Introduction to the Discrete Fourier Transform

Lucas J. van Vliet

www.ph.tn.tudelft.nl/~lucas

TU Delft

TNW: Faculty of Applied Sciences

IST: Imaging Science and technology

PH: Pattern Recognition Group

Linear Shift Invariant System

A discrete image can be decomposed into a weighted field of equi-spaced impulses.

$$f(x) = \sum_{n=-\infty}^{+\infty} f(n)\,\delta(x-n)$$

$\delta(x-n)$: impulse at position *n*

$f(n)$: amplitude at position *n*

$f(x) \longrightarrow$ [LSI system $h(x)$] $\longrightarrow g(x)$

$\delta(x) \longrightarrow h(x)$ — *h*(*x*) = impulse response or Point Spread Function (PSF)

$a\,\delta(x) \xrightarrow{\text{linear}} a\,h(x)$

$\delta(x-n\Delta) \xrightarrow{\text{shift inv.}} h(x-n)$

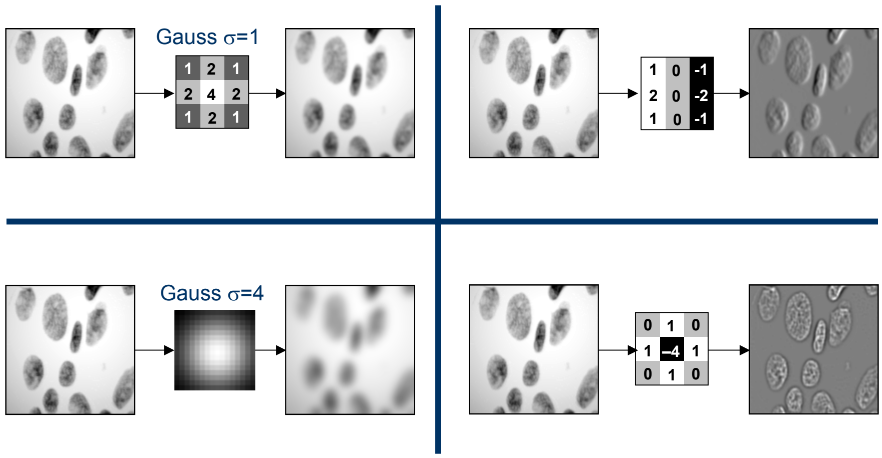
$$f(x) = \sum_{n=-\infty}^{+\infty} f(n\Delta)\,\delta(x-n\Delta) \xrightarrow{\text{superposition}} g(x) = \sum_{n=-\infty}^{+\infty} f(n)\,h(x-n) \equiv f(x) * h(x)$$

Image *g* is the result of a ***convolution*** between image *f* and *h*

Convolution revisited

TU Delft
Pattern Recognition Group

- Convolution: Replace the central pixel by a weighted sum of the gray-values inside an $n{\times}n$ neighborhood. Impulse response $h(x)$ is the filter.



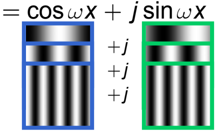
Eigenfunction of LSI systems

TU Delft
Pattern Recognition Group

- Eigenfunctions of LSI systems are complex exponentials $\varphi(x)$.

$$\varphi_\omega(x) = e^{j\omega x} = \cos\omega x + j\sin\omega x$$

$$\varphi_{\omega=\omega 1}(x) = \quad +j$$
$$\varphi_{\omega=\omega 2}(x) = \quad +j$$
$$\varphi_{\omega=\omega 3}(x) = \quad +j$$



$\varphi_\omega(x)$ → LSI system $h(x)$ → $K\varphi_\omega(x) = \sum_{n=-\infty}^{+\infty} \varphi(n)h(x-n)$

$e^{j\omega x}$ →

$$g(x) = \sum_{n=-\infty}^{+\infty} e^{j\omega n}h(x-n) = e^{j\omega x}\underbrace{\sum_{n=-\infty}^{+\infty} h(m)e^{-j\omega m}}_{H(\omega)}$$

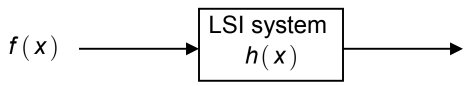
$$= e^{j\omega x} H(\omega)$$

- An LSI system multiplies each eigenfunction $\exp(j\omega x)$ by its corresponding eigenvalue $H(\omega)$.
- The Fourier transform decomposes an image into a weighted set of complex exponentials of varying frequency ω: “The Fourier spectrum”
- The system function $H(\omega)$ is the Fourier Transform of $h(x)$

