

Лаборатори №2 (Сүүлийн хэсэг)

Ц. Цолмон

ХШУИС-МКУТ, Компьютерийн Ухаан 3-р түвшний оюутан

tsoomoo446@gmail.com

Лаборатори №2 (Сүүлийн хэсэг)	1
1. ОРШИЛ	2
2. ЗОРИЛГО	2
3. ОНОЛЫН СУДАЛГАА	2
3.1 Median Filter	2
3.2 Using Second-Derivative for Image Sharpening	2
3.3 Unsharp masking and Highboost filtering	3
3.4 Low-Pass filter (LPF)	4
4. Хэрэгжүүлэлт	4
4.1 Median Filter	4
4.2 Using Second-Derivative for Image Sharpening	6
4.3 Unsharp masking and High-boost filtering	7
4.4 Combining Spatial Enhancement Methods	10
4.5 Notch Filtering	13
5. Эх сурвалж	14
6. Хавсралт	14

1. ОРШИЛ

Лекцээр үзсэн шүүлтүүрүүдийг задгайгаар (өөрөө бичих) болон бэлэн функц ашиглан хэрэгжүүлэх

2. ЗОРИЛГО

- OpenCV2 ба numpy сангуйудын талаар судлах
- Python хэл дээр cv2, numpy, matplotlib ашиглан өгөгдсөн даалгавруудыг гүйцэтгэх

3. ОНОЛЫН СУДАЛГАА

3.1 Median Filter

Smoothing spatial filter - зургийг бүдгэрүүлийх ба шуугианыг арилгахад ашигладаг.

Дотор нь шугаман ба шугаман бус гэж 2 хуваадаг. Медиан филтер нь шугаман бус филтер маск дахь утгуудыг эрэмбэлэн тухайн эрэмбэ дээр үндэслэн голын пикселийн утгыг тодорхойлдог.

Жишээ нь: 3x3 маскын бүх утгыг эрэмбэлж үзээд мэдийн нь 20 тул голын пикселд 20 гэсэн утгыг өгнө.

