# 2D image compression

## 2D image compression

• General methods not satisfactory for images:

RLE – short runs

Statistical methods – similar probabilities for different colors/shades of gray

Dictionary based methods – short repeating patterns (digitalization…)

## 2D image compression

• Fourier transform (FT), Discrete FT, Wavelet transform (WT), Space filling curves (SPC) …

- exploit pixel correlation (2D)
- allow lose of information

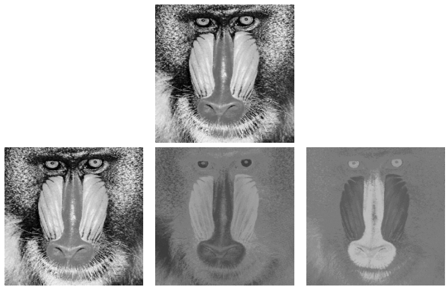
## Image compression

• Representation
• Transformation
• Quantization
• Encoding

# Representation

- RGB, YUV, CIE, HSV …
- Y = 0.299R + 0.587G + 0.114B
- U = 0.492(B – Y)
- V = 0.877(R – Y)
- Chrominance (UV) can be lossy → highly compressed

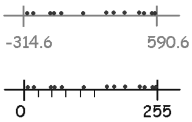
# Representation - YUV

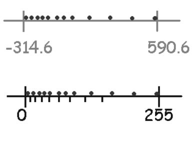


# Quantization

Uniform:
PxQrange/Orange

Px255/905.2




Non - Uniform:

# Fourier transform

Transform = mathematical tool to solve problems

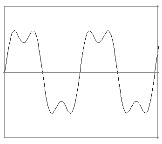
Change a quantity to another form that may exhibit useful features : XCVI x XII → 96 x 12 = 1152 → MCLII

Signal in **time domain** → **frequency domain**

(Time = image space)

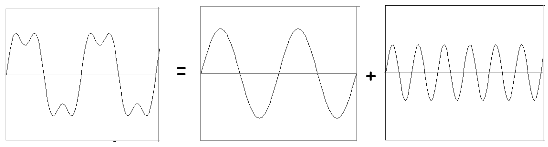
# Time & Frequency

example : $g(t) = \sin(2\pi f t) + (1/3)\sin(2\pi(3f)t)$



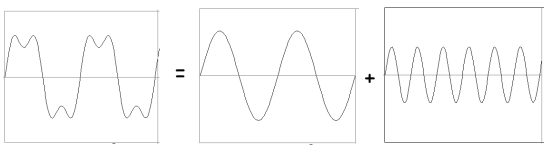
# Time & Frequency

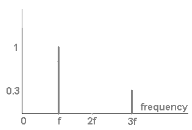
example : $g(t) = \sin(2\pi f t) + (1/3)\sin(2\pi(3f)t)$




# Time & Frequency

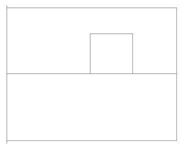
example : $g(t) = \sin(2\pi f t) + (1/3)\sin(2\pi(3f)t)$



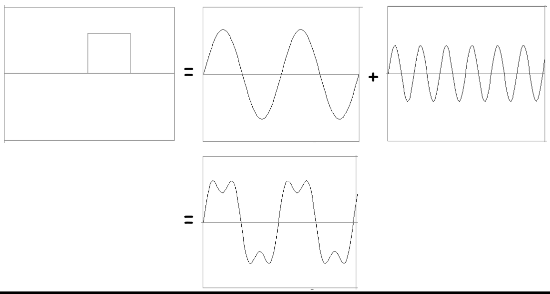

# Time & Frequency

example: $g(t)=\begin{cases} 1, & -a/2 < t < a/2 \\ 0, & \text{elsewhere} \end{cases}$



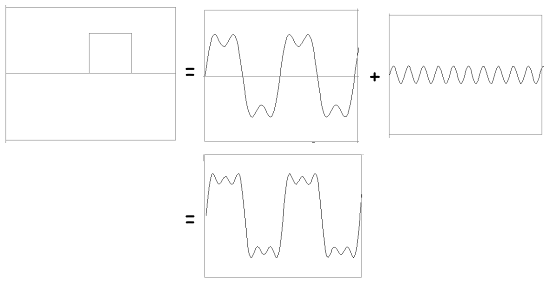
## Time & Frequency

example: $g(t)=\begin{cases} 1, & -a/2 < t < a/2 \\ 0, & \text{elsewhere} \end{cases}$



