

# **Basic Image Processing Algorithms**

---

PPKE-ITK

Lecture 4

# Texture analysis

---

# Textures - definition

---

- Textures demonstrate the difference between an artificial world of objects whose surfaces are only characterized by their color and reflectivity properties to that of real world imagery
- How we can define texture: microstructure
  - certainly not: an arbitrary pattern that extends over a large image
- Basic properties
  - Small elementary pattern which is repeated periodically or quasi-periodically in space (like pattern on a wall paper)
- It is sufficient to describe:
  - Small elementary pattern
  - Repetition rules (characteristic scales)
- Types
  - Artificial (Julesz, Pratt, Gagalowic)
  - Natural (Brodatz)

# Textures - definition

---

## ◎ Hawkins:

- Some local 'order' is repeated over a region which is large in comparison to the order's size,
- The order consists in the nonrandom arrangement of elementary parts
- The parts are roughly uniform entities having approximately the same dimensions everywhere within the textured region

## ◎ Description:

- *coarseness* ~ period of repetition
  - e.g. wool is "coarser" than silk, under the same conditions
- fineness / rudeness, contrast, orientation, arrangement ...

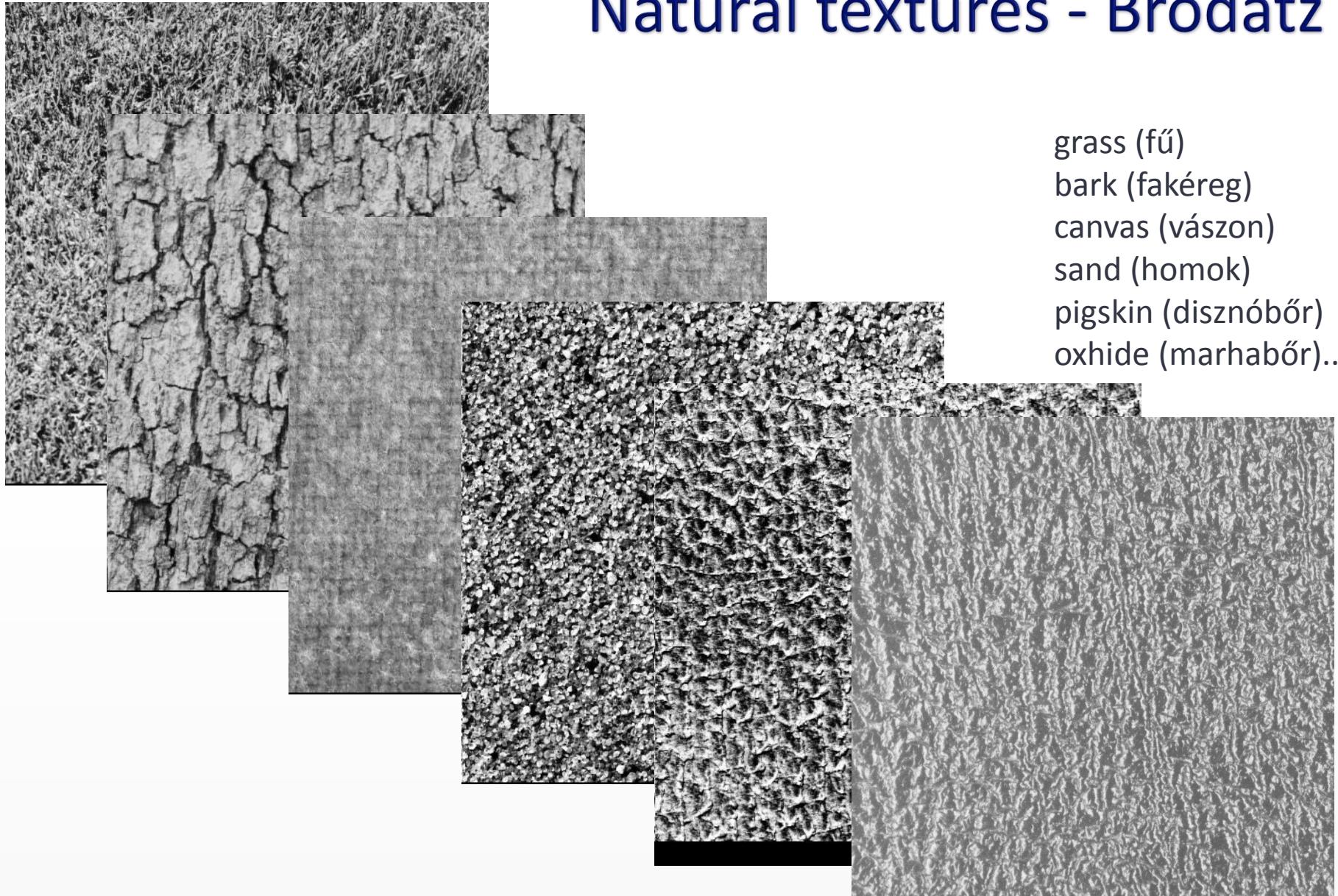
# Texture

---



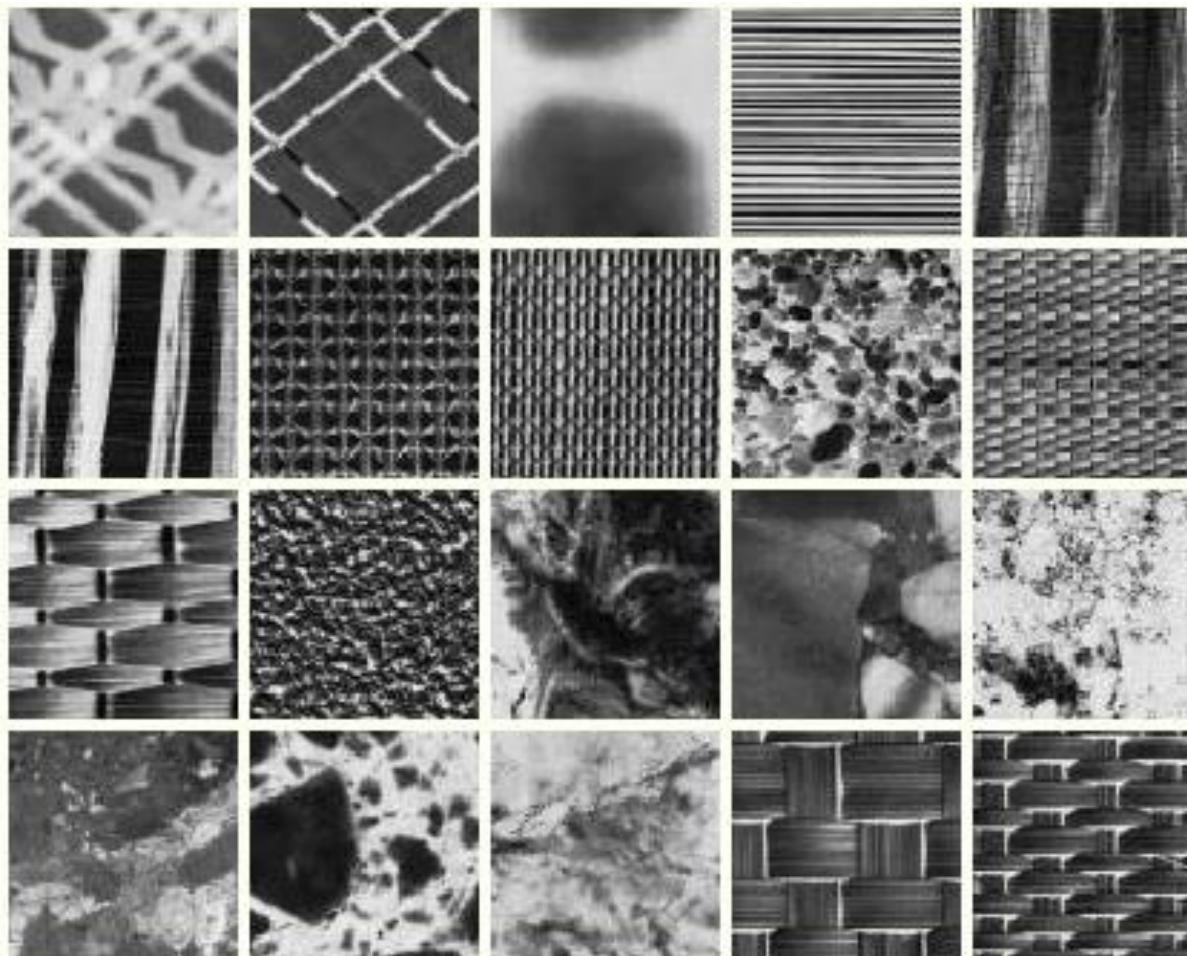
- A texture is an image that follows some statistical properties
- It has similar structures repeated over and over again

# Natural textures - Brodatz

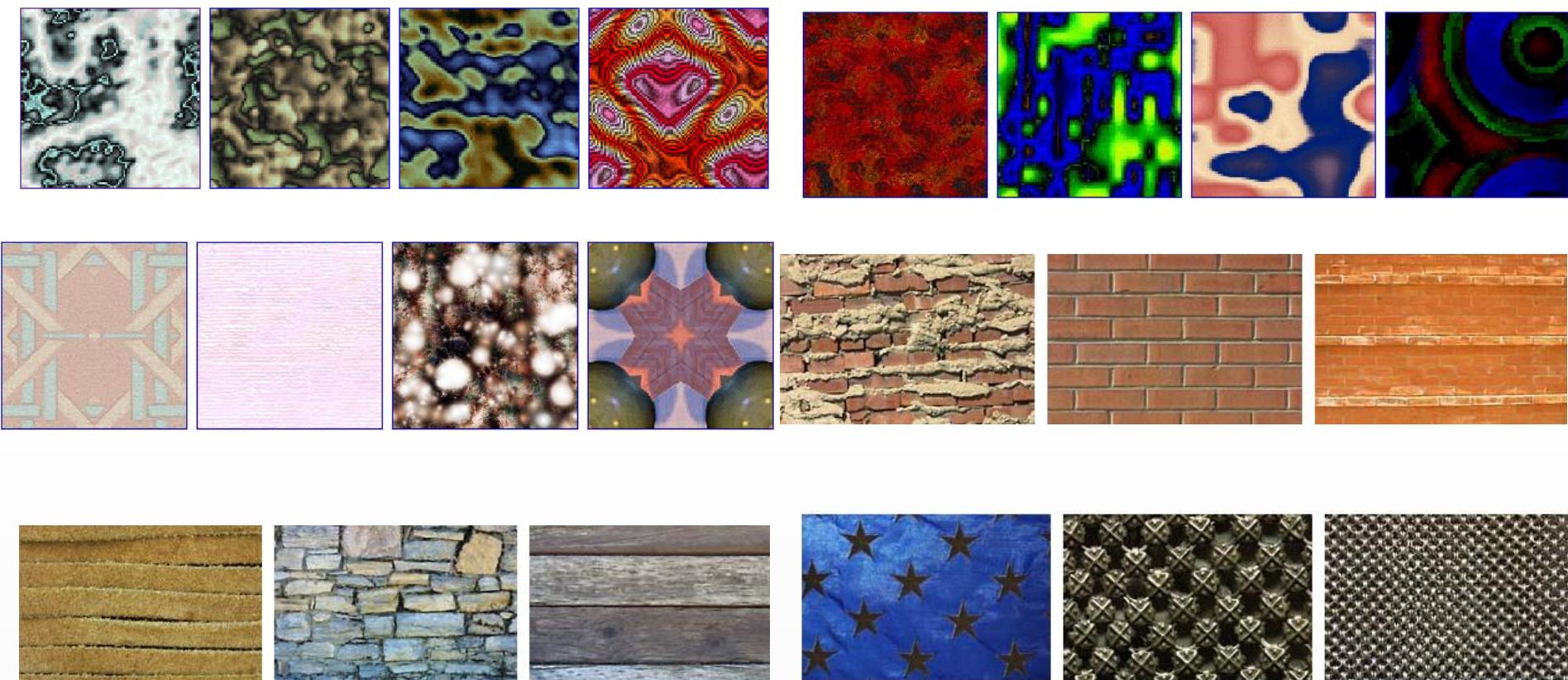


# Further natural textures from Brodatz

---



# Artificial textures



# Application Areas of Texture Analysis

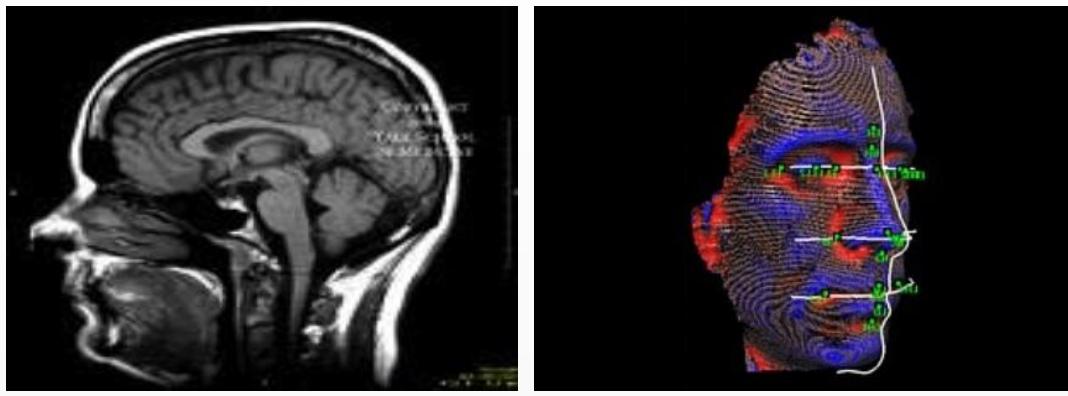
Food processing industry



Biometrics analysis  
(fingerprint, iris or retina, etc.)



Medical image analysis



Global information system (GIS)  
(for land, etc. analysis)



## Texture analysis:

- ⦿ There are various primary issues in texture analysis:
  - *TEXTURE CLASSIFICATION*
  - *TEXTURE SEGMENTATION*
  - *SHAPE RECOVERY FROM TEXTURE, and*
  - *MODELING.*

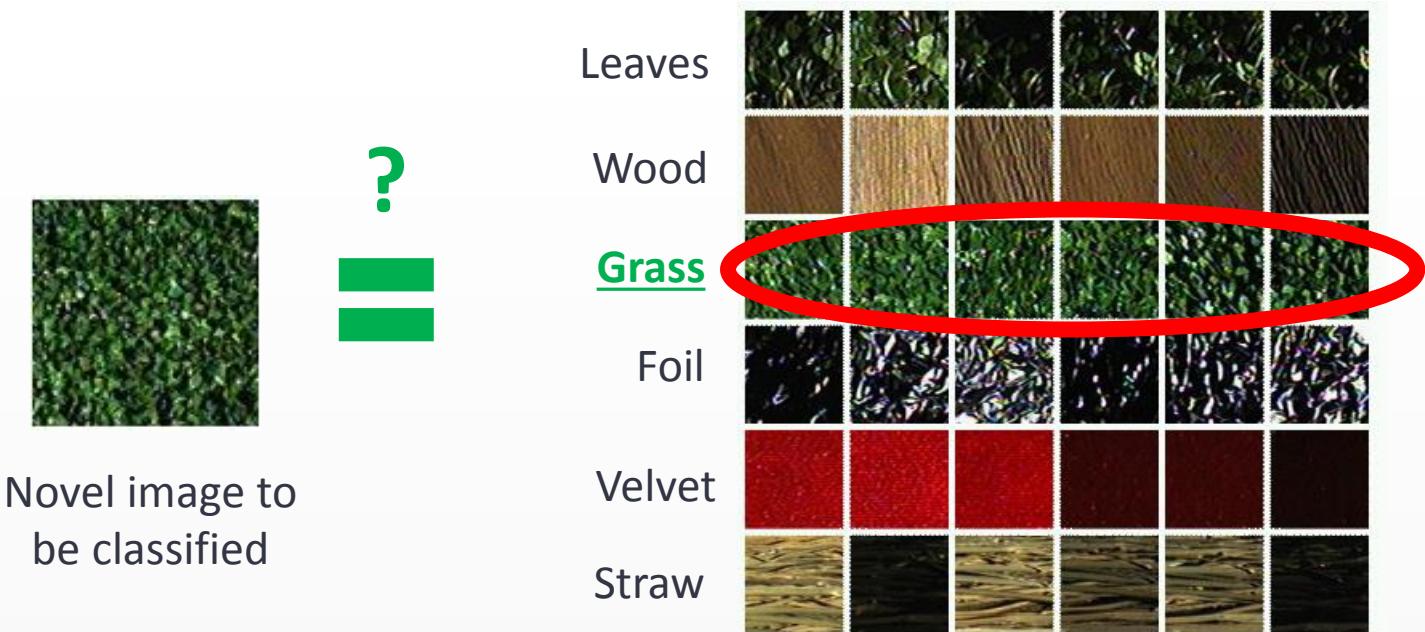
# Texture classification

- ◎ In texture **classification**, the problem is **identifying** the given textured region from a given set of texture classes.
  - The texture analysis algorithms **extract distinguishing feature** from each region to facilitate classification of such patterns.



