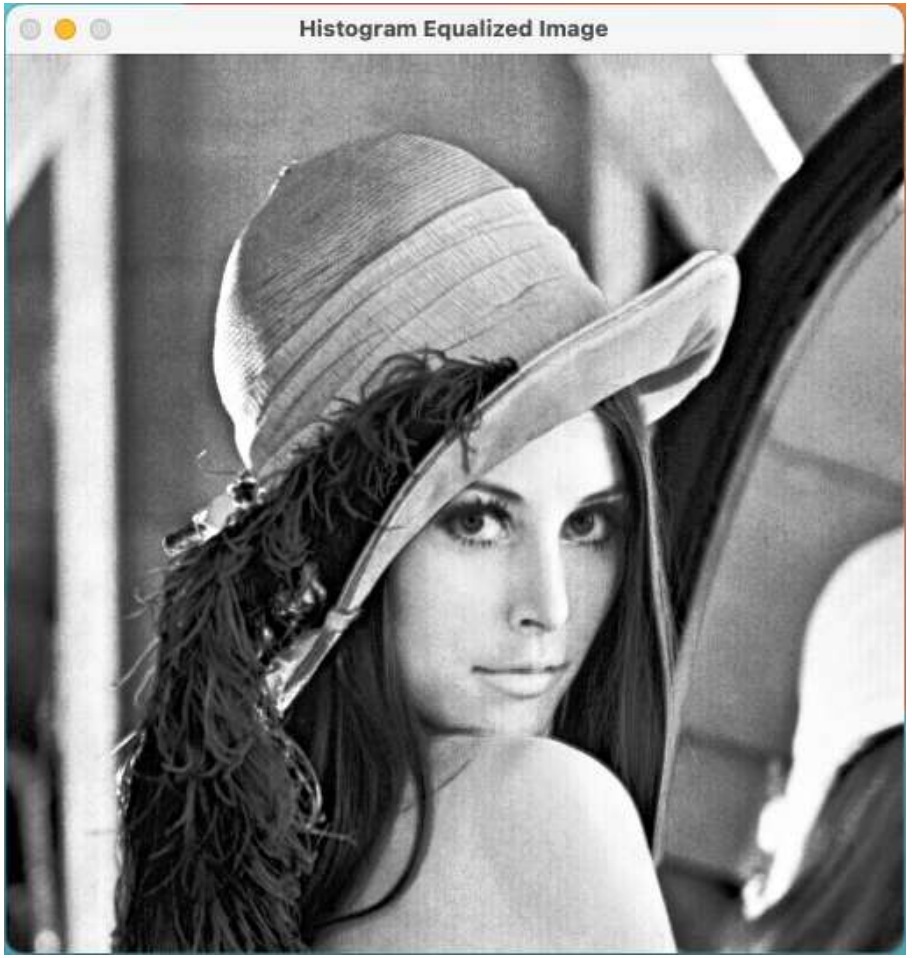
1. 히스토그램 평탄화

INDUSTRIAL AI RESEARCH CENTER

- 결과화면

  R 입력시



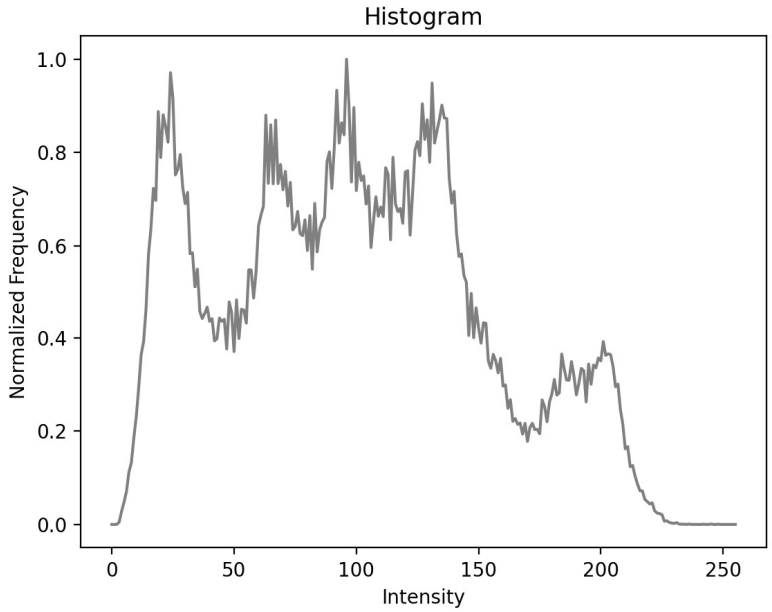


1. 히스토그램 평탄화

INDUSTRIAL AI RESEARCH CENTER

- 결과화면

  G 입력시

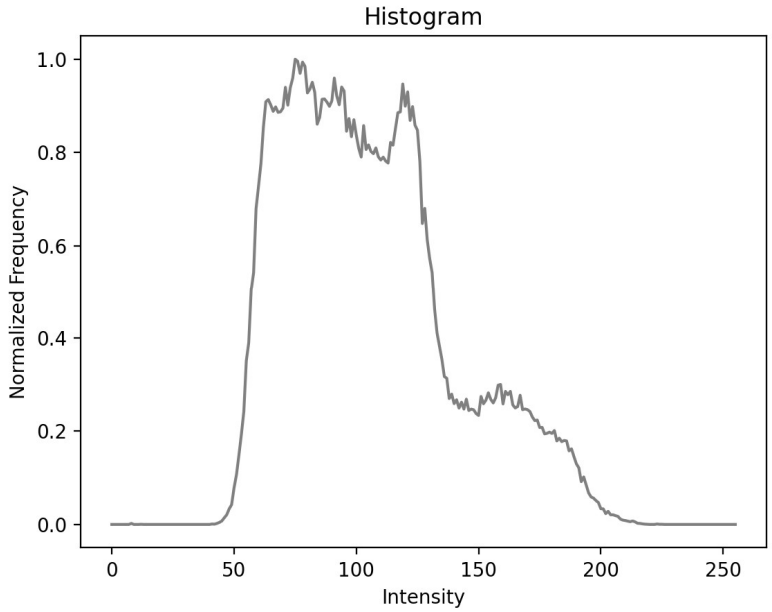


1. 히스토그램 평탄화

INDUSTRIAL AI RESEARCH CENTER

- 결과화면

  B 입력시



2. 공간 도메인 필터링

INDUSTRIAL AI RESEARCH CENTER

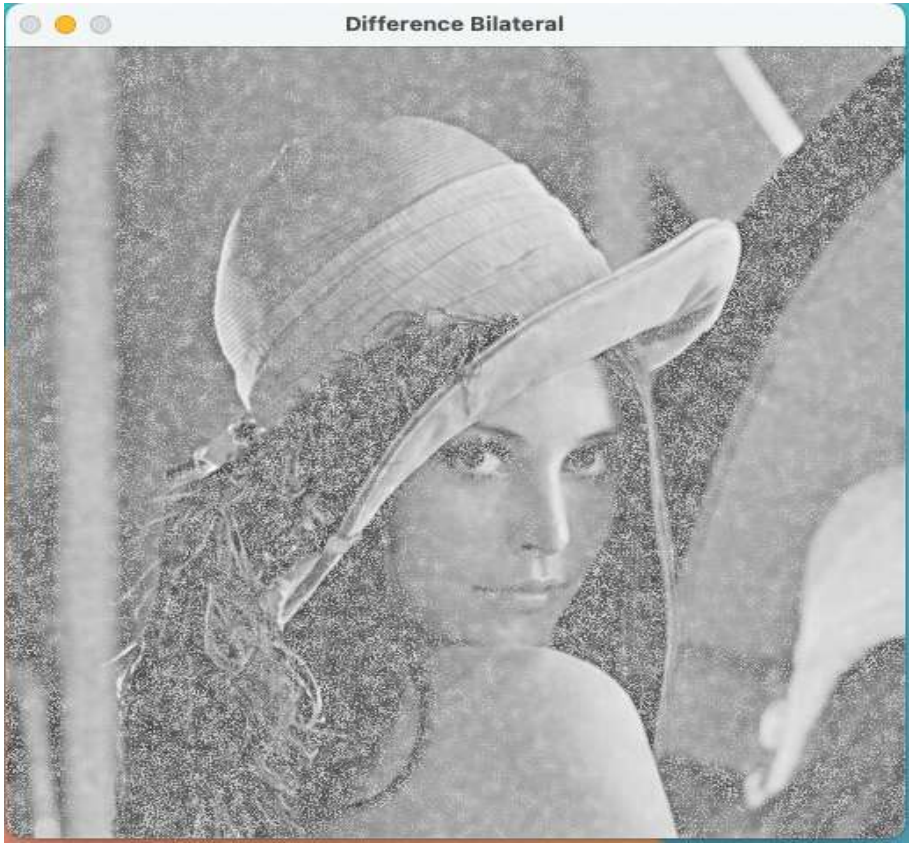
소스코드

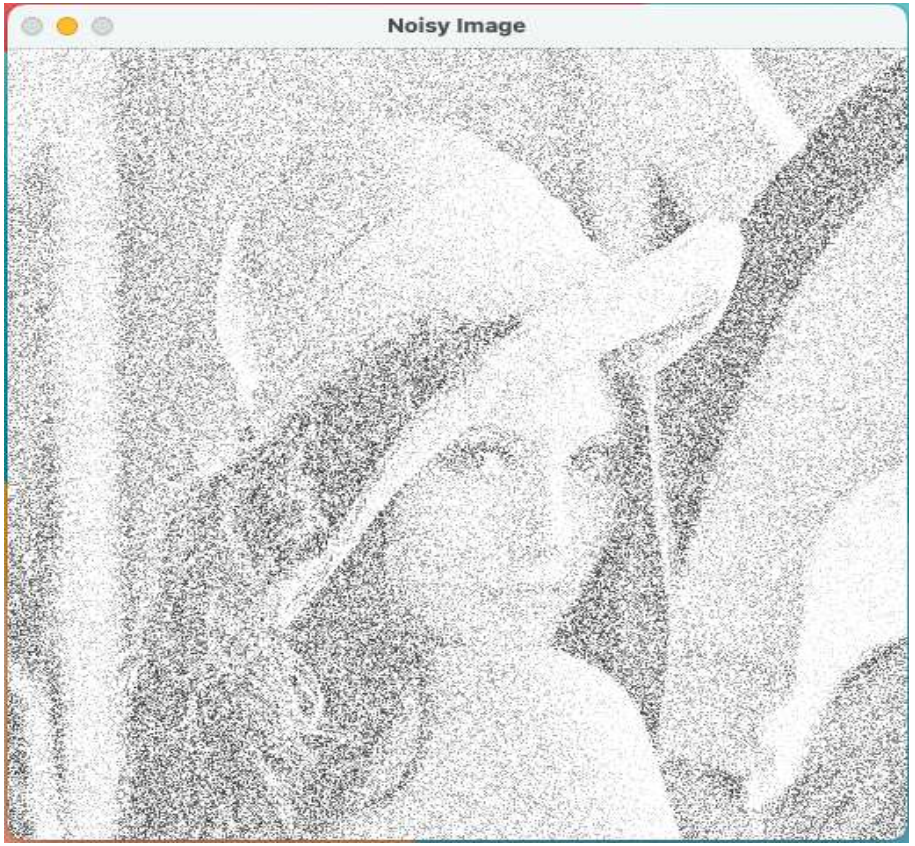
```python
import cv2
import numpy as np

