BMB5113
COMPUTER VISION

TRANSFORMS

Transforms

- Different representations of the images are generated
  - An array of pixel values converted to a different form
- Helps extraction of targeted information in the transformed domain
- Sometimes an inverse transform is needed to obtain the original image
- Depending on the context different transforms can be useful
- Many different types of transforms exist
- Frequently used transforms
  - Hough transform, Fourier transform, Distance transform, Haar transform, Wavelet transform, …

Hough Transform

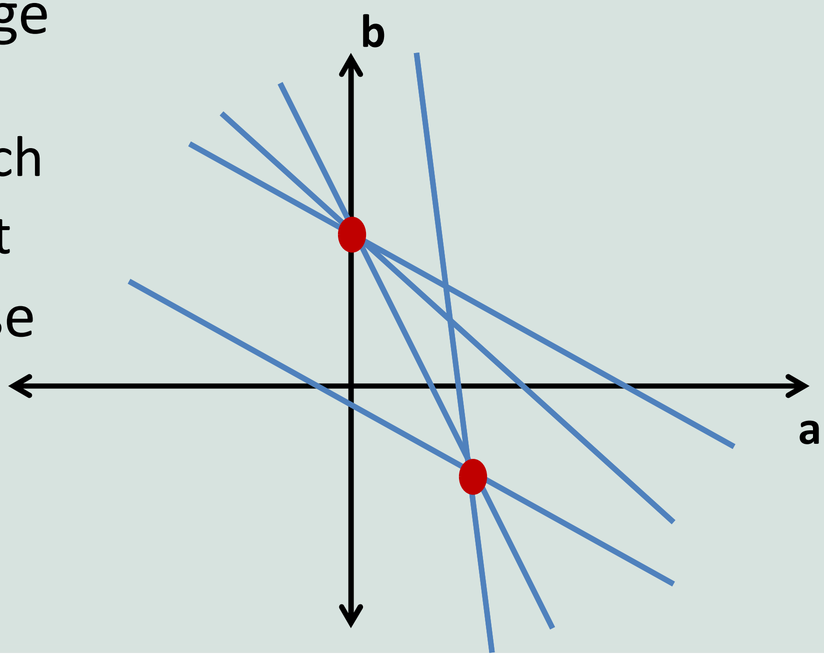
- Performed after edge detection
- It is a technique to isolate the curves of a given shape or shapes in a given image
- Classical Hough Transform can locate regular curves like straight lines, circles, parabolas, ellipses, etc.
  - Requires that the curve be specified in some parametric form
- Generalized Hough Transform can be used where a simple analytic description of feature is not possible

Advantages of Hough Transform

- The Hough transform is tolerant of gaps in the edges
- It is relatively unaffected by noise
- It is also unaffected by occlusion in the image

Hough Transform for Straight Line Detection

- A straight line can be represented as
  - y = ax + b
  - Each point in image is mapped to a line in the transform
    - Point $(2,3) \rightarrow 3 = 2a + b \rightarrow b = 3 - 2a$
    - $b = 3 - 2a$ is an equation of line
  - For multiple points in the image
    - Find all possible equations of (a,b)
  - Choose parameters (a,b) for which highest number of lines intersect
  - This representation fails in case of vertical lines
    - a cannot be ∞



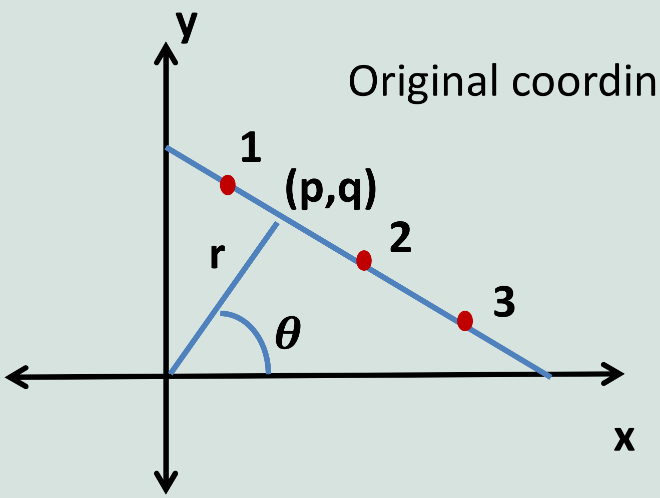
Hough Transform for Straight Line Detection

- Vertical lines are where $\theta = 0$
- If r can have negative values $-90 < \theta \leq 90$
- $(\mathrm{p}, \mathrm{q}) = (\mathrm{r}.\cos\theta, \mathrm{r}.sin\theta), tan\theta = \frac{sin\theta}{cos\theta}$
- For any point (x,y) on line gradient of line is found using $l_1 \perp l_2 \rightarrow m_1.m_2 = -1$

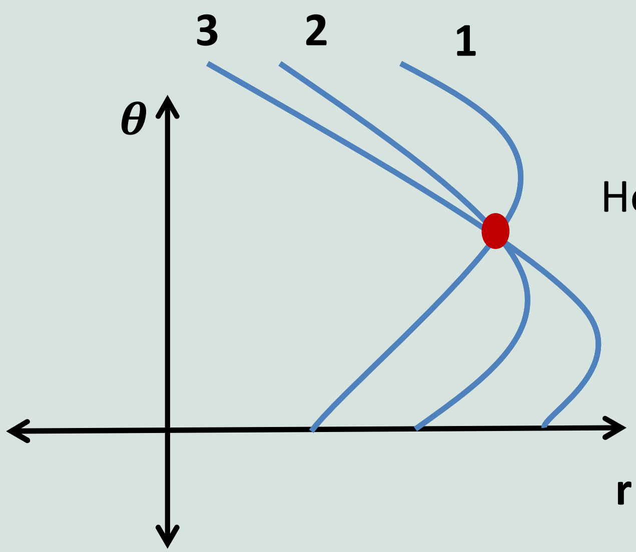
$$\frac{y - r.sin\theta}{x - r.cos\theta} = \frac{-cos\theta}{sin\theta}$$

- A more useful representation in this case is

$$x.\cos(\theta) + y.\sin(\theta) = r$$



Original coordinate plane



Hough plane

Hough Transform for Straight Lines

- Advantages of parameterization
  - Values of r and θ become bounded
- How to find intersection of the parametric curves
  - Use of accumulator arrays – concept of "voting"
  - To reduce the computational load use gradient information

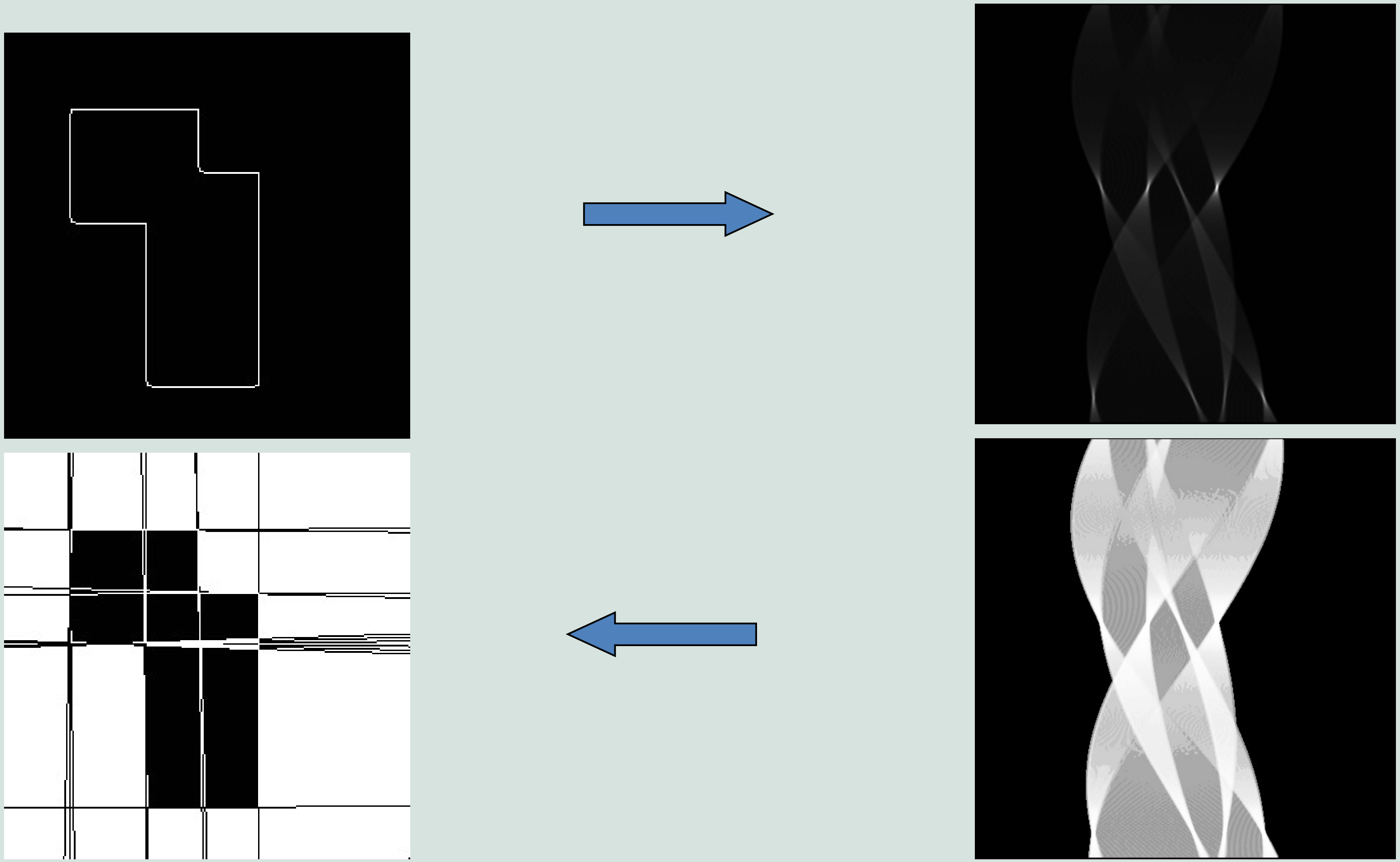
Computational Load

- Image size = 512 X 512
- Maximum value of $r = 512 * 2\sqrt{2}$
- With a resolution of 1°, maximum value of $\theta = 360°$
- Accumulator size = $512 * 2\sqrt{2} * 360$
- Use of direction of gradient reduces the computational load by 1/360

Hough Transform for Straight Lines Algorithm

- Quantize the Hough Transform space:
  - identify the maximum and minimum values of r and θ
- Generate an accumulator array $A(r, \theta)$
  - set all values of $A(r, \theta)$ to zero
- For all edge points (x_i, y_i) in the image
  - Use gradient direction for θ
  - Compute r from the equation $x.\cos(\theta) + y.\sin(\theta) = r$
  - Increment $A(r, \theta)$ by one
- For all cells in $A(r, \theta)$
  - Search for the maximum value of $A(r, \theta)$
  - Calculate the equation of the line
- To reduce the effect of noise more-than-one-element indices in a neighborhood in the accumulator array are increased

Line Detection by Hough Transform



Hough Transform for Detection of Circles

- The parametric equation of the circle can be written as

$$(x-a)^2 + (y-b)^2 = r^2$$

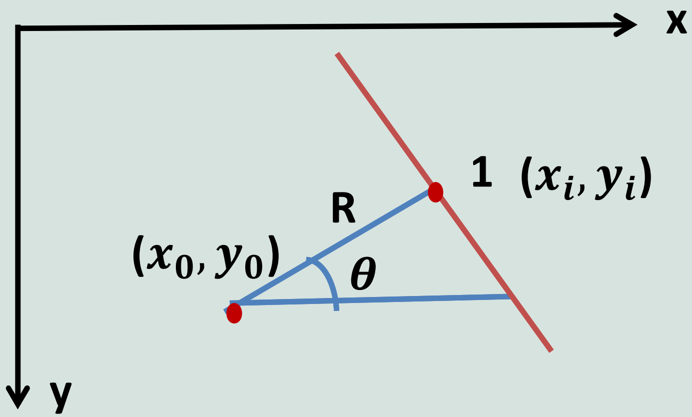
- The equation has three parameters: a, b, r
- The curve obtained in the Hough Transform space for each edge point will be a right circular cone
- Point of intersection of the cones gives the parameters a, b, r

Hough Transform for Circles

- Gradient at each edge point is known
- Then the line on which the center lies

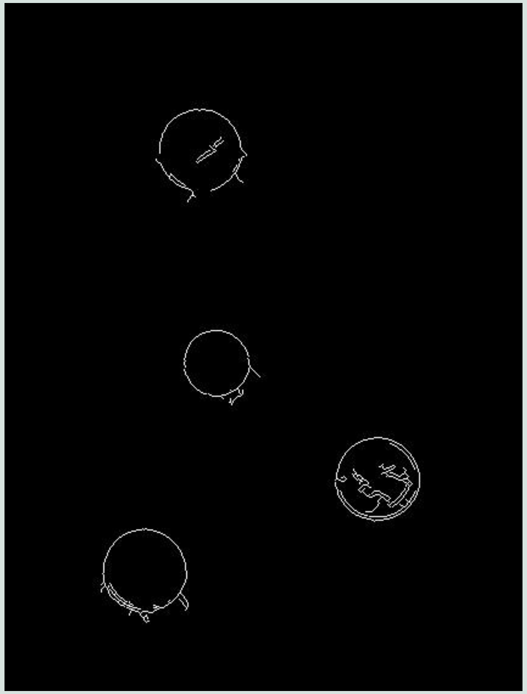
$$x_0 = x_i - R\cos\theta$$
$$y_0 = y_i - R\sin\theta$$

- If the radius is also known then center of the circle can be located
- Therefore accumulator array can store $(x, y) \times R$ values



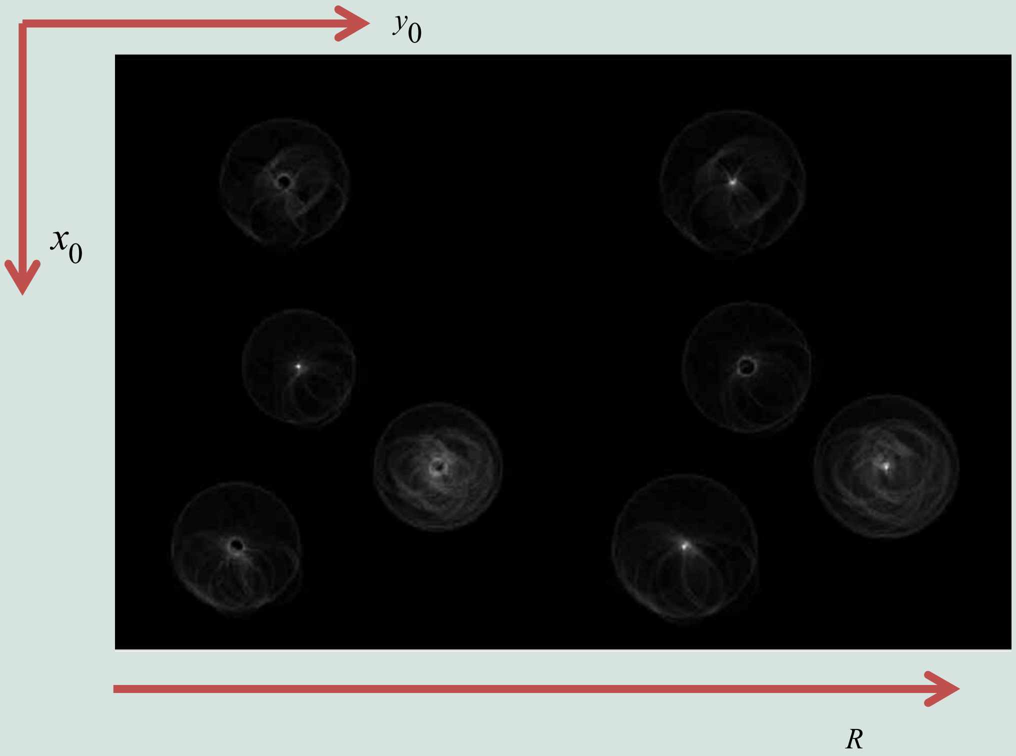
Detection Of Circle By Hough Transform: Example