# Texture classification

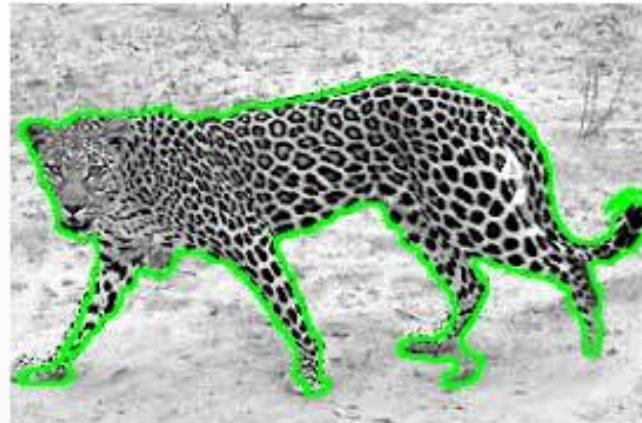
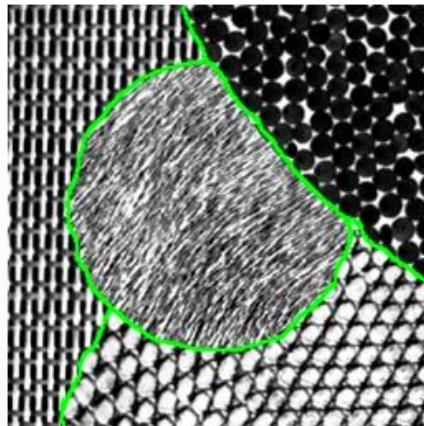
- In texture **classification**, the problem is **identifying** the given textured region from a given set of texture classes.
  - The texture analysis algorithms **extract distinguishing feature** from each region to facilitate classification of such patterns.



# Texture segmentation

---

- Unlike texture classification, texture **segmentation** is concerned with automatically determining the **boundaries** between various textured regions in an image.
- Both region-based methods and boundary-based methods have been attempted to segments texture images.



# Shape recovery from texture

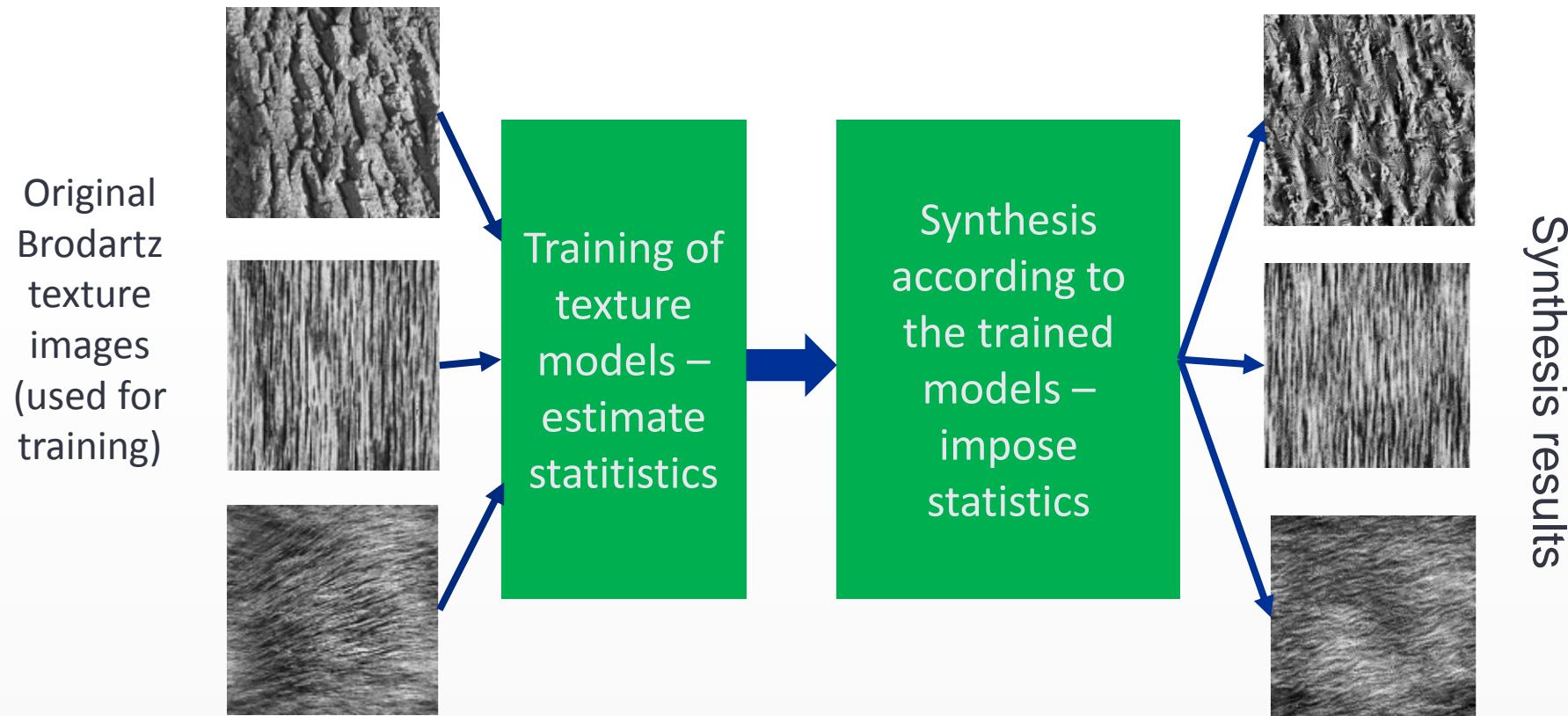
---

- Image plane variation in the texture properties, such as density, size and orientation of texture primitives, are the cues exploited by shape –from-texture algorithms.
- Quantifying the changes in the shape of texture elements is also useful to determine surface orientation.



# Texture modeling

- Specify a model that clearly identifies the given pattern sample



# Techniques for Texture Extraction

---

- There are various techniques for texture extraction. Texture feature extraction algorithms can be grouped as follows:
  - Statistical
  - Geometrical
  - Model based
  - Signal Processing

# Statistical method includes

---

## A. *Local features*

- Grey level of central pixels,
- Average of grey levels in window,
- Median,
- Standard deviation of grey levels,
- Difference of maximum and minimum grey levels,
- Difference between average grey level in small and large windows,
- Kirsch feature,
- Combine features

## B. *Galloway*

- run length matrix

## C. *Haralick*

- co-occurrence matrix

# Geometrical method includes

---

- First threshold images into binary images of n grey levels.
- Then calculate statistical features of connected areas.

# Model based method includes

---

- These involve building mathematical models to describe textures:
  - Markov random fields (see: later – image segmentation lecture)
  - Fractals:

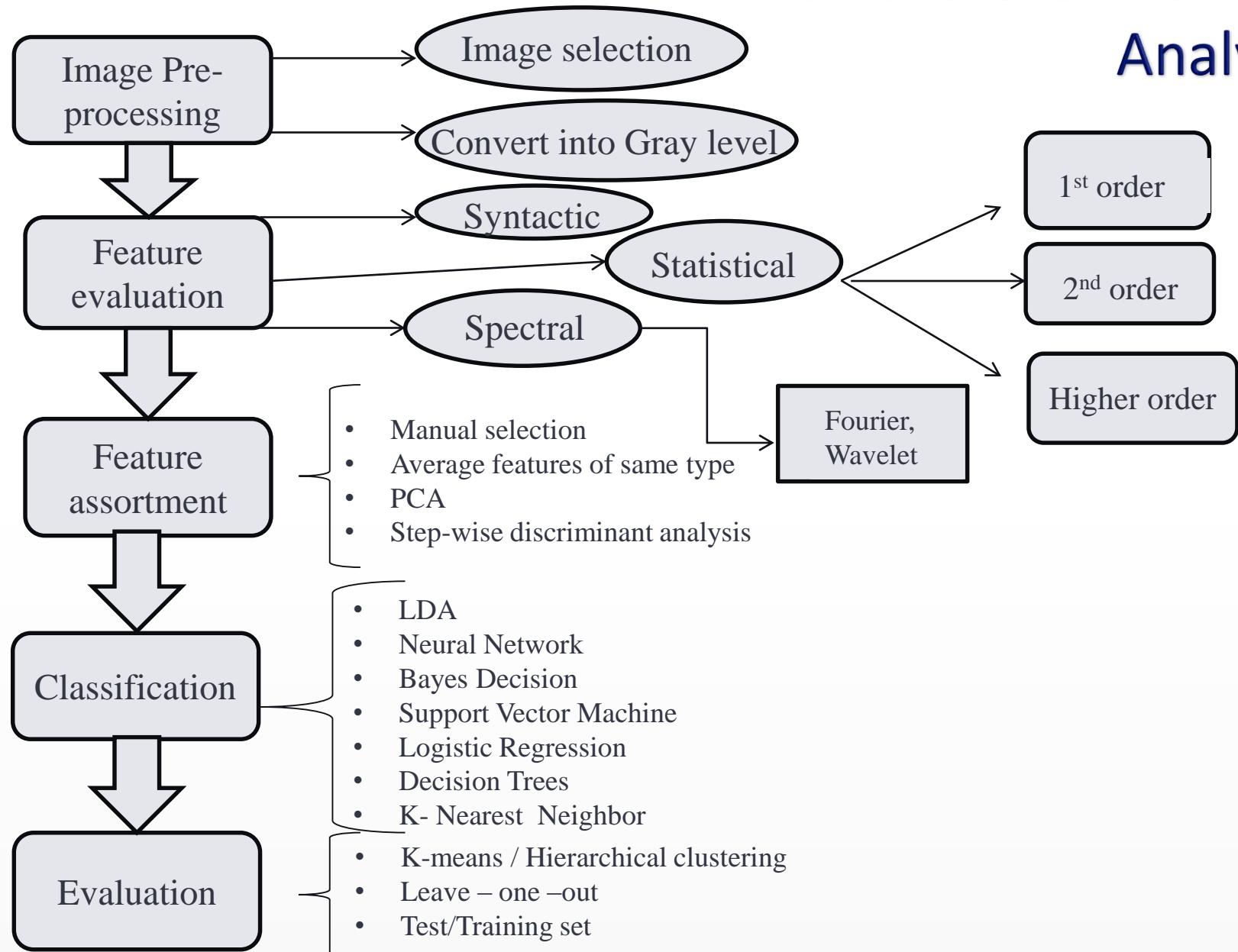


# Signal processing includes

---

- These methods involve transforming original images using filters and calculating the energy of the transformed images.
  - Law's masks (see: today – later)
  - Laines – Daubechies wavelets
  - Fourier transform (see: today – earlier)
  - Gabor filters (see: today – later)

# Flowchart for Texture Analysis



# Discrimination vs. classification of textures

---

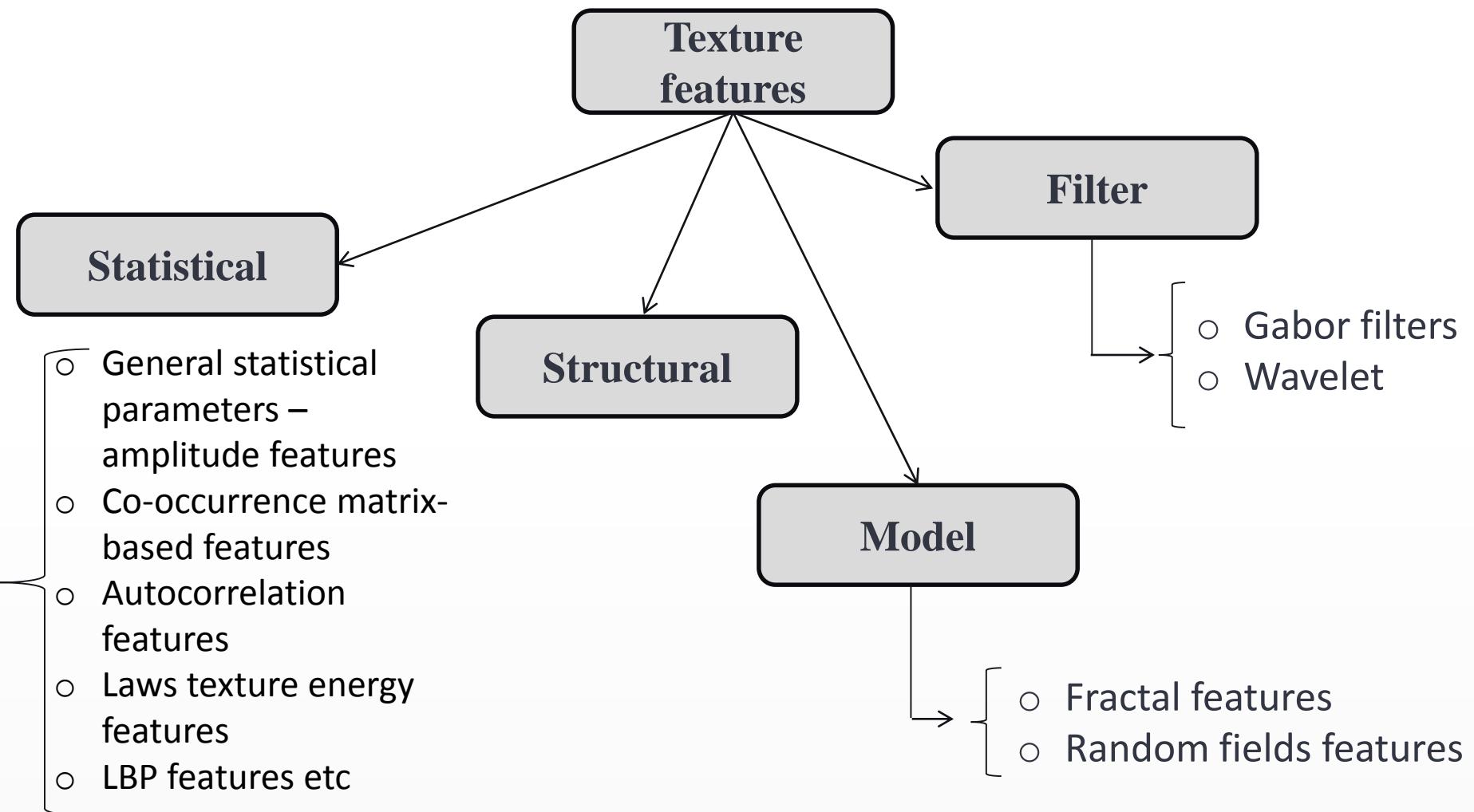
## ○ Discrimination

- Classification - grouping of blobs or points, and classifying them into various classes (local attributes)
- Segmentation - separation of spots / areas (local properties + neighborhoods)

## ○ Classification

- Supervised approach
  - take samples of textures (statistics, metrics) then
  - examine the similarity of new textures
- Unsupervised
  - evaluate statistics
  - sample categorization into classes

# Methods for Texture Features



# Statistical features

## Amplitude features

---

### ◎ Amplitude-features

- Mean

$$M(j, k) = \frac{1}{(2w+1)^2} \sum_{m=-w}^w \sum_{n=-w}^w F(j+m, k+n)$$

- Deviation

$$S(j, k) = \frac{1}{(2w+1)^2} \left[ \sum_{m=-w}^w \sum_{n=-w}^w [F(j+m, k+n) - M(j+m, k+n)]^2 \right]^{1/2}$$

