2D Image Processing and Frequency Domain (DICOM/X-ray)

Image Processing for 2D and Frequency Domain

Overview:

This documentation covers OpenCV-based techniques for processing 2D images and frequency-domain analysis. This is useful in areas like medical imaging (e.g. X-ray, CT, MRI), especially with DICOM image formats.

Required Libraries:

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

1. 2D Spatial Filtering:

These are pixel-based filters that operate directly on image values.

a) Gaussian Blur:

Used to smooth an image and reduce noise.

```
    blurred = cv2.GaussianBlur(image, (5, 5), 0)
```

b) Median Blur:

Preserves edges while reducing salt-and-pepper noise.

```
    median = cv2.medianBlur(image, 5)
```

c) Sharpening:

Highlights edges and fine details.

```
    kernel = np.array([[0, -1, 0],
                       [-1, 5, -1],
                       [0, -1, 0]])
    sharpened = cv2.filter2D(image, -1, kernel)
```

2D Image Processing and Frequency Domain (DICOM/X-ray)

d) Edge Detection with Sobel:

Captures gradients in horizontal and vertical directions.

```
sobelx = cv2.Sobel(image, cv2.CV_64F, 1, 0, ksize=3)
sobely = cv2.Sobel(image, cv2.CV_64F, 0, 1, ksize=3)
```

e) Laplacian Edge Detection:

Detects edges by computing second derivative.

```
laplacian = cv2.Laplacian(image, cv2.CV_64F)
```

2. Frequency Domain Filtering (Fourier Transform):

a) Convert to frequency domain:

```
dft = cv2.dft(np.float32(image), flags=cv2.DFT_COMPLEX_OUTPUT)
dft_shift = np.fft.fftshift(dft)
```

b) Apply High-Pass Filter (removes low-frequency background):

```
rows, cols = image.shape
crow, ccol = rows // 2 , cols // 2
mask = np.ones((rows, cols, 2), np.uint8)
mask[crow-30:crow+30, ccol-30:ccol+30] = 0
fshift = dft_shift * mask
```

c) Convert back to spatial domain:

```
f_ishift = np.fft.ifftshift(fshift)
img_back = cv2.idft(f_ishift)
img_back = cv2.magnitude(img_back[:, :, 0], img_back[:, :, 1])
```

3. DICOM Image Use Case (X-ray):

Though OpenCV doesnt support DICOM directly, you can use pydicom:

```
import pydicom
```

2D Image Processing and Frequency Domain (DICOM/X-ray)

```
dicom = pydicom.dcmread("xray.dcm")
image = dicom.pixel_array
image = cv2.normalize(image, None, 0, 255, cv2.NORM_MINMAX).astype(np.uint8)
```

You can then apply the same filters and Fourier transform to this image as described above.

4. Summary of Use Cases:

- Gaussian/Median: noise reduction in medical scans
- Laplacian/Sobel: edge structure for tumor or bone detection
- Frequency filters: background removal, texture analysis