# 이미지 불러오기
image = cv2.imread( filename: './data/Lena.png', cv2.IMREAD_GRAYSCALE)

# 임의의 노이즈 생성
noise = np.random.normal(loc=0, scale=150, size=image.shape).astype(np.uint8)
noisy_image = cv2.add(image, noise)

# 필터링
gaussian_filtered = cv2.GaussianBlur(noisy_image, ksize: (5, 5), sigmaX: 0)
median_filtered = cv2.medianBlur(noisy_image, ksize: 5)
bilateral_filtered = cv2.bilateralFilter(noisy_image, d: 9, sigmaColor: 75, sigmaSpace: 75)

# 결과 출력
cv2.imshow( winname: 'Noisy Image', noisy_image)
cv2.imshow( winname: 'Gaussian Filtered', gaussian_filtered)
cv2.imshow( winname: 'Median Filtered', median_filtered)
cv2.imshow( winname: 'Bilateral Filtered', bilateral_filtered)

# 입력 영상과의 차이 계산 및 출력
cv2.imshow( winname: 'Difference Gaussian', np.abs(image - gaussian_filtered))
cv2.imshow( winname: 'Difference Median', np.abs(image - median_filtered))
cv2.imshow( winname: 'Difference Bilateral', np.abs(image - bilateral_filtered))

cv2.waitKey(0)
cv2.destroyAllWindows()
```