## Statistical features

# Gray Level Co-occurrence Matrix (GLCM)

---

- Also referred as **co-occurrence distribution**.
- It is the most classical second-order statistical method for texture analysis.
- An image is composed of **pixels** each with an intensity (a specific gray level), the GLCM is a tabulation of how often different combinations of gray levels co-occur in an image or image section.
- **Gray Level Co-occurrence Matrix** (GLCM) filters operate by computing, for each filter window position, how often specific pairs of image cell values occur in neighboring cell positions (such as one cell to the right).
- The results are tabulated in a co-occurrence matrix, and specific statistical measures are computed from this matrix to produce the filtered value for the target cell

# Statistical features

## First vs. second order statistics examples

---

- First order statistics example: simple histogram

- Measures the number of different gray value occurrences of independent pixels
- $h_i$ : number of pixels in the image with gray value  $i$

- Second order statistics example: GLCM

- Measures the frequencies of joint gray value occurrences of different pixel pairs with a pre-defined spatial offset
- $p_{ij}(\Delta x, \Delta y)$ : frequency of pixel pairs with an offset  $(\Delta x, \Delta y)$ , where the gray value of the first pixel is  $i$  and the gray value of the second pixel is  $j$
- For example:  $\Delta x=1, \Delta y = 0, i=0, j=255$ : frequency of one-pixel-wide vertical black-white transitions in an image



# Statistical: Co-occurrence Matrix-based Features

---

- It is a matrix of *frequencies* at which → two pixels, separated by a certain vector, occur in the image.
- Co-occurrence matrix is defined as,

$$p_{ij}(\Delta x, \Delta y) = W \cdot Q(i, j | \Delta x, \Delta y)$$

where,

$$W = \frac{1}{(M - \Delta x)(N - \Delta y)} \quad Q(i, j | \Delta x, \Delta y) = \sum_{n=1}^{N-\Delta y} \sum_{m=1}^{M-\Delta x} A$$

where,

$$A = \begin{cases} 1 & \text{if } f(m, n) = i \text{ and } f(m + \Delta x, n + \Delta y) = j \\ 0 & \text{otherwise} \end{cases}$$

# Computation of Co-occurrence Matrix

---

- It has size  $N \times N$  ( $N$  = Number of gray-values) i.e., the rows & columns represent the set of possible pixel values.
- Polar representation of the offset:
  - two parameters  $d, \theta$  (instead of  $\Delta x, \Delta y$ ):

$d \rightarrow$  Relative **distance** between the pixel pair

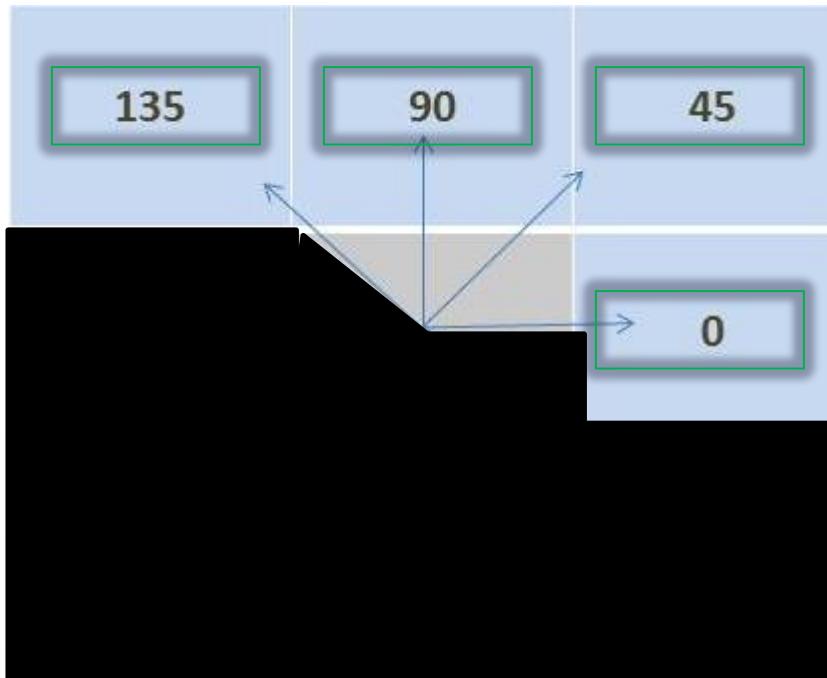
(measured in pixel number. e.g., 1, 2, ...)

$\theta \rightarrow$  Relative **orientation** / rotational angle.

(e.g.,  $0^\circ, 45^\circ, 90^\circ, 135^\circ, \dots$ )

# 8 Directions/orientations ( $\theta$ ) of Adjacency

---



we consider  $\theta$  as horizontal ( $0^\circ$ ), front diagonal ( $45^\circ$ ), vertical ( $90^\circ$ ) and back diagonal ( $135^\circ$ )

# Computation of Co-occurrence Matrix

Image matrix

0	0	1	1
0	0	1	1
0	2	2	2
2	2	3	3

Pixel values: 0,1,2,3. Thus,  $N= 4$

Thus, size of CM = 4x4

$$d = 1$$

$$\theta = \text{horizontal } (0^\circ)$$

Find the number of co-occurrences of pixel  $i$  to the neighboring pixel value  $j$

$i/j$	0	1	2	3
0	$\#(0,0)$	$\#(0,1)$	$\#(0,2)$	$\#(0,3)$
1	$\#(1,0)$	$\#(1,1)$	$\#(1,2)$	$\#(1,3)$
2	$\#(2,0)$	$\#(2,1)$	$\#(2,2)$	$\#(2,3)$
3	$\#(3,0)$	$\#(3,1)$	$\#(3,2)$	$\#(3,3)$

## Example: Computation (contd.)

---

$i/j$	0
0	$\#(0,0)$ 

$$d = 1$$

$$\theta = \text{horizontal } (0^\circ)$$

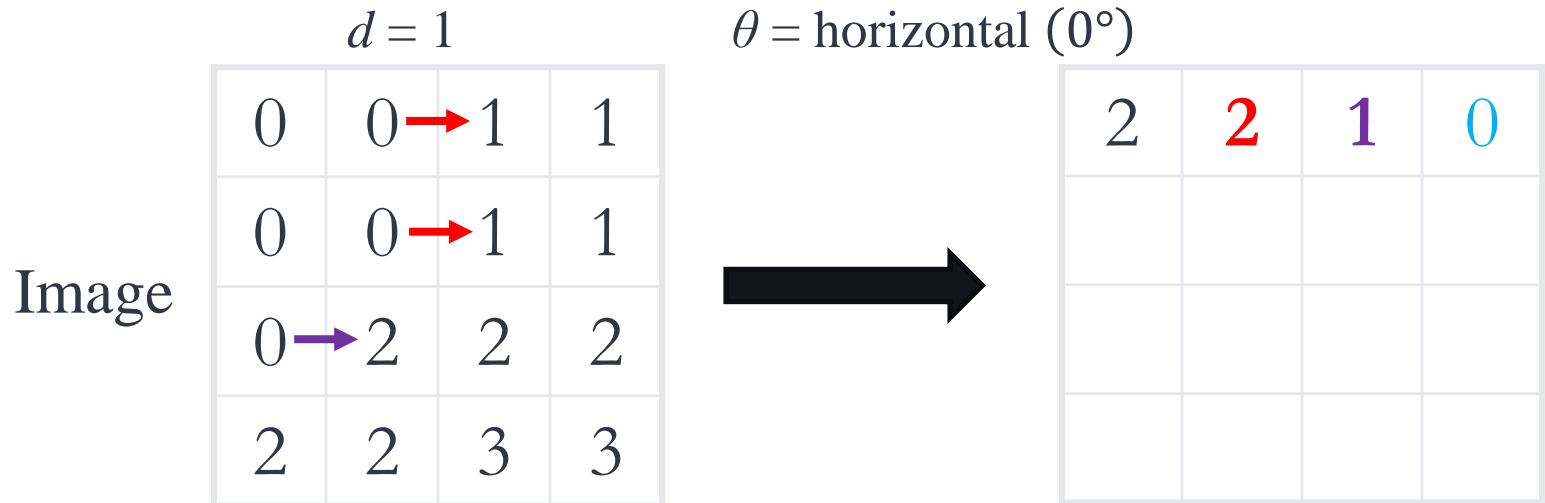


0	0	1	1
0	0	1	1
0	2	2	2
2	2	3	3



2			

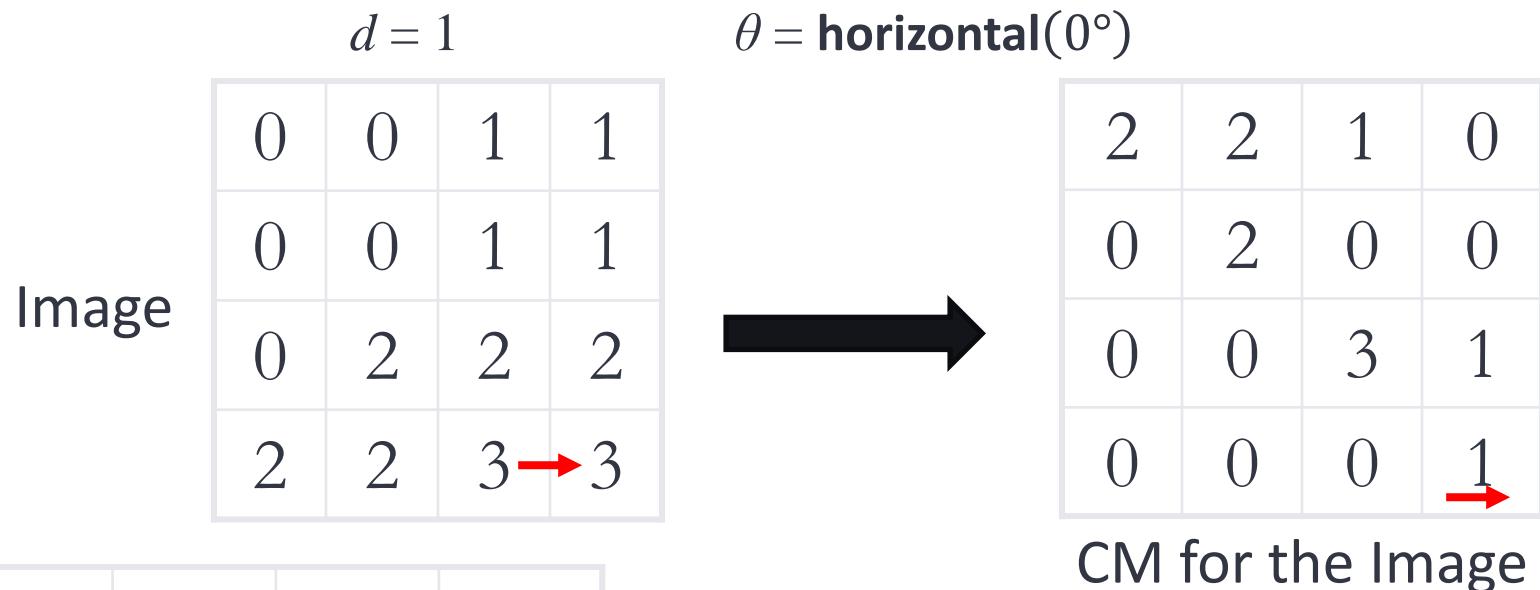
# Example: Computation (contd.)



$i/j$	0	1	2	3
0		#(0,1)	#(0,2)	#(0,3)
1				
2				
3				

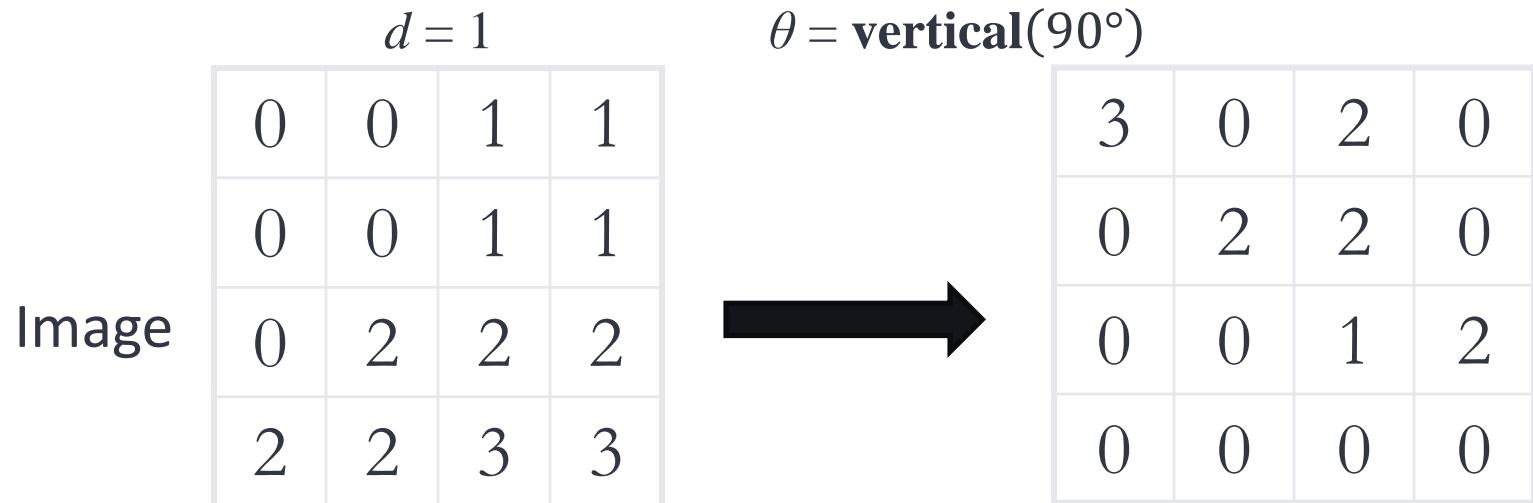
CM for the Image

# Example: Computation (contd.)



$i/j$	0	1	2	3
0	#(0,0)	#(0,1)	#(0,2)	#(0,3)
1	#(1,0)	#(1,1)	#(1,2)	#(1,3)
2	#(2,0)	#(2,1)	#(2,2)	#(2,3)
3	#(3,0)	#(3,1)	#(3,2)	#(3,3)

# Example: Computation (contd.)



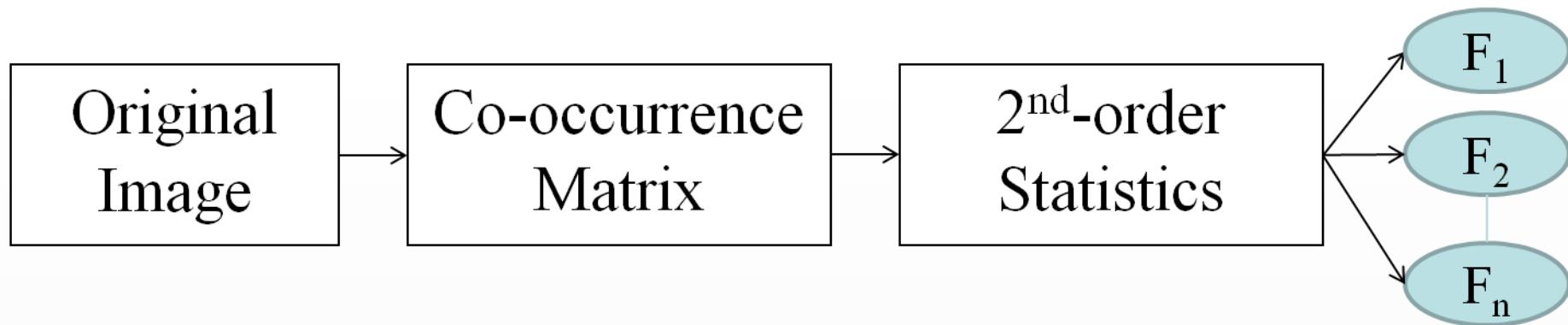
$i/j$	0	1	2	3
0	#(0,0)	#(0,1)	#(0,2)	#(0,3)
1	#(1,0)	#(1,1)	#(1,2)	#(1,3)
2	#(2,0)	#(2,1)	#(2,2)	#(2,3)
3	#(3,0)	#(3,1)	#(3,2)	#(3,3)

