Spectral representation of images

Paweł Forczmański
Chair of Multimedia Systems, Faculty of Computer Science and Information Technology

Agenda

1. image spectrum
2. FFT
3. filtering
4. JPEG / JFIF

DIP scheme

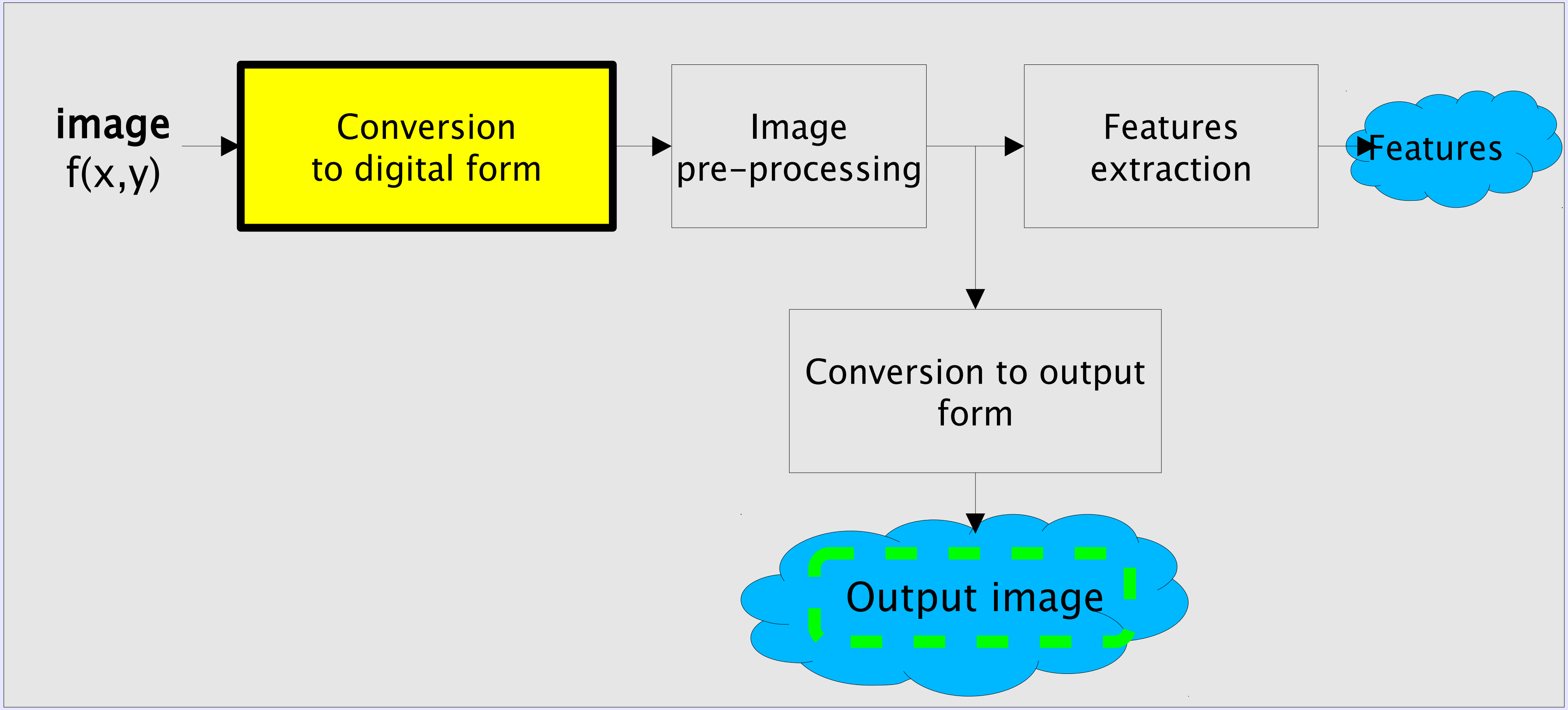


image
f(x,y)
Conversion
to digital form
Image
pre-processing
Features
extraction
Features
Conversion to output
form
Output image

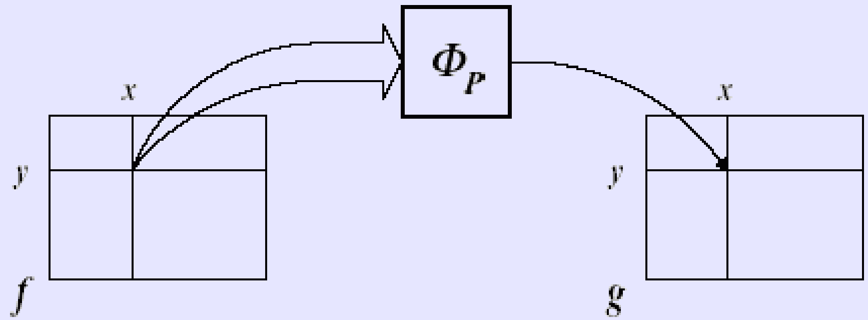
Jean Baptiste Joseph Fourier

From Wikipedia, the free encyclopedia

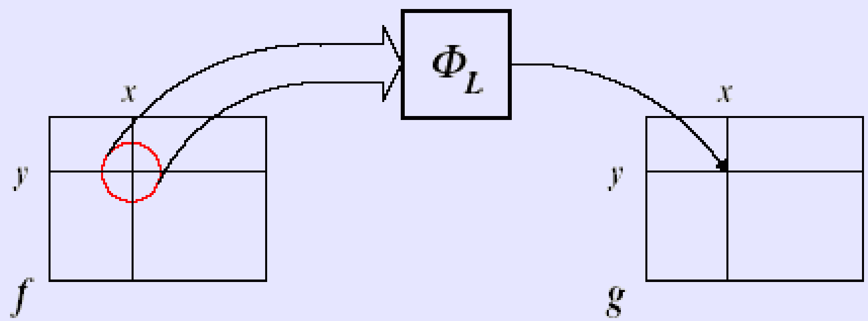
Jean Baptiste Joseph Fourier (21 March 1768 – 16 May 1830) was a French mathematician and physicist born in Auxerre and best known for initiating the investigation of Fourier series and their applications to problems of heat transfer and vibrations. The Fourier transform and Fourier's Law are also named in his honour. Fourier is also generally credited with the discovery of the greenhouse effect.[1]

Transformations

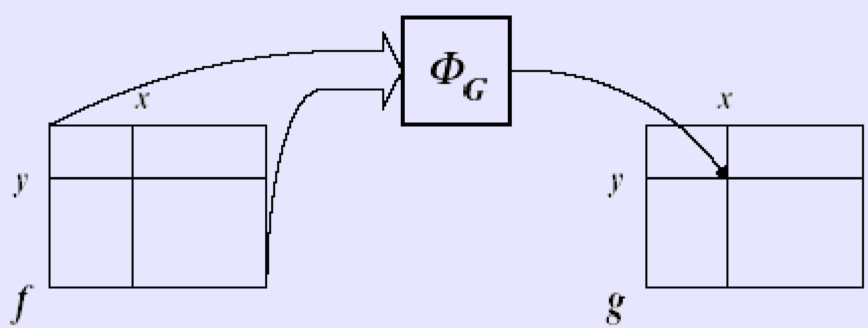
point transform



local transform



global transform



Global, linear transformation

$$\mathbf{f}_{H\times W} = \begin{bmatrix} f(0,0) & f(0,1) & \dots & f(0,W-1) \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ f(H-1,0) & f(H-1,1) & \dots & f(H-1,W-1) \end{bmatrix}$$

$$\mathbf{F}_{H\times W} = \mathbf{P}_{H\times H}\mathbf{f}_{H\times W}\mathbf{Q}_{W\times W};\ det\mathbf{P} \neq 0,\ det\mathbf{Q} \neq 0$$

$$F(u,v) = \sum_{m=0}^{H-1}\sum_{n=0}^{W-1} P(u,m)f(m,n)Q(n,v)$$

General properties

separability

$$\mathbf{F} = (\mathbf{P}\,\mathbf{f})\,\mathbf{Q} = \mathbf{P}\,(\mathbf{f}\,\mathbf{Q})$$

P and Q are real, orthogonal and symmetric:

$$\begin{aligned} \mathbf{F} &= \mathbf{P}\,\mathbf{f}\,\mathbf{Q} \\ \mathbf{f} &= \mathbf{P}\,\mathbf{F}\,\mathbf{Q} \end{aligned}$$

Properties:

$\mathbf{A}$ *Is symmetric* $\Leftrightarrow \mathbf{A}^T = \mathbf{A}$

$\mathbf{A}$ *Is orthogonal* $\Leftrightarrow \mathbf{A}^T\mathbf{A} = \mathbf{1}$