Original Image

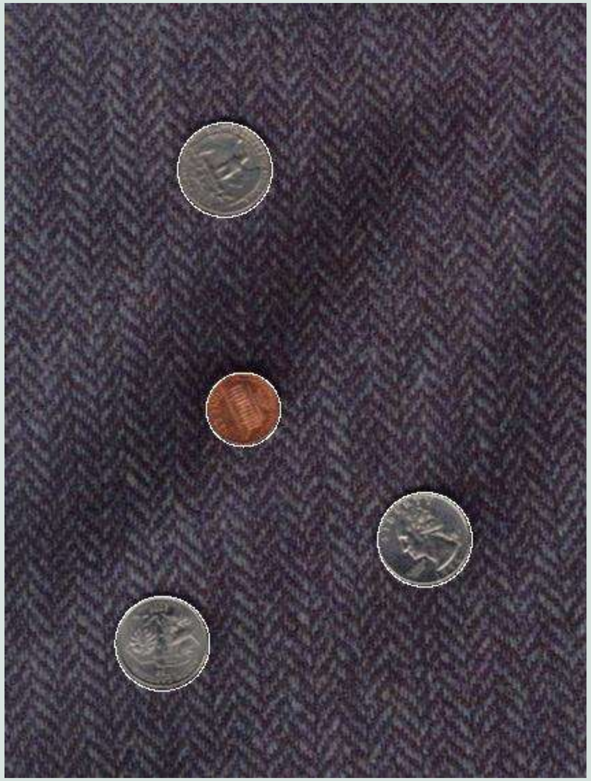


Circles detected by Canny Edge Detector

Detection Of Circle By Hough Transform Cont’d



Hough Transform of the edge detected image



Detected Circles

Recap

- In detecting lines
  - The parameters r and θ are found out relative to the origin (0,0)
- In detecting circles
  - The radius and center are found out
- In both the cases the shape is known
  - Line, circle etc.
  - Aim to find out its location and orientation in the image
- The idea can be extended to shapes like ellipses, parabolas, etc.

Parameters for Analytic Curves

Analytic Form	Parameters	Equation
Line	ρ, θ	$x\cos\theta+y\sin\theta=\rho$
Circle	x_0, y_0, ρ	$(x-x_o)^2+(y-y_0)^2=\rho^2$
Parabola	x_0, y_0, ρ, θ	$(y-y_0)^2=4\rho(x-x_o)$
Ellipse	x_0, y_0, a, b, θ	$(x-x_o)^2/a^2+(y-y_0)^2/b^2=1$

Generalized Hough Transform

- The Generalized Hough transform can be used to detect arbitrary shapes
- Complete specification of the exact shape of the target object is required in the form of the R-Table
- Information that can be extracted are
  - Location
  - Size
  - Orientation
  - Number of occurrences of that particular shape

Generating the R-Table

- Algorithm
  - Choose a reference point
  - Draw a vector from the reference point to an edge point on the boundary
  - Store the information of the vector against the gradient angle in the R-Table
  - There may be more than one entry in the R-Table corresponding to a gradient value

Generalized Hough Transform Algorithm

- Form an Accumulator array to hold the candidate locations of the reference point
- For each point on the edge
  - Compute the gradient direction and determine the row of the R-Table it corresponds to
  - For each entry on the row calculate the candidate location of the reference point

$$x_c = x_i + r\cos\theta$$

$$y_c = y_i + r\sin\theta$$

  - Increase the Accumulator value for that point
- The reference point location is given by the highest value in the accumulator array

Generalized Hough Transform Size and Orientation

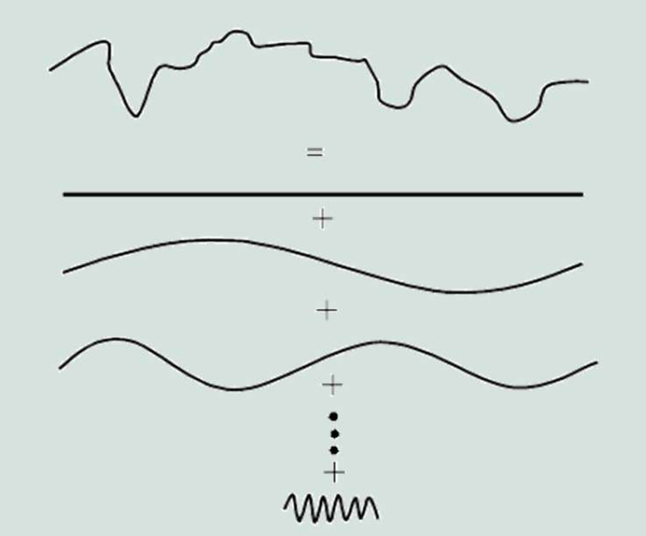
- The size and orientation of the shape can be found out by simply manipulating the R-Table
- For scaling by factor S multiply the R-Table vectors by S
- For rotation by angle θ, rotate the vectors in the R-Table by angle θ

Generalized Hough Transform Advantages and Disadvantages

- Advantages
  - A method for object recognition
  - Robust to partial deformation in shape
  - Tolerant to noise
  - Can detect multiple occurrences of a shape in the same pass
- Disadvantages
  - Lot of memory and computation is required

Fourier Transform

- Represents horizontal and vertical intensity variations in image using sinusoidal (sinus+cosinus) components



- Edge like structures: High frequency components
- Large homogenous regions: Low frequency components

Fourier Transform

- Continuous 2-D Fourier transform

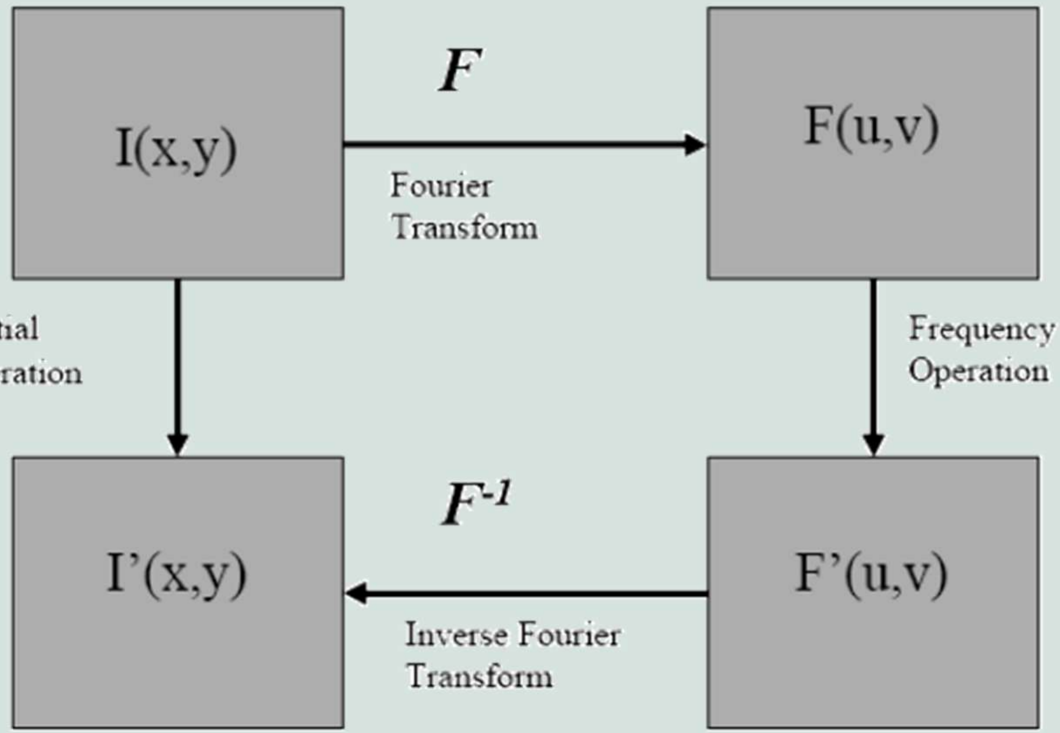
$$FT\big(I(x,y)\big) = F(u,v) = \iint_{-\infty}^{\infty} I(x,y) * e^{[-j2\pi(ux+vy)]} dx.dy$$

$$e^{[-j2\pi(ux+vy)]} = \cos 2\pi(ux+vy) - \mathrm{j}sin2\pi(ux+vy)$$

- Euler's formula: $e^{j\theta} = \cos\theta + \mathrm{j}sin\theta$
- Discrete 2-D Fourier transform

$$F(u,v) = \frac{1}{N^2}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} I(x,y)\, e^{[-j2\pi(ux+vy)]/N}$$

Forward and Inverse Fourier



Fourier Coefficients

- Fourier coefficients are complex numbers

$$H(u,v) = R(u,v) + jI(u,v)$$

- Amplitude (magnitude) and phase

$$M(u,v) = \sqrt{R(u,v)^2 + I(u,v)^2}$$

$$\varphi = \tan^{-1}\left(\frac{I(u,v)}{R(u,v)}\right)$$

Matlab Functions

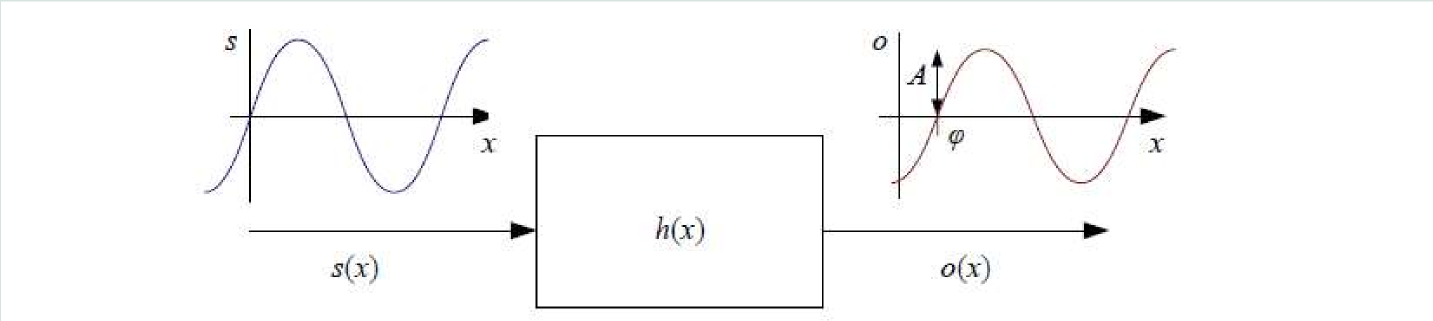
- fft
  - DFT of a vector
- ifft
  - inverse DFT of a vector
- fft2
  - DFT of a matrix
- ifft2
  - inverse DFT of a matrix
- fftshift
  - shifts DC component of frequency in center
- imagesc, imshow
  - for display purposes

Python Functions

- cv2.fft
  - DFT of a vector
- np.fft.ifft
  - inverse DFT of a vector
- cv2.fft2, np.fft.fft2
  - DFT of a matrix
- cv2.ifft2, np.fft.ifft2
  - inverse DFT of a matrix
- cv2.fftshift, np.fft.fftshift
  - shifts DC component of frequency in center
- cv2.magnitude
- plt.imshow
  - for display purposes

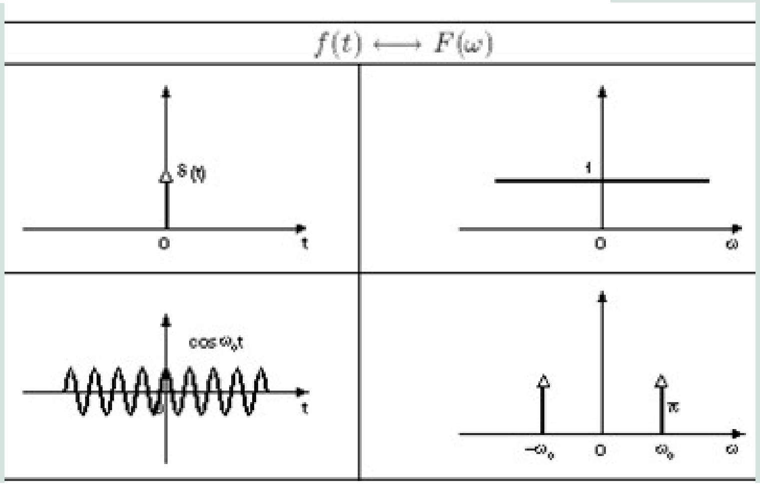
Background: Fourier Analysis

- Fourier transform as the response of a filter h(x)
  - to an input sinusoid $s(x) = e^{j\omega x}$
  - yielding an output sinusoid $o(x) = h(x) * s(x) = Ae^{j\omega x + \varphi}$



- Note symmetry in magnitude

$F(\omega) = F(-\omega)$

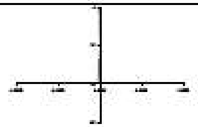
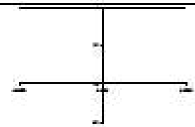
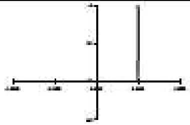
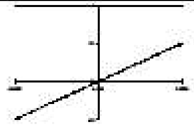
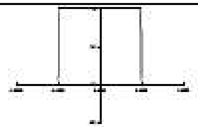
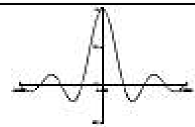
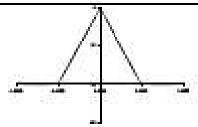
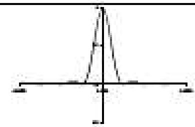
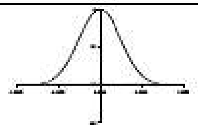
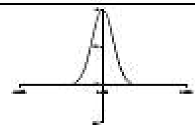
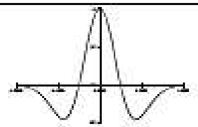
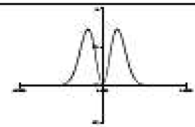
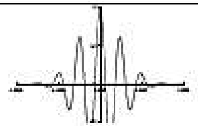
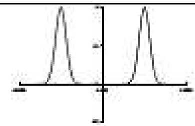