Convolution property

TU Delft
Pattern Recognition Group

- A convolution between an image *f*(*x*) and an impulse response *h*(*x*) in space, corresponds to a multiplication of the Fourier spectra *F*(ω) and *H*(ω) in the Fourier domain.



$$f(x) * h(x) = \sum_{n=-\infty}^{+\infty} f(n\Delta) h(x - n\Delta)$$

$$\begin{aligned} F\{f(x) * h(x)\} &= \sum_{x=-\infty}^{+\infty} \left[\sum_{n=-\infty}^{+\infty} f(n) h(x-n) \right] e^{-j\omega x} \\ &= \underbrace{\sum_{n=-\infty}^{+\infty} f(n) e^{-j\omega n}}_{F(\omega)} \underbrace{\sum_{x=-\infty}^{+\infty} h(m) e^{-j\omega m}}_{H(\omega)} \\ &= F(\omega) H(\omega) \end{aligned}$$

Gaussian derivatives

TU Delft
Pattern Recognition Group

- How to compute a derivative in digital space?

$$f_x[x,y] \triangleq \frac{\partial}{\partial x} f[x,y] = ?$$

- Introduce (Gaussian) scale

$$\begin{aligned} f^{(\sigma)}[x,y] &= f^{(0)}[x,y] \otimes g^{(\sigma)}[x,y] \\ f^{(\sigma=5)}[x,y] &= f^{(\sigma=3)}[x,y] \otimes g^{(\sigma=4)}[x,y] \end{aligned}$$

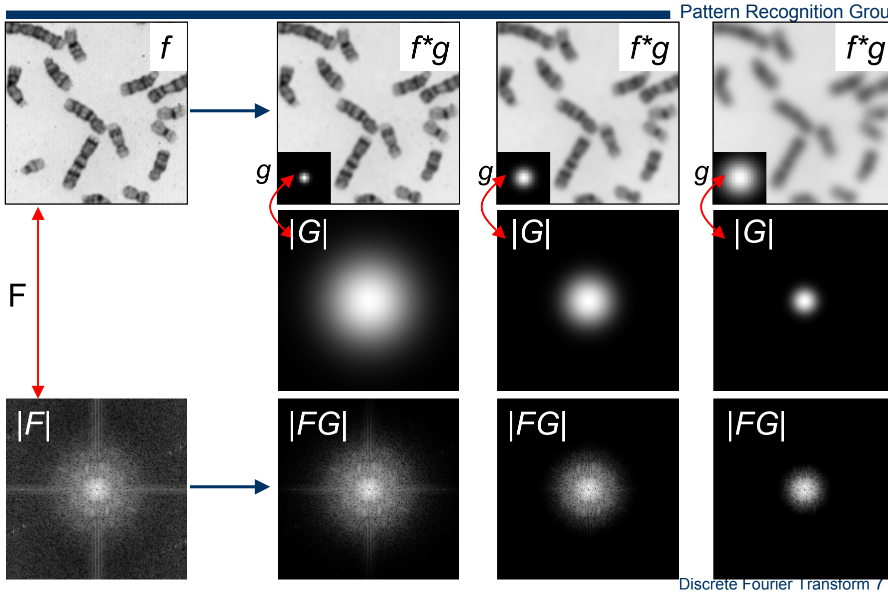
- Derivative at scale σ is computed by convolution of the discrete image *f*[x,y] with discrete Gaussian derivative

$$f_x^{(\sigma)}[x,y] = f^{(0)}[x,y] \otimes g_x^{(\sigma)}[x,y]$$

- Note that the Gaussian function is known analytically, compute the derivative(s), and then sample to produce a discrete filter. The sampling of the Gaussian should be high enough, i.e. $\sigma > 0.9$ pixels

Fourier filters: Gaussian

TU Delft
Pattern Recognition Group



Gaussian derivative filters

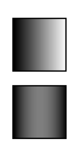
TU Delft
Pattern Recognition Group

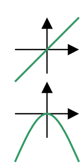
- In continuous space: the derivative operator corresponds to multiplication of the Fourier spectrum with $j\omega$

$$f(x) \stackrel{F}{\longleftrightarrow} F(\omega)$$

$$\frac{d}{dx} f(x) \stackrel{F}{\longleftrightarrow} j\omega F(\omega)$$

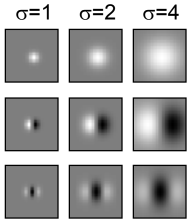
$$\frac{d^2}{dx^2} f(x) \stackrel{F}{\longleftrightarrow} -\omega^2 F(\omega)$$