CM for the Image

# Features on co-occurrence matrix

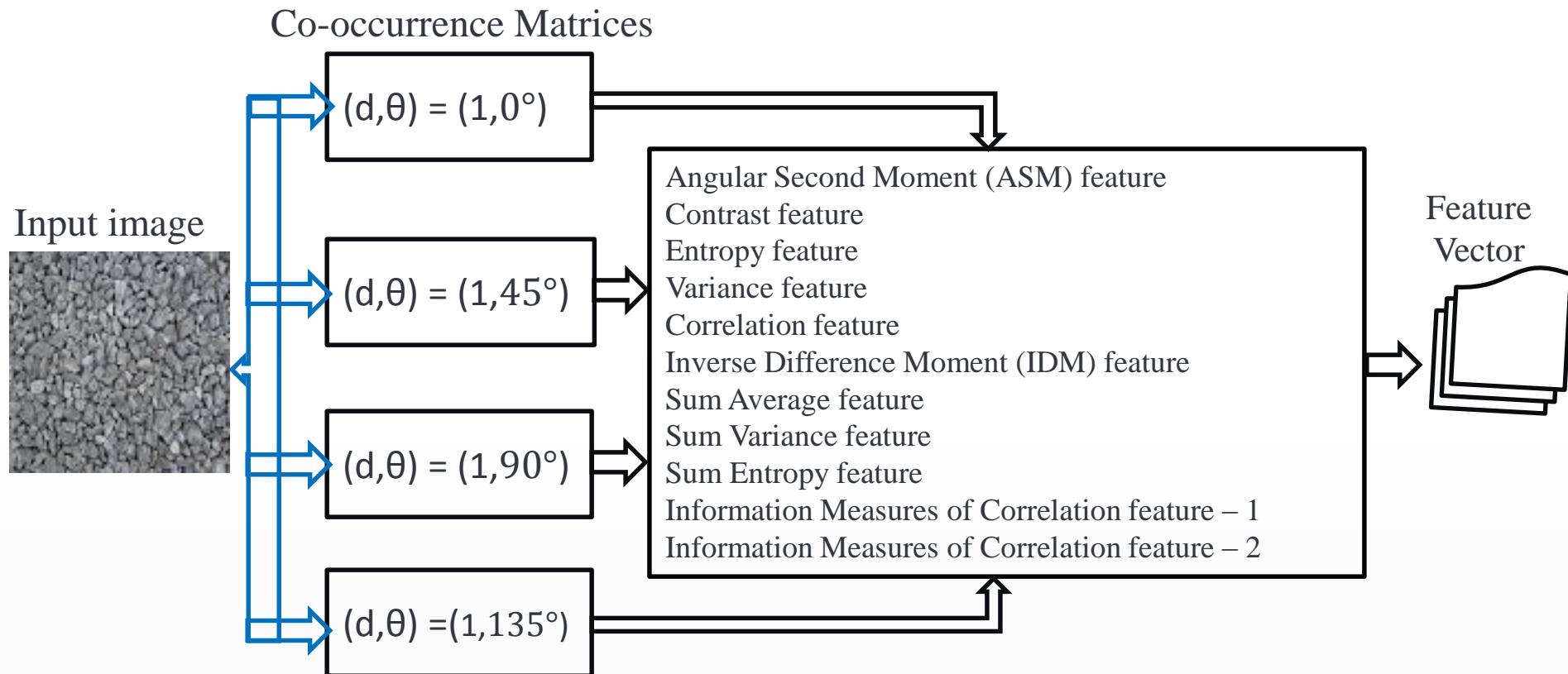
---

- Co-occurrence matrices capture properties of a texture
- But they are *not directly useful* for further analysis  
(e.g., comparison of two textures)



**11 Numeric features** are computed from a matrix

# Features on co-occurrence matrix



# Energy

---

- Also called **Uniformity** or **Angular second moment**.
- **Measures the textural uniformity that is pixel pair repetitions.**
- Detects disorders in textures.
- Energy reaches a maximum value equal to one

$$Energy = \sum_i \sum_j p_{ij}^2$$

# Entropy

---

- **Measures the disorder or complexity of an image.**
- The entropy is large when the image is not texturally uniform.
- Complex textures tend to have high entropy.
- Entropy is strongly, but inversely correlated to energy.

$$Entropy = - \sum_i \sum_j p_{ij} \log_2 p_{ij}$$

# Contrast

---

- Measures the spatial frequency of an image and is difference moment of GLCM.
- It is the difference between the highest and the lowest values of a contiguous set of pixels.
- It measures the amount of local variations present in the image.

$$Contrast = \sum_i \sum_j (i - j)^2 p_{ij}$$

# Homogeneity

---

- Also called as **Inverse Difference Moment**.
- **Measures image homogeneity as it assumes larger values for smaller gray tone differences in pair elements.**
- It is more sensitive to the presence of near diagonal elements in the GLCM.
- It has maximum value when all elements in the image are same.
- Homogeneity decreases **if** contrast increases while energy is kept constant.
- $\text{Homogeneity}(\text{hom}) = \sum_i \sum_j \frac{1}{1 + (i - j)^2} p_{ij}$

# Variance

---

- ◎ This statistic is a measure of heterogeneity and is strongly correlated to first order statistical variable such as standard deviation.
- ◎ Variance increases when the gray level values differ from their mean

$$\text{Variance}(\text{var}) = \sum_i \sum_j (i - \mu)^2 p_{ij}$$

where  $\mu$  is the mean of  $p_{ij}$

## ◎ Contrast (Sum of Squares Variance)

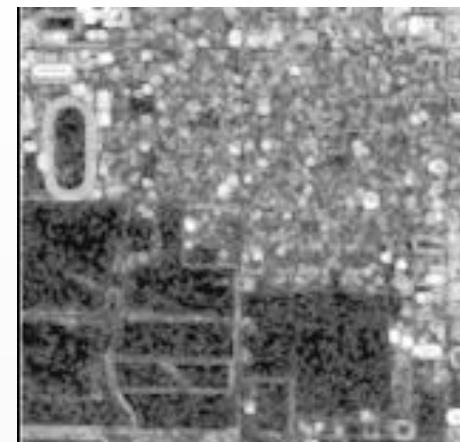
- measures gray-level contrast by using GLCM weighting factors equal to the square of the gray level difference. Thus the averaging weights are 0 for matrix position on the main diagonal and increase exponentially away from the diagonal. The filter result is 0 for areas with identical image values and is high where there are large differences in tone.



Thresholded Contrast



Input image



Contrast filter output

# Features on co-occurrence matrix

## Examples

---

Feature	Comment
F2: Contrast	<ul style="list-style-type: none"><li>- Have discriminating ability.</li><li>- Rotationally-variant.</li></ul>
F3: Entropy	<ul style="list-style-type: none"><li>- Have strong discriminating ability.</li><li>- Almost rotational-invariant.</li></ul>
F4: Variance	<ul style="list-style-type: none"><li>- Have discriminating ability.</li><li>- Rotational-invariant.</li></ul>
F5: Correlation	<ul style="list-style-type: none"><li>- Have strong discriminating ability.</li><li>- Rotational-dependent feature.</li></ul>

# Features on co-occurrence matrix

---

Feature	Comment
F7: Sum average	<ul style="list-style-type: none"><li>- Characteristics are similar to ‘variance’/F4</li><li>- Rotational-invariant.</li></ul>
F10: Information Measure of Correlation–1	<ul style="list-style-type: none"><li>- It has almost similar pattern of ‘sum average’/F7 but vary for various classes</li><li>- Varies significantly with rotation</li></ul>
F11: Information Measure of Correlation–2	<ul style="list-style-type: none"><li>- It is computationally expensive compare to others.</li><li>- Rotation-variant</li></ul>



# Features on co-occurrence matrix

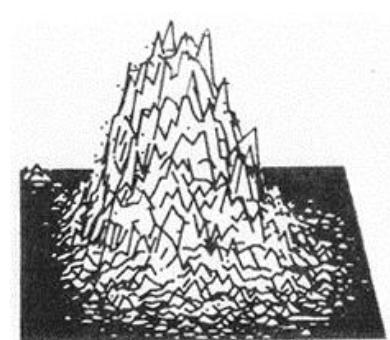
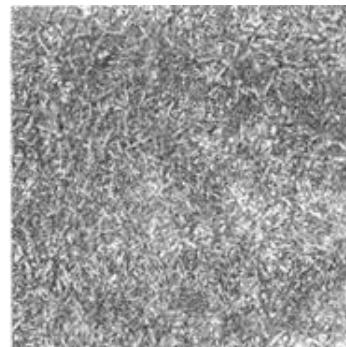
---

Feature	Comment
F1: Angular Second Moment / Energy	- No distinguishing ability
F6: Inverse Different Moment	- Similar to ‘angular second moment’/F1
F8: Sum Variance	- Similar to ‘variance’/F4
F9: Sum Entropy	- Similar to ‘entropy’/F3

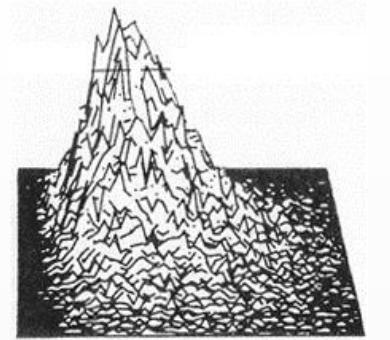
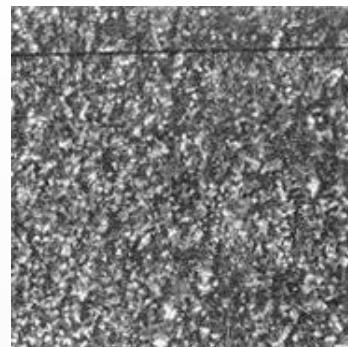
# Visualization of co-occurrence histograms

## ◉ Dependency matrices

- co-occurrence histograms
- calculated in a given direction, and distance
- smoother texture implies more steady response (less dependencies)
- Coarse texture: dominant response along the main diagonal



Grass



Ivy (borostyán)

Grayscale dependency matrices for  $r = 4, \theta = 0^\circ$

# Statistical features

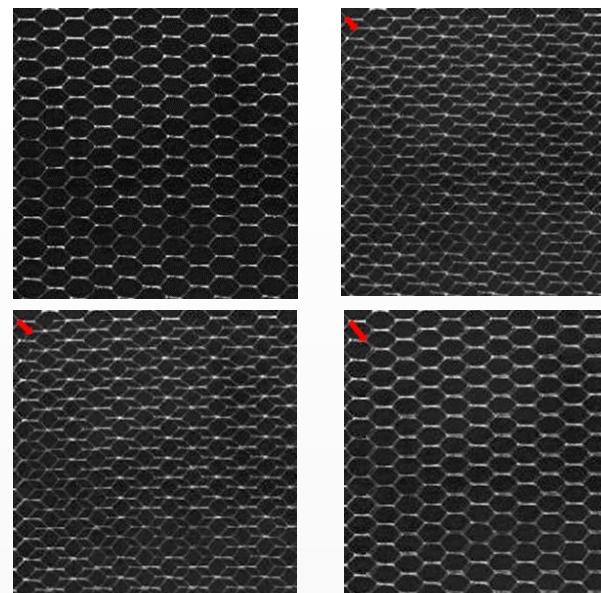
## Autocorrelation function

- **Definition of autocorrelation:** compare the dot product (energy) of non shifted image with a shifted image

$$R_{II}(\Delta x, \Delta y) = \frac{\sum_{x=0}^N \sum_{y=0}^N I(x, y)I(x + \Delta x, y + \Delta y)}{\sum_{x=0}^N \sum_{y=0}^N I^2(x, y)}$$

- Features:

- Autocorrelation function can detect repetitive patterns of texels
- Also defines fineness/coarseness of the texture

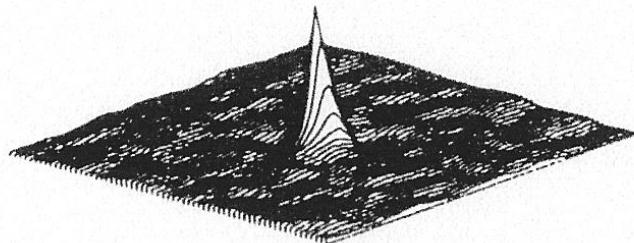


Different shift values

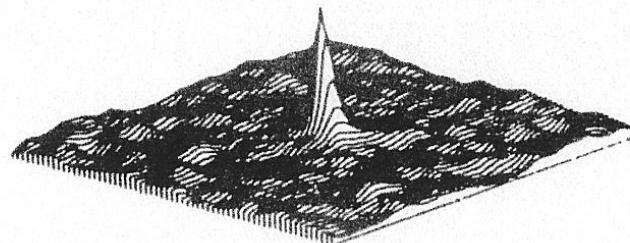
$\Delta x, \Delta y$

# Textures – image features

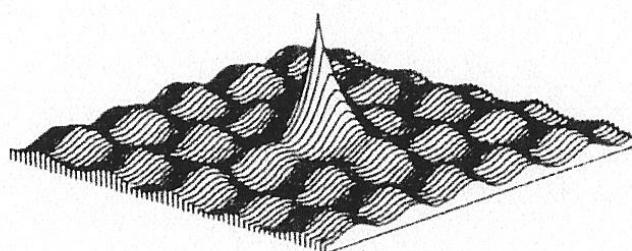
---



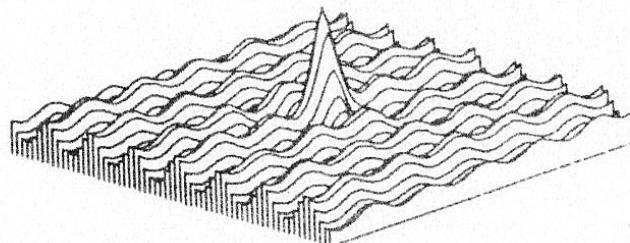
(a) Sand



(b) Grass



(c) Wool



(d) Raffia

Principle: a coarse pattern texture for the same shift value shows greater autocorrelation than a finer pattern one

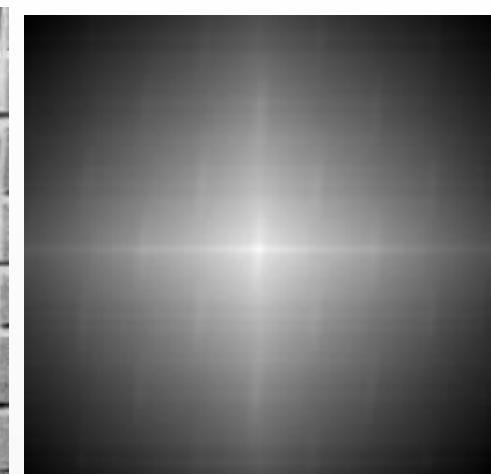
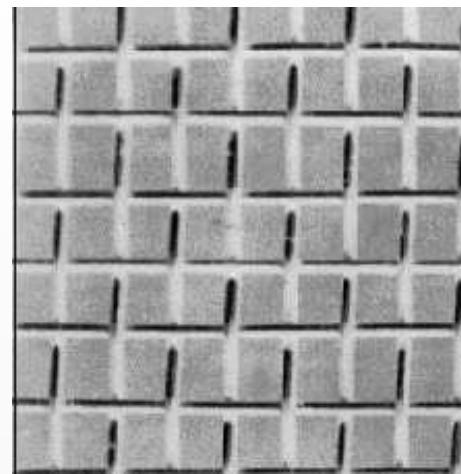
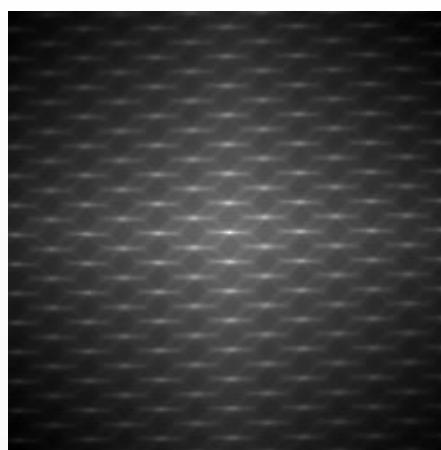
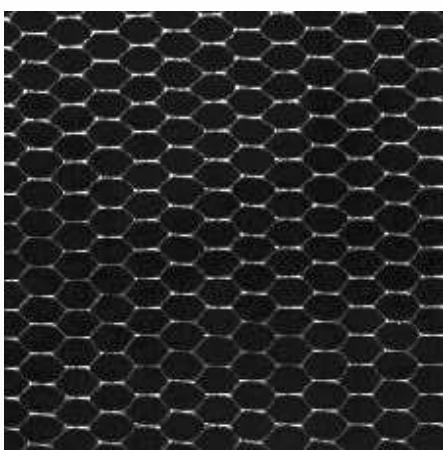
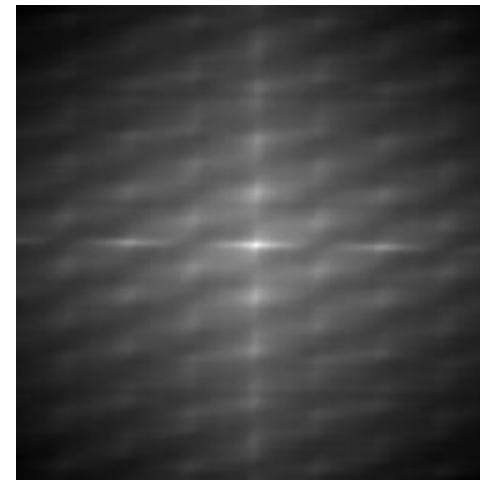
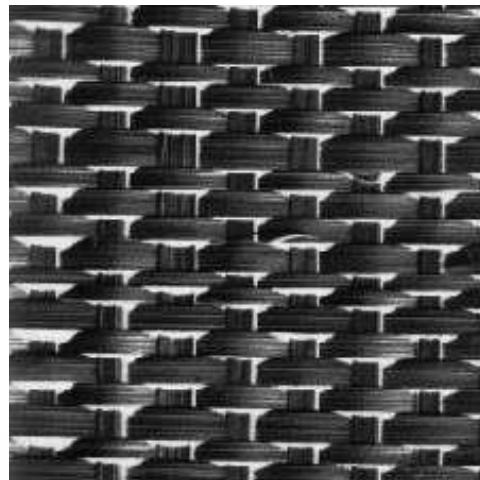
# Interpreting autocorrelation