Fourier transform for 1D

For continous functions

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

direct

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

inverse

Fourier transform for 2D

For continous functions

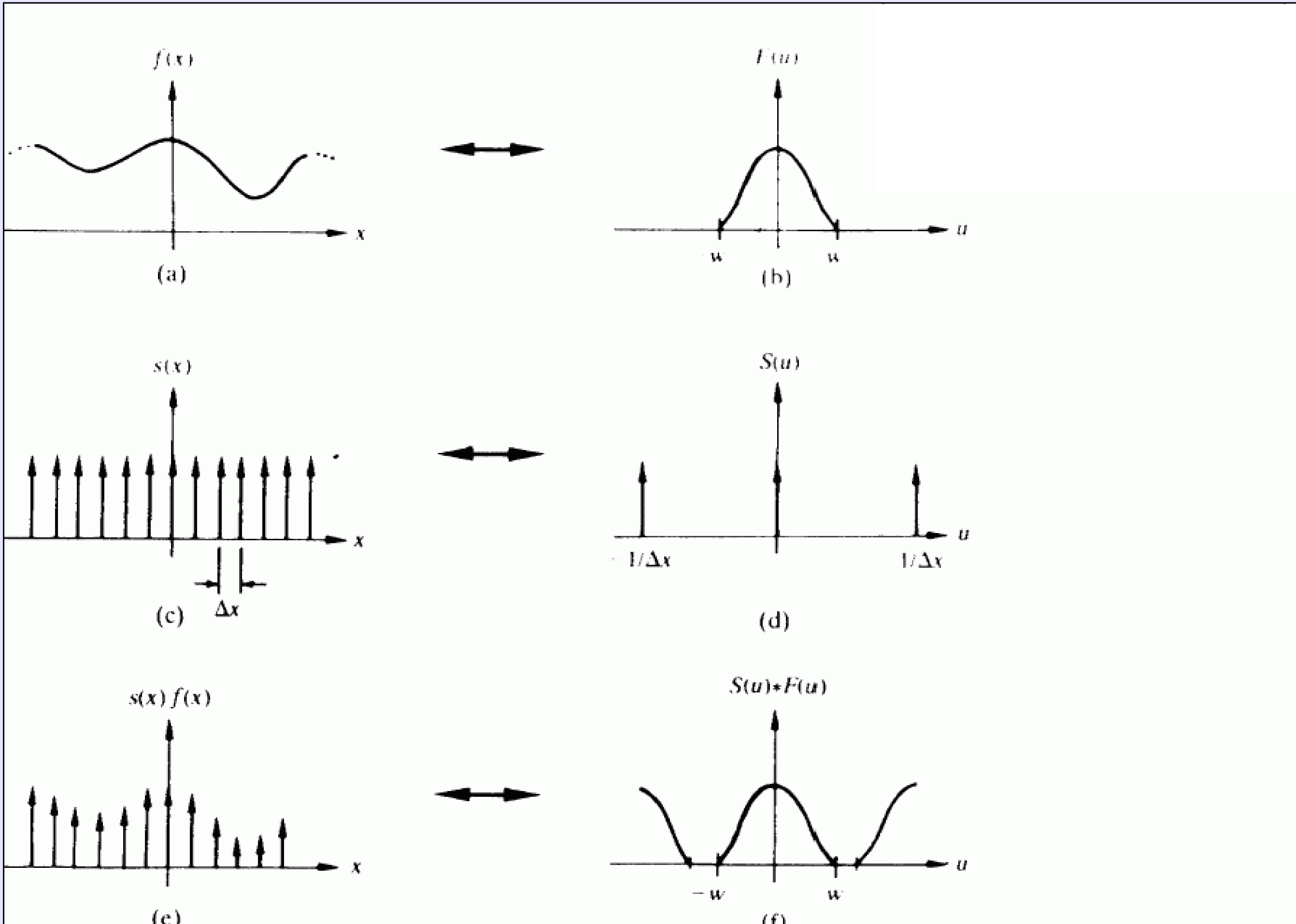
$$F(u,v)=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)e^{-j2\pi(ux+vy)}dxdy$$

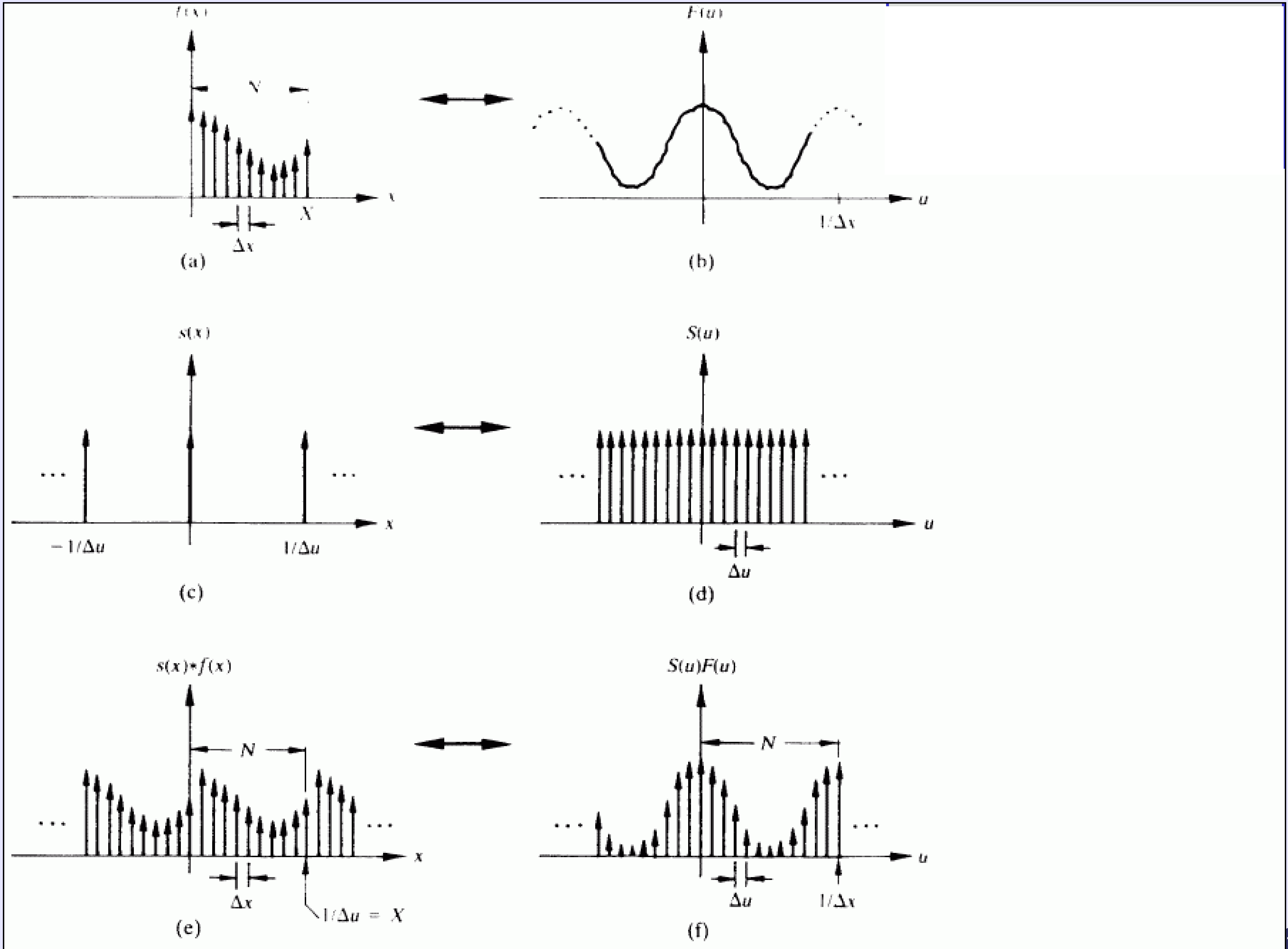
direct

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

inverse

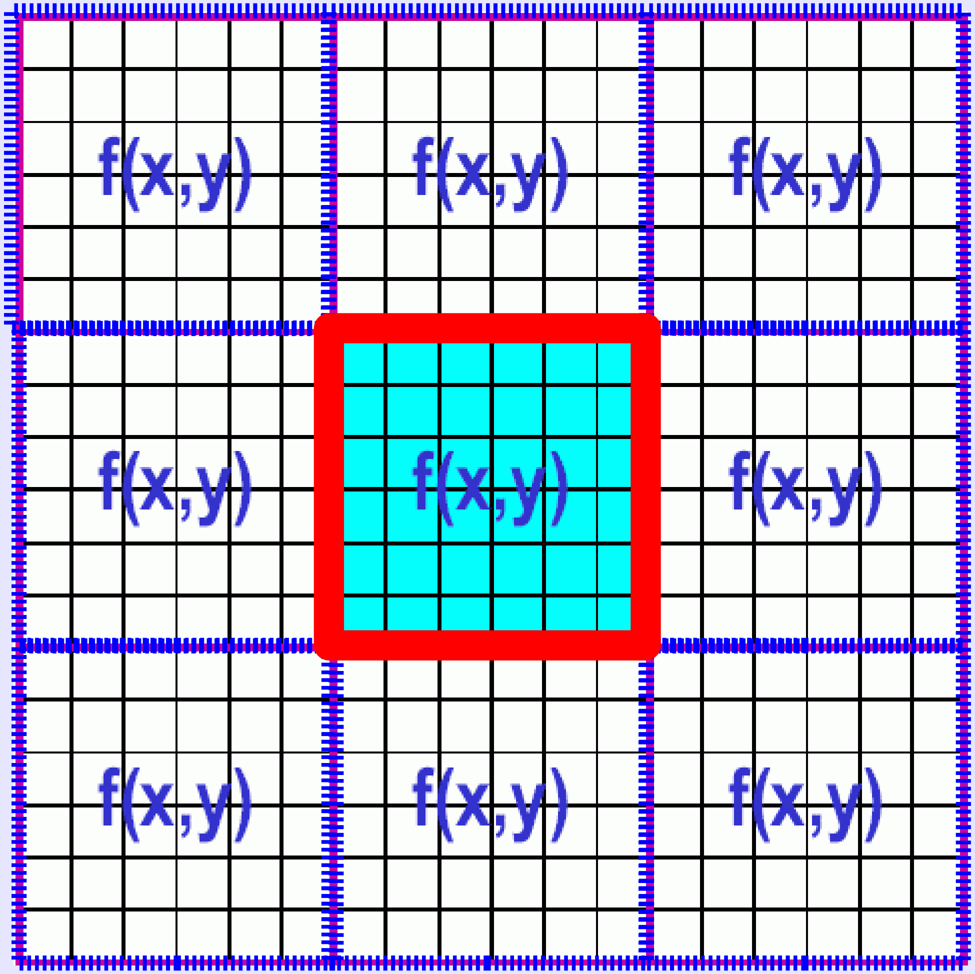
Exemplary transforms




f(x)
F(u)
X
Δx
1/Δx
(a)
(b)
s(x)
S(u)
−1/Δu
1/Δu
Δu
(c)
(d)
s(x)*f(x)
S(u)F(u)
N
Δx
1/Δu = X
Δu
1/Δx
(e)
(f)

Digital image as a discrete function

An image is assumed to be a discrete function with a size of (N,N), and periodical!