Background: Fourier Analysis

- Some useful properties of Fourier transform
  - The original transform pair is $F(\omega) = \mathcal{F}\{f(x)\}$

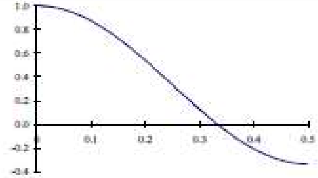
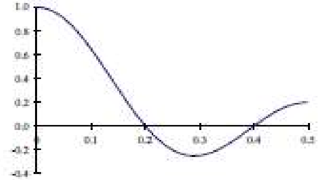
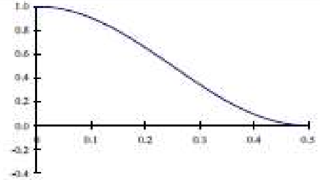
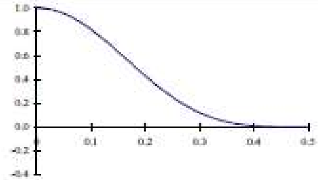
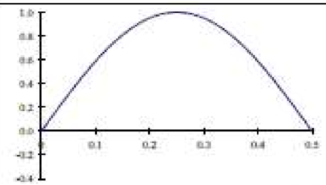
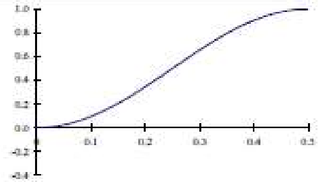
Property	Signal	Transform
superposition	$f_1(x) + f_2(x)$	$F_1(\omega) + F_2(\omega)$
shift	$f(x - x_0)$	$F(\omega)e^{-j\omega x_0}$
reversal	$f(-x)$	$F^*(\omega)$
convolution	$f(x) * h(x)$	$F(\omega)H(\omega)$
correlation	$f(x) \otimes h(x)$	$F(\omega)H^*(\omega)$
multiplication	$f(x)h(x)$	$F(\omega) * H(\omega)$
differentiation	$f'(x)$	$j\omega F(\omega)$
domain scaling	$f(ax)$	$1/aF(\omega/a)$
real images	$f(x) = f^*(x) \Leftrightarrow F(\omega) = F(-\omega)$	
Parseval's Thm.	$\sum_x [f(x)]^2 = \sum_\omega [F(\omega)]^2$	

Some Useful (Continuous) Fourier Transform Pairs

Name	Signal		Transform	
impulse		$\delta(x)$	1	
shifted impulse		$\delta(x-u)$	$e^{-j\omega u}$	
box filter		$\text{box}(x/a)$	$a\text{sinc}(a\omega)$	
tent		$\text{tent}(x/a)$	$a\text{sinc}^2(a\omega)$	
Gaussian		$G(x;\sigma)$	$\frac{\sqrt{2\pi}}{\sigma}G(\omega;\sigma^{-1})$	
Lapl. of Gauss.		$(\frac{x^2}{\sigma^4}-\frac{1}{\sigma^2})G(x;\sigma)$	$-\frac{\sqrt{2\pi}}{\sigma}\omega^2G(\omega;\sigma^{-1})$	
Gabor		$\cos(\omega_0 x)G(x;\sigma)$	$\frac{\sqrt{2\pi}}{\sigma}G(\omega\pm\omega_0;\sigma^{-1})$	

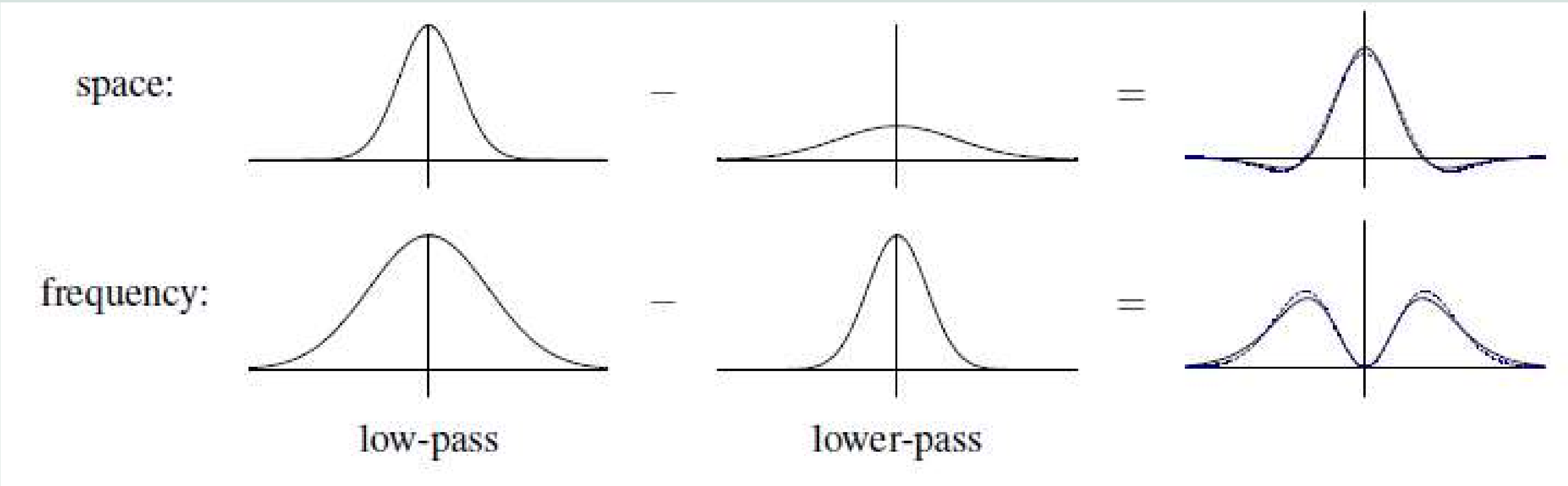
The dashed line in the Fourier transform of the shifted impulse indicates its (linear) phase. All other transforms have zero phase (they are real valued).

Fourier Transforms of Separable Kernels

Name	Kernel	Transform	Plot
box-3	$\frac{1}{3}\boxed{1 \mid 1 \mid 1}$	$\frac{1}{3}(1 + 2\cos\omega)$	
box-5	$\frac{1}{5}\boxed{1 \mid 1 \mid 1 \mid 1 \mid 1}$	$\frac{1}{5}(1 + 2\cos\omega + 2\cos 2\omega)$	
linear	$\frac{1}{4}\boxed{1 \mid 2 \mid 1}$	$\frac{1}{2}(1 + \cos\omega)$	
binomial	$\frac{1}{16}\boxed{1 \mid 4 \mid 6 \mid 4 \mid 1}$	$\frac{1}{4}(1 + \cos\omega)^2$	
Sobel	$\frac{1}{2}\boxed{-1 \mid 0 \mid 1}$	$\sin\omega$	
"Laplacian"	$\frac{1}{2}\boxed{-1 \mid 2 \mid -1}$	$\frac{1}{2}(1 - \cos\omega)$	

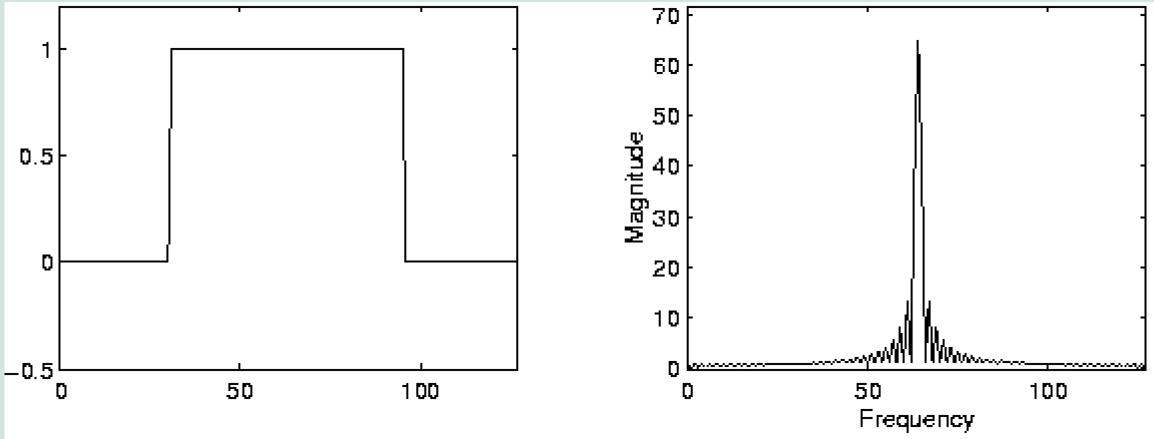
1D D.O.G.

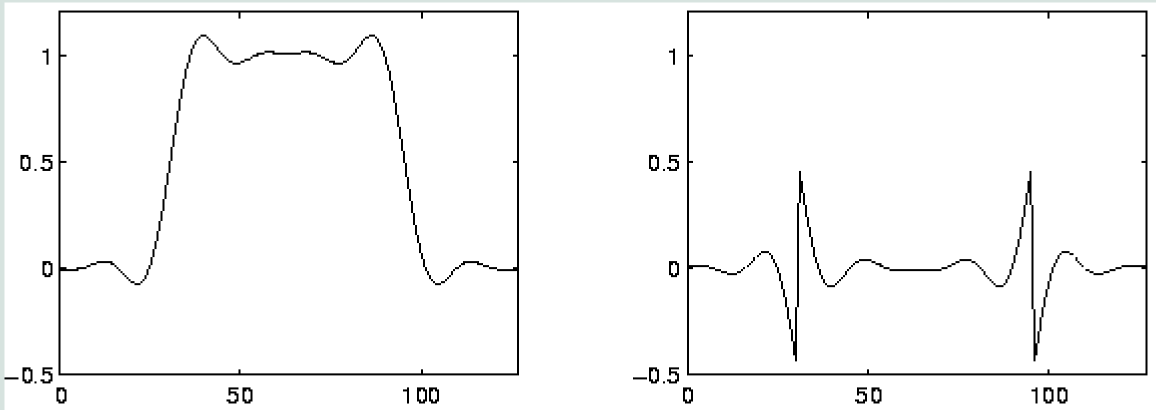
- The difference of two low-pass filters results in a band pass filter.
- The dashed lines show the close fit to a half octave Laplacian of Guaussian