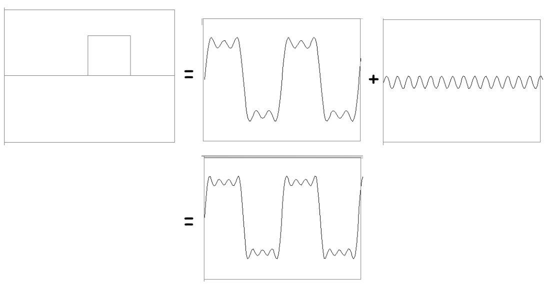
## Time & Frequency

example: $g(t)=\begin{cases} 1, & -a/2 < t < a/2 \\ 0, & \text{elsewhere} \end{cases}$



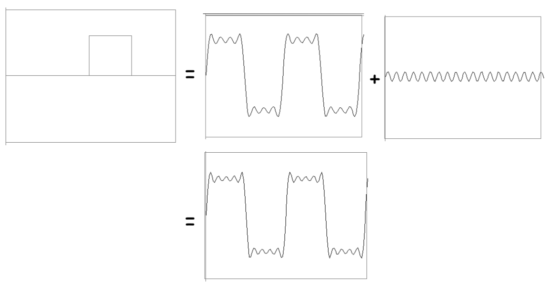
## Time & Frequency

example: $g(t)=\begin{cases} 1, & -a/2 < t < a/2 \\ 0, & \text{elsewhere} \end{cases}$



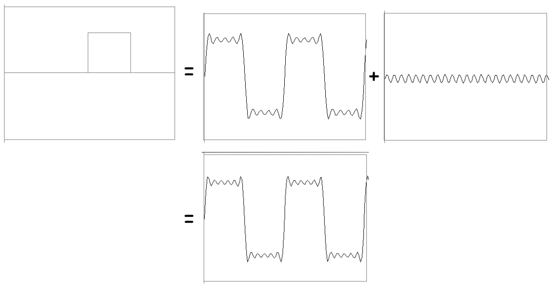
## Time & Frequency

example: $g(t)=\begin{cases} 1, & -a/2 < t < a/2 \\ 0, & \text{elsewhere} \end{cases}$



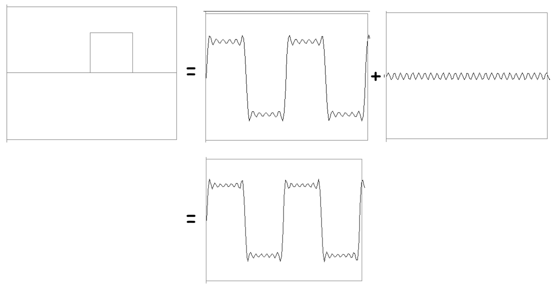
# Time & Frequency

example: $g(t)=\begin{cases} 1, & -a/2 < t < a/2 \\ 0, & \text{elsewhere} \end{cases}$



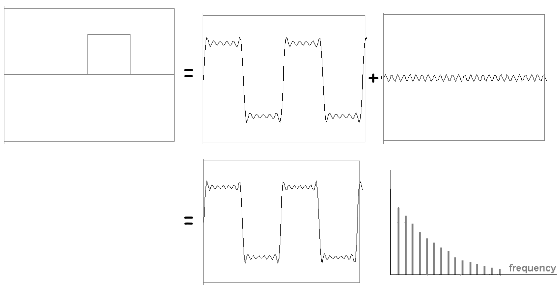
# Time & Frequency

example: $g(t)=\begin{cases} 1, & -a/2 < t < a/2 \\ 0, & \text{elsewhere} \end{cases}$ $A\sum_{k=1}^{\infty}(1/k)\sin(2\pi kft)$



# Time & Frequency

example: $g(t)=\begin{cases} 1, & -a/2 < t < a/2 \\ 0, & \text{elsewhere} \end{cases}$ $A\sum_{k=1}^{\infty}(1/k)\sin(2\pi kft)$

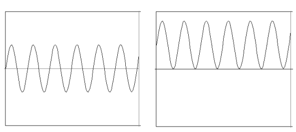



# Frequency domain

Spectrum of freq. domain : range of freq.