FT for discrete functions

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

$$f(x,y)=f(x_0+x\Delta x, y_0+y\Delta y), \qquad x,y=0,\dots,N-1$$

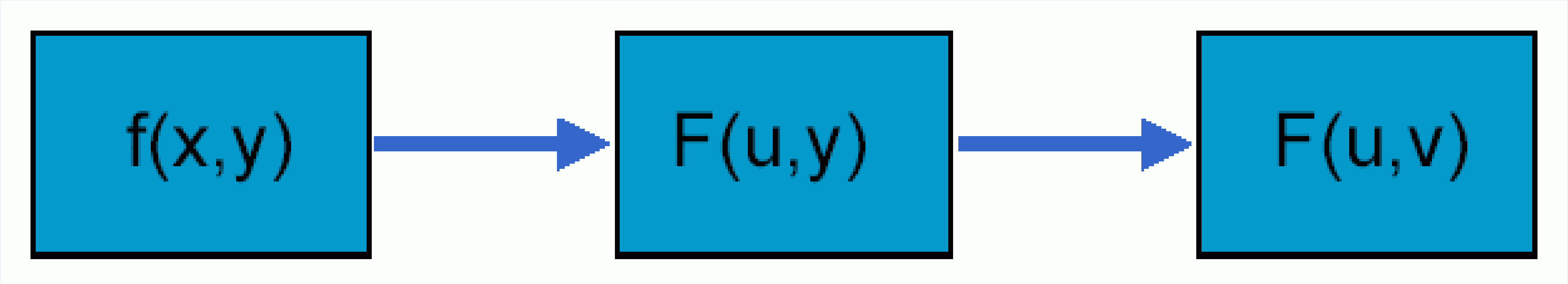
$$f(x,y)=\frac{1}{N}\sum_{u=0}^{N-1}\sum_{v=0}^{N-1} F(u,v)e^{j2\pi(ux+vy)/N}$$

$$F(u,v)=F(u_0+u\Delta u, v_0+v\Delta v), \qquad u,v=0,\dots,N-1$$

$$\Delta u=\frac{1}{N\Delta x}, \qquad \Delta v=\frac{1}{N\Delta y}$$

Practical property

Separability of Fourier Transform means that:



2D FT = FT along rows → FT along columns of an image

General properties

$$
\begin{aligned}
F(u,-v) &= F(u,W-v) \\
F(-u,v) &= F(H-u,v) \\
F(-u,-v) &= F(H-u,W-v) \\
F(aH+u,bW+v) &= F(u,v);\ a,b\varepsilon Z
\end{aligned}
$$

$$
\begin{aligned}
f(-m,n) &= f(H-m,n) \\
f(m,-n) &= f(m,W-n) \\
f(-m,-n) &= f(H-m,W-n) \\
f(aH+m,bW+n) &= f(m,n);\ a,b\varepsilon Z
\end{aligned}
$$

Phase and a module of spectrum

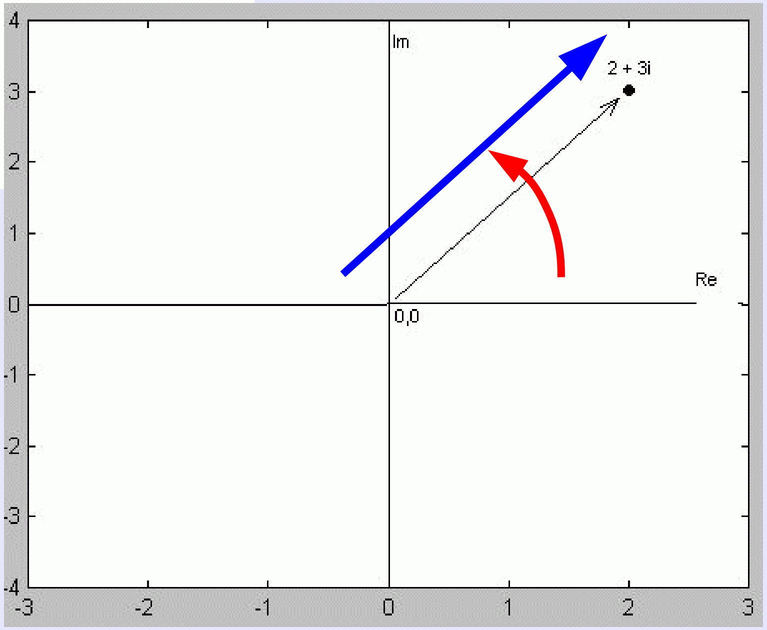
Image spectrum can be decomposed into module (abs) and phase

$$F(u,v) = |F(u,v)|\, e^{-j\arg[F(u,v)]}$$

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

$$\arg(F(u,v)) = arctan\frac{\mathrm{Im}(F(u,v))}{\mathrm{Re}(F(u,v))}$$

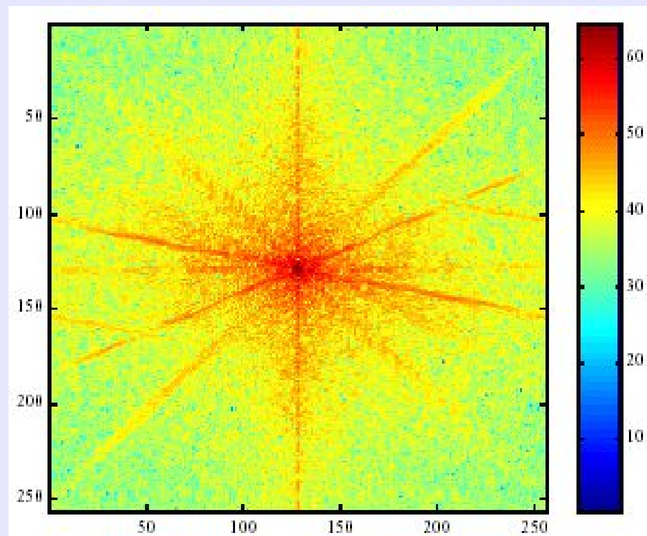
Graphical representation of typical complex numbers →



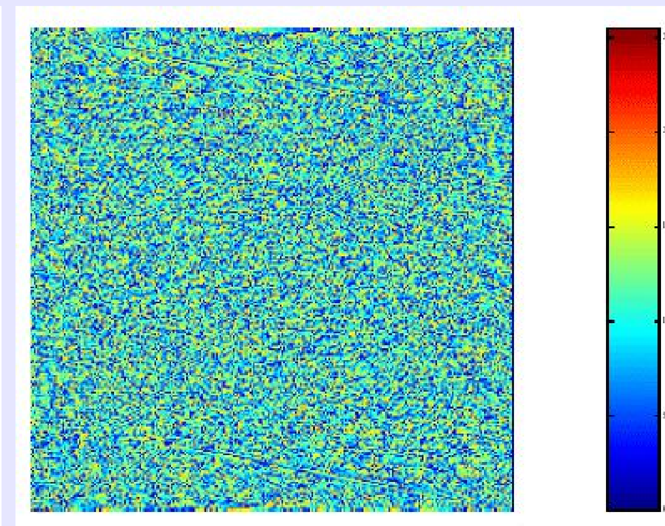
Phase and module

Module is often named „amplitude spectrum” or „power spectrum” while phase is known as „phase spectrum”

Sample image



module



phase

Fast Fourier Transform (FFT)

We start from general formula:

$$C(k) = \sum_{n=0}^{N-1} x(n) \exp\left(-j \frac{2\pi}{N} kn\right), \quad \forall k \in \overline{0, N-1}$$

The calculations require N*N computations with complex numbers.

In order to reduce the computations number we transform above formula...

FFT (2)

Input vector X is decomposed into two vectors:
$X^{(p)}$ i $X^{(np)}$ according to the following :

$$\left.\begin{aligned} X^{(p)}(n) &= x(2n) \\ X^{(np)}(n) &= x(2n+1) \end{aligned}\right\}, \text{for } n \in 0, \frac{N}{2} - 1$$

where “p” means even alements and “np” - odd elements.

Hence we decompose the vector using above assumption:

FFT (3)

$$C(k) = \sum_{n=0}^{N/2-1}\left[x(2n)\exp\left(-j\frac{2\Pi}{N}k\,2n\right) + x(2n+1)\exp\left(-j\frac{2\Pi}{N}k(2n+1)\right)\right] =$$

$$= \sum_{n=0}^{N/2-1} x(2n)\exp\left(-j\frac{2\Pi}{N}k\,2n\right) + \sum_{n=0}^{N/2-1} x(2n+1)\exp\left(-j\frac{2\Pi}{N}k(2n+1)\right) =$$

$$= \sum_{n=0}^{N/2-1} x(2n)\exp\left(-j\frac{2\Pi}{N}k\,2n\right) + \sum_{n=0}^{N/2-1} x(2n+1)\exp\left(-j\left(\frac{2\Pi}{N}k\,2n + \frac{2\Pi}{N}k\right)\right) =$$

$$= \sum_{n=0}^{N/2-1} x(2n)\boxed{\exp\left(-j\frac{2\Pi}{\frac{N}{2}}kn\right)} + \exp\left(-j\frac{2\Pi k}{N}\right)\sum_{n=0}^{N/2-1} x(2n+1)\boxed{\exp\left(-j\frac{2\Pi}{\frac{N}{2}}kn\right)}$$