---

- Coarse texture → function drops off slowly
- Fine texture → function drops off rapidly
- Regular textures → function will have peaks and valleys; peaks can repeat far away from [0, 0]
- Random textures → only peak at [0, 0]; breadth of peak gives the size of the texture

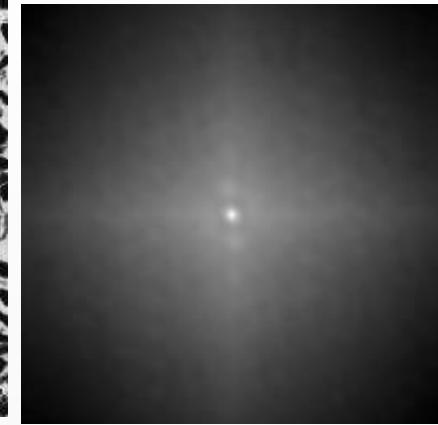
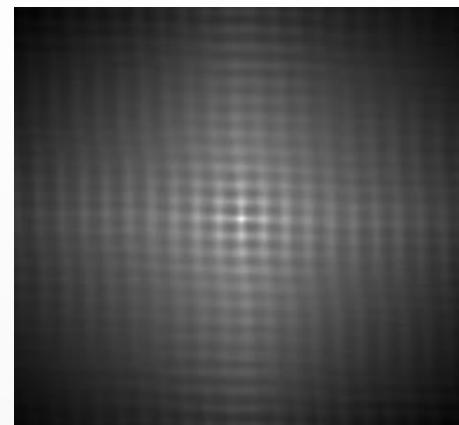
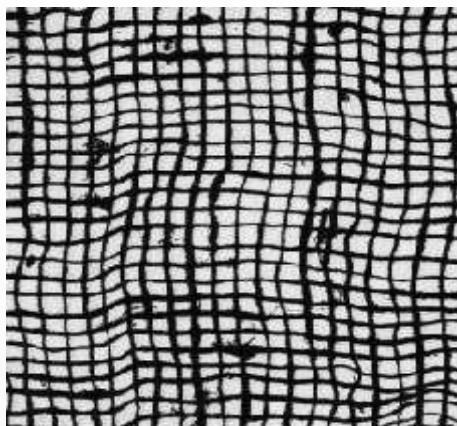
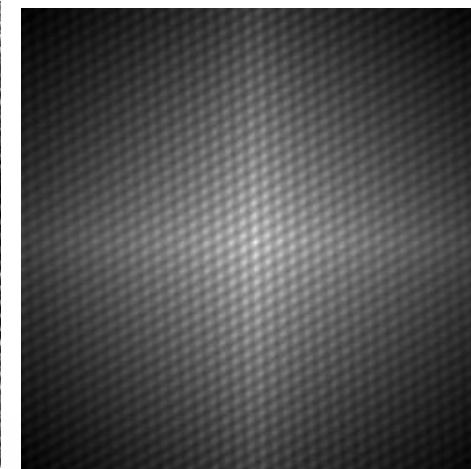
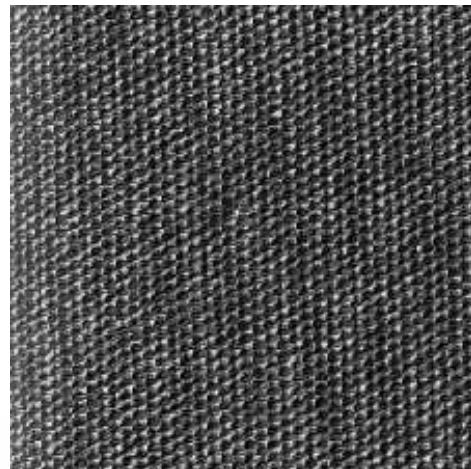
# Autocorrelation

---



# Autocorrelation

---



# Autocorrelation calculation the Fourier domain

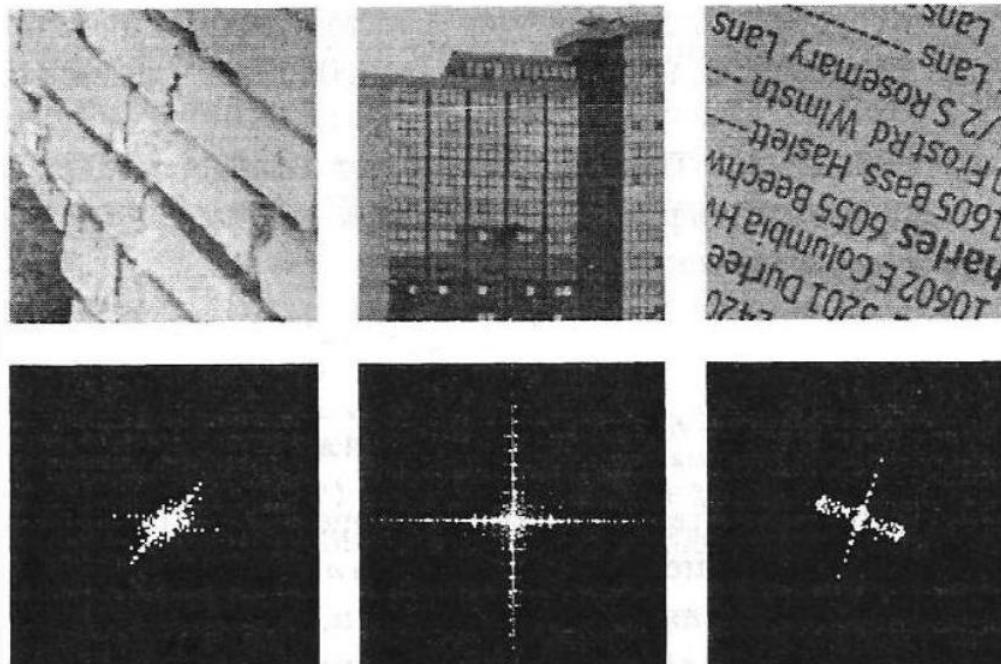
---

## ◎ Wiener-Khinchin Theorem

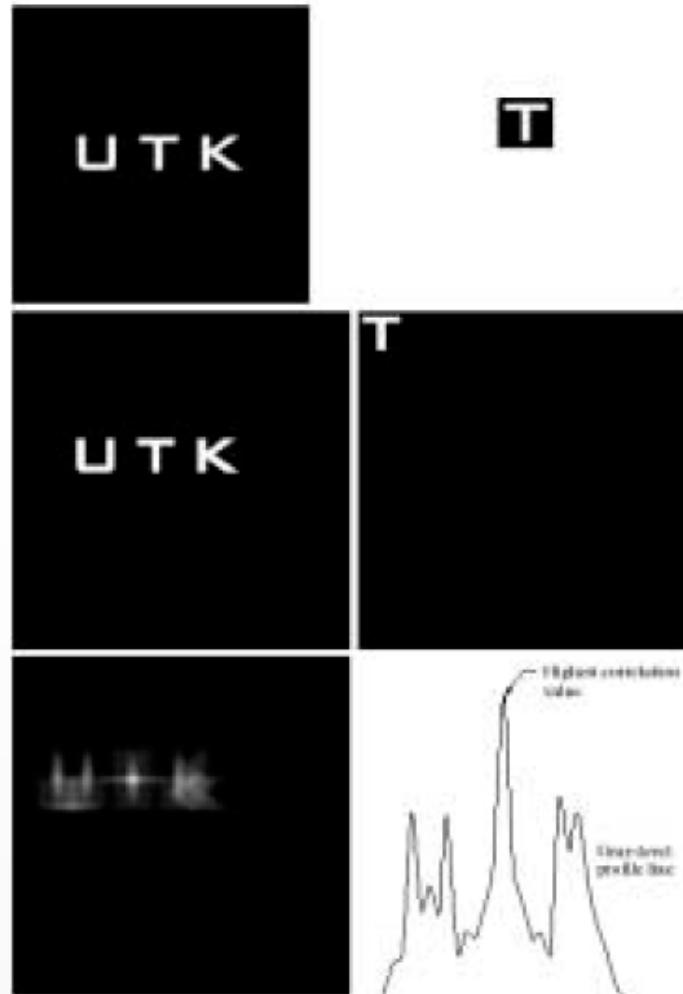
- Input image:  $I(x, y)$
- Fourier transform:  $\{X(\omega_1, \omega_2)\} = \text{DFT}\{I(x, y)\}$
- Power spectrum:  $P_{XX}(\omega_1, \omega_2) = X(\omega_1, \omega_2) \cdot X^*(\omega_1, \omega_2) = |X(\omega_1, \omega_2)|^2$
- Autocorrelation: inverse Fourier transform of the power spectrum:  
 $\{R_{II}(\Delta x, \Delta y)\} = \text{IDFT}\{P_{XX}(\omega_1, \omega_2)\}$

# Fourier domain analysis

- Power spectrum:  $X\{\omega_1, \omega_2\} \cdot X^*\{\omega_1, \omega_2\} = |X\{\omega_1, \omega_2\}|^2$
- Concentrated power → regularity
- High frequency power → fine texture
- Directionality → directional texture



# Correlation (pattern recognition)



# Textures – image features

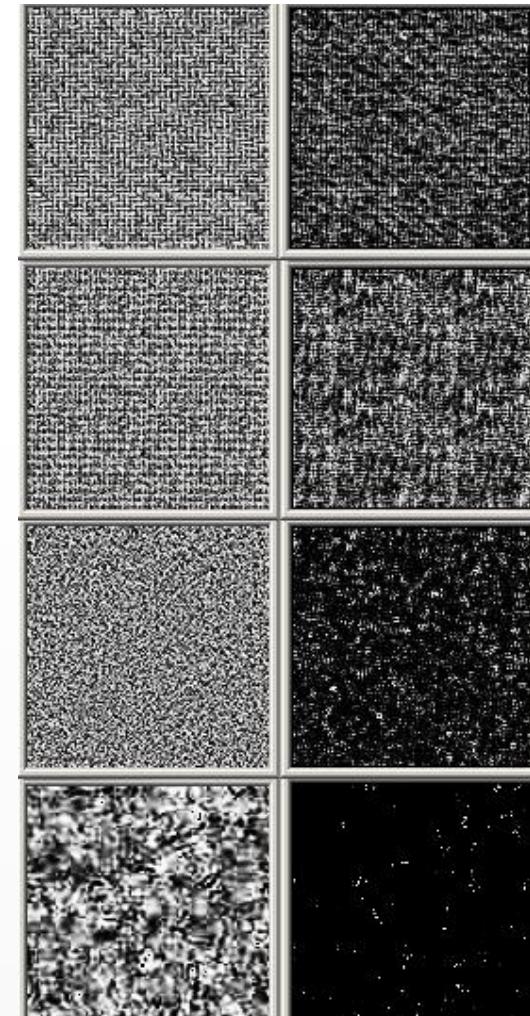
- Edge detection based procedures

$E(j,k)$  = binary edge image obtained  
by some edge detector (eg. Sobel)

Use low threshold for binarization

Measure of local edge content

$$T(j,k) = \frac{1}{(2w+1)^2} \sum_{m=-w}^w \sum_{n=-w}^w E(j+m, k+n)$$



# Laws filters

---

- Enhancing micro-structure of the texture
- Main elements:
  - Averaging
  - Edges
  - Points

$$L_3 = \frac{1}{6} \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} \quad E_3 = \frac{1}{2} \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix} \quad S_3 = \frac{1}{2} \begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix}$$

# Laws filters

---

$$H_1(\ ) = \frac{1}{36} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

$$H_2(\ ) = \frac{1}{12} \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$$

$$H_3(\ ) = \frac{1}{12} \begin{bmatrix} -1 & 2 & -1 \\ -2 & 4 & -2 \\ -1 & 2 & -1 \end{bmatrix}$$

$$H_4(\ ) = \frac{1}{12} \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

$$H_5(\ ) = \frac{1}{4} \begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$$

$$H_6(\ ) = \frac{1}{4} \begin{bmatrix} -1 & 2 & -1 \\ 0 & 0 & 0 \\ 1 & -2 & 1 \end{bmatrix}$$

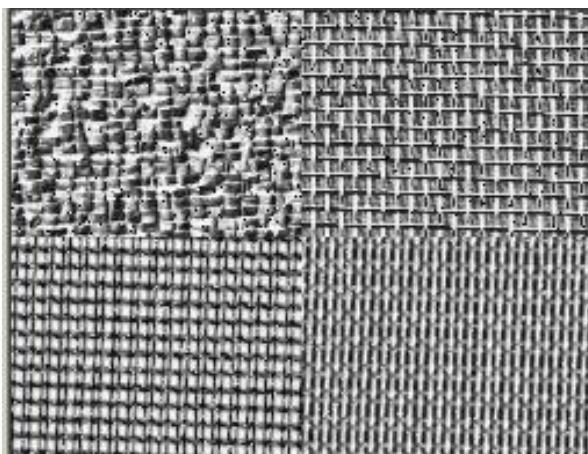
$$H_7(\ ) = \frac{1}{12} \begin{bmatrix} -1 & -2 & -1 \\ 2 & 4 & 2 \\ -1 & -2 & -1 \end{bmatrix}$$

$$H_8(\ ) = \frac{1}{4} \begin{bmatrix} -1 & 0 & 1 \\ 2 & 0 & -2 \\ -1 & 0 & 1 \end{bmatrix}$$

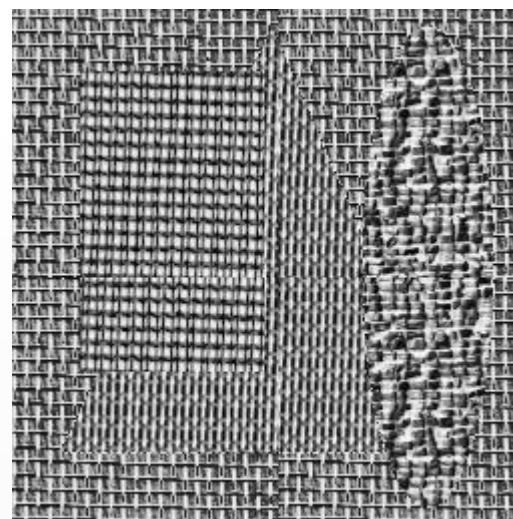
$$H_9(\ ) = \frac{1}{4} \begin{bmatrix} 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{bmatrix}$$