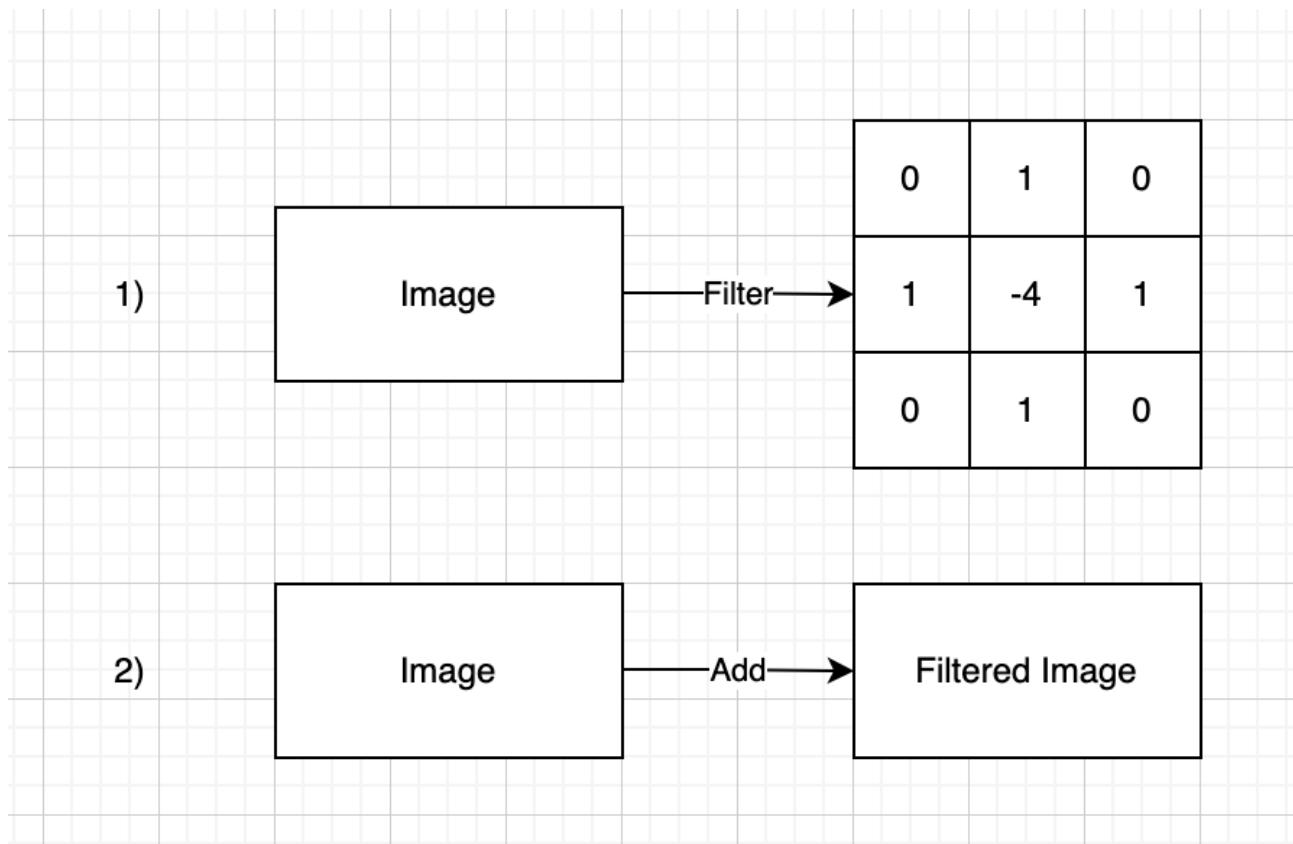
20	10	15
20	100	20
20	20	25

10, 15, 20, 20, **20**, 20, 20, 25, 100

3.2 Using Second-Derivative for Image Sharpening

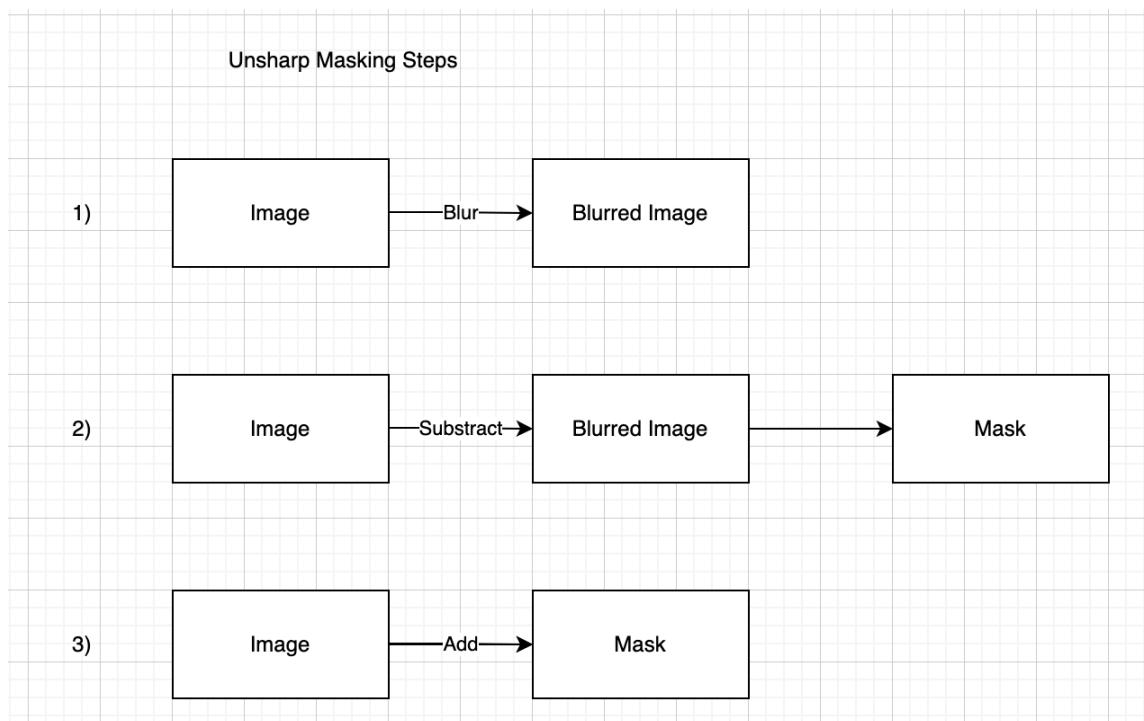
Image Sharpening - нь зураг дээрх өнцөг, нарийн шугаман деталь гэх мэт зүйлсийг илүү тодотгодог. Гол санаа нь пикселүүдийн intensity-гын зөрүү гарч байгаа хэсгүүдийг олж sharpening хийдэг.