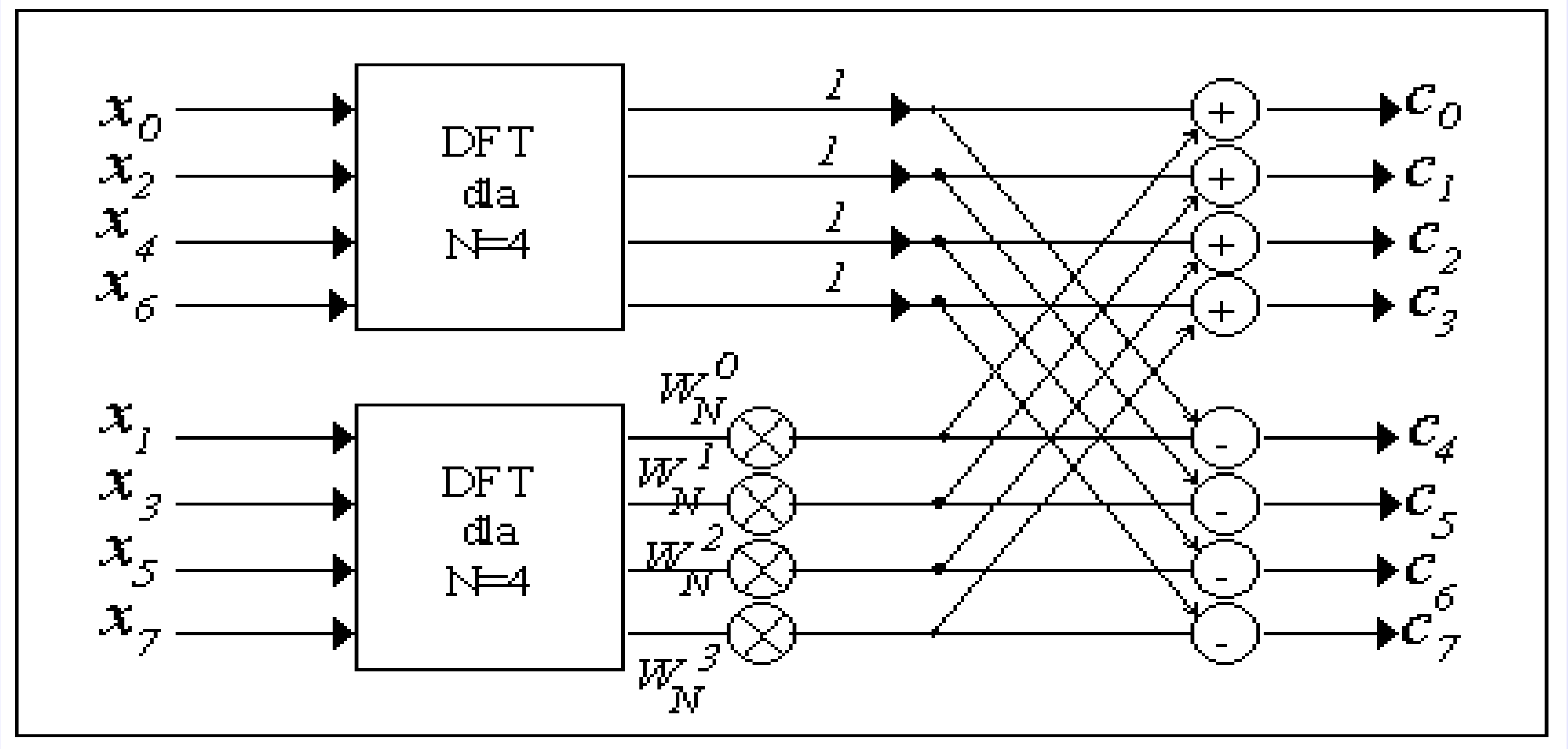
Cooley-Tuckey - FFT (4)

FFT = descrete FT (DFT), where a period is equal to *N/2* próbek,

FFT for $N = 8$

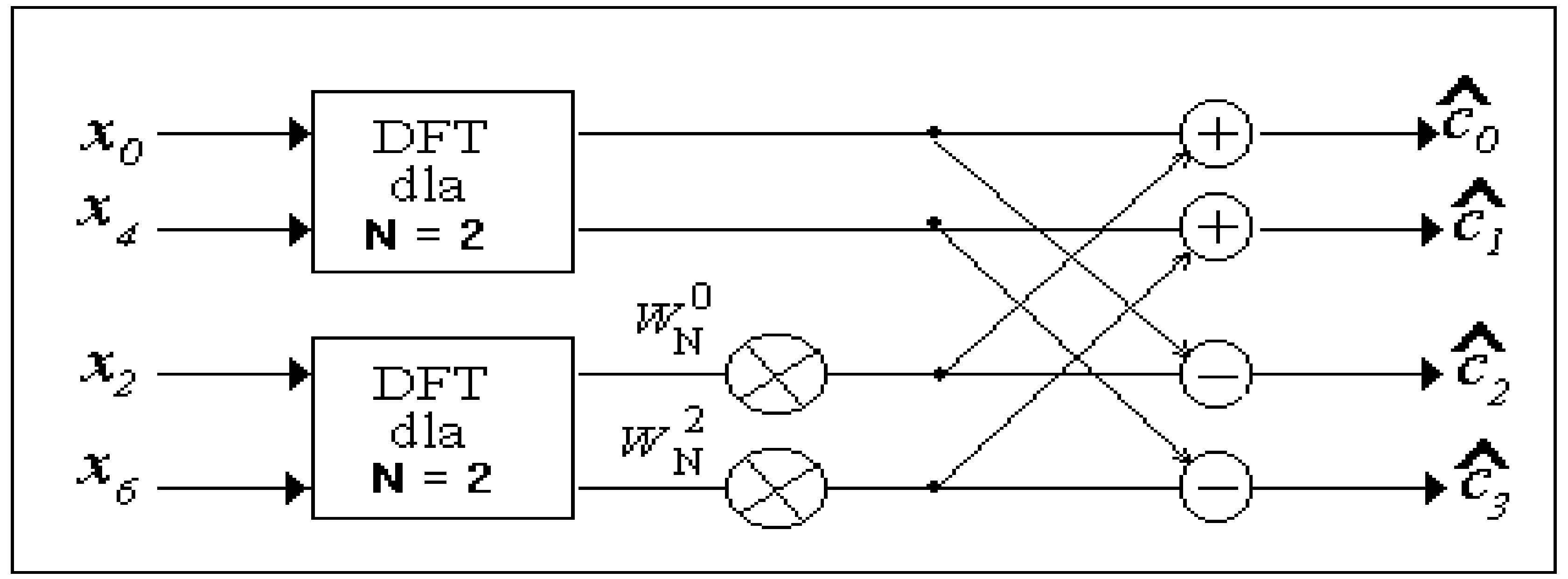
$$W_N^k = \exp(-j2\pi k / N), \quad \forall k \in \overline{0, N-1}$$

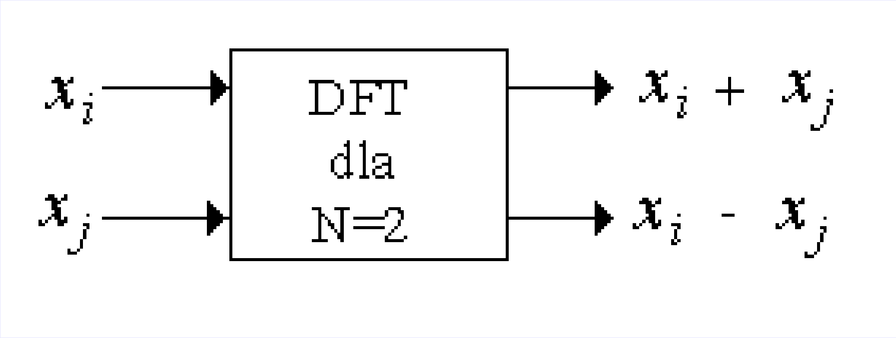
$$k \geq (N/2), \quad W_N^{k+N/2} = -W_N^k$$



FFT (5)

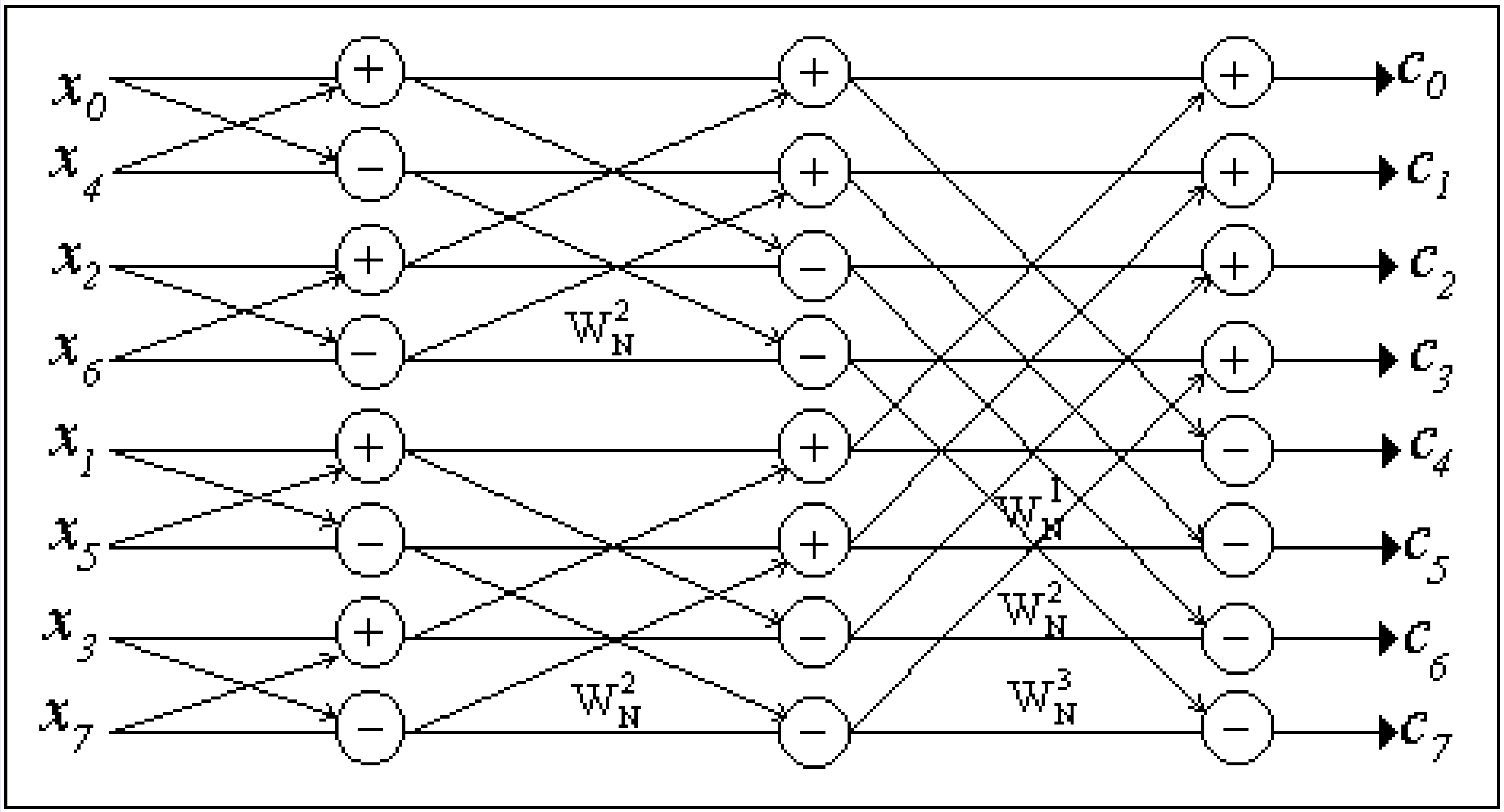
DFT for $N = 4$ is simple:





FFT (6)

When we join all previous elements, we get:



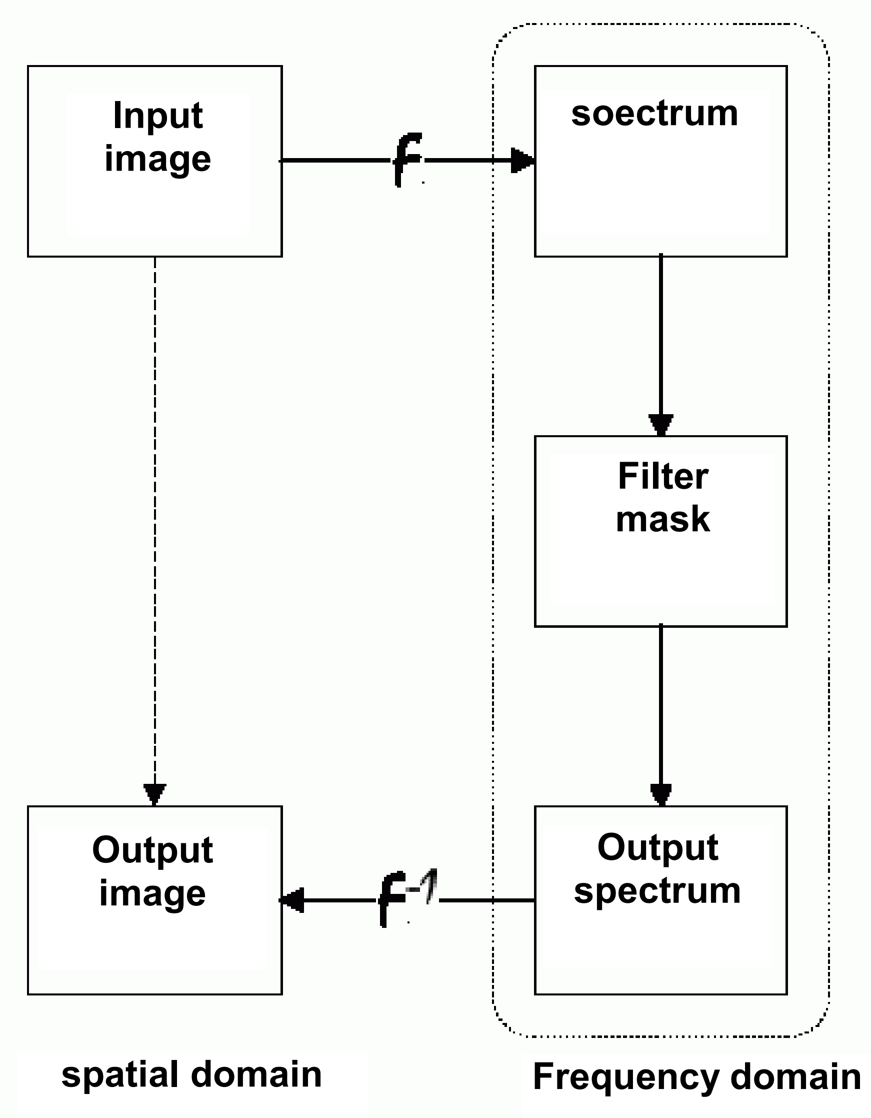
FFT for $N = 8$, $W_N^k = \exp\left(-j2\pi k / N\right), \quad \forall k \in \overline{0, N-1}$

Spectrum and its applications:

Filtering
Analysis of features (recognition)
Coding
Compression
Digital watermarking

Still images and video

The general idea of filtering in the frequency domain



DCT (ang. discrete cosine transform)

Transforms limited number N of real elements
g(0), … , g(N-1)

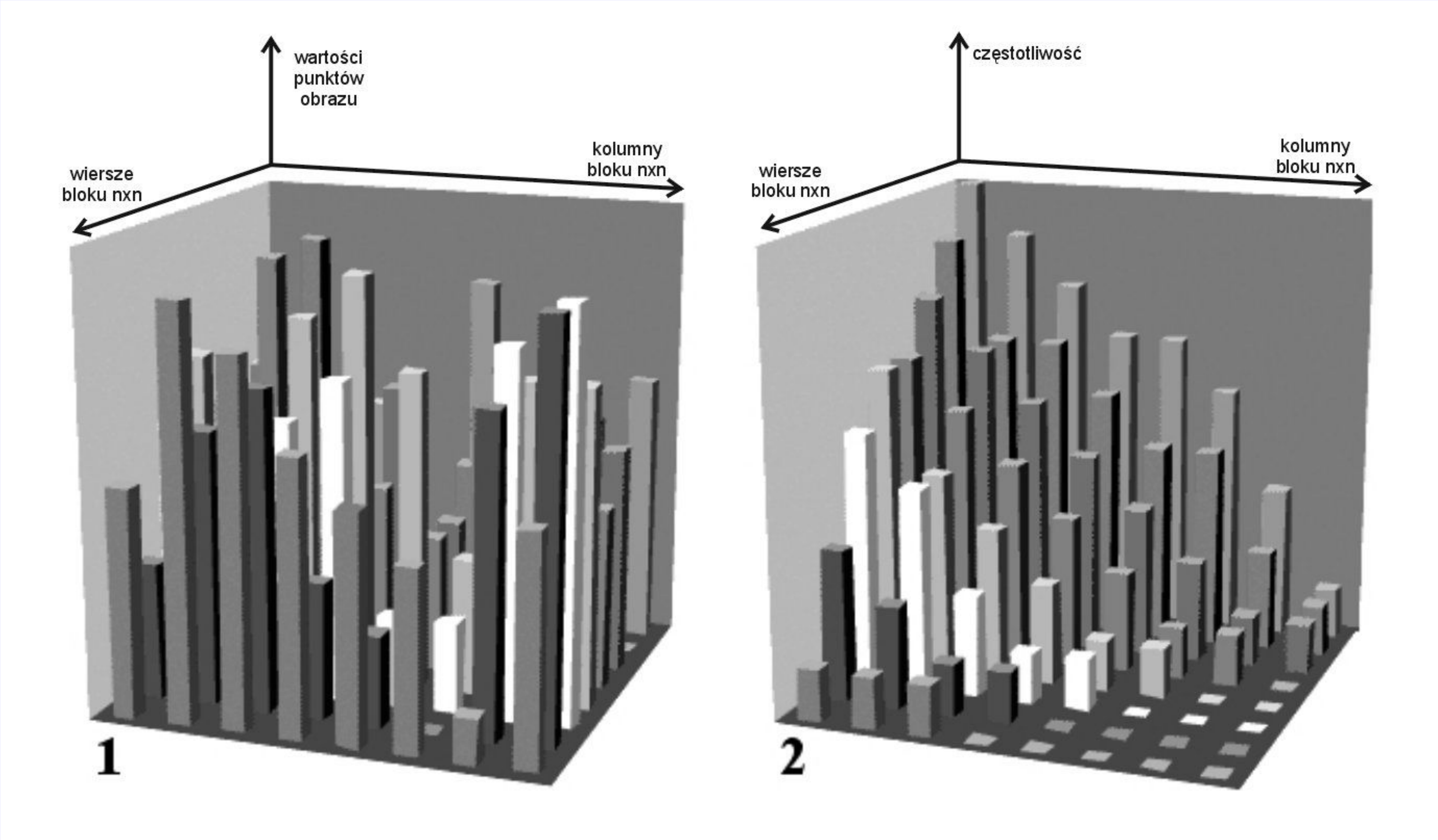
into another N real elements
G(0), ..., G(N-1)

According to:

$$G(0) = \frac{1}{\sqrt{N}} \sum_{m=0}^{N-1} g(m)$$

$$G(k) = \sqrt{\frac{2}{N}} \sum_{m=0}^{N-1} g(m) \cos \frac{\pi k(2m+1)}{2N} \quad \text{dla} \quad k = 1, 2, \ldots, N-1$$