- In discrete space: combine the derivative operator with Gaussian smoothing

σ=1 σ=2 σ=4

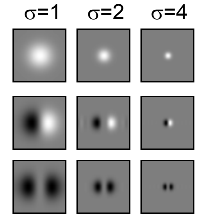


$$g^{(\sigma)}(x) * f(x) \stackrel{F}{\longleftrightarrow} G(\omega) F(\omega)$$

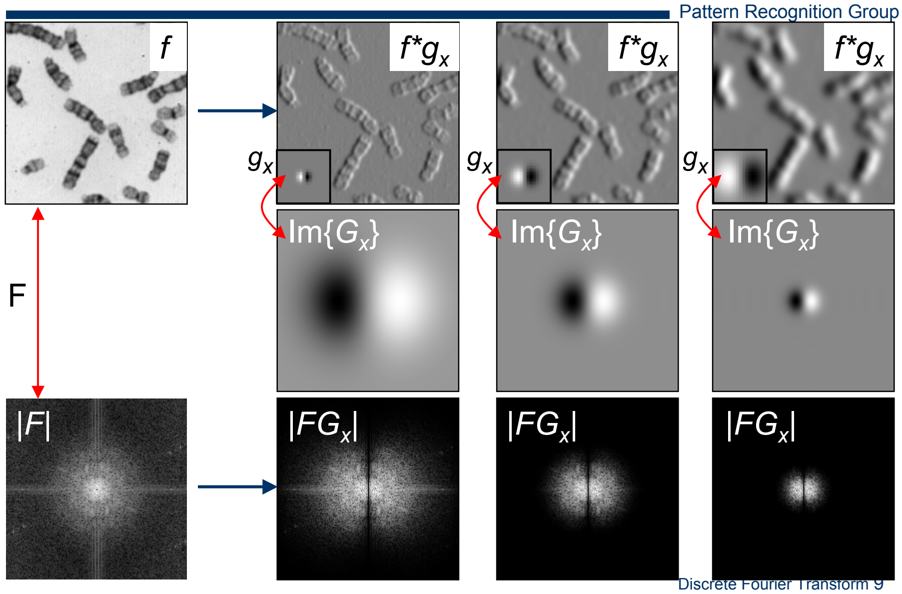
$$\frac{d}{dx} g^{(\sigma)}(x) * f(x) \stackrel{F}{\longleftrightarrow} j\omega\, G(\omega) F(\omega)$$

$$\frac{d^2}{dx^2} g^{(\sigma)}(x) * f(x) \stackrel{F}{\longleftrightarrow} -\omega^2 G(\omega) F(\omega)$$

σ=1 σ=2 σ=4

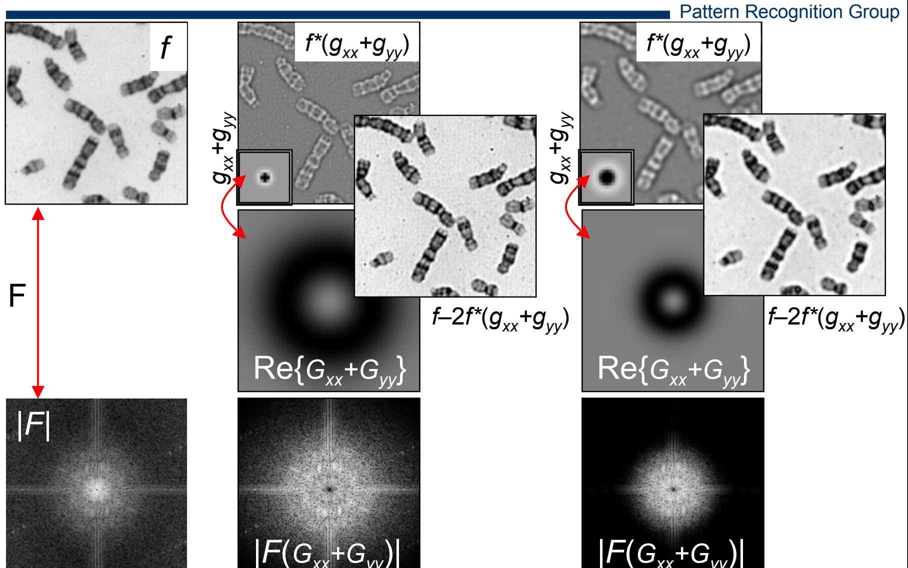


Fourier: Gaussian derivative



Fourier: Laplace & sharpening

TU Delft
Pattern Recognition Group



Discrete Fourier Transform

- Each image can be decomposed in weighted sum of complex exponentials (sines and cosines) of frequency *f* and angle ϕ. (or two frequency components *u* and *v*)