High-Pass Low-Pass Filtering

Original signal   Fourier analysis of the signal

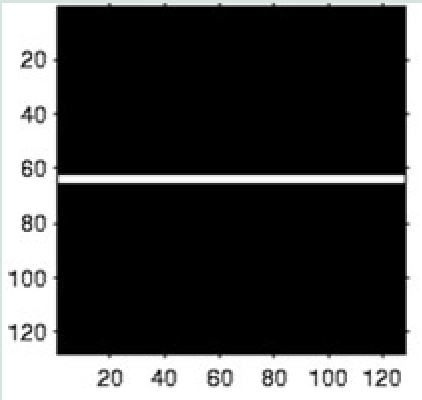




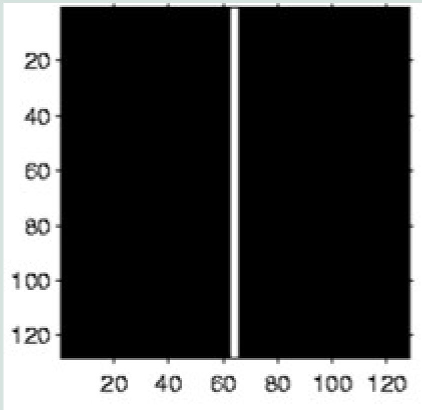
Low passed signal   High passed signal

2D FT Example

Original signal



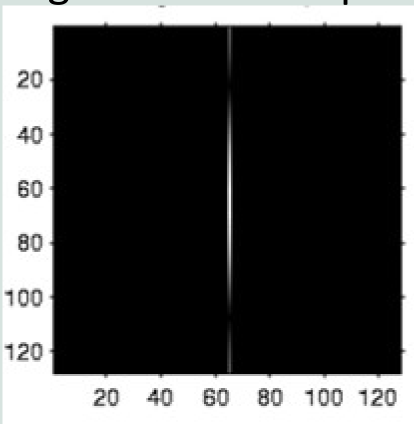
Original signal



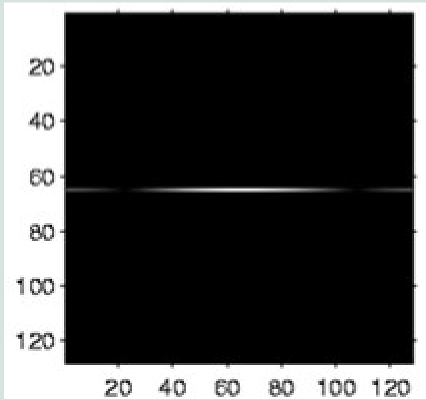
High vertical frequency

High vertical frequency

High horizontal frequency



High horizontal frequency



High horizontal frequency

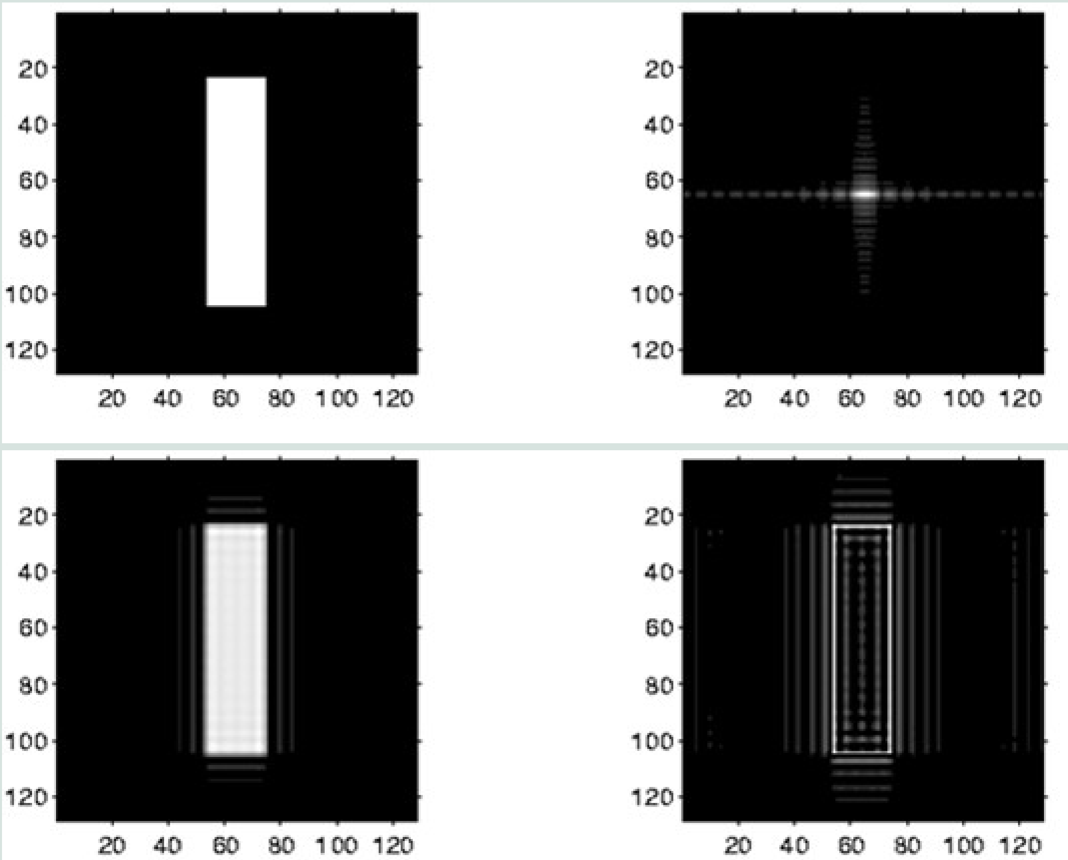
High vertical frequency

High vertical frequency

2D High-Pass Low-Pass Filtering

Original signal

Fourier analysis of the signal



Low passed signal

High passed signal

Magnitude versus Phase

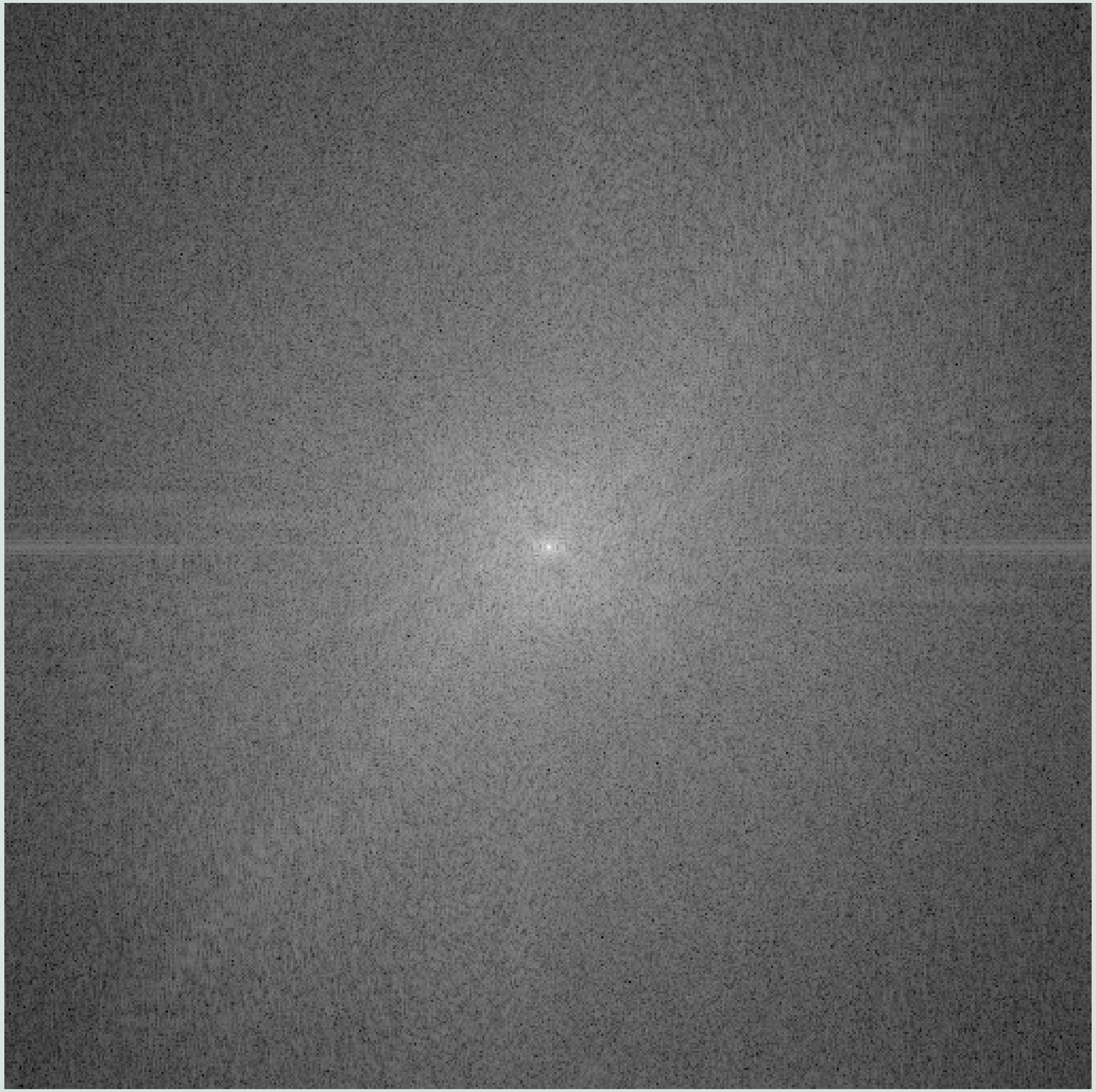
- Mostly considered magnitude spectra so far
- Sufficient for many vision methods:
  - high-pass/low-pass channel coding
  - simple edge detection, focus/defocus models
  - certain texture models
- May discard perceptually significant structure!

Phase and Magnitude

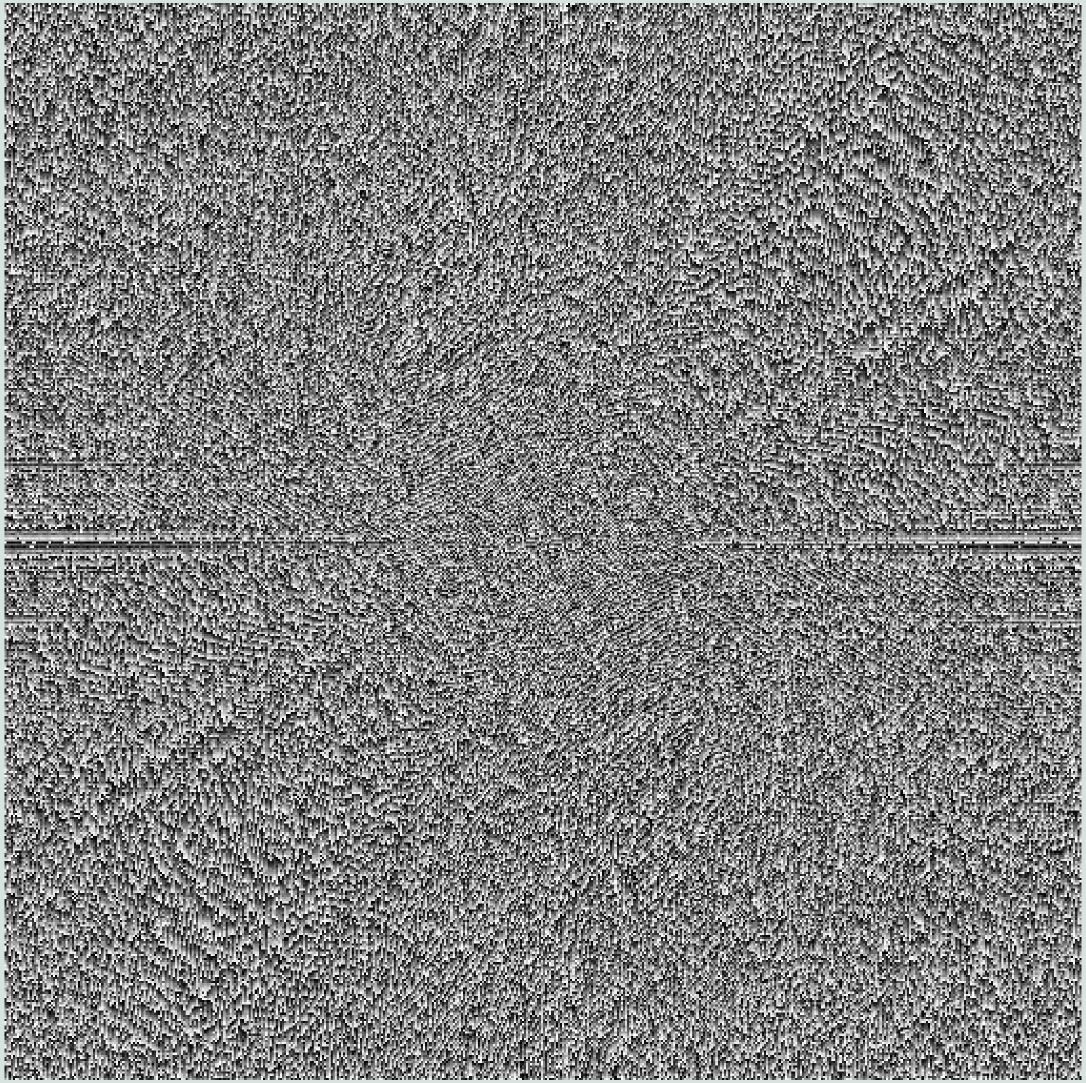
- Fourier transform of a real function is complex
  - difficult to plot, visualize
  - instead, we can think of the phase and magnitude of the transform
- Phase is the phase of the complex transform
- Magnitude is the magnitude of the complex transform
- Curious fact
  - all natural images have about the same magnitude transform
  - hence, phase seems to matter, but magnitude largely doesn’t
- Demonstration
  - Take two pictures, swap the phase transforms, compute the inverse - what does the result look like?