2. 공간 도메인 필터링

INDUSTRIAL AI RESEARCH CENTER

결과화면

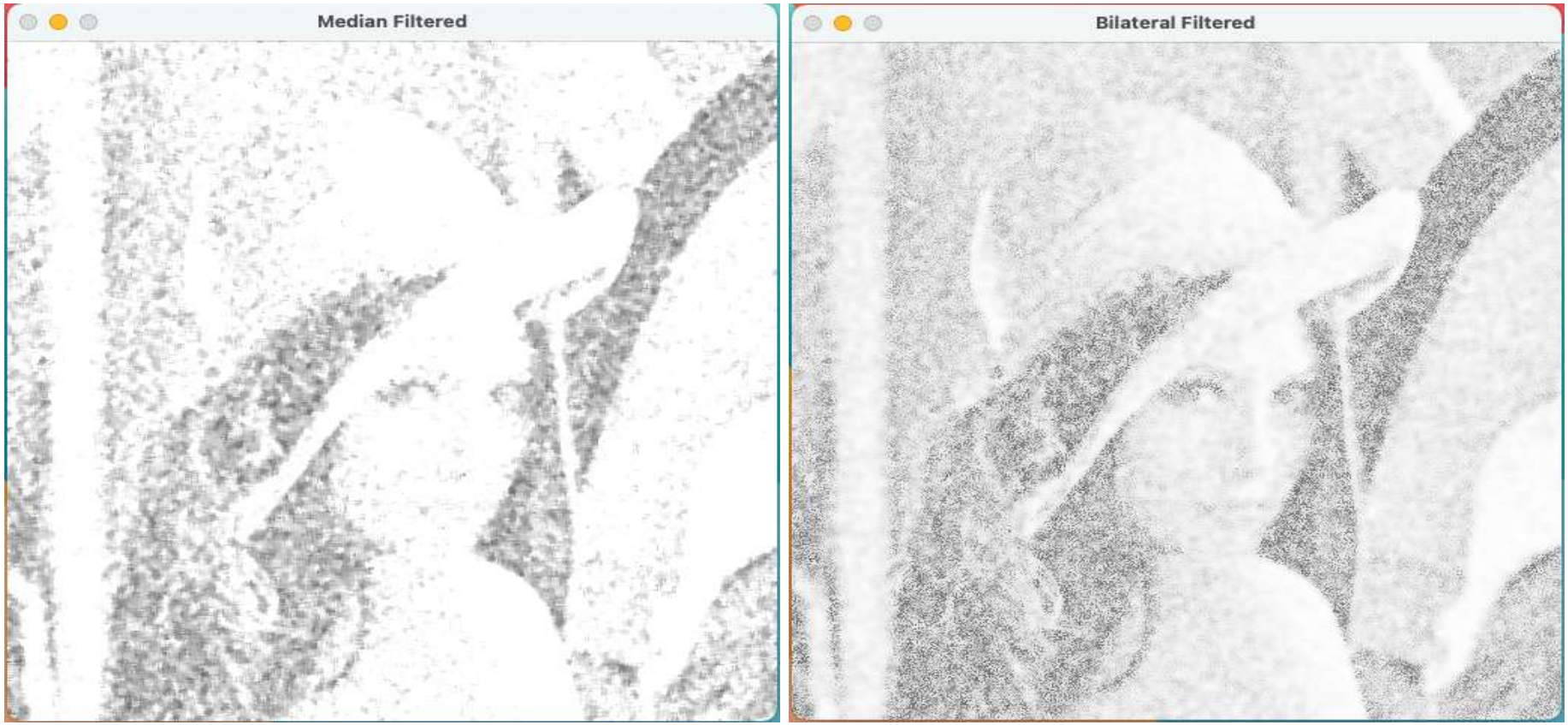




2. 공간 도메인 필터링

INDUSTRIAL AI RESEARCH CENTER

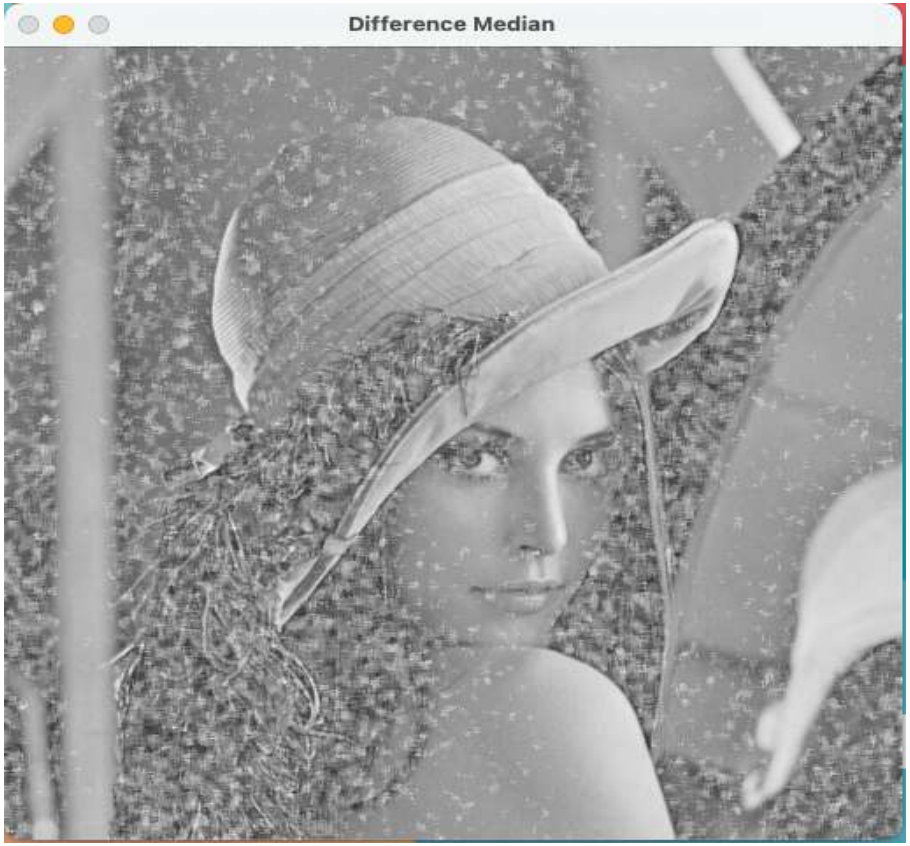
■ 결과화면



2. 공간 도메인 필터링

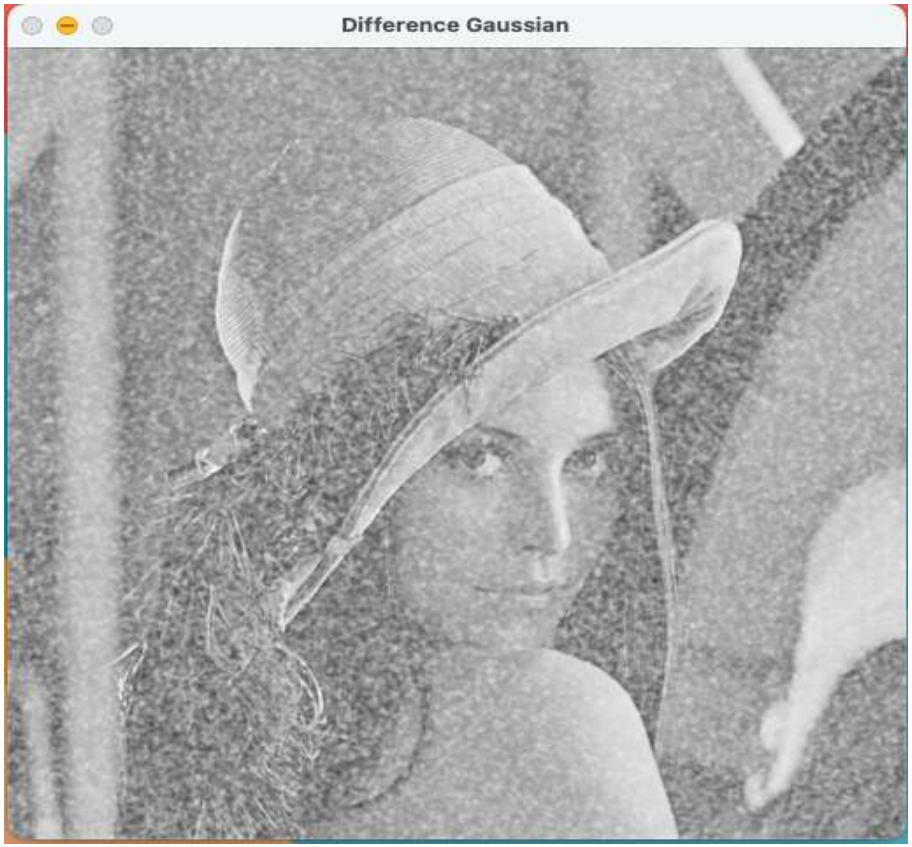
INDUSTRIAL AI RESEARCH CENTER

결과화면



2. 공간 도메인 필터링

결과화면



3. 주파수 도메인 필터링

INDUSTRIAL AI RESEARCH CENTER

소스코드

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

# 이미지 불러오기
image_path = './data/Lena.png'
image = cv2.imread(image_path, cv2.IMREAD_GRAYSCALE)

# 이미지 크기 축소
resized_image = cv2.resize(image,  dsize: (0, 0), fx=0.5, fy=0.5)

# DFT를 위한 함수 정의
# 2개의 사용 위치
def dft(img):
    # 이미지 데이터 타입을 float32로 변환