DCT spectrum

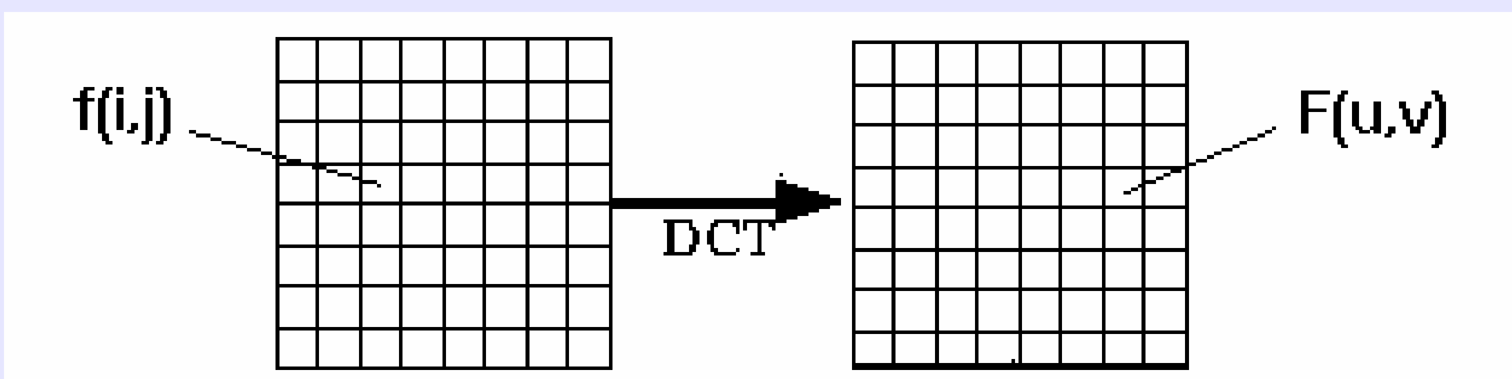


DCT 1D

Let us assume that *f*(*i*, *j*) is a pixel at the coordinates (*i*, *j*), and *F*(*u*, *v*) is a transformation result (where *U and V* represent *I* i *J*):

$$F(u,v) = \frac{C(u)}{2}\frac{C(v)}{2}\sum_i \sum_j f(i,j) * \cos\frac{(2x+1)u\pi}{16} * \cos\frac{(2y+1)v\pi}{16}$$

$C(u) = \frac{1}{\sqrt{2}}$ for $u = 0$ or $C(u) = 1$ for $u > 0$ $C(v) = \frac{1}{\sqrt{2}}$ for $v = 0$ or $C(v) = 1$ for $v > 0$



DCT2

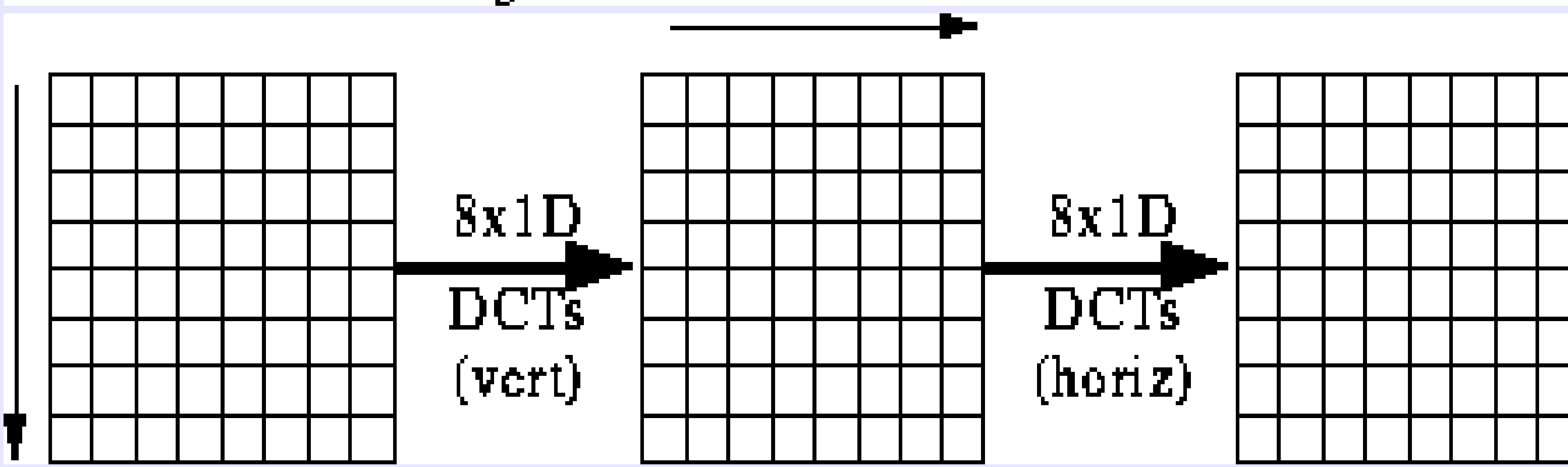
$$F[u,v] = \frac{1}{4}\sum_{i,j}\Lambda(u)\Lambda(v)\cos\frac{(2i+1)u\pi}{16}\cos\frac{(2j+1)v\pi}{16}\,f(i,j)$$

$$\Lambda(\xi) = \begin{cases} \frac{1}{\sqrt{2}} & \text{for } \xi = 0 \\ 1 & \text{otherwise} \end{cases}$$

Obliczanie DCT2D na podstawie DCT 1D

$$F[u,v] = \frac{1}{2}\sum_{i} \Lambda(u)\cos\frac{(2i+1)u\pi}{16}G[i,v]$$

$$G[i,v] = \frac{1}{2}\sum_{j} \Lambda(v)\cos\frac{(2j+1)v\pi}{16}f[i,j]$$



JPEG ?

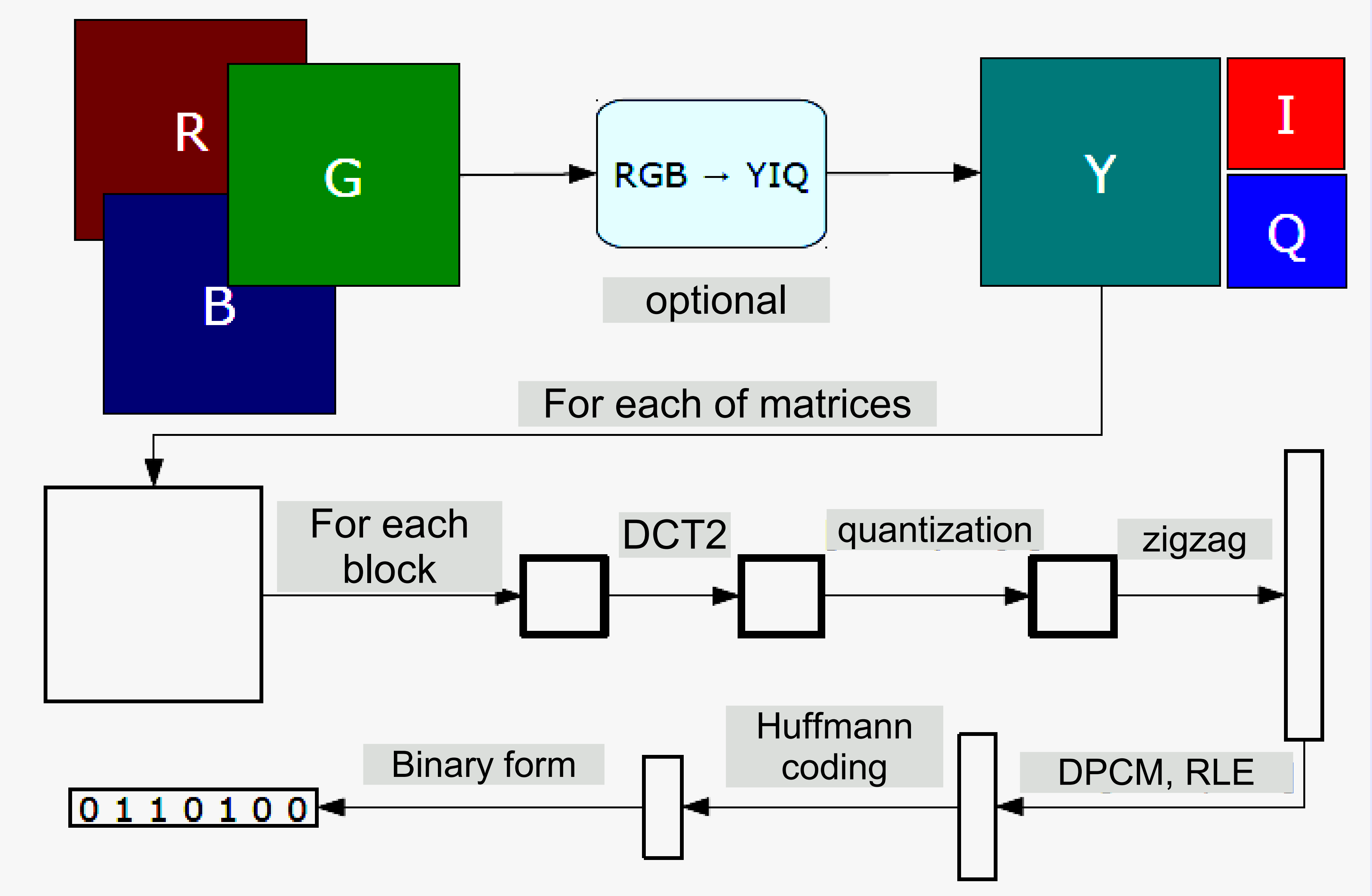
In computing, JPEG - named after its creator the Joint Photographic Experts Group - (/ˈdʒeɪpɛg/ JAY-peg) (seen most often with the .jpg extension) is a commonly used method of lossy compression for digital photography (i.e. images). The degree of compression can be adjusted, allowing a selectable tradeoff between storage size and image quality. JPEG typically achieves 10:1 compression with little perceptible loss in image quality, and is the file type most often produced in digital photography.