Cheetah and Zebra

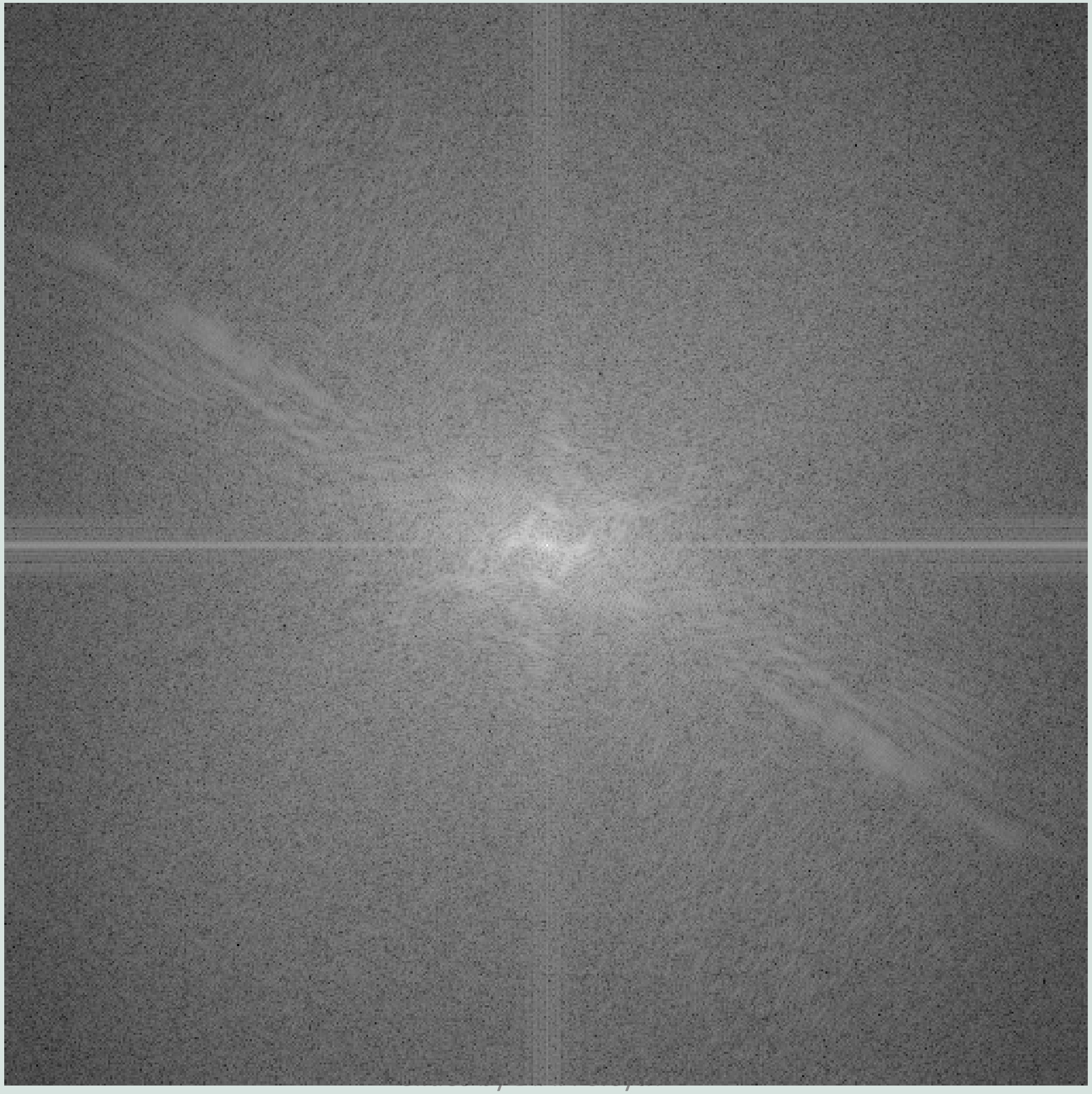
Magnitude of the Transform for Cheetah



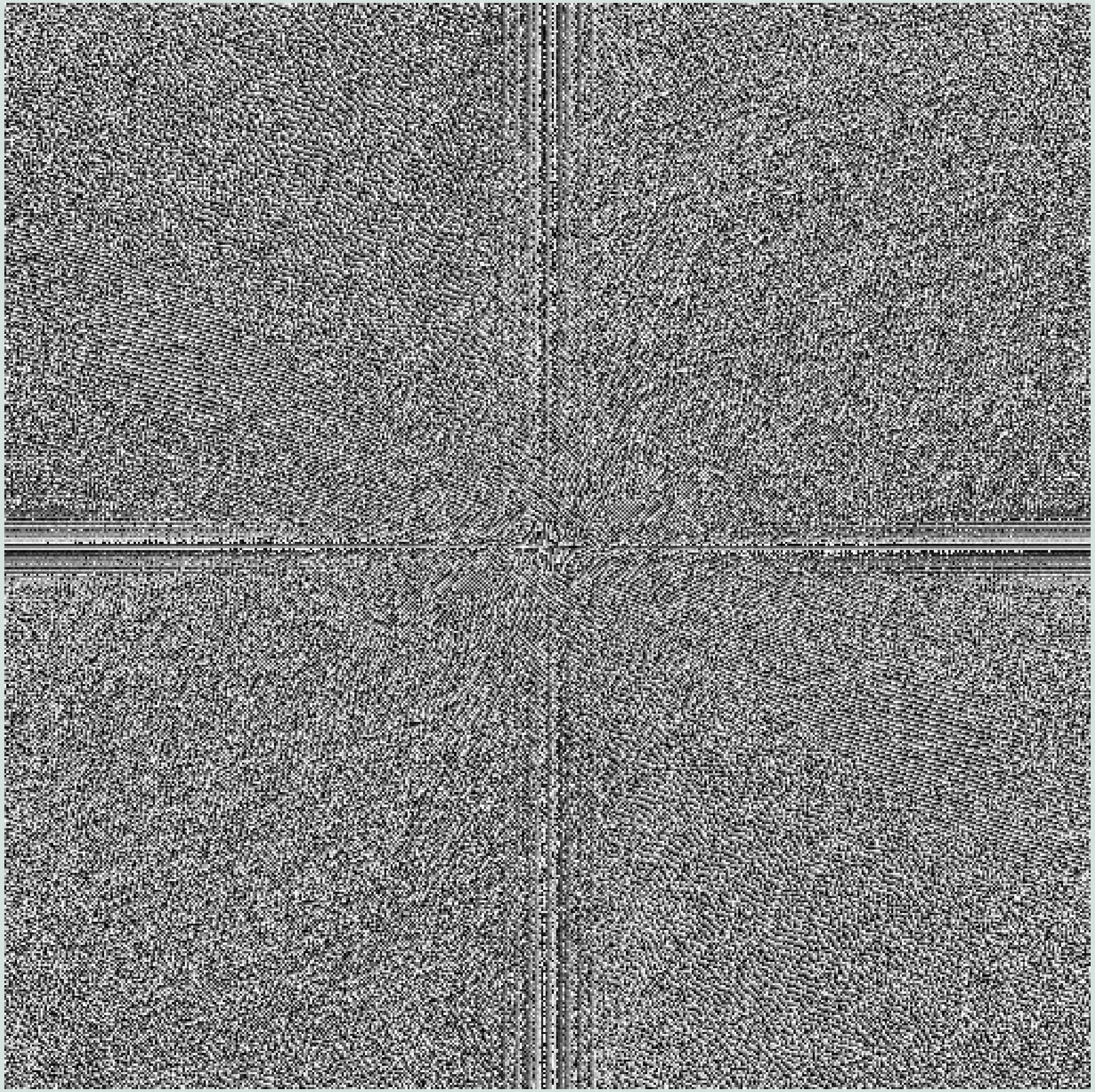
Phase of the Transform for Cheetah



Magnitude of the Transform for Zebra

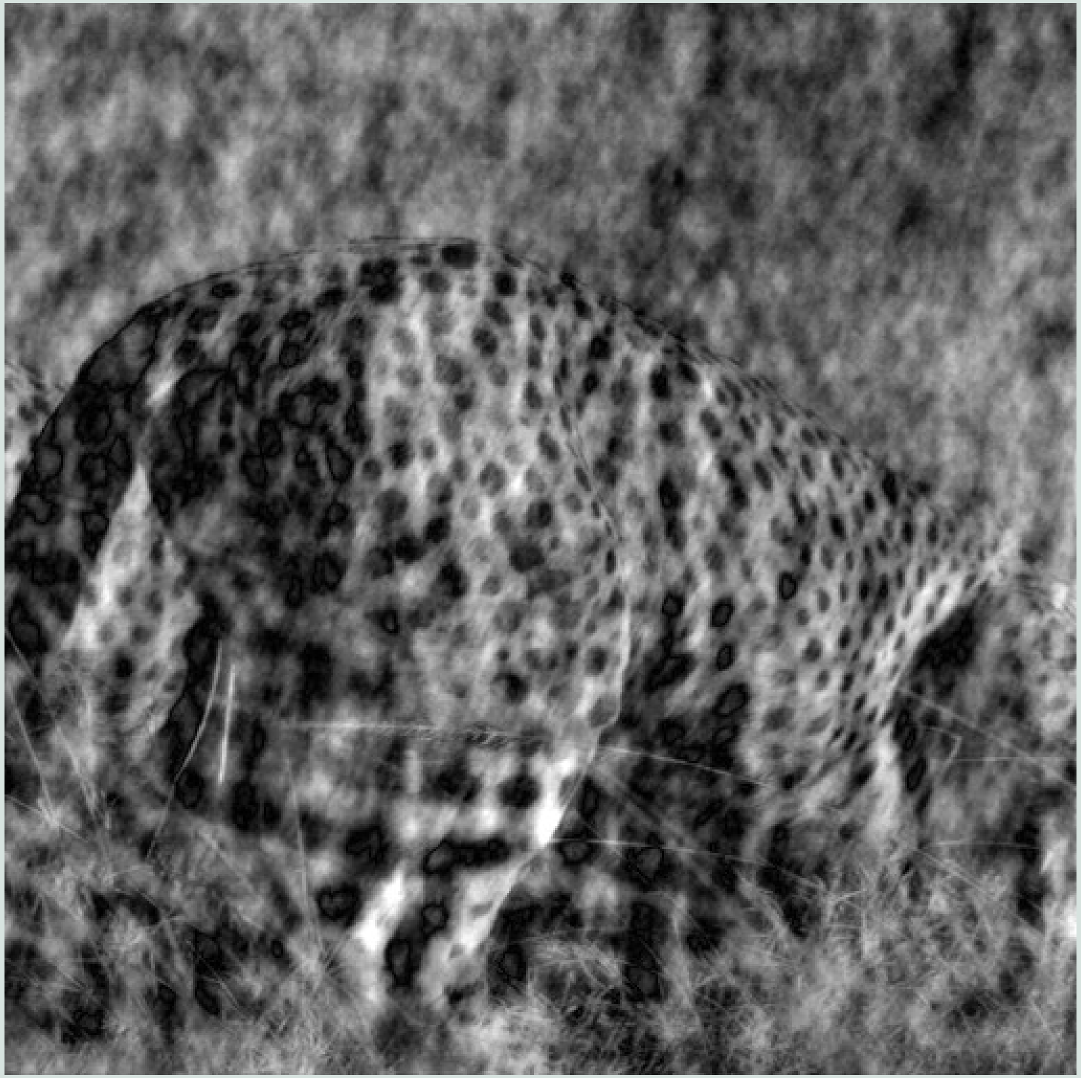


Phase of the Transform for Zebra



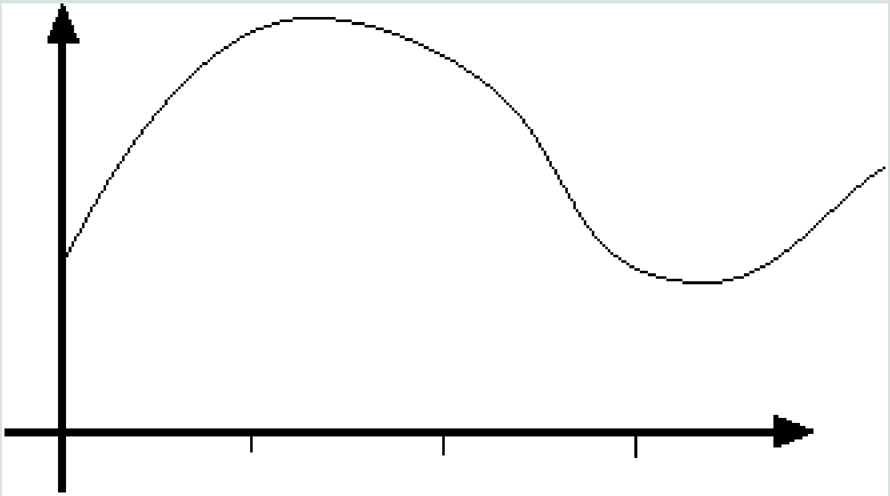
Reconstruction with Zebra Phase and Cheetah Magnitude

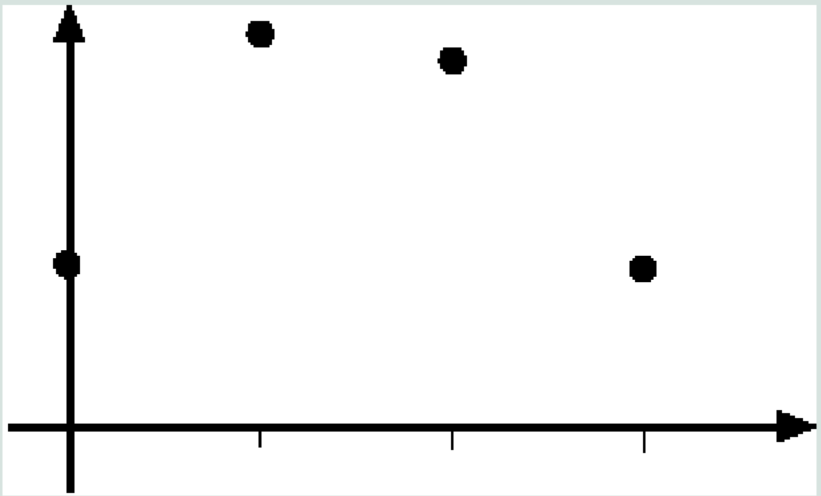
Reconstruction with Cheetah Phase and Zebra Magnitude



Sampling and Aliasing

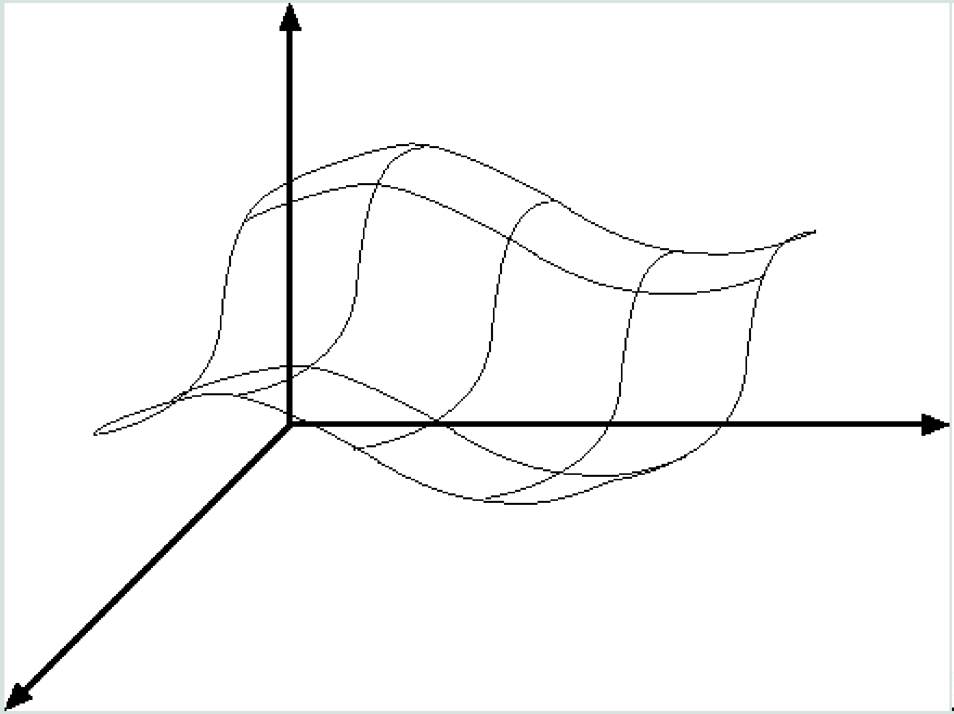
- Sampling in 1D takes a continuous function and replaces it with a vector of values, consisting of the function’s values at a set of sample points.
- These sample points are assumed to be on a regular grid, and can place one at each integer for convenience.

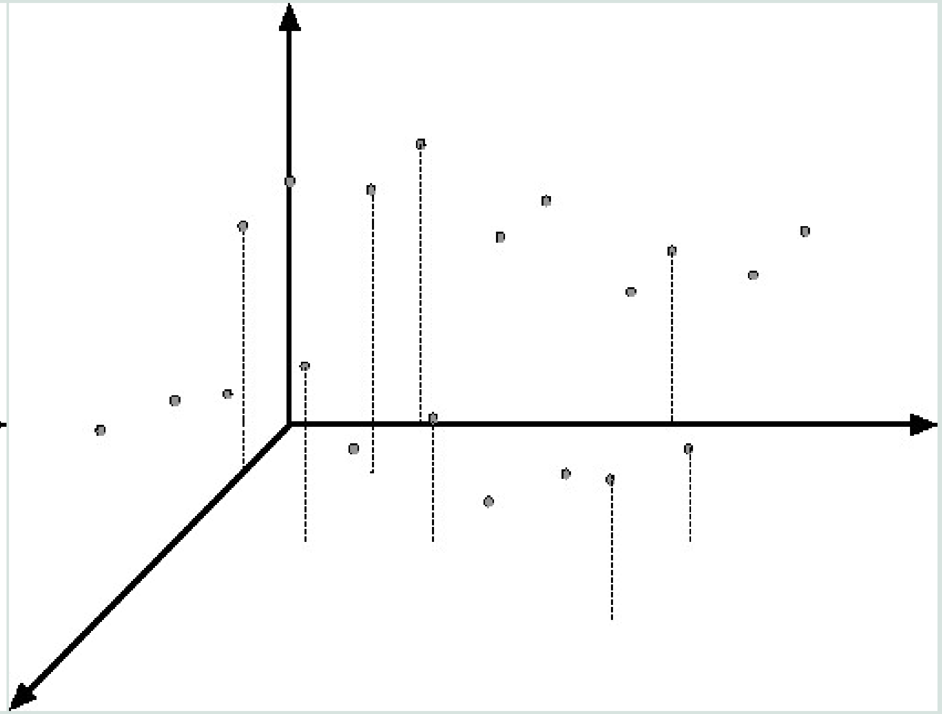




Sampling and Aliasing

- Sampling in 2D does the same thing, only in 2D.
- These sample points are assumed to be on a regular grid, and can place one at each integer point for convenience.