    img_float32 = np.float32(img)
    # 이미지 데이터를 FFT 수행
    f = cv2.dft(img_float32, flags=cv2.DFT_COMPLEX_OUTPUT)
    # 결과의 크기 스펙트럼 계산
    magnitude_spectrum = 20 * np.log(cv2.magnitude(f[:, :, 0], f[:, :, 1]))
    return magnitude_spectrum

# 주파수 도메인으로 변환
dft_img = dft(resized_image)

# 입력 받은 반지름
r1 = int(input("첫 번째 원의 반지름 입력: "))
r2 = int(input("두 번째 원의 반지름 입력: "))

# 중심 좌표 및 크기 계산
rows, cols = resized_image.shape
center_row, center_col = rows // 2, cols // 2
x, y = np.ogrid[:rows, :cols]

# 첫 번째 원의 반지름을 기준으로 원 밖의 영역을 선택하는 마스크 생성
mask1 = np.sqrt((x - center_row)**2 + (y - center_col)**2) > r1

# 두 번째 원의 반지름을 기준으로 원 안의 영역을 선택하는 마스크 생성
mask2 = np.sqrt((x - center_row)**2 + (y - center_col)**2) < r2

# 원 밖의 영역을 선택하는 마스크와 원 안의 영역을 선택하는 마스크를 AND 연산하여 bandpass 필터 생성
bandpass_filter = np.logical_and(mask1, mask2)

# 필터링
dft_img_filtered = dft_img * bandpass_filter

# 역 DFT
f_ishift = np.fft.ifftshift(dft_img_filtered)
img_back = np.fft.ifft2(f_ishift)
img_back = np.abs(img_back)

# 결과 출력
plt.subplot(121),plt.imshow(resized_image, cmap_='gray')
plt.title('Input Image'), plt.xticks([]), plt.yticks([])
plt.subplot(122),plt.imshow(img_back, cmap_='gray')
plt.title('Band Pass Filtered Image'), plt.xticks([]), plt.yticks([])
plt.show()
```

3. 주파수 도메인 필터링

INDUSTRIAL AI RESEARCH CENTER

- 결과화면

4. 모폴로지 필터

INDUSTRIAL AI RESEARCH CENTER

소스코드