Хамгийн түгээмэл аргуудын нэг нь анхны зураг дээр Лапласиан гэх мэт кернелүүдийг ашиглан өнцөг, шугам нь илүү тодорсон зураг үүсгээд уг зурагаа анхны зураг дээр нэмж sharpening хийдэг.



3.3 Unsharp masking and Highboost filtering

Unsharp masking нь sharpening хийсэн зургийг буцаан илүү smooth болгох буюу sharpening-ын яг эсрэг үйлдэл юм.



Маскаа үржүүлэхдээ $k > 1$ үед Highboost filtering болно.

3.4 Low-Pass filter (LPF)

Low-Pass Filter нь зургийг smoothing хийхэд ашигладаг шүүлтүүр бөгөөд Фурьеийн хувиргалт ашиглан зургийн давтамжийн хэсгүүдийг өөрчлөдөг.

4. Хэрэгжүүлэлт

4.1 Median Filter

cv2.median(array) - OpenCV медиан олох функц

Параметрүүд:

- array - олох гэж буй медианы хүснэгт

np.zeros_like(img,dtype) - Numpy сангийн шинэ 0 массив үүсгэх функц

Параметрүүд:

- img - img-тэй адилхан урт, өргөнтэй массив үүсгэх параметр
- dtype - массивийн элементийн төрлийг зааж өгөх параметр

cv2.hconcat - OpenCV олон зурагнуудыг horizontal буюу хэвтээ тэнхлэгийн дагуу нийлүүлэх функц