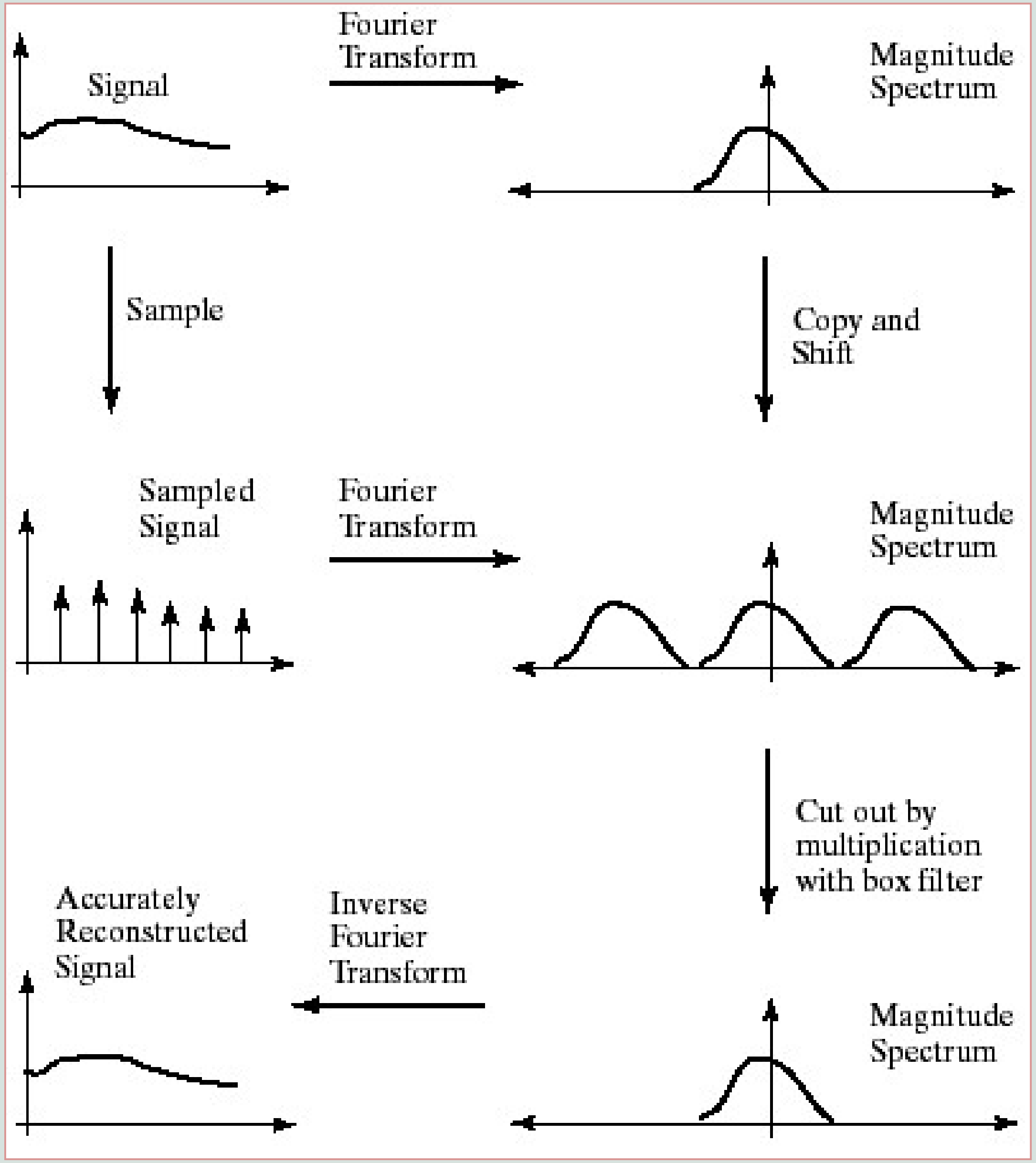


The Fourier Transform of a Sampled Signal

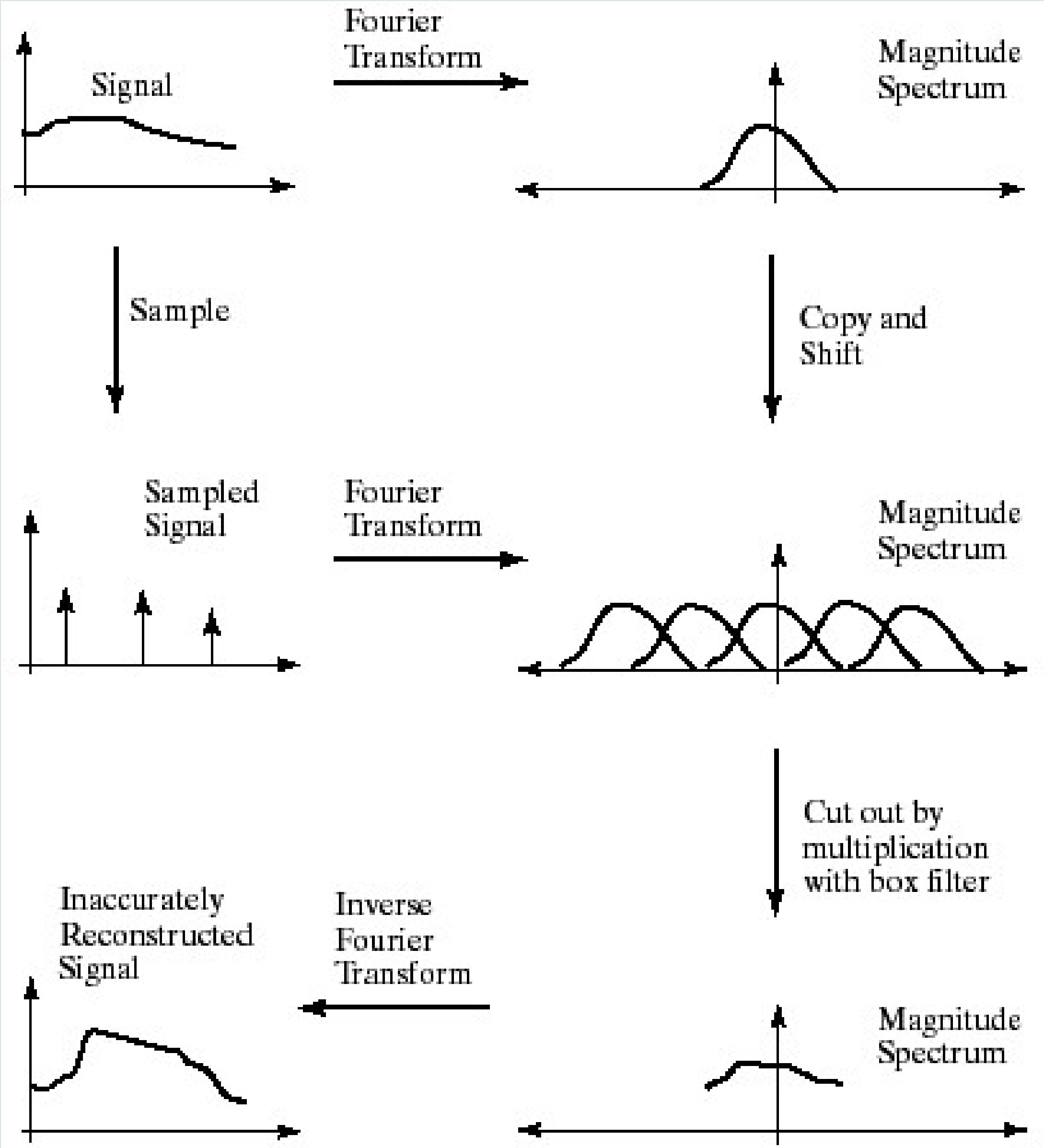
- Sampling in 2D does the same thing, only in 2D.
- These sample points are assumed to be on a regular grid, and can place one at each integer point for convenience.

$$
\begin{aligned}
F\left(\text{Sample}_{2D}(f(x,y))\right) &= F\left(f(x,y) \sum_{i=-\infty}^{\infty} \sum_{i=-\infty}^{\infty} \delta(x-i, y-j) \right) \\
&= F(f(x,y)) * F\left(\sum_{i=-\infty}^{\infty} \sum_{i=-\infty}^{\infty} \delta(x-i, y-j) \right) \\
&= \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} F(u-i, v-j)
\end{aligned}
$$

The Fourier Transform of a Sampled Signal



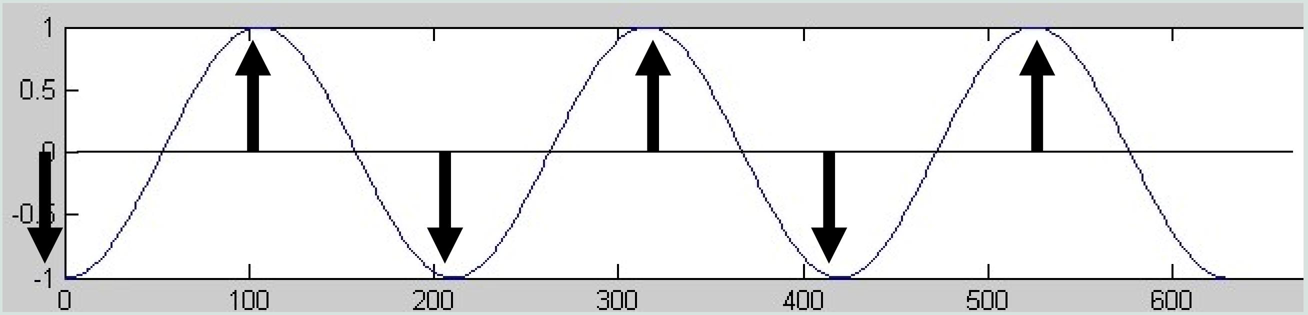
The Fourier Transform of a Sampled Signal



Space Domain Explanation of Nyquist Sampling

- You need to have at least two samples per sinusoid cycle to represent that sinusoid.

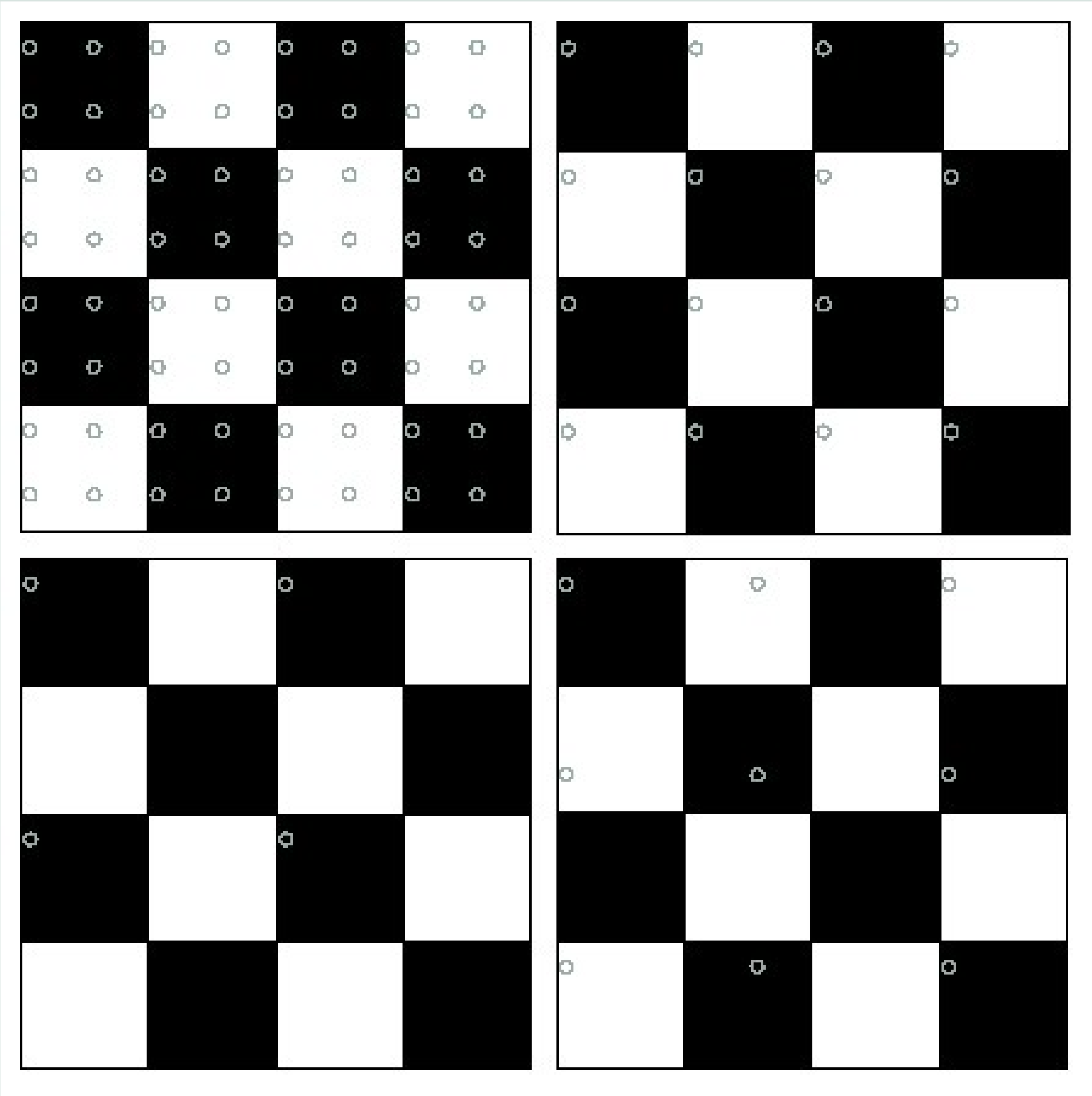
$$f_s \geq 2x f_{max}$$



Aliasing

- Can’t shrink an image by taking every second pixel
- If we do, characteristic errors appear
  - Typically, small phenomena look bigger; fast phenomena can look slower
  - Common phenomenon
    - Wagon wheels rolling the wrong way in movies
    - Checkerboards misrepresented in ray tracing

Aliasing



- Resample the checkerboard by taking one sample at each circle. In the case of the top left board, new representation is reasonable.
- Top right also yields a reasonable representation.
- Bottom left is all black (dubious) and bottom right has checks that are too big.

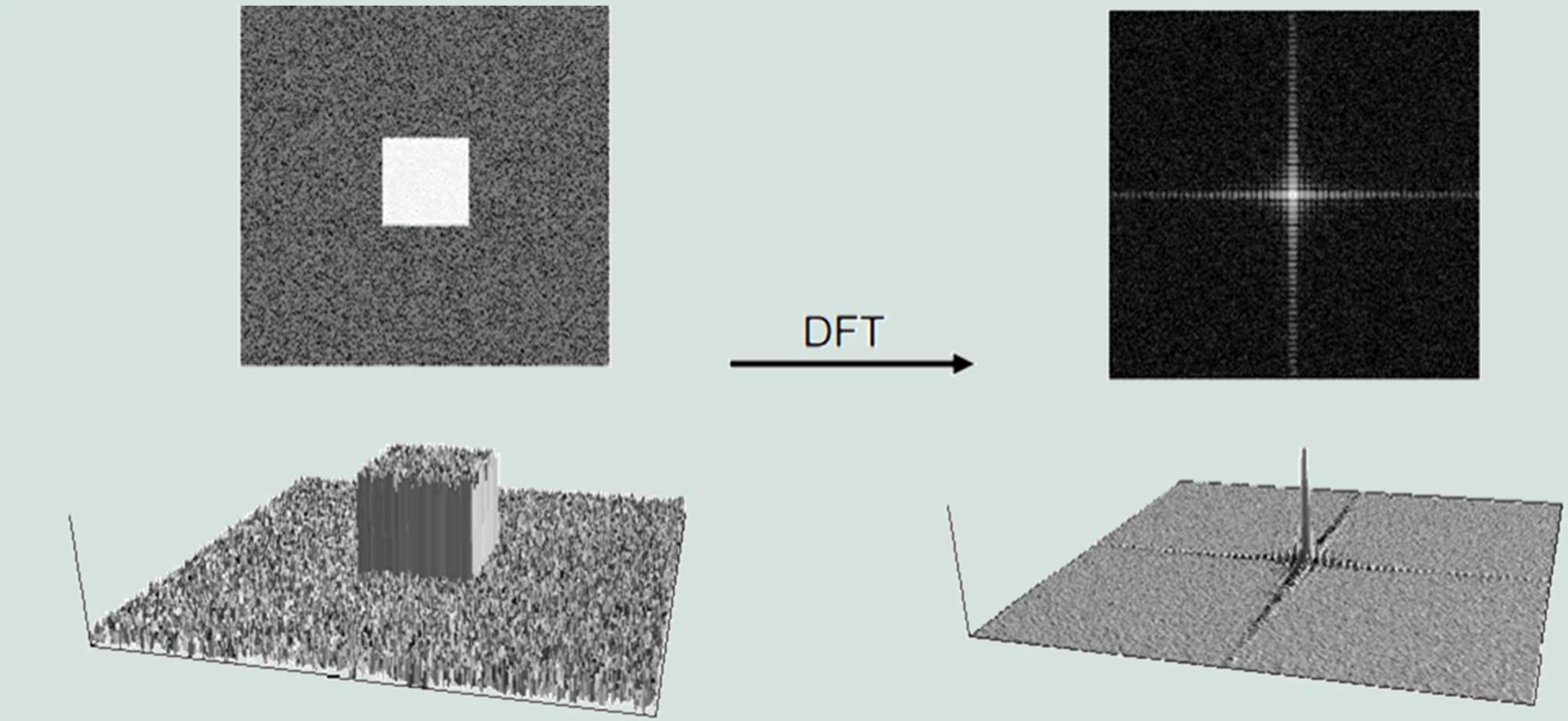
Smoothing as Low-Pass Filtering

- The message of the FT is that high frequencies lead to trouble with sampling.
- Solution: suppress high frequencies before sampling
  - multiply the FT of the signal with something that suppresses high frequencies
  - or convolve with a low-pass filter
- A filter whose FT is a box is bad, because the filter kernel has infinite support
- Common solution: use a Gaussian
  - multiplying FT by Gaussian is equivalent to convolving image with Gaussian.