Bandwidth of freq. domain : width of the spectrum

Dc component (direct current): component of zero freq.



# Fourier transform

$$G(f) = \int_{-\infty}^{\infty} g(t)[\cos(2\pi ft) - i \sin(2\pi ft)]dt$$

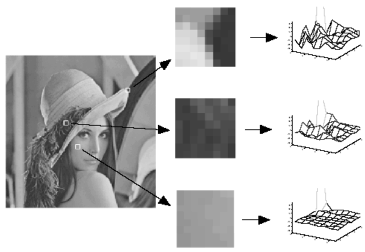
$$g(t) = \int_{-\infty}^{\infty} G(f)[\cos(2\pi ft) + i \sin(2\pi ft)]df$$

- Discrete

$$G(f) = (1/n)\sum_{t=0}^{n-1} g(t)[\cos(2\pi ft/n) - i \sin(2\pi ft/n)] , \quad 0<f<n-1$$

$$g(t) = (1/n)\sum_{f=0}^{n-1} G(f)[\cos(2\pi ft/n) + i \sin(2\pi ft/n)] , \quad 0<t<n-1$$

# FT for digitized image

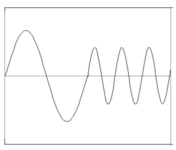


- neighboring pixels "close" values → surface almost flat → most FT coeff. small (large DC small AC coeff.)
- Loose unimportant image info buy cut $G_{ij}$ at right bottom

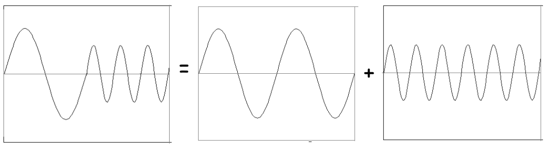
# FT for digitized image

- image = (continuous) signal of intensity function i(t)
- Sampling: store a finite sequence in memory i(1)…i(n)
- The bigger the sample, the better the quality? - no
- Sampling theory: sample an image and reconstruct it without loss of quality if we can :
  - Transform i(t) function from time to freq. Domain
  - Find the max freq. $F_m$
  - Sample i(t) at rate $\geq 2f_m$
  - Store the sampled values in a bitmap

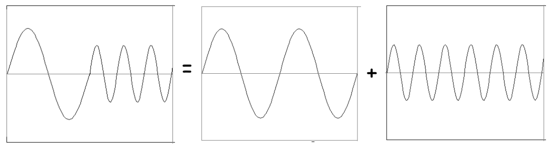
# Fourier transform

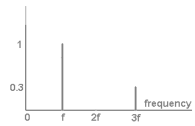


# Fourier transform

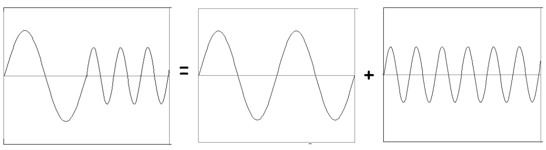


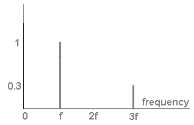

# Fourier transform

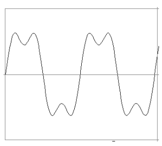


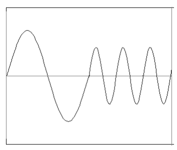

# Fourier transform

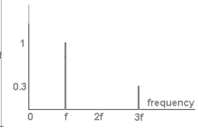


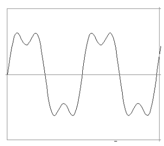



# Fourier transform

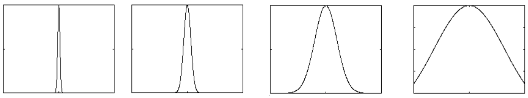




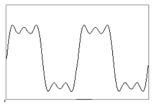
Either time or frequency

## Short time FT

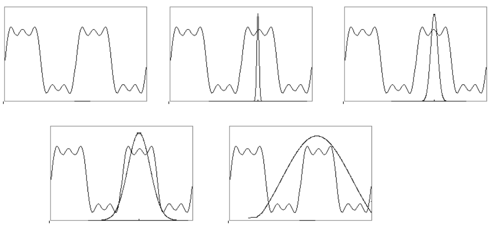


- narrow window – good time resolution, poor freq. resolution
- wide window – good freq. resolution, poor time resolution
- Fixed resolution