# Texture segmentation - Laws

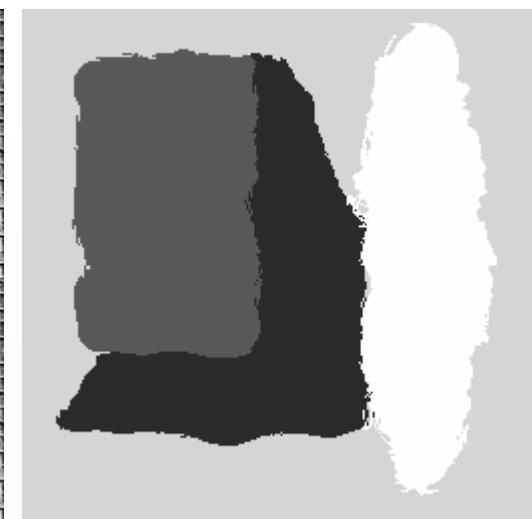
- Training step, thereafter recognizing the trained textures
- Utilizing Law matrices



Training image

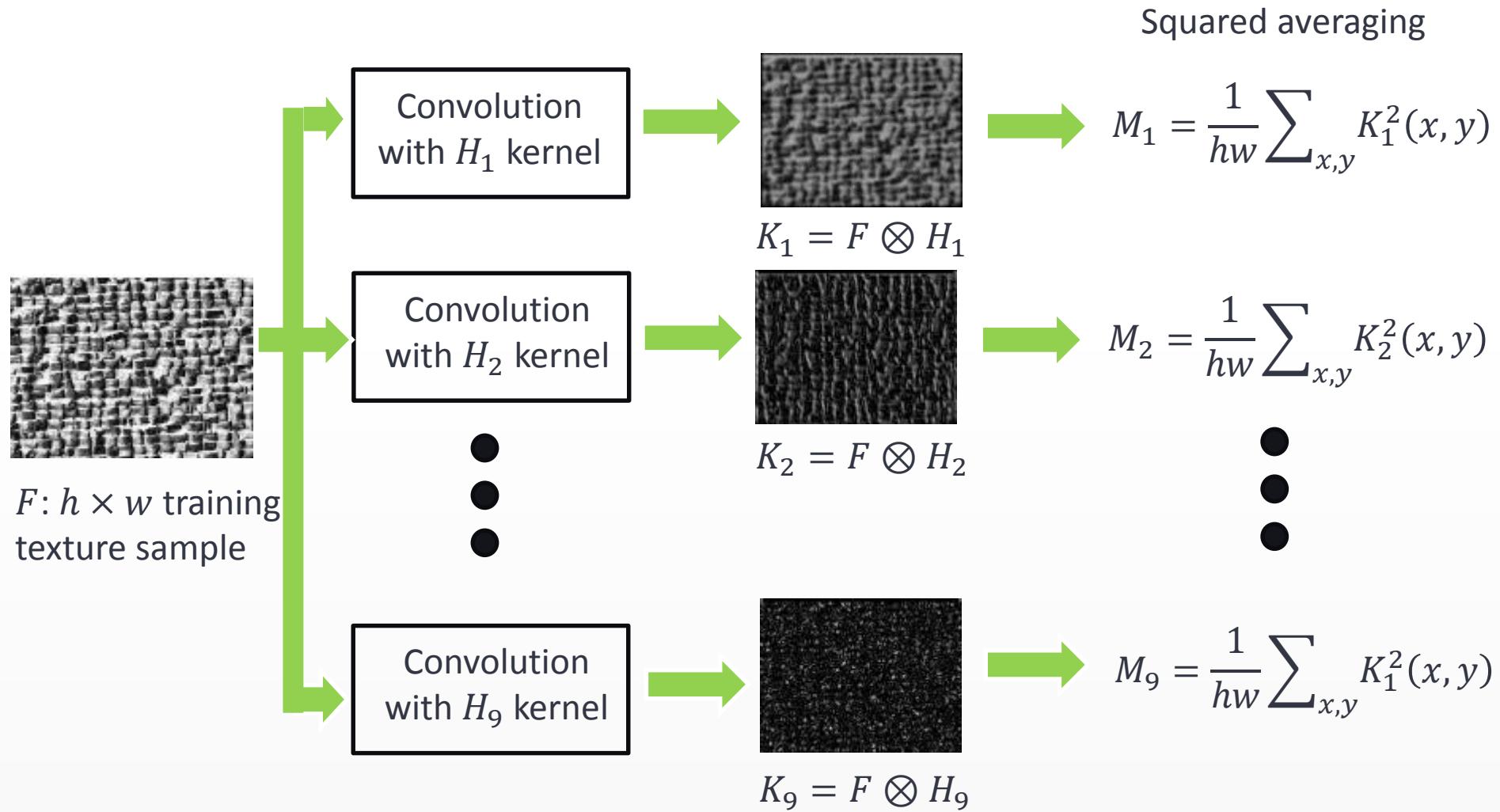


Input test image



Ground truth for  
the test image

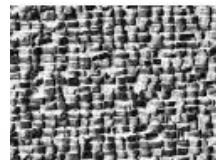
# Learning a training texture model



# Laws filter– training phase

---

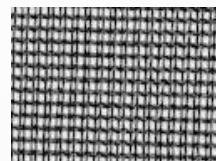
- Each training texture  $j$  is represented by 9 scalars:  $M_1^j \dots M_9^j$   
$$M_i^j = \frac{1}{hw} \sum_{x,y} (K_i^j(x,y))^2, \text{ where } i = 1, \dots, 9, j = 1, \dots, 4, \text{ and } K_i^j = F_j \otimes K_i,$$



$$M_1^1, M_2^1, M_3^1, M_4^1, M_5^1, M_6^1, M_7^1, M_8^1, M_9^1$$



$$M_1^2, M_2^2, M_3^2, M_4^2, M_5^2, M_6^2, M_7^2, M_8^2, M_9^2$$



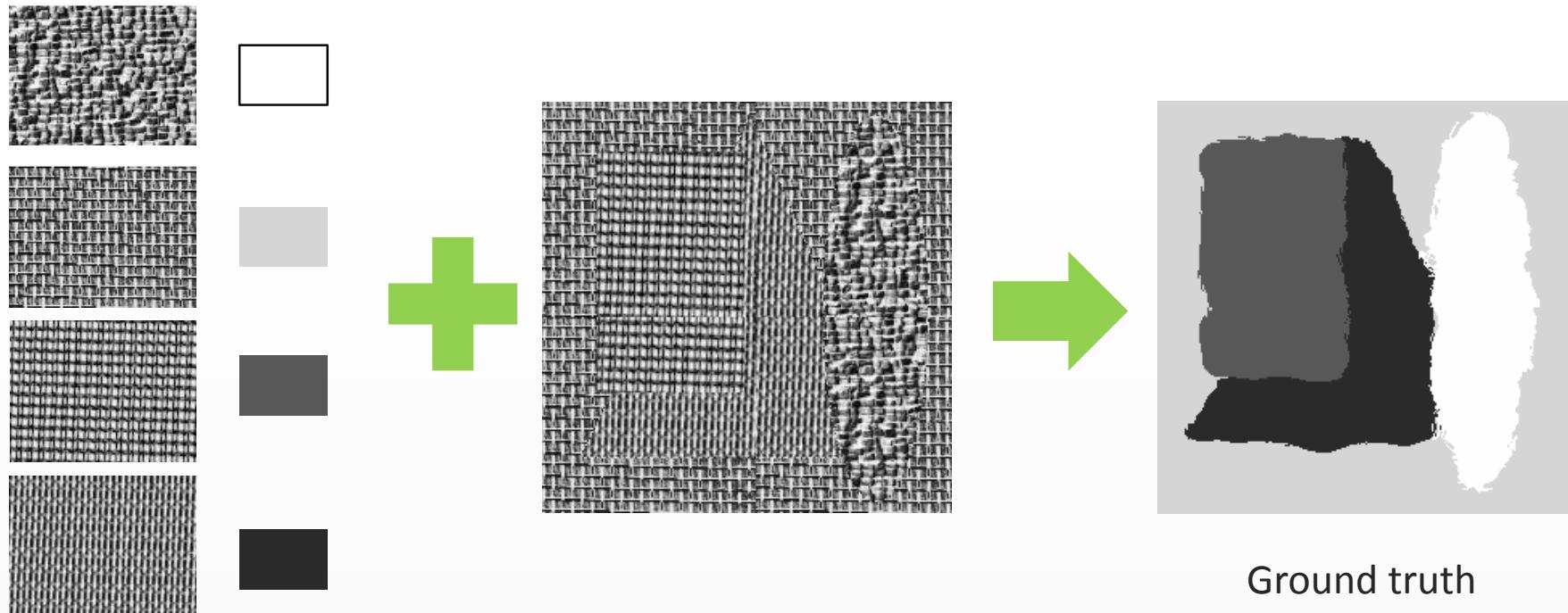
$$M_1^3, M_2^3, M_3^3, M_4^3, M_5^3, M_6^3, M_7^3, M_8^3, M_9^3$$



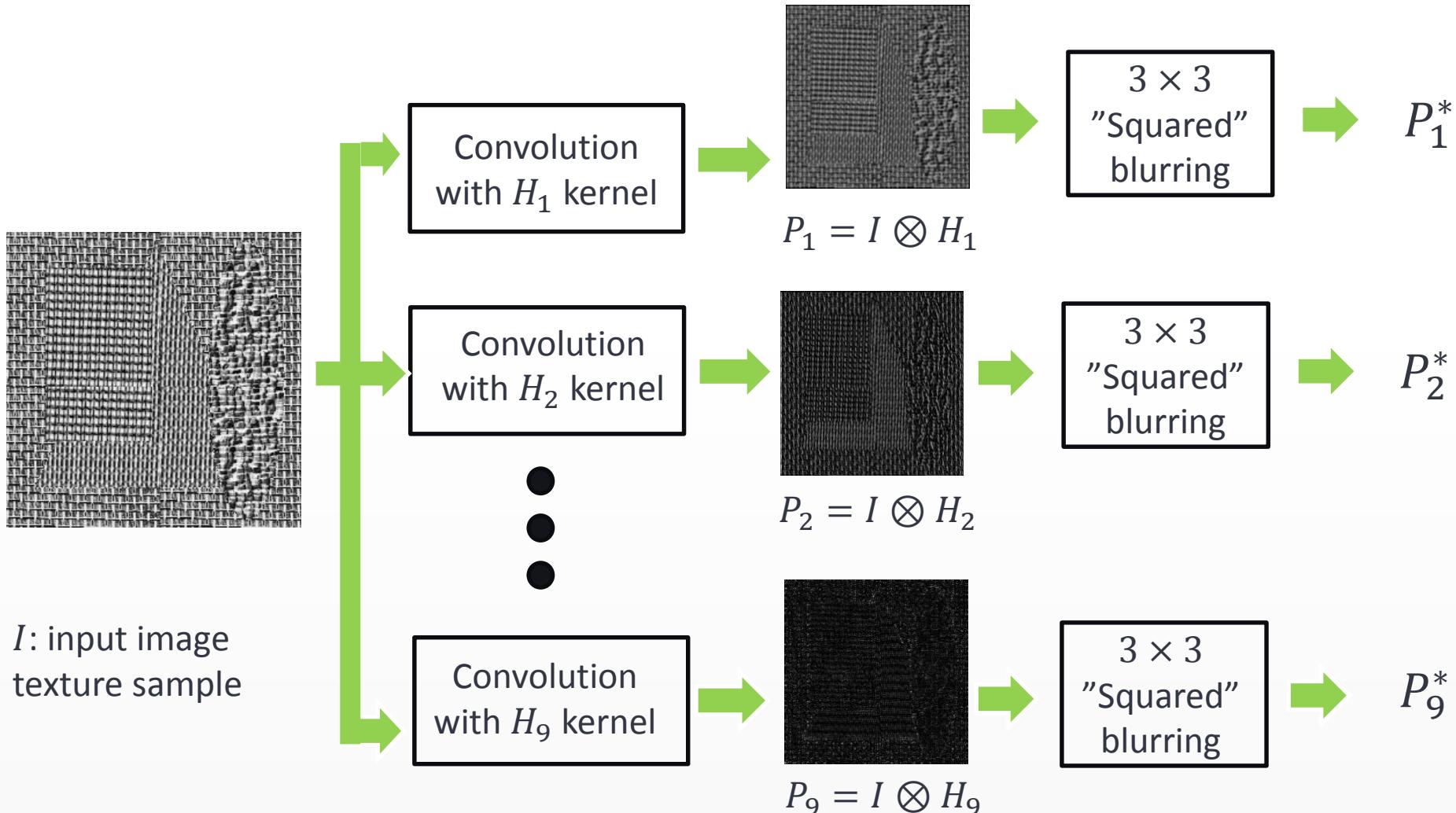
$$M_1^4, M_2^4, M_3^4, M_4^4, M_5^4, M_6^4, M_7^4, M_8^4, M_9^4$$

# Laws filter – recognition phase

- Input image: consists of arbitrary regions of pre-trained textures



# Laws filter – recognition phase



# Laws filter– recognition phase

---

- Pixel level decision of the input map

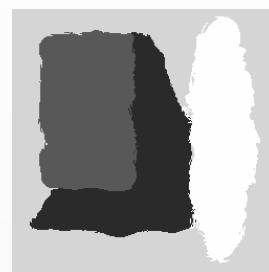
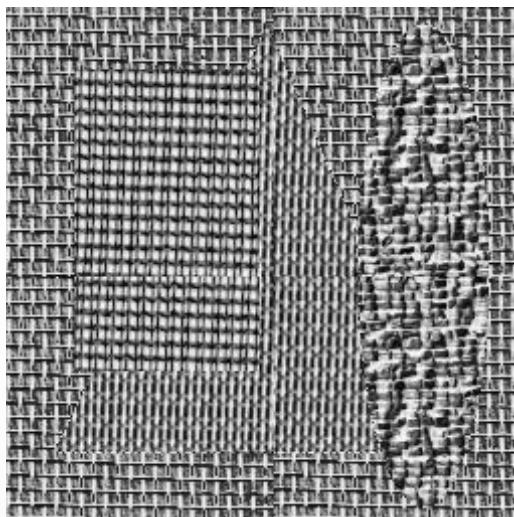
$$\text{ClassMap}(x, y) = \operatorname{argmin}_{j=1 \dots 4} \sum_{i=1 \dots 9} |P_i^*(x, y) - M_i^j|$$

where

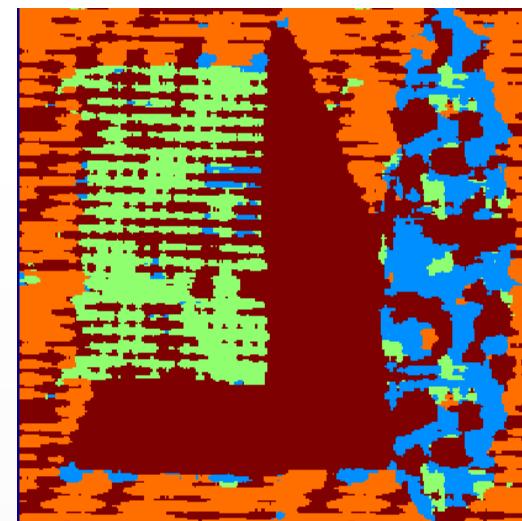
$$P_i^*(x, y) = \frac{1}{9} \sum_{r=x-1}^{x+1} \sum_{s=y-1}^{y+1} P_i^2(x + r, y + s)$$

# Texture segmentation result

- Output „winner class” maps for the four textures – enhanced with some morphology



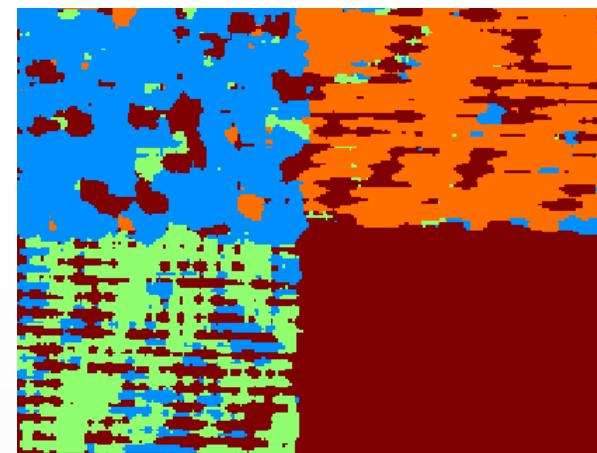
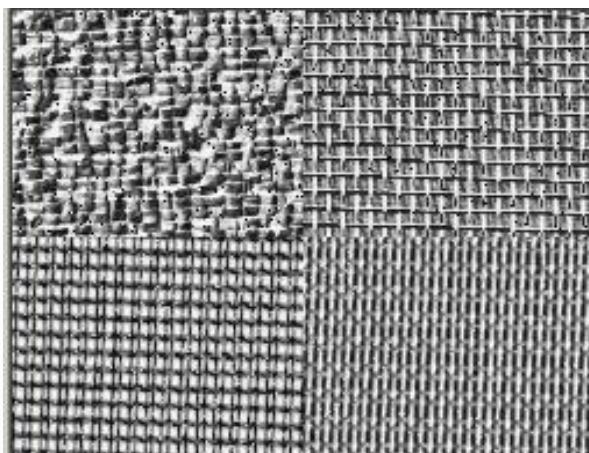
Ground truth



