Image Filtering in Frequency Domain using DFT

Introduction:

A technique where image processing operations (like filtering) are performed in the frequency domain instead of the spatial domain using the **Discrete Fourier Transform (DFT)**.

Importances:

- To analyze the frequency components of an image.
- To apply filters (like low-pass or high-pass) more effectively.
- Some operations (like convolution) are faster in the frequency domain.

Working :

1. **Convert Image to Float32**
    - Required by `cv.dft()` function.
2. **Apply DFT**

```
dft = cv.dft(np.float32(img), flags=cv.DFT_COMPLEX_OUTPUT)
```

    - Converts the image from spatial to frequency domain.
    - Output has **2 channels**: Real and Imaginary.
3. **Shift the Zero-Frequency Component to Center**

```
dft_shift = np.fft.fftshift(dft)
```

    - Moves low frequencies to the center for better visualization.
4. **Compute Magnitude Spectrum**

```
magnitude = cv.magnitude(dft_shift[:,:,0], dft_shift[:,:,1])
```

5. **Convert to Log Scale**

```
magnitude_spectrum = 20 * np.log(magnitude + 1)
```

    - Compresses wide-ranging values.
    - Makes small details visible.
6. **Display**

```
plt.imshow(magnitude_spectrum, cmap='gray')
```

Formula:

2D Discrete Fourier Transform (DFT) Formula

$$F(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) \cdot e^{-j2\pi\left(\frac{ux}{M} + \frac{vy}{N}\right)}$$

Where:

f (x , y) f(x,y) = input image pixel value at position (x , y) (x,y)

F (u , v) F(u,v) = frequency domain value at (u , v) (u,v)

M , N M,N = image size

j j = imaginary unit (-1) (-1)

Magnitude Spectrum Formula

$$|F(u,v)| = \sqrt{\text{Re}(F(u,v))^2 + \text{Im}(F(u,v))^2}$$

Log Scaling (for better visualization)

$$\text{Magnitude Spectrum (dB)} = 20 \cdot \log_{10}(|F(u,v)| + 1)$$

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

```
img = cv.imread('//content/Lenna_fourierinput.png',0)
dft = cv.dft(np.float32(img),flags = cv.DFT_COMPLEX_OUTPUT)
```

```
dft_shift = np.fft.fftshift(dft)
```

```
magnitude_spectrum = 20*np.log(cv.magnitude(dft_shift[:,:,0],dft_shift[:,:,1]))
```

```
plt.subplot(121),plt.imshow(img, cmap = 'gray')
plt.title('Input Image'), plt.xticks([]), plt.yticks([])
plt.subplot(122),plt.imshow(magnitude_spectrum, cmap = 'gray')
plt.title('Magnitude Spectrum'), plt.xticks([]), plt.yticks([])
plt.show()
```