cv2.vconcat - OpenCV олон зурагнуудыг vertical буюу босоо тэнхлэгийн дагуу нийлүүлэх функц

```
Median filtering

import cv2
import numpy as np

# Load the input image
image = cv2.imread("../Images/motherboard.tif", cv2.IMREAD_GRAYSCALE)

def custom_avg(image, kernel):
    img_height, img_width = image.shape
    kernel_height, kernel_width = kernel.shape

    # padding олно. Мөн Zero-padding арга хэрэглэнэ.
    pad_height = kernel_height // 2
    pad_width = kernel_width // 2
    output_image = np.zeros_like(image)
    for y in range(pad_height, img_height - pad_height):
        for x in range(pad_width, img_width - pad_width):
            # Хөрш пикселүүдийн утгыг олоод кернелээрээ үржүүлнэ.
            neighborhood = image[y - pad_height:y + pad_height + 1, x - pad_width:x + pad_width + 1]
            result = np.sum(neighborhood * kernel)
            output_image[y, x] = result

def custom_median(image, kernel):
    img_height, img_width = image.shape
    kernel_height, kernel_width = kernel.shape

    # padding олно. Мөн Zero-padding арга хэрэглэнэ.
    pad_height = kernel_height // 2
    pad_width = kernel_width // 2
    output_image = np.zeros_like(image)
    for y in range(pad_height, img_height - pad_height):
        for x in range(pad_width, img_width - pad_width):
            # Хөрш пикселүүдийн утгыг олоод кернелээрээ үржүүлнэ.
            neighborhood = image[y - pad_height:y + pad_height + 1, x - pad_width:x + pad_width + 1]
            output_image[y, x] = np.median(neighborhood)

    return output_image

size = 3

results = [image]
results.append(image)

kernel = np.ones((size, size), np.float32) / (size * size)

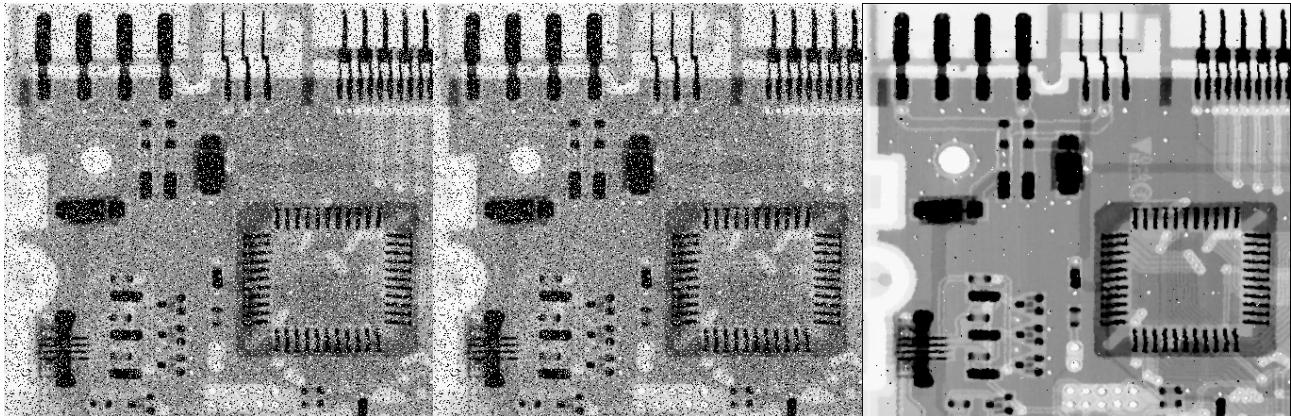
# Задгай
custom_median_img = custom_median(image, kernel)
custom_avg_img = custom_avg(image, kernel)

results.append(custom_avg_img)
results.append(custom_median_img)

result = cv2.hconcat(results)

cv2.imwrite("cv_result.png", result)
```

ҮР ДҮН :



4.2 Using Second-Derivative for Image Sharpening

cv2.filter2D(image, ddepth, kernel) - OpenCV зураг resize хийх функц

Параметрүүд :

- image - filter хийх зураг
- kernel - filter хийх kernel

```
Median filtering

import cv2
import numpy as np

image = cv2.imread('../Images/moon.tif', cv2.IMREAD_GRAYSCALE)

scale_factor = 10.0
kernel = np.array([[0, 1, 0], [1, -4, 1], [0, 1, 0]])
kernel_d = np.array([[1, 1, 1], [1, -8, 1], [1, 1, 1]])

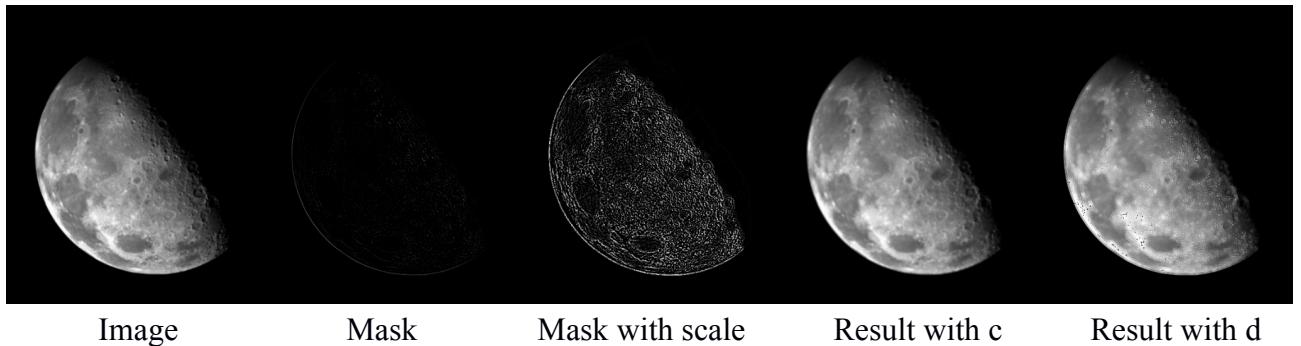
laplacian_result = cv2.filter2D(image, -1, kernel)
scaled = scale_factor * laplacian_result
laplacian_result = np.uint8(np.absolute(laplacian_result))
scaled_result = np.uint8(np.absolute(scaled))

laplacian_c = cv2.filter2D(image, -1, kernel)
laplacian_d = cv2.filter2D(image, -1, kernel_d)
result_c = image + np.uint8(np.absolute(laplacian_c))
result_d = image + np.uint8(np.absolute(laplacian_d))

result = cv2.hconcat([image, laplacian_result, scaled_result, result_c, result_d])

cv2.imwrite('result.png', result)
```