Input data for **JPEG:**

Monochormatic image – single matrix of natural numbers

Color image (RGB mostly) – three matrices of natural numbers, responsible for three elementary channels

Algorithm


R
G
B
RGB → YIQ
optional
Y
I
Q
For each of matrices
For each block
DCT2
quantization
zigzag
DPCM, RLE
Huffmann coding
Binary form
0 1 1 0 1 0 0

YIQ

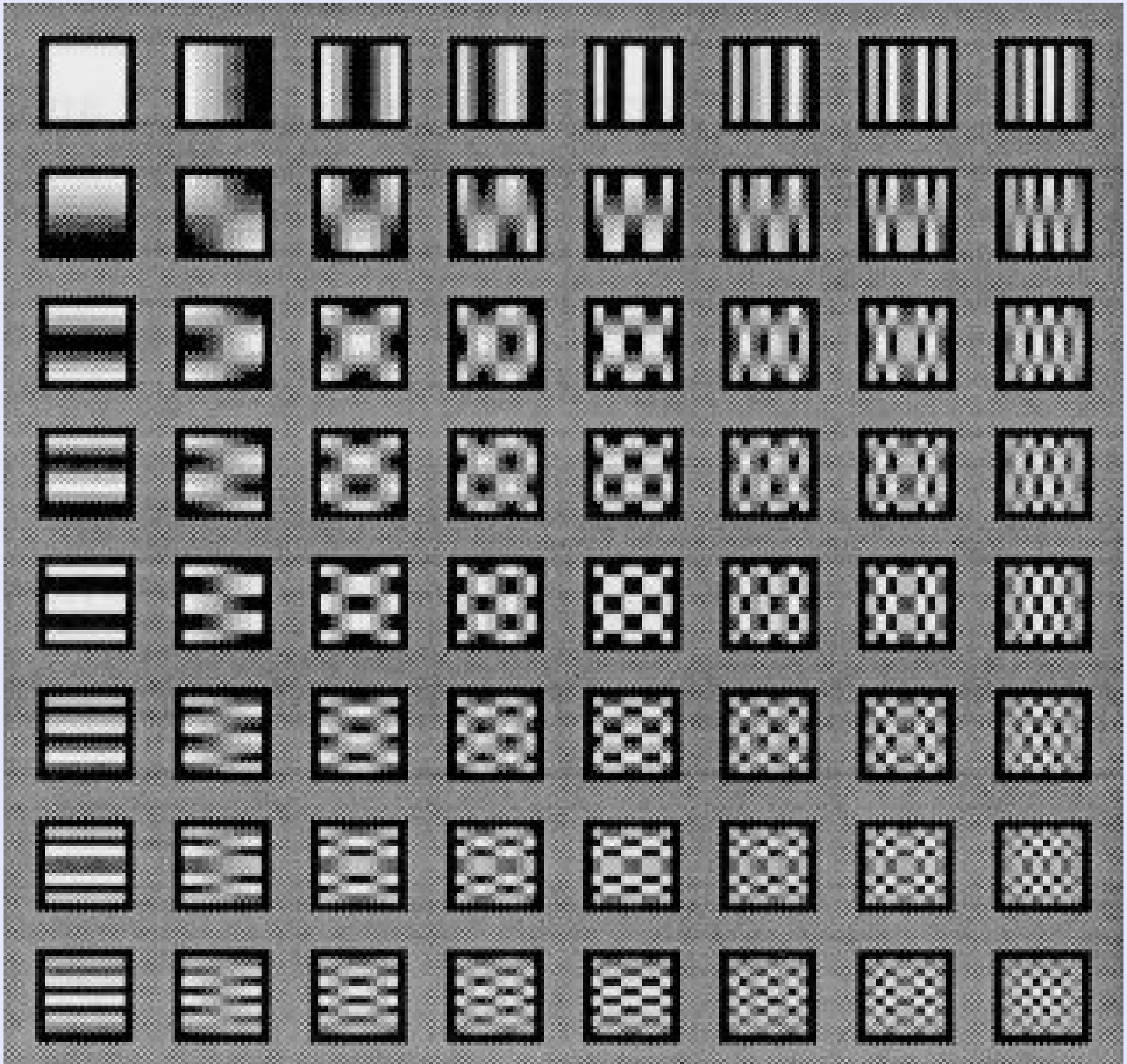
Input image is defined as 3 matrices:

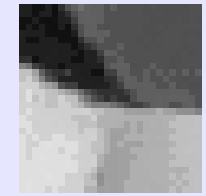
$R=[\ r_{ij}\ ],\ G=[\ g_{ij}\ ],\ B=[\ b_{ij}\ ];$

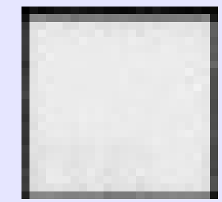
After conversion we have 3 matrices Y,I,Q:

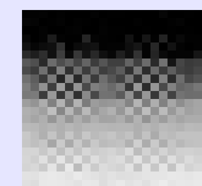
$$\begin{bmatrix} y_{ij} \\ i_{ij} \\ q_{ij} \end{bmatrix} = \begin{bmatrix} 0.229 & 0.587 & 0.114 \\ -0.168 & -0.257 & -0.321 \\ 0.212 & -0.528 & 0.311 \end{bmatrix} \begin{bmatrix} r_{ij} \\ g_{ij} \\ b_{ij} \end{bmatrix}$$

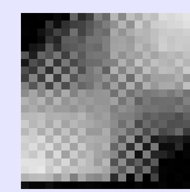
Base functions



= 

+ W_1 * 

+ W_2 * 

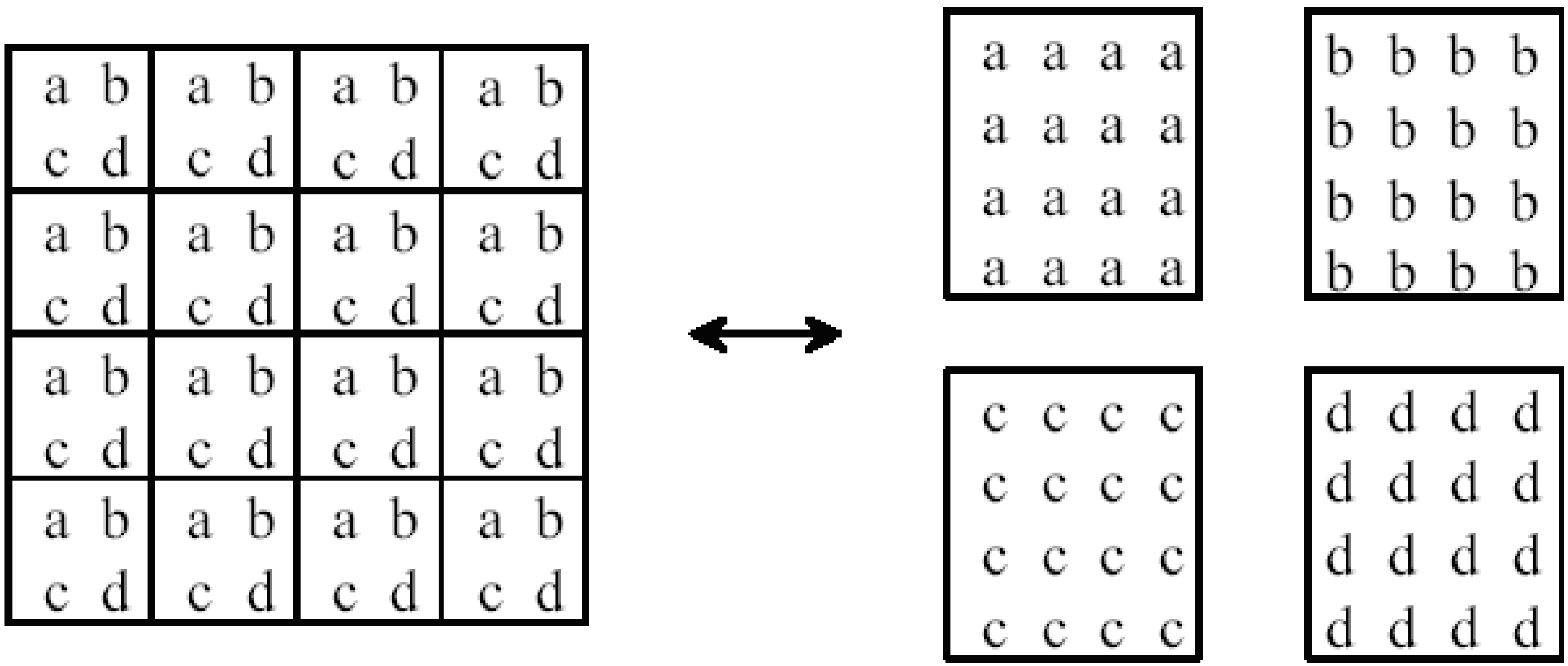
+ W_3 * 

+ . . .

