# Outlines

# Introduction

## Deal with images in

- Spatial domain - Processing in pixel/spatial domain.
- Frequency Domain - Processing in the transformed domain.

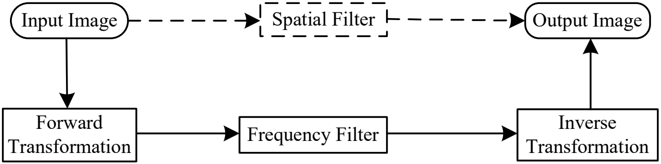


Figure: Frequency domain filtering

# Frequency

**Time series:**

- In 1-D time series, frequency is related to $\frac{dA}{dt}$.
- If rate of change of amplitude higher then frequency is also higher

**Images:**

- For images, spatial frequency is related to $\frac{dI}{dx}$.
- For edges frequency is higher, smooth regions frequency is lower.

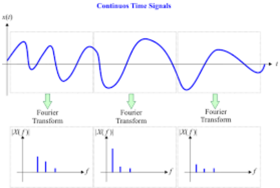


Figure: 1-D Time Signal

Figure: An Image.

# Why Frequency Domain Processing?

- It is computationally faster to perform 2D Fourier transforms and a filter multiply than to perform a convolution in the image (spatial) domain.
- Frequency domain has a established suit of processes and tools that can be borrowed directly from signal processing in other domains.
- Easy to remove periodic noise in images.
- The parameters are easily manipulated.
- Very useful in compression - few coefficients represent most of the information.

# Image Transforms

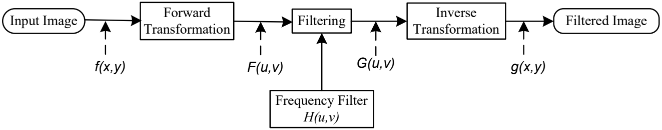


Figure: Frequency domain filtering

- Convolution in spatial domain $\leftrightarrow$ Multiplication in frequency domain.
- $G(u,v) = H(u,v).F(u,v)$

# Image Transforms

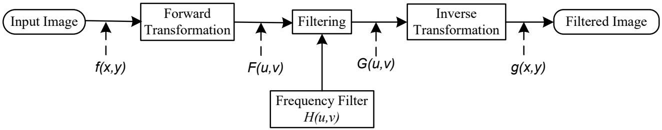


Figure: Frequency domain filtering

- Convolution in spatial domain $\leftrightarrow$ Multiplication in frequency domain.
- $G(u,v) = H(u,v).F(u,v)$

**Standard Transforms - Kernel is image independent**

- Discrete Fourier Transform (DFT).
- Discrete Cosine Transform (DCT).
- Discrete Sine Transform (DST).
- Discrete Hadamard Transform (DHT).

**Image dependent Transforms - Kernel is image dependent**

- K-L Transform.

# Discrete Fourier Transform

**1-D Continuous Fourier Transform**

$$F(u) = \int_{-\infty}^{\infty} f(x) e^{-j2\pi ux} dx \tag{1}$$

**2-D Continuous Discrete Fourier Transform**

$$F(u,v) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x,y) e^{-j2\pi(ux+vy)} dxdy \tag{2}$$

# Discrete Fourier Transform

**1-D Continuous Fourier Transform**

$$F(u) = \int_{-\infty}^{\infty} f(x) e^{-j2\pi ux} dx \quad (1)$$

**2-D Continuous Discrete Fourier Transform**

$$F(u,v) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x,y) e^{-j2\pi(ux+vy)} dxdy \quad (2)$$

**2-D Discrete Fourier Transform**

$$F(u,v) = \frac{1}{N} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y) e^{-j\frac{2\pi}{N}(ux+vy)} \quad (3)$$

**2-D Inverse Discrete Fourier Transform**

$$f(x,y) = \frac{1}{N} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} F(u,v) e^{j\frac{2\pi}{N}(ux+vy)} \quad (4)$$

For a $512 \times 512$ image $\rightarrow 512^4$ complex multiplications. Huge complexity!

# Separability Property

**2-D Discrete Fourier Transform**

$$F(u,v)=\frac{1}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}f(x,y)e^{-j\frac{2\pi}{N}(ux+vy)}=\frac{1}{N}\sum_{x=0}^{N-1}e^{-j\frac{2\pi}{N}ux}.\sqrt{N}.\frac{1}{\sqrt{N}}\sum_{y=0}^{N-1}f(x,y)e^{-j\frac{2\pi}{N}vy}$$

$$=\frac{1}{N}\sum_{x=0}^{N-1}e^{-j\frac{2\pi}{N}ux}.\sqrt{N}.F(x,v)=\frac{1}{\sqrt{N}}\sum_{x=0}^{N-1}F(x,v)e^{-j\frac{2\pi}{N}ux}=F(u,v)$$

# Separability Property

**2-D Discrete Fourier Transform**

$$F(u,v) = \frac{1}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)e^{-j\frac{2\pi}{N}(ux+vy)} = \frac{1}{N}\sum_{x=0}^{N-1} e^{-j\frac{2\pi}{N}ux}.\sqrt{N}.\frac{1}{\sqrt{N}}\sum_{y=0}^{N-1} f(x,y)e^{-j\frac{2\pi}{N}vy}$$

$$= \frac{1}{N}\sum_{x=0}^{N-1} e^{-j\frac{2\pi}{N}ux}.\sqrt{N}.F(x,v) = \frac{1}{\sqrt{N}}\sum_{x=0}^{N-1} F(x,v)e^{-j\frac{2\pi}{N}ux} = F(u,v)$$

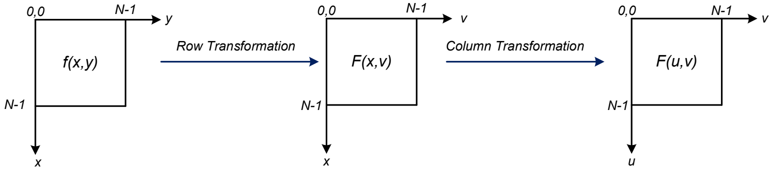


Figure: Separability - 2-D Transform as combination of 1-D Transform

# Fast Fourier Transform

- Faster implementation of DFT.
- Exploiting Periodicity and Symmetry properties of DFT.

$$F(u) = \frac{1}{N}\sum_{x=0}^{N-1} f(x)e^{-j\frac{2\pi}{N}ux} = \frac{1}{N}\sum_{x=0}^{N-1} f(x)W_N^{ux} \tag{5}$$

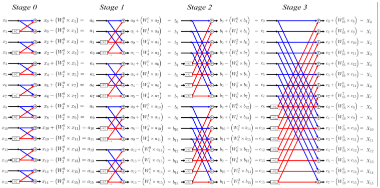


Figure: 16 point FFT

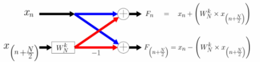


Figure: Butterfly structure.

# Fast Fourier Transform

- For an N-Point FFT, computational complexity would by $O(NlogN)$.
- For an image we can have a computational complexity of $O(N^2 logN)$.
- For $512 \times 512$ image - $128 \times 512 \times 9$ complex multiplications.
- Similarly we have faster implementation of IDFT.
- Implement 1-D FFT as an recursive function.