Discrete Fourier Transform

image

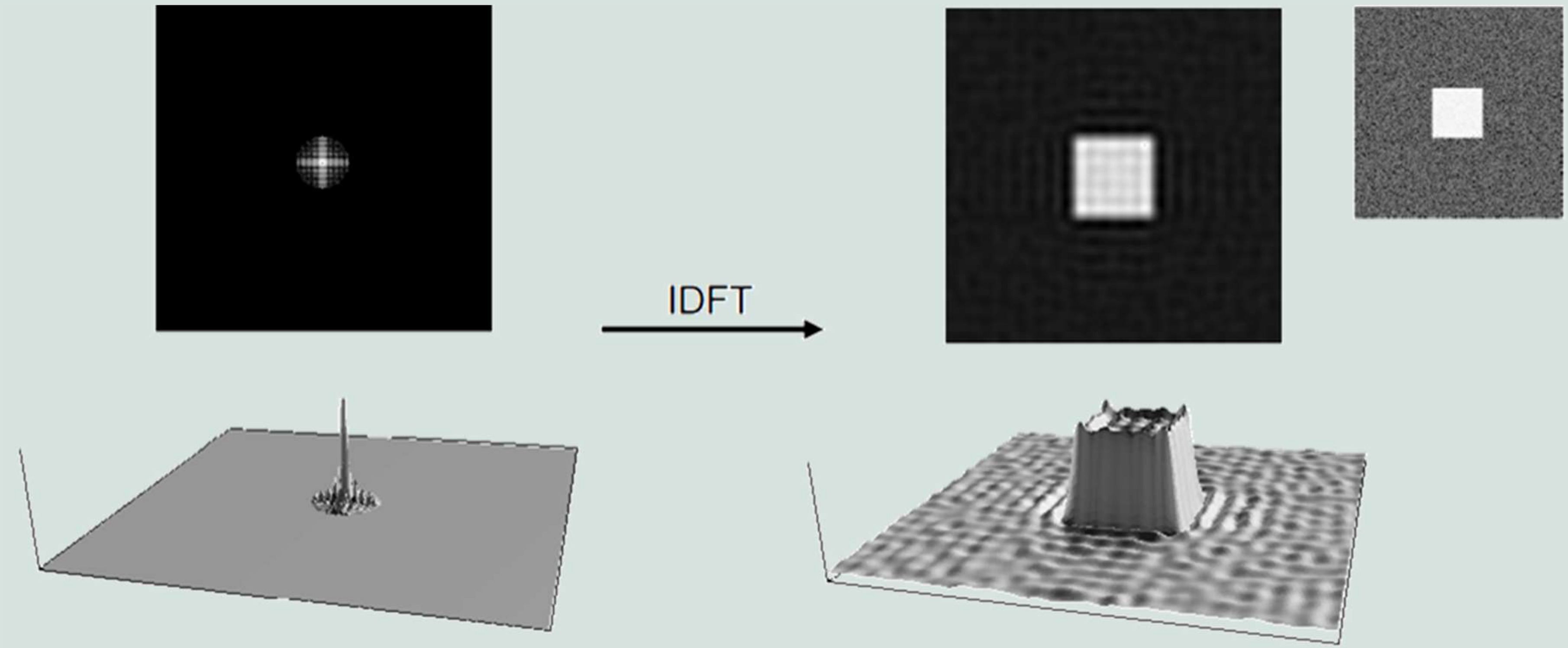
magnitude



Low-Pass Filtering

magnitude

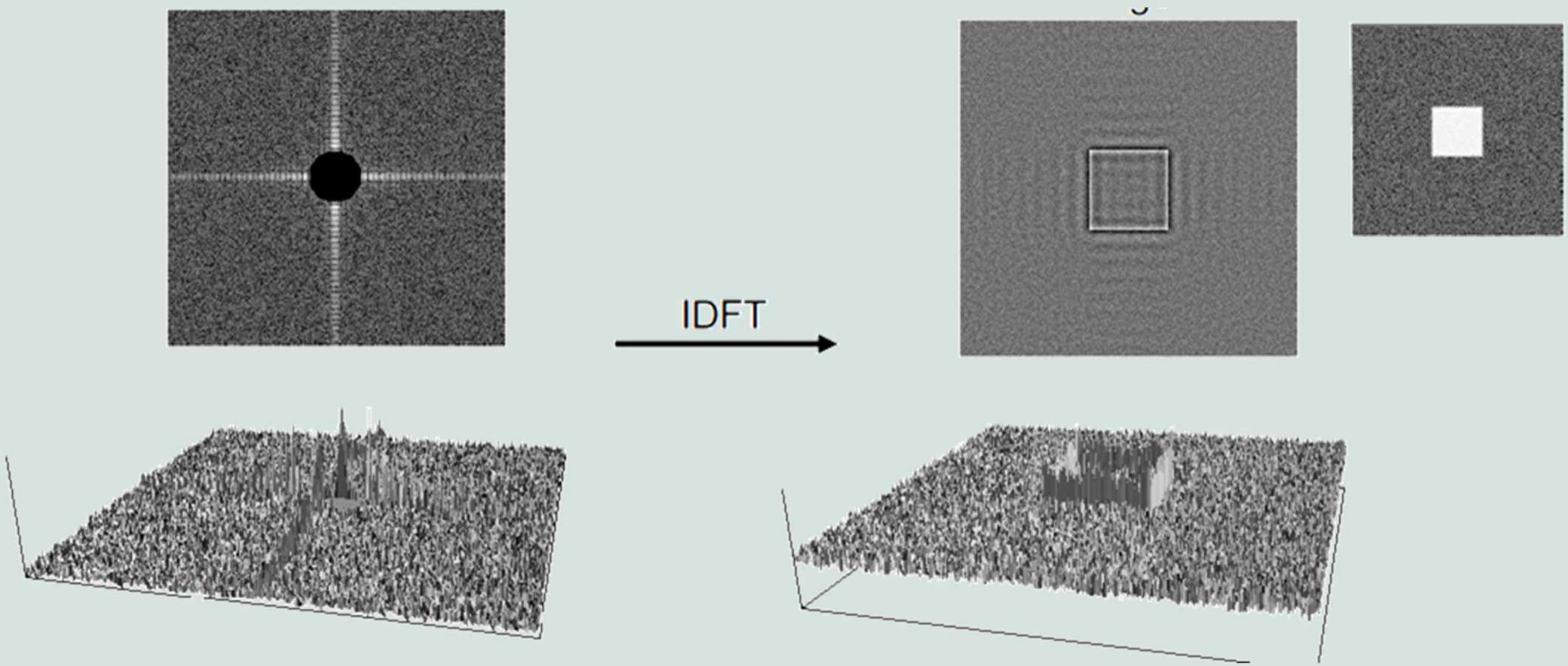
image



High-Pass Filtering

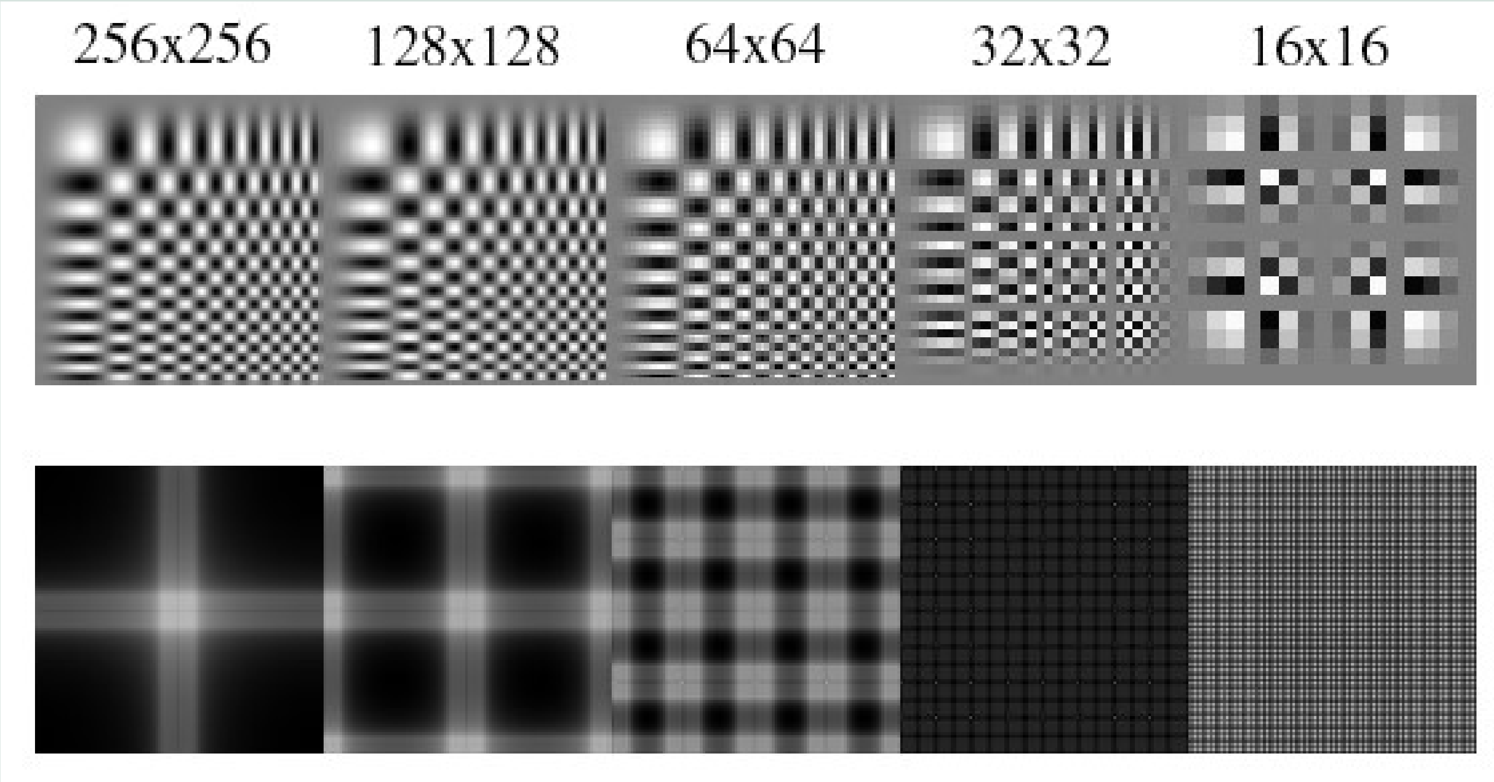
magnitude

image



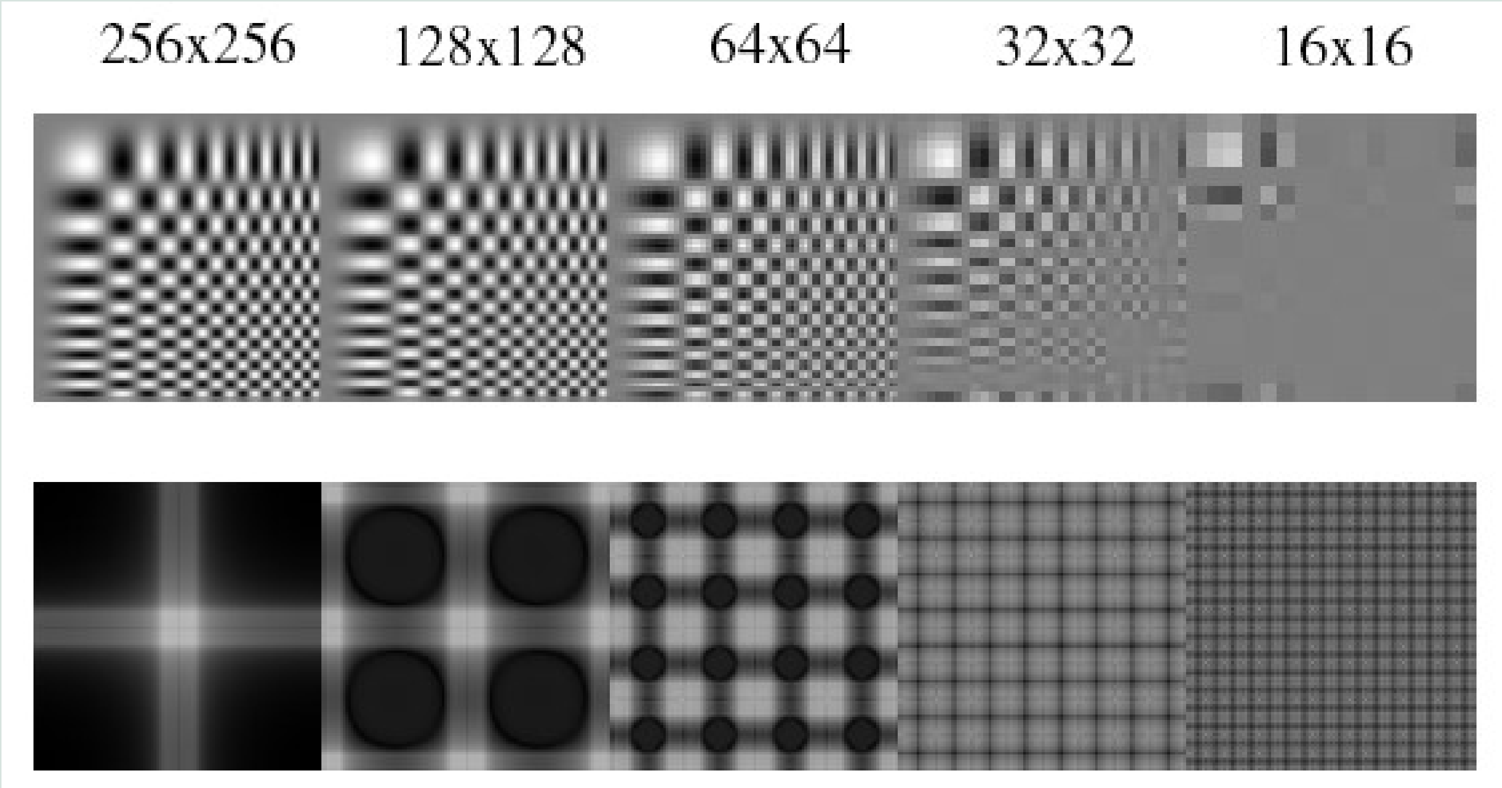
Sampling without Smoothing

- Top row shows the images, sampled at every second pixel to get the next.