```python
import cv2
import numpy as np

# 이미지 불러오기
image = cv2.imread( filename: './data/Lena.png', cv2.IMREAD_GRAYSCALE)

# 사용자 입력 받기
binary_method = input("이진화 방법을 선택하세요 (otsu 또는 adaptive median): ")
morphology_operation = input("적용할 모폴로지 연산을 선택하세요 (erosion, dilation, opening, closing): ")
iterations = int(input("모폴로지 연산을 적용할 횟수를 입력하세요: "))

# 이진화
if binary_method == 'otsu':
    _, binary_image = cv2.threshold(image, thresh: 0, maxval: 255, cv2.THRESH_BINARY + cv2.THRESH_OTSU)
else:
    binary_image = cv2.adaptiveThreshold(image, maxValue: 255, cv2.ADAPTIVE_THRESH_MEAN_C, cv2.THRESH_BINARY, blockSize: 11, C: 2)

# 모폴로지 연산 수행
kernel = np.ones( shape: (3,3),np.uint8)
morph_operations = {'erosion': cv2.erode, 'dilation': cv2.dilate, 'opening': cv2.morphologyEx, 'closing': cv2.morphologyEx}
if morphology_operation in morph_operations:
    result = morph_operations[morphology_operation](binary_image, kernel, iterations=iterations)

# 결과 출력
cv2.imshow( winname: 'Original Image', image)
cv2.imshow( winname: 'Binary Image', binary_image)
cv2.imshow( winname: 'Morphology Result', result)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

4. 모폴로지 필터

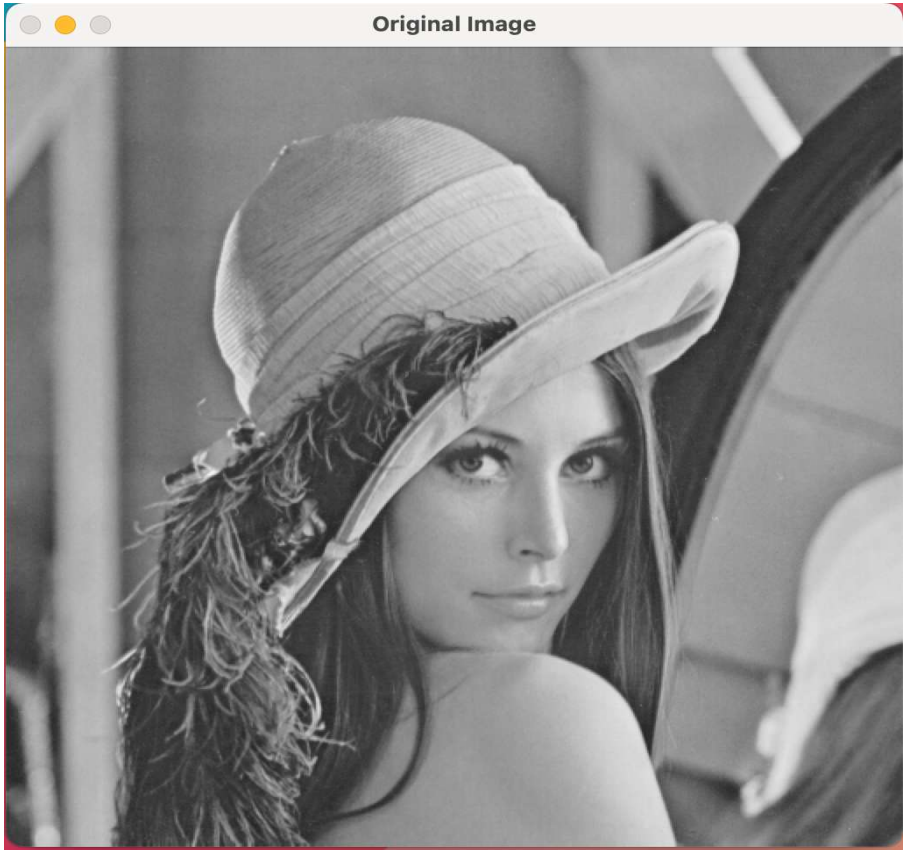
INDUSTRIAL AI RESEARCH CENTER

결과화면

4. 모폴로지 필터

INDUSTRIAL AI RESEARCH CENTER

결과화면



Thank You!