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

image size *N*x*N*

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

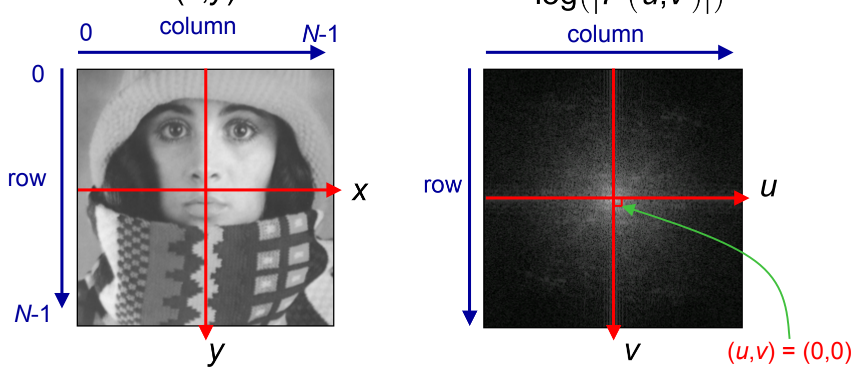
- For real-valued images: $\mathrm{Ev}\{g(x,y)\} \xleftrightarrow{F} \mathrm{Re}\{G(u,v)\}$

$$\mathrm{Od}\{g(x,y)\} \xleftrightarrow{F} j\,\mathrm{Im}\{G(u,v)\}$$

Fourier spectrum

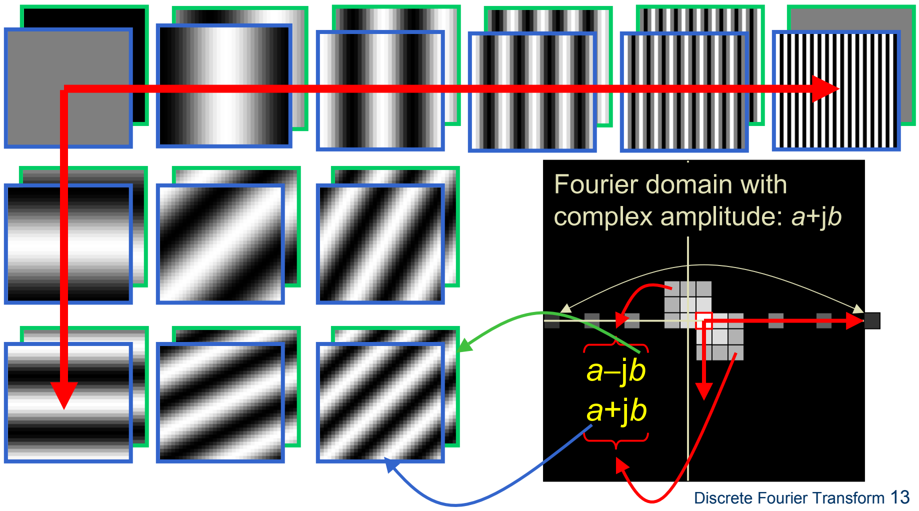
- $F(u,v)$ is the complex amplitude of the eigenfunction $\exp(j\,(2\pi/N)(ux+vy))$
  Note that $\mathbf{exp}(j\,(2\pi/N)(ux+vy)) = \mathbf{cos}((2\pi/N)(ux+vy)) + j\,\mathbf{sin}((2\pi/N)(ux+vy))$
- Standard display is the logarithm of the magnitude: $\log(|F(u,v|)$

$f(x,y)$ $\qquad$ ${}^{10}\log(|F(u,v)|)$

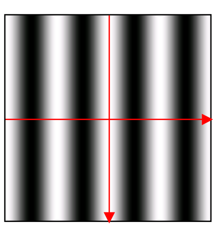


Getting used to Fourier (2)

An image is a weighted sum of cos (even) and sin (odd) images.



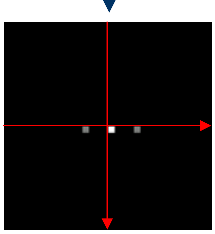
Eigenfunctions: Even & Odd



Real & even

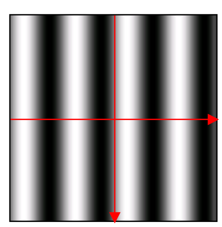
$$1+\cos\left(\frac{2\pi}{N}u_0x\right)=1+\frac{e^{j\frac{2\pi}{N}u_0x}+e^{j\frac{2\pi}{N}(-u_0)x}}{2}$$

$\mathcal{F}$



Real & even

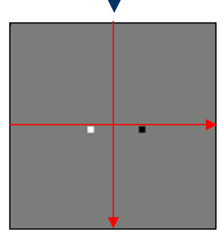
$$\frac{1}{N}\left(\begin{array}{l}\delta(u,v)+\\ \frac{1}{2}\delta(u-u_0,v)+\\ \frac{1}{2}\delta(u+u_0,v)\end{array}\right)$$