ҮР ДҮН :



4.3 Unsharp masking and High-boost filtering

cv2.add(target, source) - OpenCV 2 зурагны нийлбэр олох функц

cv2.subtract(target, source) - OpenCV 2 зурагны ялгавар олох функц

Параметрүүд:

target - 1-р зураг

target - 2-р зураг

```
● ● ● Local Enhancement

import cv2
from matplotlib import pyplot as plt
import numpy as np

image = cv2.imread("../Images/enhacement.tif", cv2.IMREAD_GRAYSCALE)

grid_size = (3,3)
image_height, image_width = image.shape

#local_enhancement тухайн region хэсэг дээр хийх функц
def local_enhancement(region):
    enhanced_region = cv2.equalizeHist(region)
    return enhanced_region

output_image_3x3 = np.zeros_like(image)
# 3x3 kernel бүрийн хувьд local_enhancement хийнэ
for i in range(grid_size[0]):
    for j in range(grid_size[1]):
        start_x = (i * image_height) // grid_size[0]
        end_x = ((i + 1) * image_height) // grid_size[0]
        start_y = (j * image_width) // grid_size[1]
        end_y = ((j + 1) * image_width) // grid_size[1]

        region = image[start_x:end_x, start_y:end_y]

        enhanced_region = local_enhancement(region)

        output_image_3x3[start_x:end_x, start_y:end_y] = enhanced_region

img_function = cv2.equalizeHist(image)

result = cv2.hconcat([image, img_function, output_image_3x3])
cv2.imwrite("result.png", result)

# Histogram харьцуулалт
hist1 = cv2.calcHist([image], [0], None, [256], [0, 256])
hist2 = cv2.calcHist([img_function], [0], None, [256], [0, 256])
hist3 = cv2.calcHist([output_image_3x3], [0], None, [256], [0, 256])

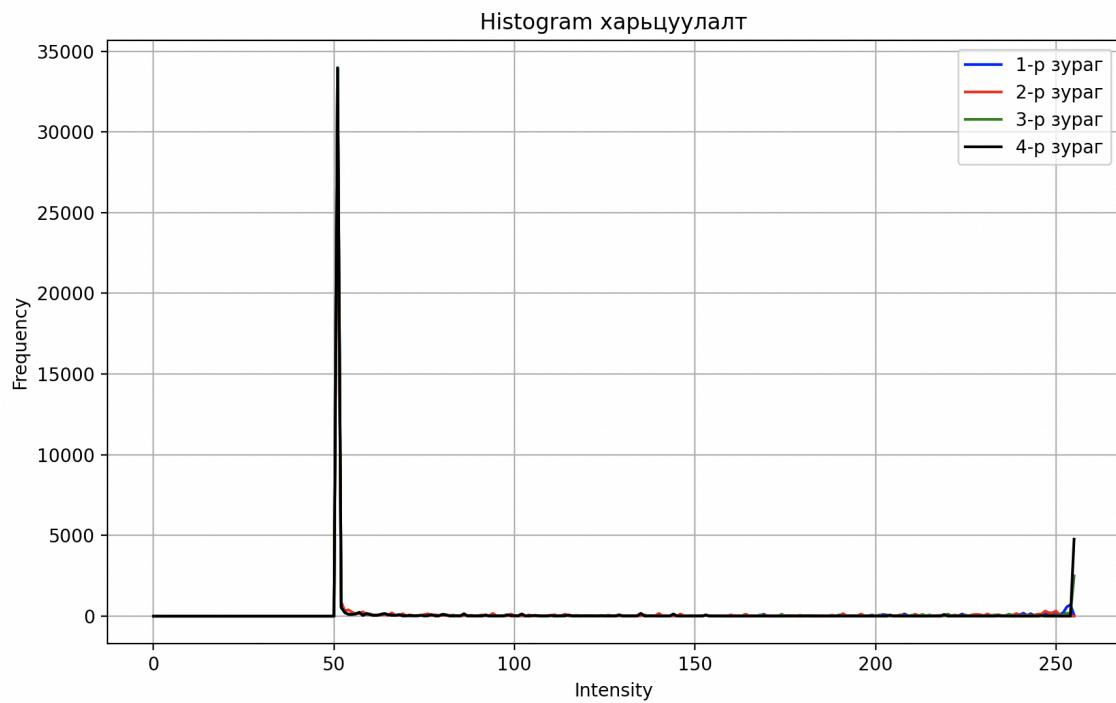
plt.figure(figsize=(10, 6))
plt.plot(hist1, color='blue', label='1-р зураг')
plt.plot(hist2, color='red', label='2-р зураг')
plt.plot(hist3, color='green', label='3-р зураг')

plt.title('Histogram харьцуулалт')
plt.xlabel('Intensity')
plt.ylabel('Frequency')
plt.legend()
plt.grid()
plt.show()
```