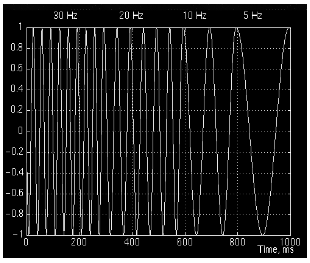
## Short time FT



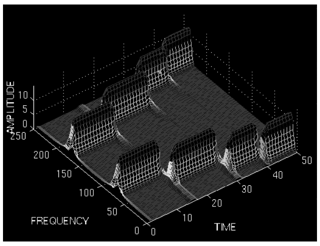
## Short time FT



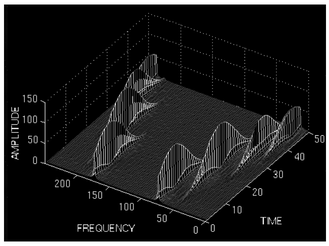
## Short time FT



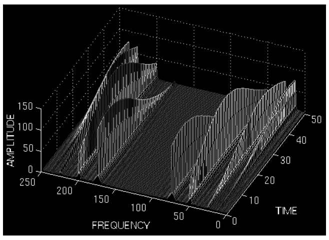
## Short time FT




## Short time FT

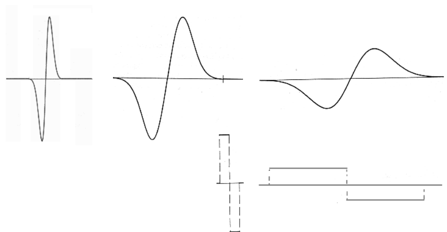



## Short time FT

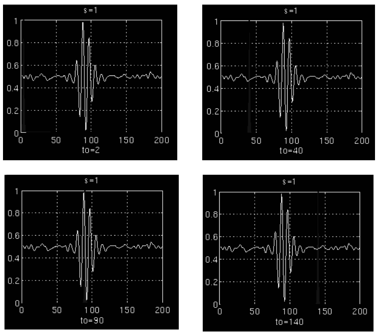



## Wavelet Transform

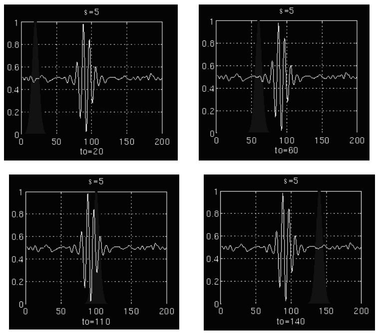
- what band of freq. exists at what time interval
- variable resolution



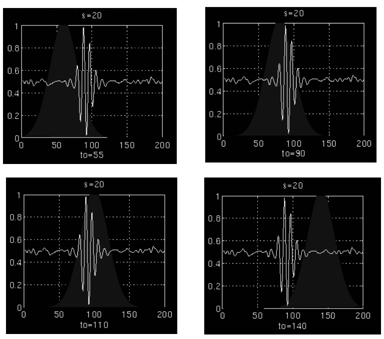
## WT – Multi resolution Analysis



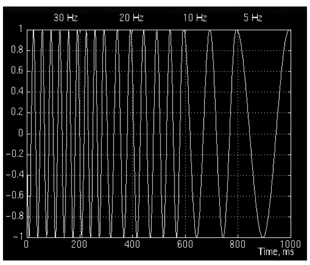

## WT – Multi resolution Analysis



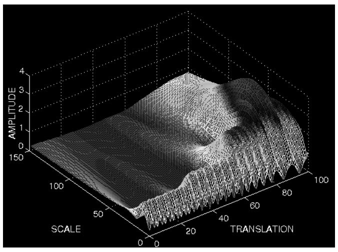

## WT – Multi resolution Analysis



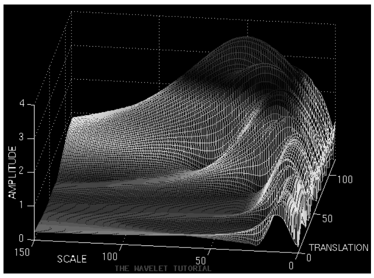

## WT – Multi resolution Analysis



## WT – Multi resolution Analysis




## WT – Multi resolution Analysis

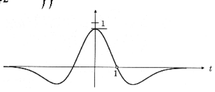



## WT – Multi resolution Analysis

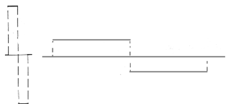
Wavelet function $\psi_{\tau,s} = \frac{1}{\sqrt{s}}\psi\left(\frac{t-\tau}{s}\right)$