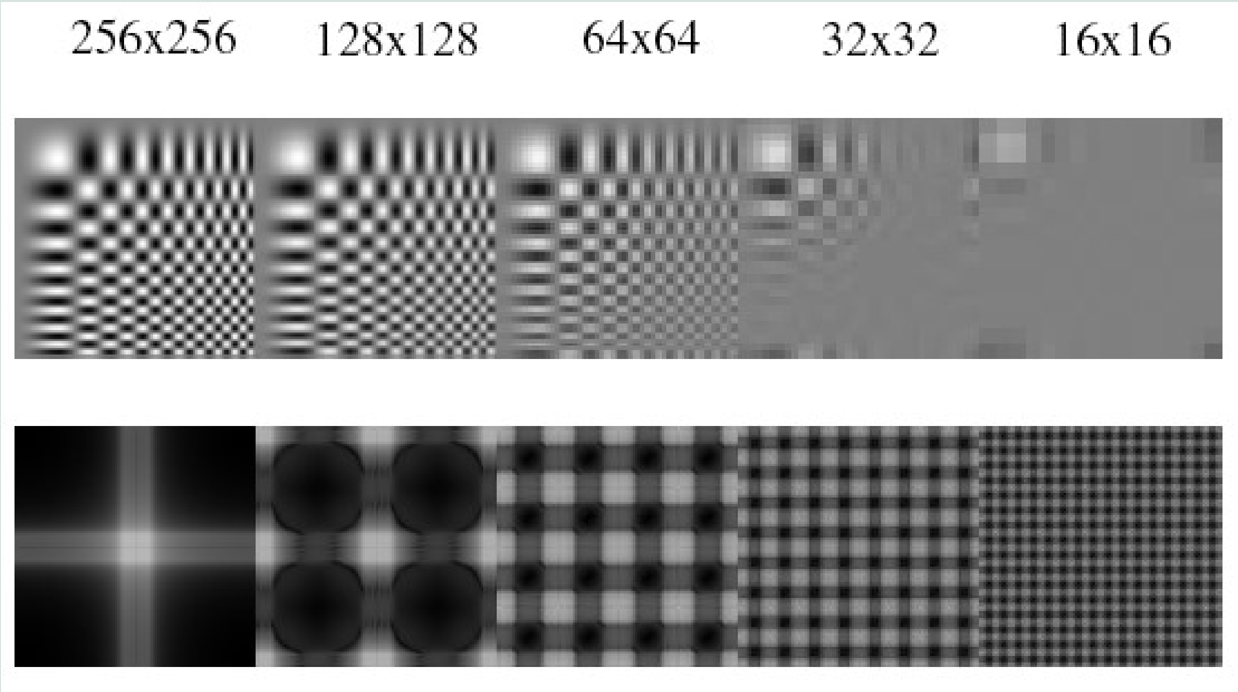
Sampling with Smoothing

- Top row shows the images. The next image is obtained by smoothing the image with a Gaussian with sigma = 1 pixel, then sampling at every second pixel to get the next.



Sampling with Smoothing

- Top row shows the images. The next image is obtained by smoothing the image with a Gaussian with sigma = 1.4 pixels, then sampling at every second pixel to get the next.
  - If the sigma is big, then the reconstruction error will be smaller
    - because the function is flatter in the relevant region of FT space
  - but aliasing will be larger, because it doesn't die off quickly enough.



Many More Transforms…

- Haar-transform
- Wavelet transform
- Distance transform

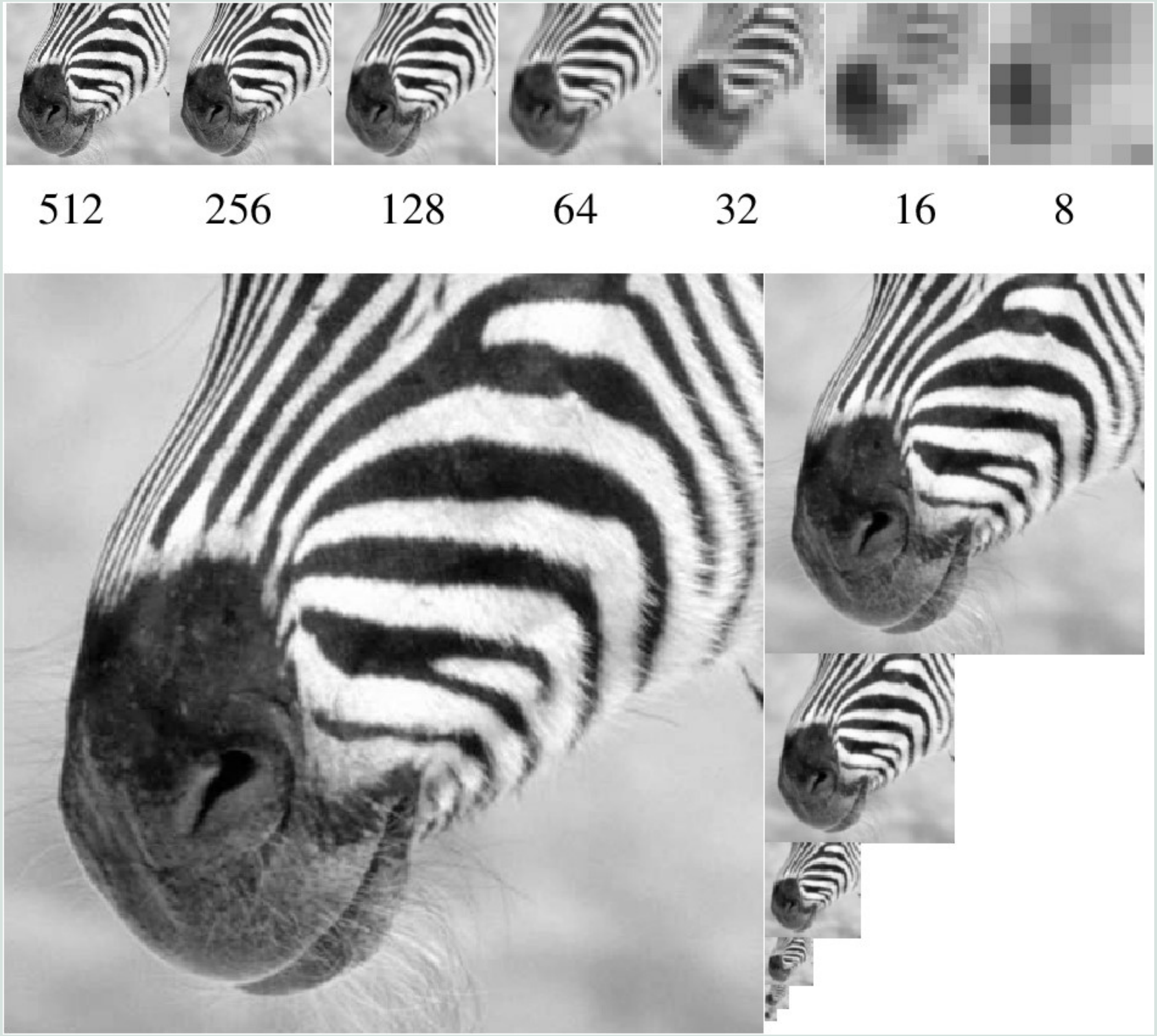
Image Pyramids

- Gaussian pyramid
- Laplacian pyramid
- Wavelet/QMF pyramid
- Steerable pyramid

The Gaussian Pyramid

- Smooth with Gaussians, because
  - a Gaussian*Gaussian=another Gaussian
- Gaussians are low pass filters, so representation is redundant
- Gaussian pyramids are used for up- or down- sampling images
- Multi-resolution image analysis
  - Look for an object over various spatial scales
  - Coarse-to-fine image processing:
    - form blur estimate or the motion analysis on very low-resolution image, upsample and repeat.
    - Often a successful strategy for avoiding local minima in complicated estimation tasks.

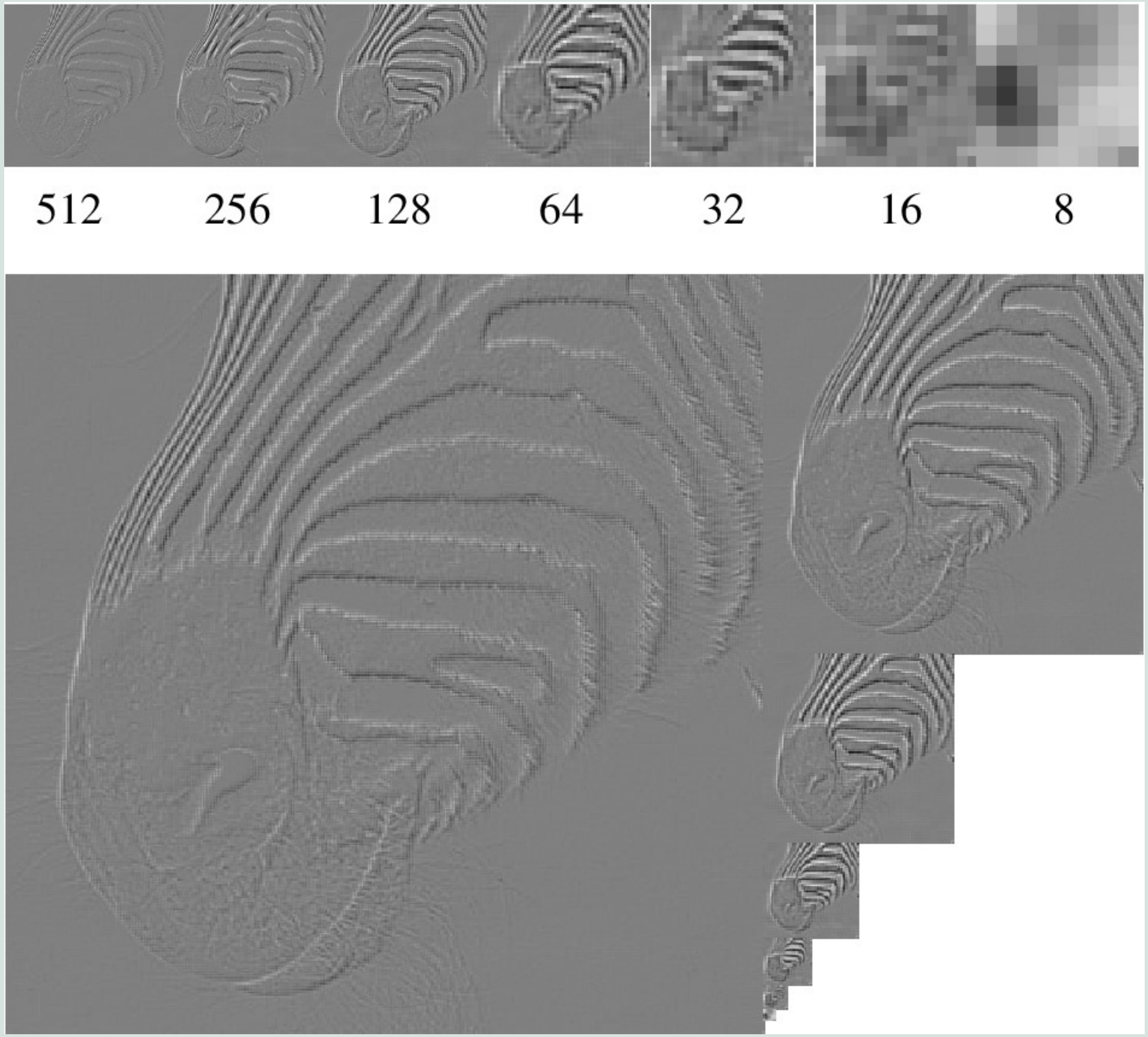
The Gaussian Pyramid



The Laplacian Pyramid

- Compute the difference between upsampled Gaussian pyramid level and Gaussian pyramid level.
- band pass filter - each level represents spatial frequencies (largely) unrepresented at other level.

The Laplacian Pyramid



Singular Value Decomposition

- Helps us to find solutions for linear systems
- With SVD any mxn matrix can be written as

$$A_{mxn} = U_{mxm}\, D_{mxn} {V^T}_{nxn}$$

- Columns of U and V are mutually orthogonal unit vectors
- Diagonal elements of D are singular values σ_i such that $\sigma_1 \geq \sigma_2 \geq \sigma_3 \ldots \geq \sigma_n \geq 0$
- Although U and V are not unique σ_i are fully determined by A

SVD Some Useful Properties

1. A is nonsingular if and only if all $\sigma_i > 0$
2. Number of non-zero σ_i equals the rank of A
3. Be A singular or not pseudoinverse of A is

$$A^+ = VD_0{}^{-1}U^T$$

$$\text{where } D_0{}^{-1} = D^{-1} \text{ for } \sigma_i \neq 0$$

$$D_0{}^{-1} = 0 \text{ for } \sigma_i = 0$$

4. Columns of U corresponding to non-zero σ_i span the range of A
5. Columns of V corresponding to zero σ_i is the null-space of A
6. Non-zero $\sigma_i{}^2$ are non-zero eigenvalues of $(A^TA)_{nxn}$ or $(AA^T)_{mxm}$
7. Columns of U are eigenvectors of AA^T
8. Columns of V are eigenvectors ofA^TA

Applications of SVD I

- Least-squares estimation:
  - To solve a system of nonhomogeneous linear equations of the form

$$Ax = b$$

  - Nonhomogeneous mean not all elements of b are 0
  - x is vector of unknowns
  - A is mxn coefficients matrix
  - b is $mx1$ data

$$A^T A x = A^T b \rightarrow x = (A^T A)^+ A^T \mathrm{b}$$

- SVD gives $(A^T A)^+$ from property 3
- When we have more equations than unknowns $(A^T A)^+$gets closer to $(A^T A)^{-1}$

Applications of SVD II

- Solving homogeneous systems
  - To solve a homogeneous system with m equations and unknowns

$$Ax = 0 \text{ with } \mathrm{m} \geq n - 1, rank(A) = n - 1$$

  - Disregarding the trivial solution $\mathrm{x} = 0$, a solution unique up to a scale factor is the eigenvector in V corresponding to the only zero eigenvalue of $A^T A$
  - Since $\mathrm{rank}(A) = n - 1$ all other eigenvalues are positive
    - Form properties of 5 and 8

Applications of SVD III

- Enforcing constraints
  - An estimate of A can be constructed to enforce some constraints on A such as independence or orthogonality as
$$\hat{A} = UD'V^T$$
  - D' is obtained by changing the singular values of D to those expected when the constraints are satisfied exactly.
  - Orthogonal matrices of fundamental matrix F is a case in point