ҮР ДҮН :



Mathplot histogram харьцуулалт :



4.4 Combining Spatial Enhancement Methods

cv2.Sobel(image,ddepth,dx,dy,ksize) -

image - Зураг

dx - X тэнхлэгийн дагуу хийх эсэх

dy - Y тэнхлэгийн дагуу хийх эсэх

ksize - кернелийн хэмжээ

cv2.blur(image,ksize) -

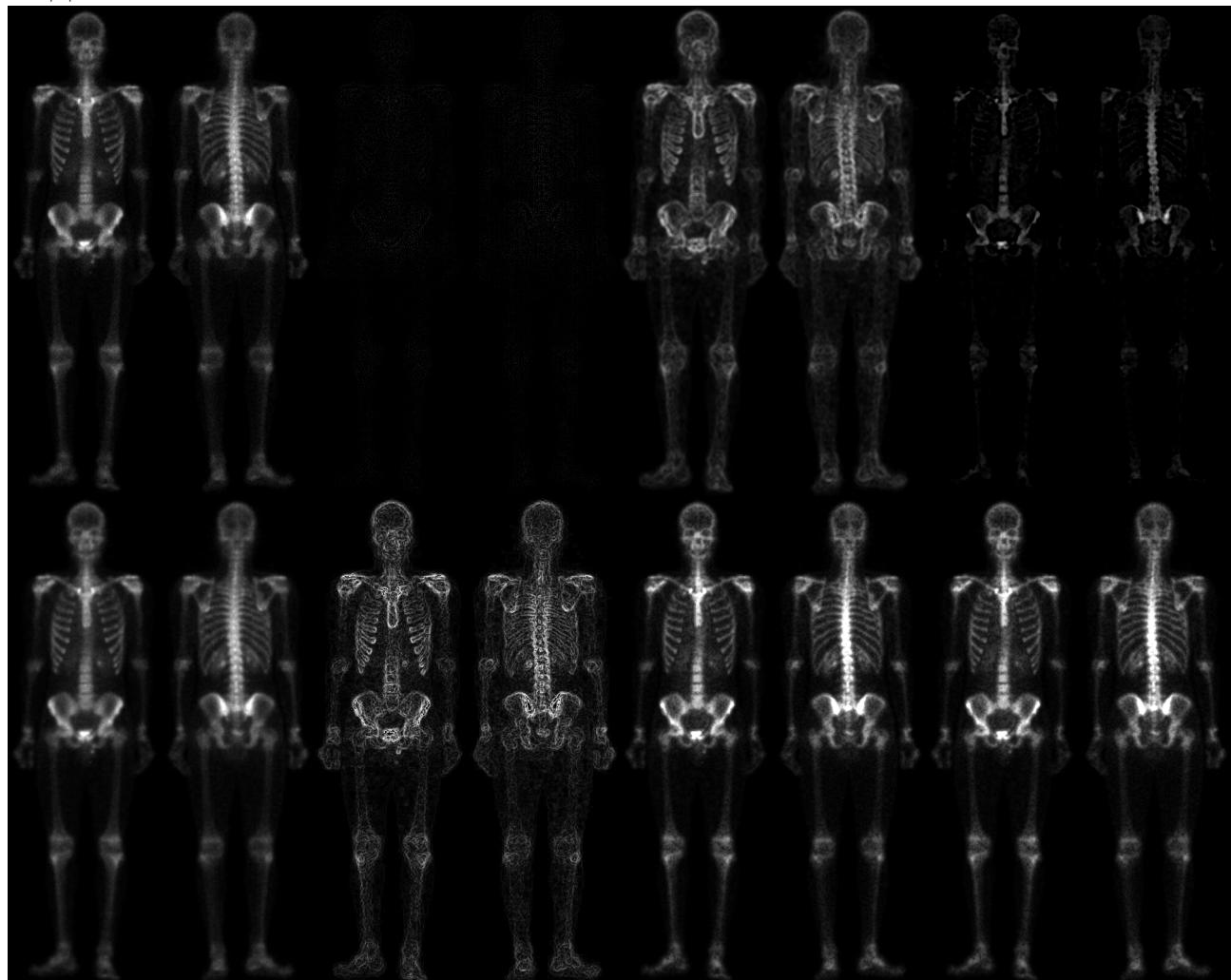
image - Зураг

ksize - кернелийн хэмжээ

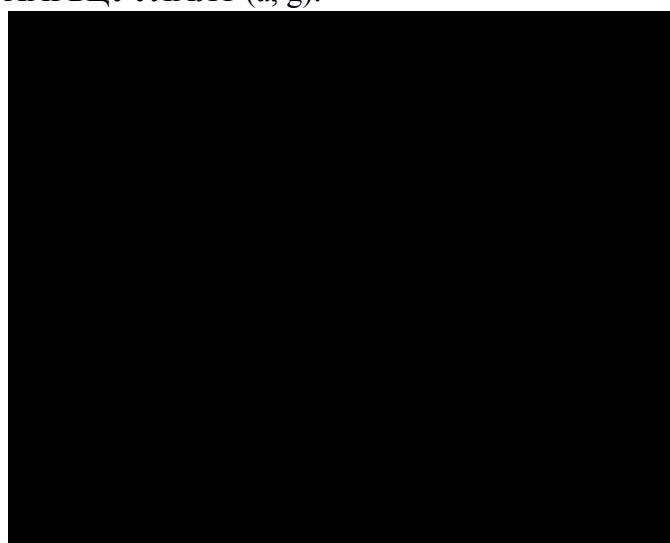
```
import cv2
import matplotlib.pyplot as plt
import numpy as np

image = cv2.imread('../Images/skeleton.tif', cv2.IMREAD_GRAYSCALE)
laplacian_kernel = np.array([[0, 1, 0], [1, -4, 1], [0, 1, 0]])
laplacian_result = cv2.filter2D(image, -1, laplacian_kernel)
sharpened = cv2.add(image, laplacian_result)
sobelx = cv2.Sobel(image, cv2.CV_64F, 1, 0, ksize=3)
sobely = cv2.Sobel(image, cv2.CV_64F, 0, 1, ksize=3)
sobel_result = np.sqrt(sobelx**2 + sobely**2).astype(np.uint8)
sobel_avg = cv2.blur(sobel_result, (5, 5))
sub = cv2.subtract(sharpened, sobel_avg)
sharpened_2 = cv2.add(image, sub)
gamma = 4.0
last_result = cv2.pow(sharpened_2 / 255., gamma) * 255
last_result = np.clip(sharpened_2, 0, 255).astype(np.uint8)
r1 = cv2.hconcat([image, laplacian_result, sobel_avg, sub])
r2 = cv2.hconcat([sharpened, sobel_result, sharpened_2, last_result])
result = cv2.vconcat([r1,r2])
cv2.imwrite('result.png', result)
comp1 = cv2.subtract(image, last_result)
comp2 = cv2.subtract(image, sharpened_2)
comp_result = cv2.hconcat([comp1, comp2])
cv2.imwrite('comp.png', comp_result)
```