Mexican hat $\psi(t) = \frac{1}{\sqrt{2\pi}\sigma^3}\left(e^{\frac{-t^2}{2\sigma^2}}\cdot\left(\frac{t^2}{\sigma^2}-1\right)\right)$

Second derivation of $w(t) = \frac{1}{\sqrt{2\pi}\sigma}e^{\frac{-t^2}{2\sigma^2}}$



Harr $\psi(x) = \begin{cases} 1 & 0 \le x < 1/2 \\ -1 & 1/2 \le x < 1 \\ 0 & x < 0,\ x > 1 \end{cases}$



## Wavelet Transform

$$CWT_x^{\psi}(\tau, s) = \Psi_x^{\psi}(\tau, s) = \frac{1}{\sqrt{|s|}}\int x(t)\psi^*\left(\frac{t-\tau}{s}\right)dt$$

$$\psi_{\tau,s} = \frac{1}{\sqrt{s}}\psi\left(\frac{t-\tau}{s}\right)$$

- inverse

$$x(t) = \frac{1}{c_{\psi}^2}\int_s\int_{\tau}\Psi_x^{\psi}(\tau, s)\frac{1}{s^2}\psi\left(\frac{t-\tau}{s}\right)d\tau\, ds$$

- scale = 1/freq.
- Discretization : non uniform sampling (high scale→ low freq. →decrease sampling)

# Wavelets

$$(1,\ 2,\ 3,\ 4,\ 5,\ 6,\ 7,\ 8)$$

$L = \sqrt{2^0}\,(\frac{1}{2},\frac{1}{2})$  $H = \sqrt{2^0}\,(-\frac{1}{2},\frac{1}{2})$ :

$$\sqrt{2^0}\ (3/2,\ 7/2,\ 11/2,\ 15/2, -1/2,\ -1/2,\ -1/2,\ -1/2)$$

$L = \sqrt{2^1}\,(\frac{1}{2},\frac{1}{2})$  $H = \sqrt{2^1}\,(-\frac{1}{2},\frac{1}{2})$ :

$$\sqrt{2^1}\ (10/4,\ 26/4,\ -4/4,\ -4/4, -1/2,\ -1/2,\ -1/2,\ -1/2)$$

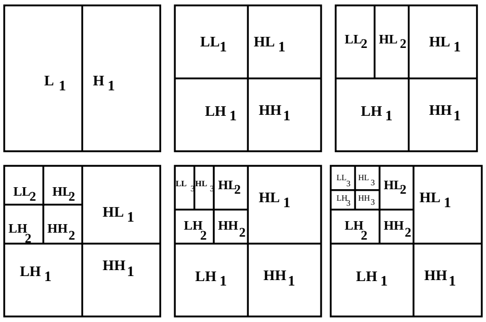
$L = \sqrt{2^2}\,(\frac{1}{2},\frac{1}{2})$  $H = \sqrt{2^2}\,(-\frac{1}{2},\frac{1}{2})$ :

$$\sqrt{2^2}\ (36/8, -16/8,\ -4/4, -4/4,\ -1/2,\ -1/2,\ -1/2,\ -1/2)$$

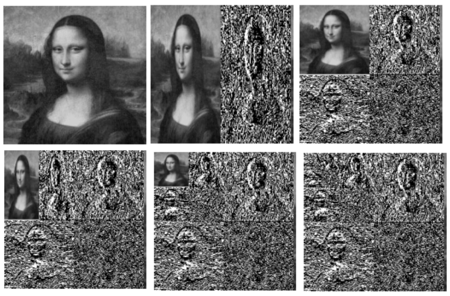
$$\left(\frac{36/8}{\sqrt{2^0}},\ \frac{-16/8}{\sqrt{2^0}},\ \frac{-4/4}{\sqrt{2^1}},\ \frac{-4/4}{\sqrt{2^1}}, \frac{-1/2}{\sqrt{2^2}},\ \frac{-1/2}{\sqrt{2^2}}, \frac{-1/2}{\sqrt{2^2}},\ \frac{-1/2}{\sqrt{2^2}}\right)$$