Real & odd

$$\sin\left(\frac{2\pi}{N}u_0x\right)=\frac{e^{j\frac{2\pi}{N}u_0x}-e^{j\frac{2\pi}{N}(-u_0)x}}{2j}$$

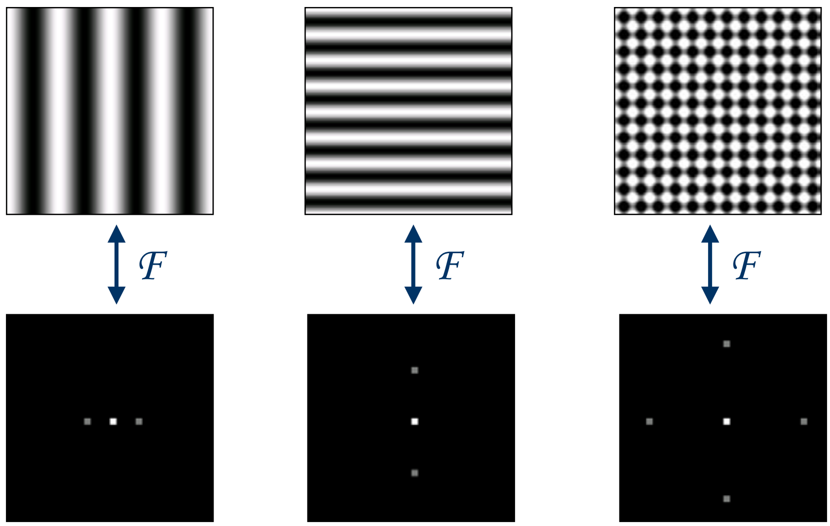
$\mathcal{F}$



Imag & odd

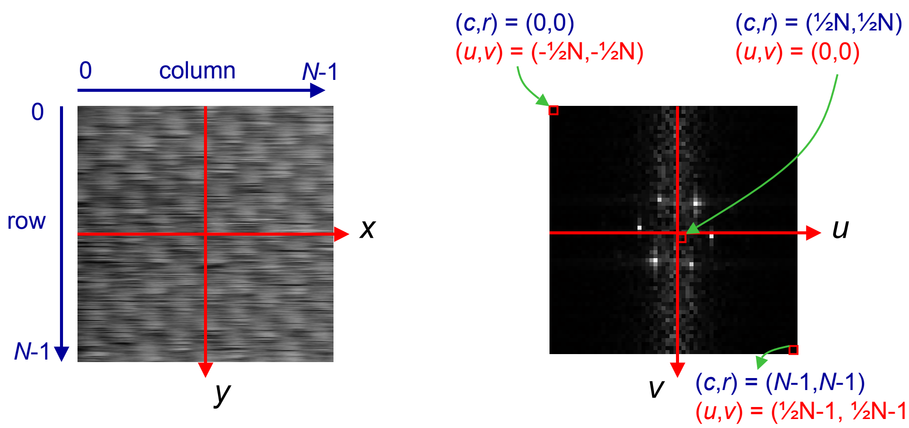
$$\frac{1}{N}j\left(\begin{array}{l}-\frac{1}{2}\delta(u-u_0,v)\\ +\frac{1}{2}\delta(u+u_0,v)\end{array}\right)$$

Orientation & frequency



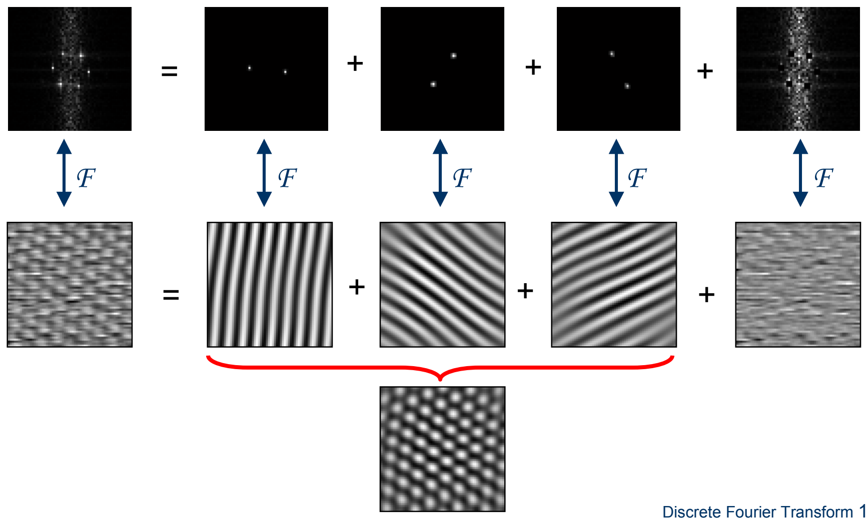
Getting used to Fourier (1)