Laws segmentation results

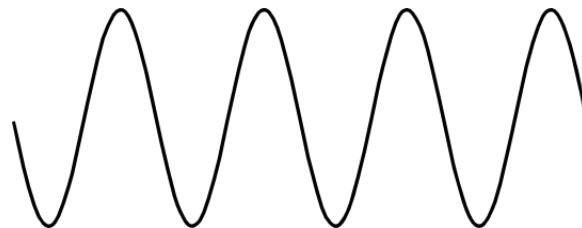
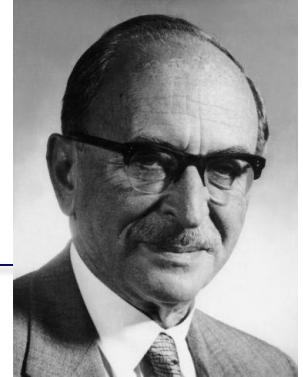
# Texture segmentation result

- Output „winner class” maps for the four textures – enhanced with some morphology



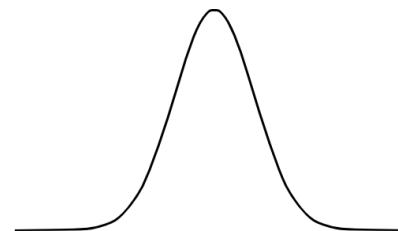
Laws segmentation results

# Gabor filters – 1D illustration

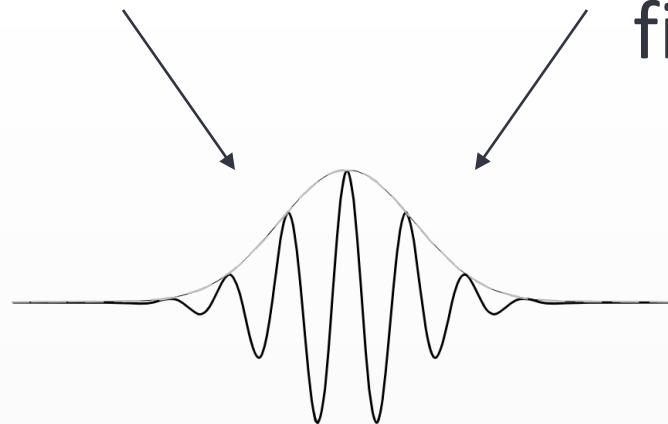


Sinusoid

×



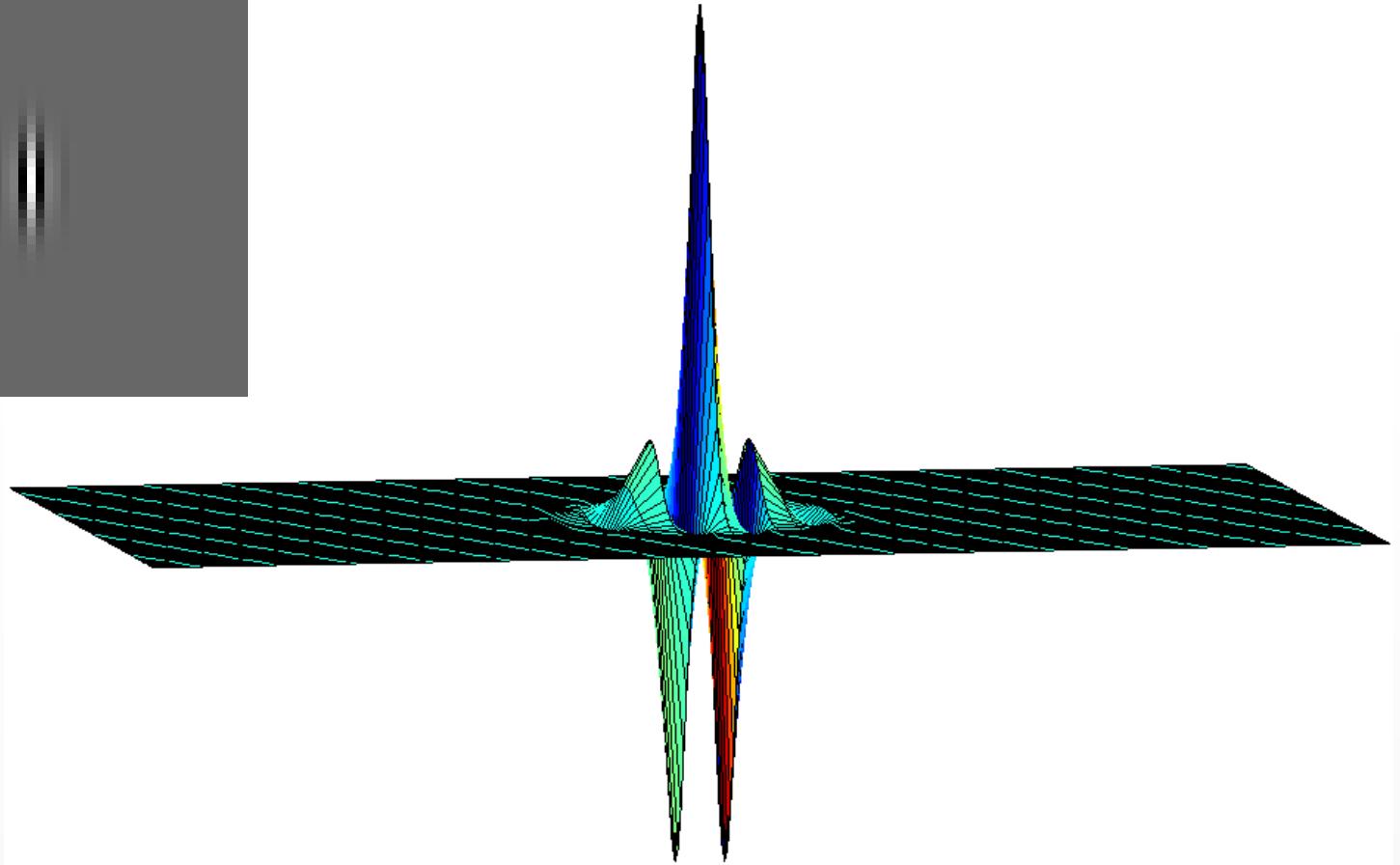
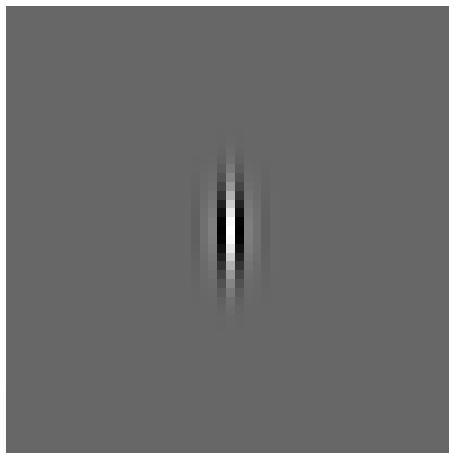
Gaussian  
filter



Gabor filter

# Gabor filters- 2D kernel example

---

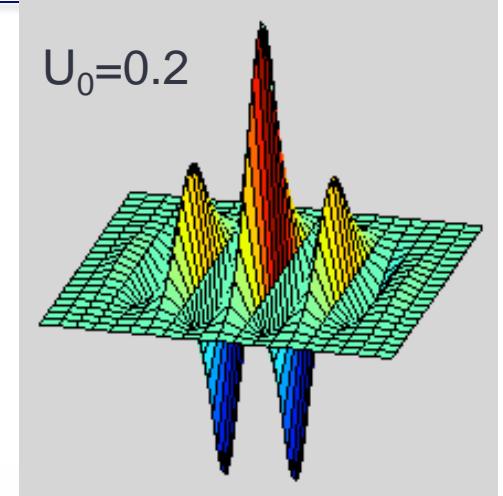
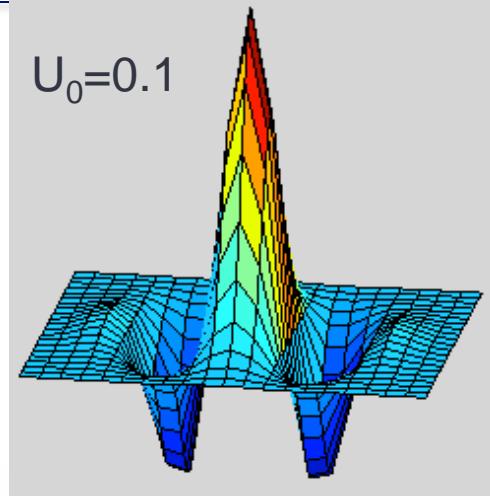
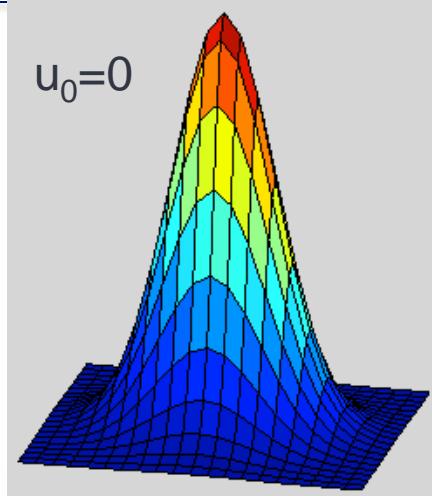




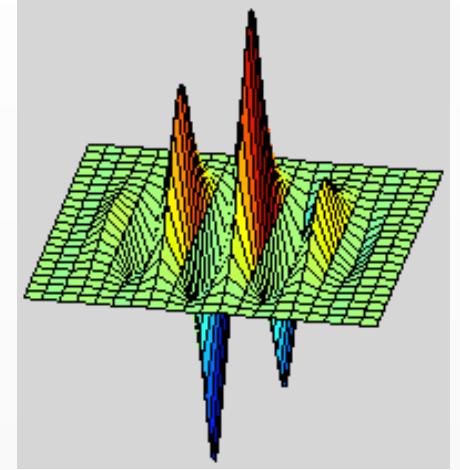
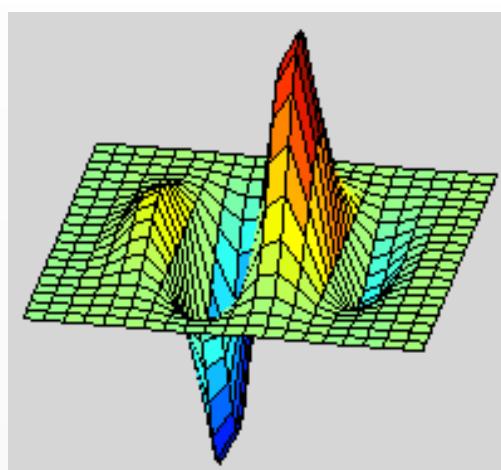


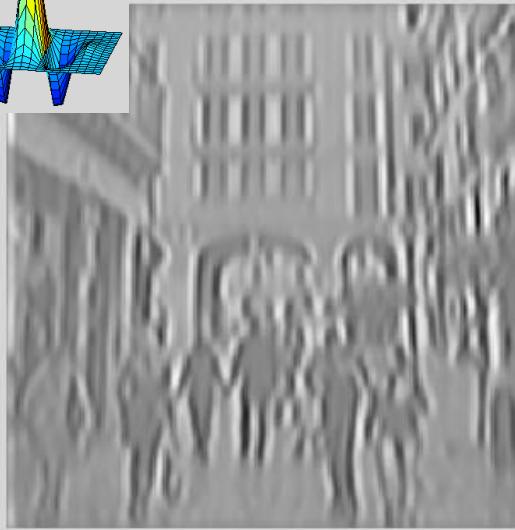
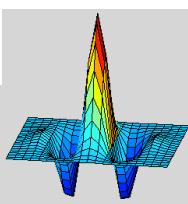
# 2D - Gabor wavelets

$$\psi_c(x,y) = e^{-\frac{x^2+y^2}{2\sigma^2}} \cos(2\pi u_0 x)$$

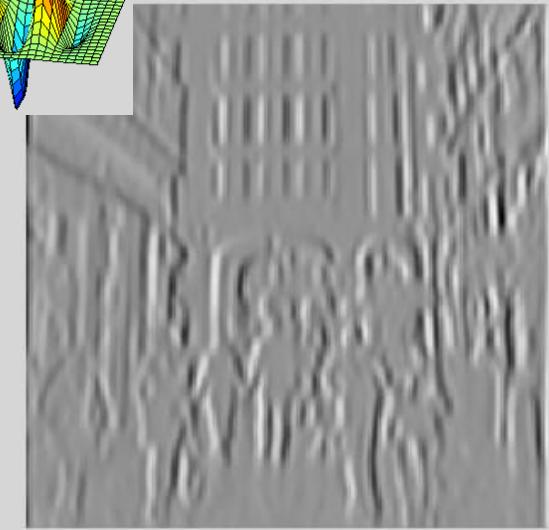
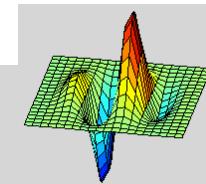


$$\psi_s(x,y) = e^{-\frac{x^2+y^2}{2\sigma^2}} \sin(2\pi u_0 x)$$

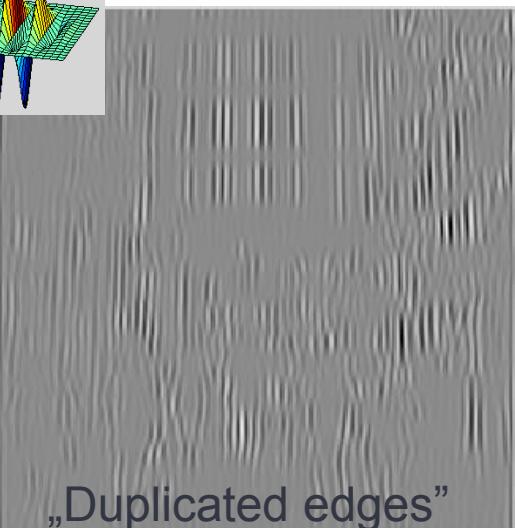
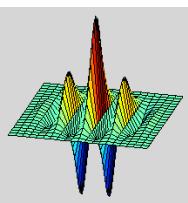




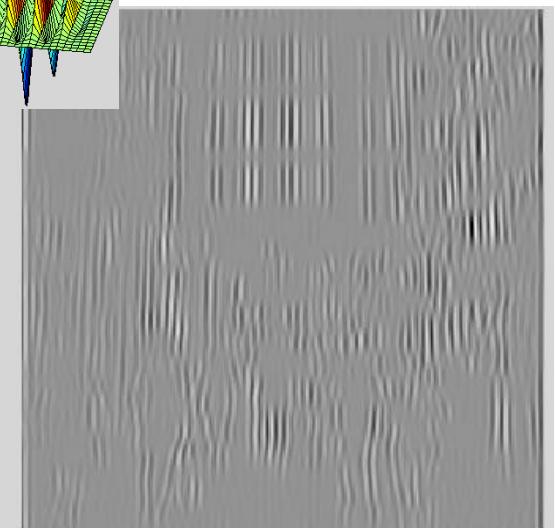
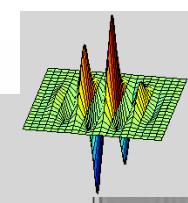
Salient bright thin vertical lines



Dark-bright transitions in horizontal direction



„Duplicated edges”