$L = (0,\ \frac{1}{4},\ \frac{1}{2},\ \frac{1}{4},\ 0)$  $H = (-1/8,\ \frac{1}{4},\ \frac{3}{4},\ \frac{1}{4},\ -1/8)$

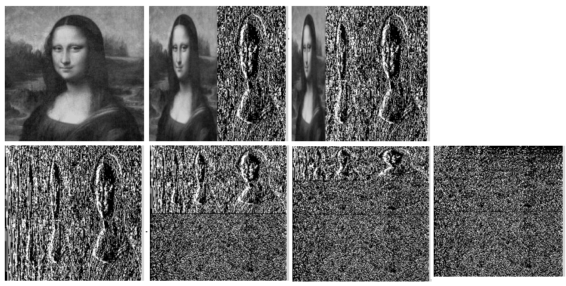
# Wavelets




# Wavelets



# Wavelets

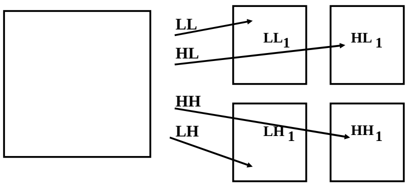


## Wavelets

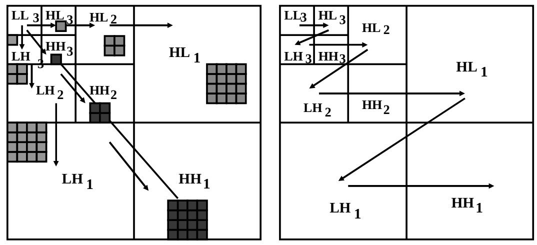
**Filters: L = (0, $\frac{1}{4}$, $\frac{1}{2}$, $\frac{1}{4}$, 0) H = (-1/8, $\frac{1}{4}$, $\frac{3}{4}$, $\frac{1}{4}$, -1/8)**

**LL : LL(i,j) = L(i)*L(j) HL : HL(i,j) = H(i)*L(j)**

**LH : LH(i,j) = L(i)*H(j) HH : HH(i,j) = H(i)*H(j)**




## Zerotrees




Zerotree : encode location of coefficients below threshold in subtrees

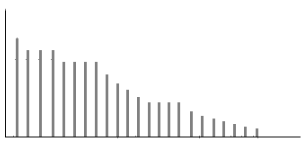
## Wavelet coding using zerotrees

•Encode = several passes with exp. decrease thresholds

• Each pass - send significance bits for new significant coeff

• Additionally send refinement bits for coeff that became significant in an earlier pass (no need to send their location)

• Decoder can reconstruct geometry associated with any bits prefix



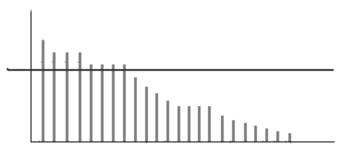
## Wavelet coding using zerotrees

•Encode = several passes with exp. decrease thresholds

• Each pass - send significance bits for new significant coeff

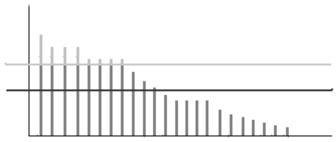
• Additionally send refinement bits for coeff that became significant in an earlier pass (no need to send their location)

• Decoder can reconstruct geometry associated with any bits prefix



## Wavelet coding using zerotrees

- Encode = several passes with exp. decrease thresholds
- Each pass - send significance bits for new significant coeff
- Additionally send refinement bits for coeff that became significant in an earlier pass (no need to send their location)
- Decoder can reconstruct geometry associated with any bits prefix



## Wavelets

- Effective decorrelation
- Correlation: Coarse level - fine level

## JPEG image compression

- lossy (also lossless mode)
- works best for continuous-tone images,
- Advantage – use many parameters, user can adjust amount of data lost (and thus compression ration)
- Progressive and hierarchical coding
- Main idea –lose data for which the human eye is not sensitive

## JPEG image compression

- Representation RGB → luminance/chrominance
- Downsampling (color images only), for chrominance components only
- DCT 8x8 blocks (data units)