Decomposition

Block after transformation

Band-images



Quantization

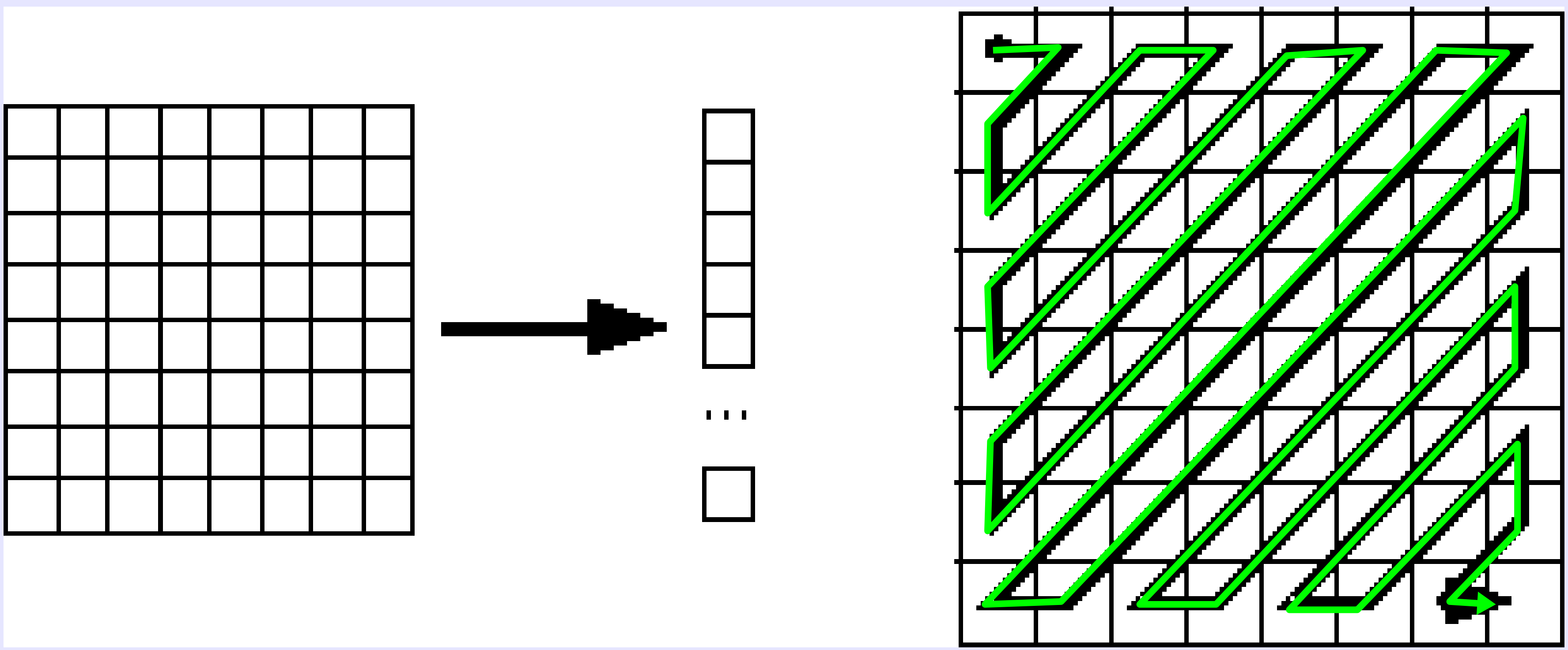
Quantization matrix Q

16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

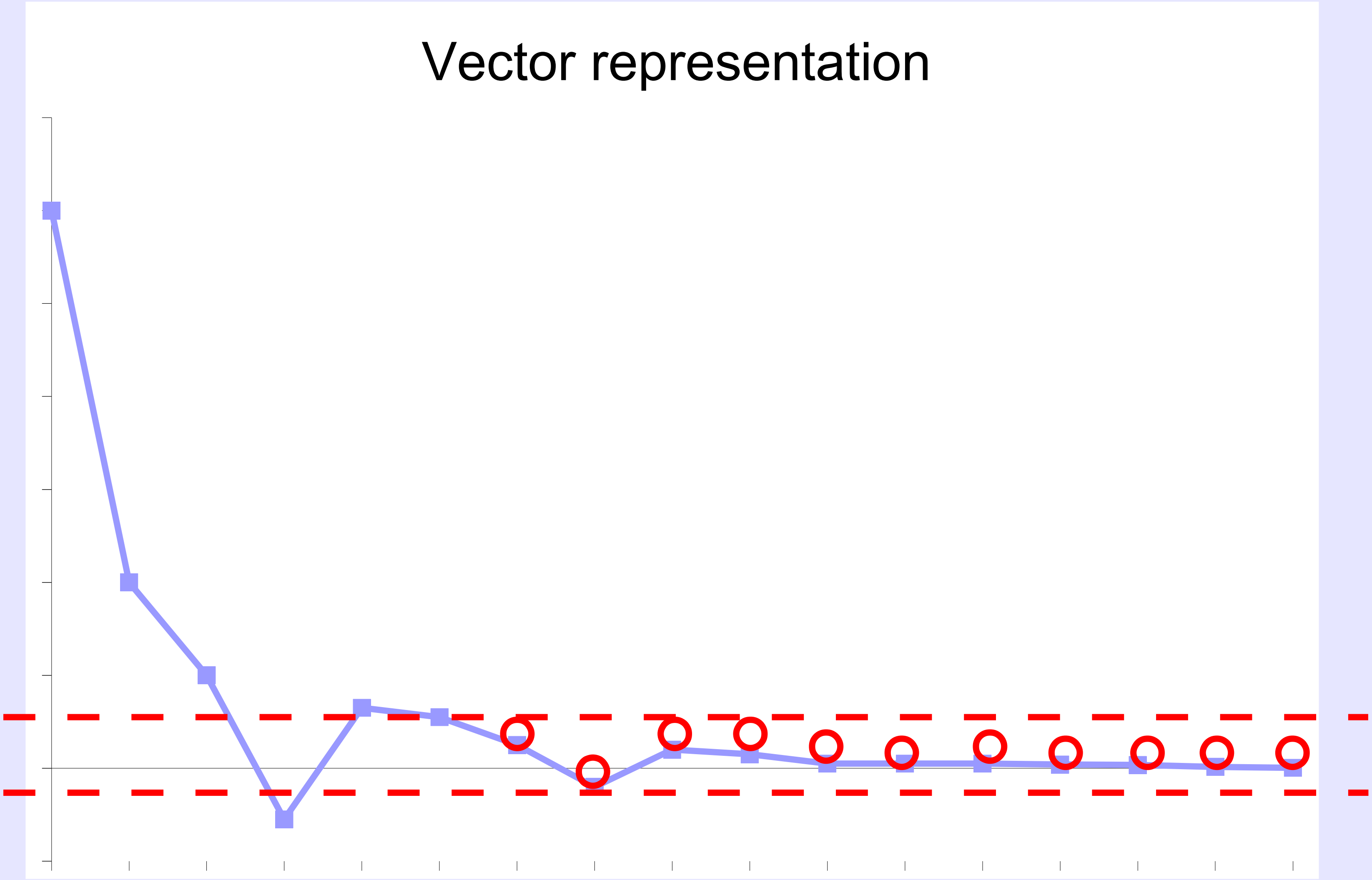
Block after DCT2 / Quant Table = Output

Zig-zag

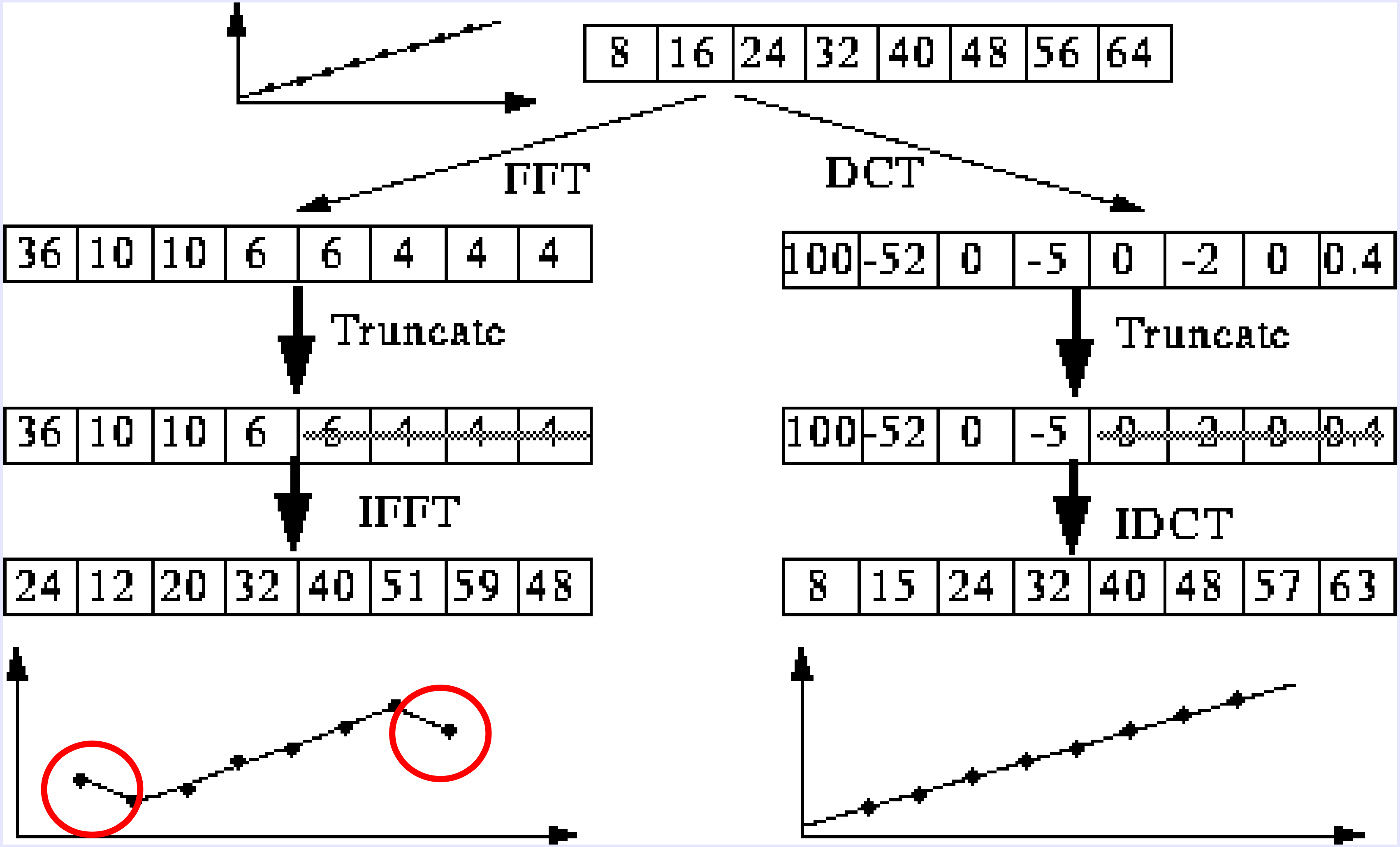
Zigzag makes vectors 1x 64 out of 8x8 matrices



After zigzag...



Why DCT not FFT?



Comparison

8kB - JPEG

45kB - JPEG

8kB - JPG2000