ҮР ДҮН :



ХАРЬЦУУЛАЛТ (а, г):



Хуудас 12
2023/12/09

4.5 Notch Filtering

fft2(image) - Фурье хувиргалт хийх функц .

- image : Зураг

fftshift(image) - Фурье хувиргалтыг shift хийх функц

- image: Зураг

```
import cv2
import numpy as np
image = cv2.imread("../Images/car.tif", cv2.IMREAD_GRAYSCALE)

f = np.fft.fft2(image)
fshift = np.fft.fftshift(f)

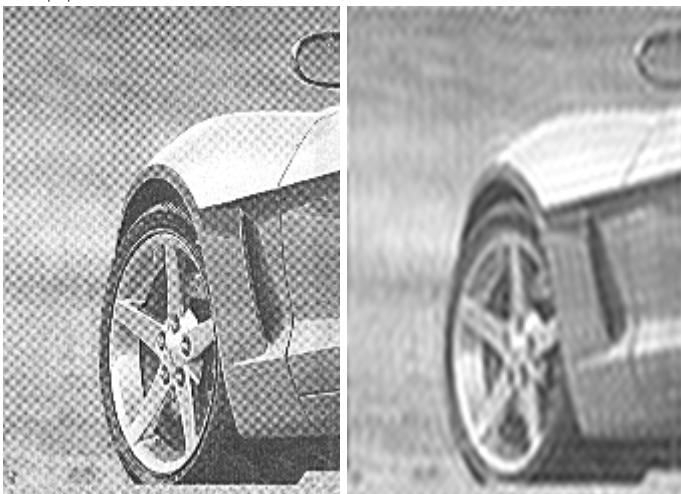
rows, cols = image.shape
crow, ccol = rows // 2, cols // 2

mask = np.zeros((rows, cols), np.uint8)
mask[crow-30:crow+30, ccol-30:ccol+30] = 1
fshift = fshift * mask

f_ishift = np.fft.ifftshift(fshift)
img_back = np.fft.ifft2(f_ishift)
img_back = np.abs(img_back)

cv2.imwrite("shift.png", fshift.astype(np.uint8))
cv2.imwrite("result1.png", image)
cv2.imwrite("result2.png", img_back)
```

ҮР ДҮН :



5. Эх сурвалж

1. <https://www.geeksforgeeks.org/difference-between-low-pass-filter-and-high-pass-filter/>
2. https://en.wikipedia.org/wiki/Median_filter
3. <https://www.geeksforgeeks.org/image-sharpening-using-laplacian-filter-and-high-boost-filtering-in-matlab/>
4. Хэв танилтын үндэс - Лекц 2,3 Б.Сувдаа
5. https://en.wikipedia.org/wiki/Unsharp_masking

6. Хавсралт

Gitlab эх код :

1. Median filter-
https://gitlab.com/tsoomo/hev-tanilt/-/tree/main/lab2-11?ref_type=heads
2. Using Second Derivative for Image sharpening -
https://gitlab.com/tsoomo/hev-tanilt/-/tree/main/lab2-12?ref_type=heads
3. Unsharp masking and High boost filtering -
https://gitlab.com/tsoomo/hev-tanilt/-/tree/main/lab2-13?ref_type=heads
4. Combining Spatial Enhancement Methods -
https://gitlab.com/tsoomo/hev-tanilt/-/tree/main/lab2-14?ref_type=heads
5. Low Pass filter -
https://gitlab.com/tsoomo/hev-tanilt/-/tree/main/lab2-15?ref_type=heads