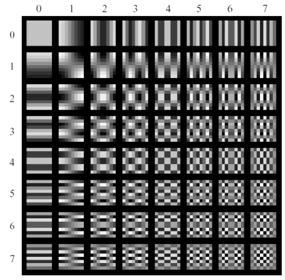
$$G_f = \left(\tfrac{1}{2}\right) C_f \sum_{t=0}^{7} P_t \cos((2t+1)f\pi/16), \quad C_f = \begin{cases} \frac{1}{\sqrt{2}} & f=0 \\ 1 & f=1\ldots7 \end{cases}$$

- real numbers
- fast implementation
- Separable (row/column)

## JPEG image compression

• DCT – cont
• DCT on blocks (not entire image):
small block
- faster
- correlation exists between neighboring pixels
large block
- better compression in “flat” regions
• Prepares data for losing information

## DCT Basis Functions

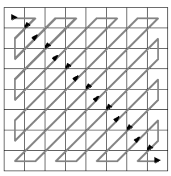



## JPEG image compression

• Quantization – 64 different coefficients
- Lossy step.
- High freq. has large QC.
- Each QC is JPEG parameter, can be specified by the user.
- Predefined tables – for each color component

## JPEG image compression

Encoding – RLE, Huffman conding, arithmetic
• Scan each block in zig-zag order



• DC component – first DC + differences (Huffman) : in continuous-tone images average of pixels in adjacent data units are close
• AC component – RLE + Huffman/Arithmetic coder: many zeros, few nonzero
• Headers

## JPEG image compression

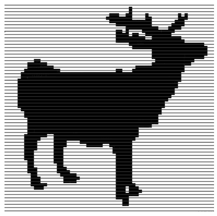
• Progressive mode (for quick coarse preview)
  - "scans" – of high freq., each – sharper image
  - Slow encoding (all step for each scan)

• Hierarchical mode (when high resolution image should be output in low resolution)
  - store image several times, several resolutions

•Lossless mode
  - differences from prediction based on neighbors
  - Huffman/arithmetic coder

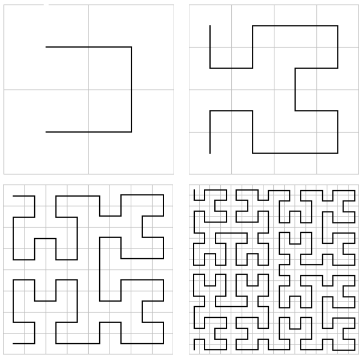
## Space Filling Curves

• order of scan



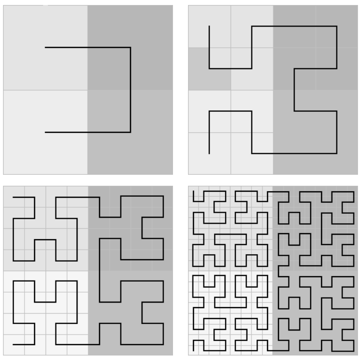
## Space Filling Curves

Hilbert curve (1-4)



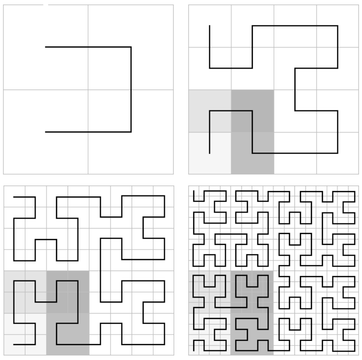
## Space Filling Curves

Hilbert curve (1-4)



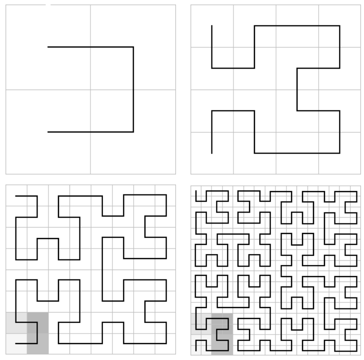
## Space Filling Curves

Hilbert curve
(1-4)



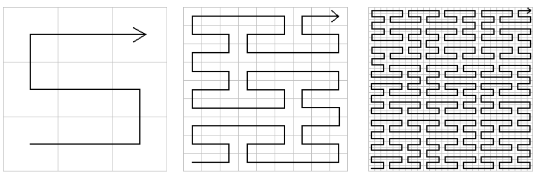
## Space Filling Curves

Hilbert curve
(1-4)



## Space Filling Curves

Peano curve



## Context Based SFC