- Graphite surface by Scanning Tunneling Microscopy
- Atomic structure of graphite shows a hexagonal surface

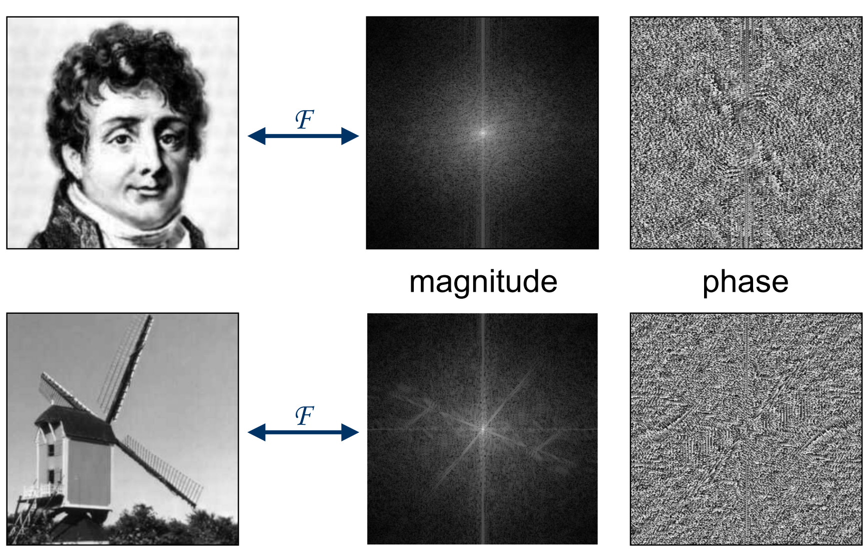


Superposition

- Fourier spectrum



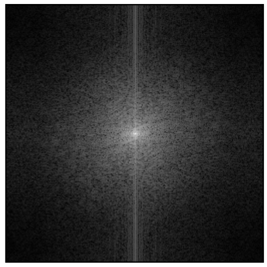
Fourier transforms



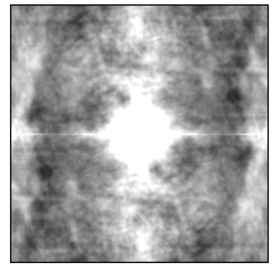
Magnitude & phase

TU Delft
Pattern Recognition Group

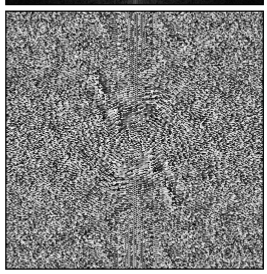
$|F(u,v)|$



$\mathcal{F}^{-1}$ ↔



$e^{j\sphericalangle F(u,v)} = \dfrac{F(u,v)}{|F(u,v)|}$



$\mathcal{F}^{-1}$ ↔

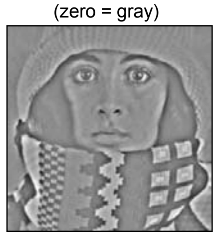
Local variance filter: power

TU Delft
Pattern Recognition Group

- *Recipe:* local variance filter (filter size = *n*)
  1. Compute the local mean (blurring filter of size *n*)
  2. Subtract the local mean.
  3. Compute the square of each pixel value
  4. Suppress the "double" response by local averaging (blurring filter of size *n*)
- Local variance is a measure for the local squared-contrast.

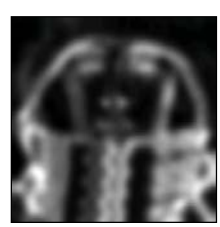
step 1

(zero = gray)



step 2

step 3



step 4

Scaling: local vs global

- Problem: Choosing the proper scale is an important, but tedious task.
  - Scale too small: Local characteristics are missed which yields an incomplete data description.
  - Scale too large: Confusion (mixing) of adjacent objects, lack of localization, and blindness for detail.
- Solution: Multi-scale analysis.
  - Analyze the image as function of scale: from fine detail to course “image-filling” objects.

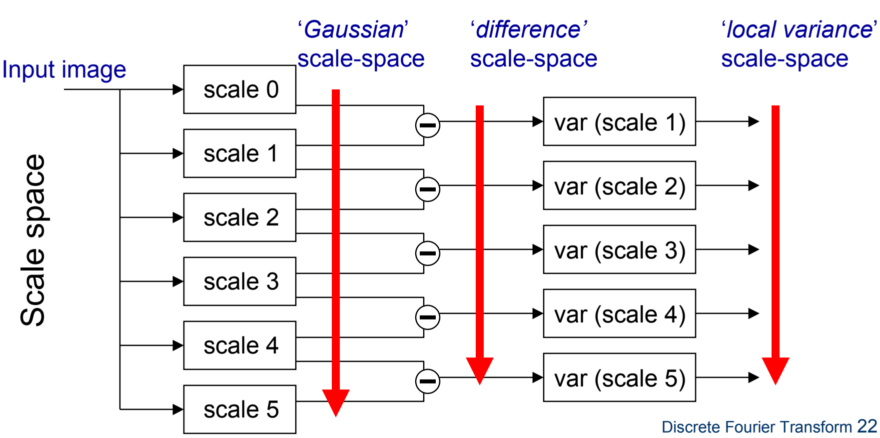
$$g(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \quad \overset{F}{\longleftrightarrow} \quad G(u,v) = e^{-\frac{\left(\left(\frac{2\pi}{N}u\right)^2 + \left(\frac{2\pi}{N}v\right)^2\right)\sigma^2}{2}}$$

Multi-Scale

Series of images of increasing scale: Scale-space

- Sample the scales logarithmically using filters of $size = base^{scale}$ yields n scales per octave

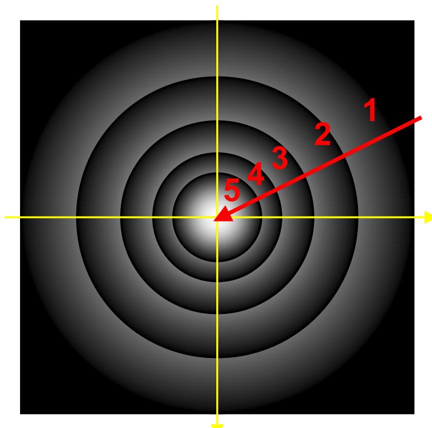
$$base \in \left\{2^1, 2^{1/2}, 2^{1/3}, \ldots, 2^{1/n}\right\}$$



Scale-spaces

TU Delft
Pattern Recognition Group

- Morphological scale-space: Use *openings* (*closings*)
- Gaussian scale-space: Use *Gaussian filters*



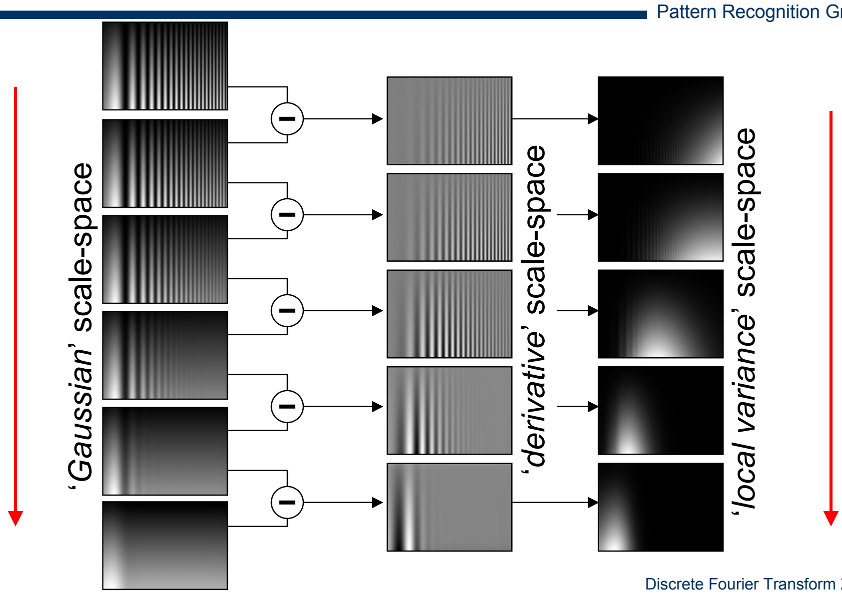
Increasing scales

Fourier domain with “footprints” of Gaussian filters of increasing scale

Filter size is inversely proportional to “footprint” in Fourier domain

Chirp example

TU Delft
Pattern Recognition Group



Gaussian derivatives

TU Delft
Pattern Recognition Group

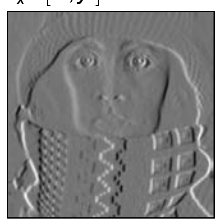
$f^{(0)}[x,y]$

$f^{(1)}[x,y]$

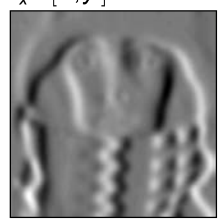
$f^{(5)}[x,y]$

$f_x^{(0)}[x,y] = ?$

$f_x^{(1)}[x,y]$

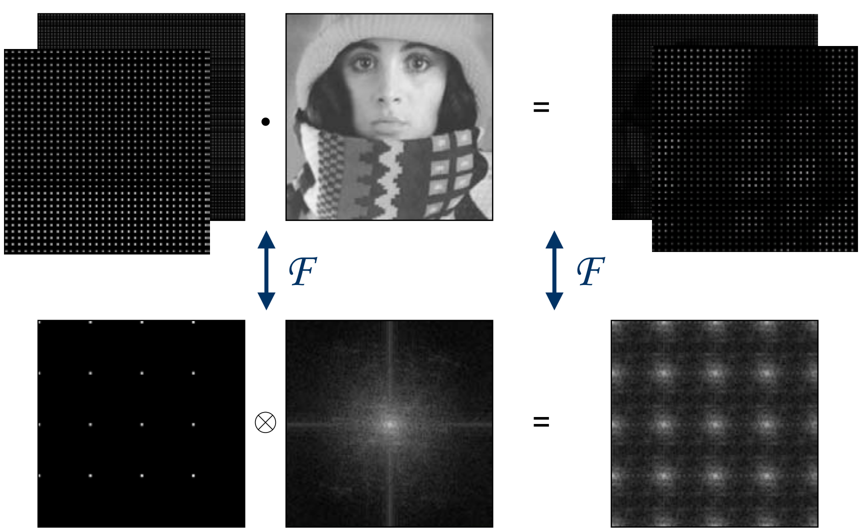


$f_x^{(5)}[x,y]$

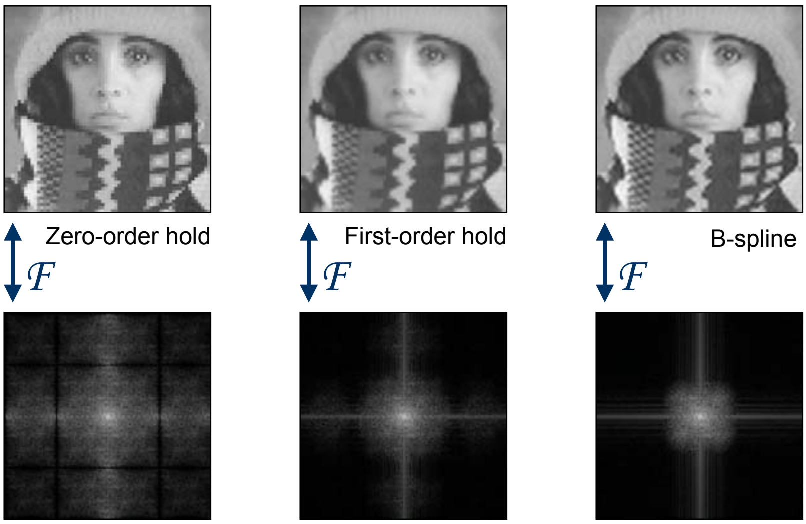


Sampling

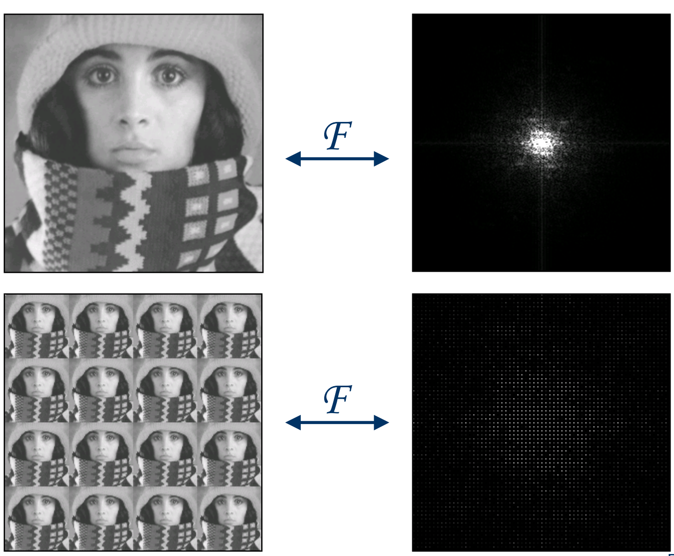
TU Delft
Pattern Recognition Group



Interpolation



Periodic images



Periodic image

yields

Fourier spectrum
with impulses