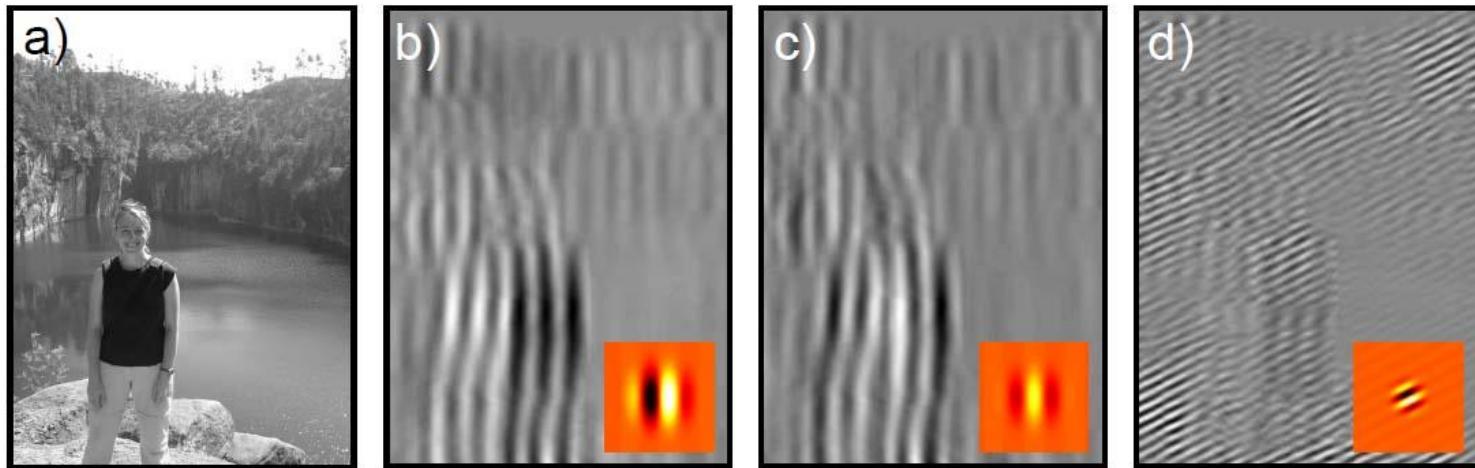


# Application of Gabor filters in image processing

---

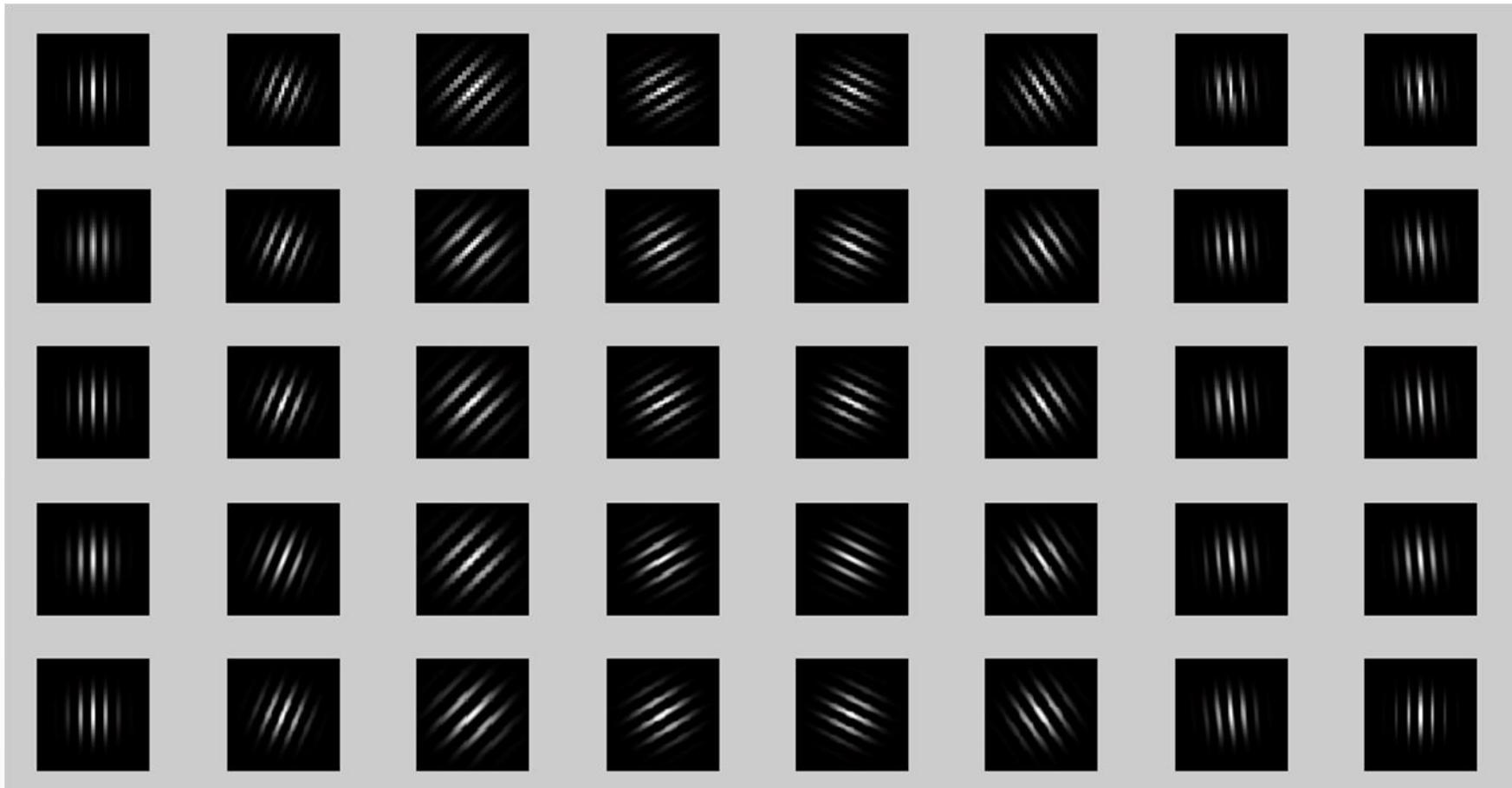
- Gabor filters can selectively highlight specific image elements according to their appropriately set frequency, orientation and phase parameters
- Invariant for additive changes of illumination (in case of asymmetric sinusoid functions)
- Motivation: vision mechanism of mammals – one of the first processing operations of visual stimuli in the brain

# Application of Gabor filters in image processing



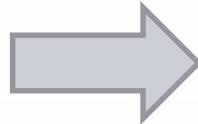
- a) Input image. b) Output of a low frequency, horizontal, asymmetric Gabor filter. c) Output of a low frequency, horizontal, symmetric Gabor filter d) Output of a diagonal Gabor filter

# Direction selective Gabor filter bank



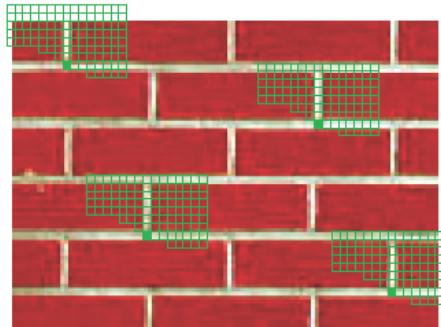
# Texture Synthesis

- Given a small sample, generate larger realistic versions of the texture

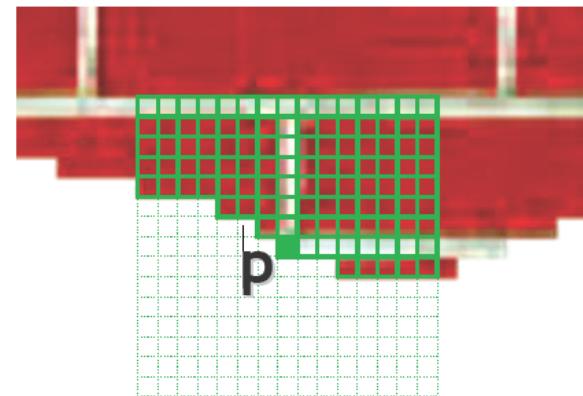


[Alexei A. Efros and Thomas K. Leung](#), “Texture Synthesis by Non-parametric Sampling,” Proc. International Conference on Computer Vision (ICCV), 1999.

# Synthesizing One Pixel



input image

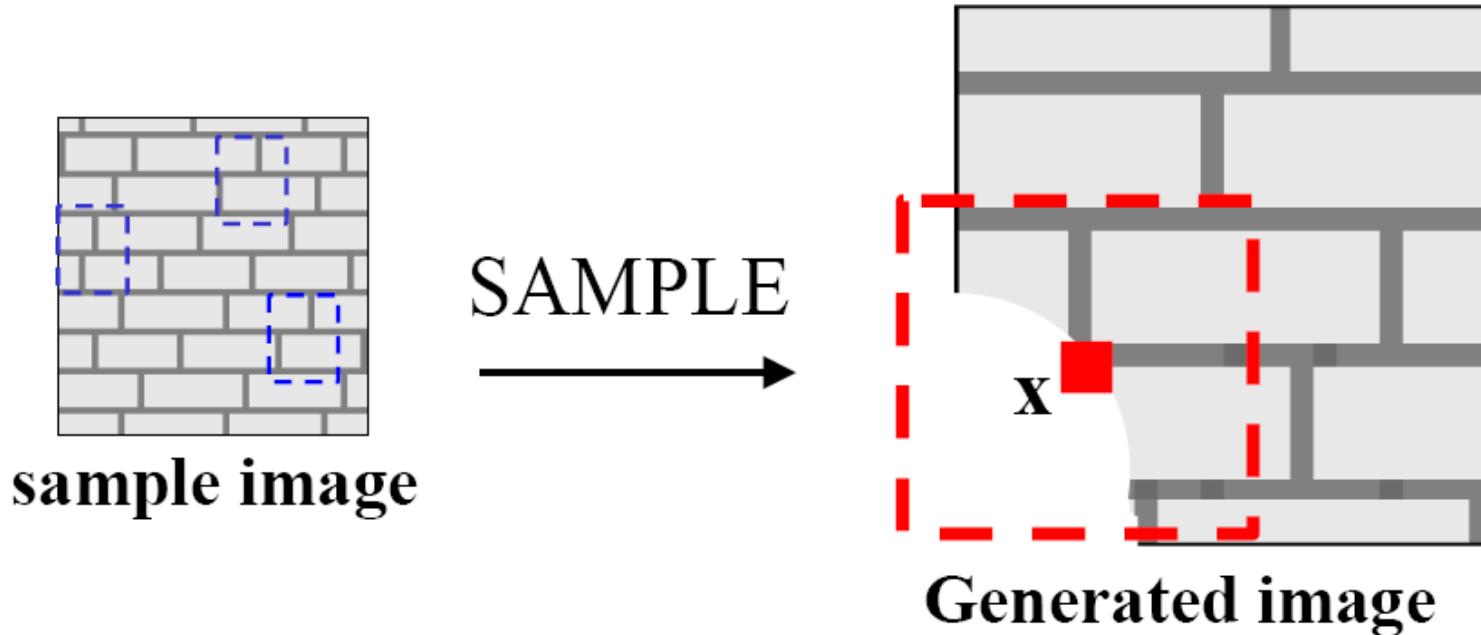


synthesized image

◎ What is  $P(x|\text{neighborhood of pixels around } x)$ ?

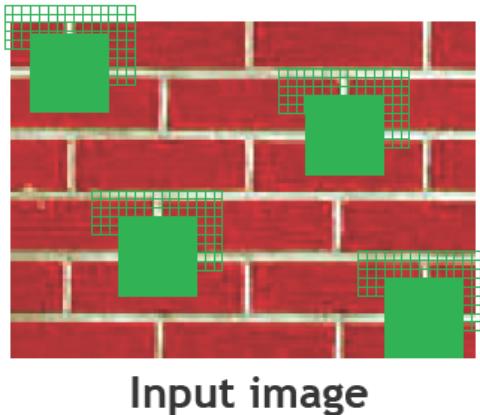
- Find all the windows in the image that match the neighborhood—consider only pixels in the neighbourhood that are already filled in
- To synthesize  $x$ 
  - pick one matching window at random
  - assign  $x$  to be the centerpixel of that window

# Really Synthesizing One Pixel

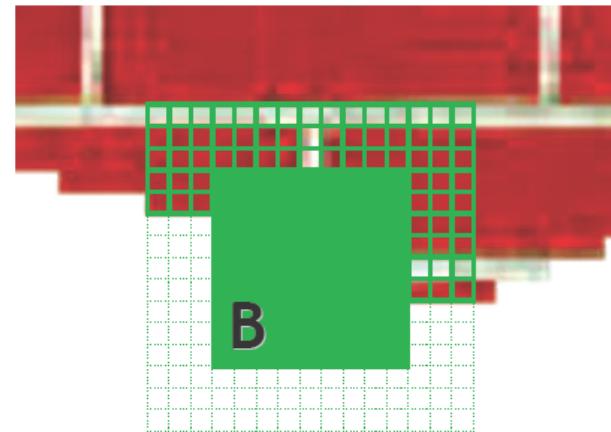


- An exact neighborhood match might not be present
- So we find the **best** matches using SSD error and randomly choose between them, preferring better matches with higher probability

# Block-based texture synthesis



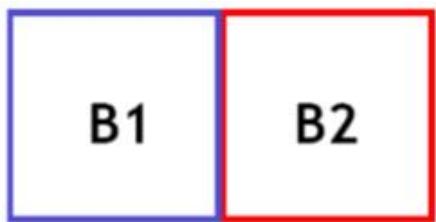
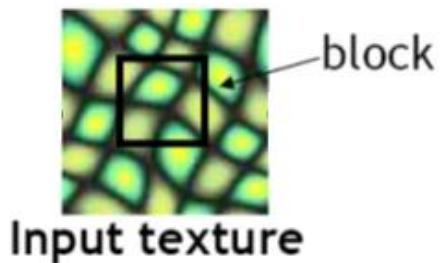
Input image



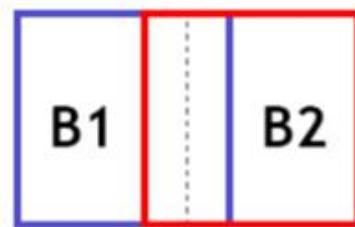
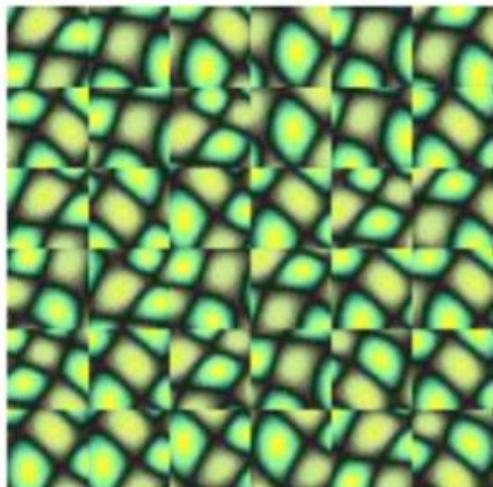
Synthesizing a block

- **Observation:** neighbor pixels are highly correlated
- **Idea:** unit of synthesis = block
  - Exactly the same but now we want  $P(B|N(B))$
  - Much faster: synthesize all pixels in a block at once

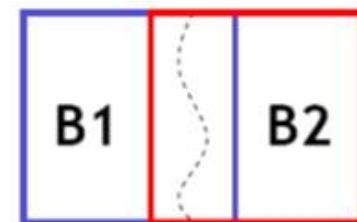
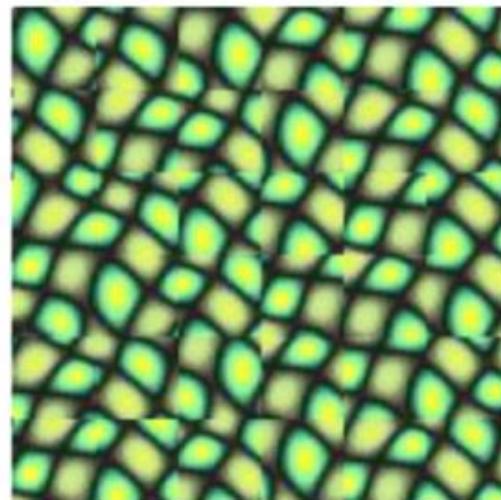
*Image Quilting for Texture Synthesis and Transfer*, Efros & Freeman, SIGGRAPH, 2001.



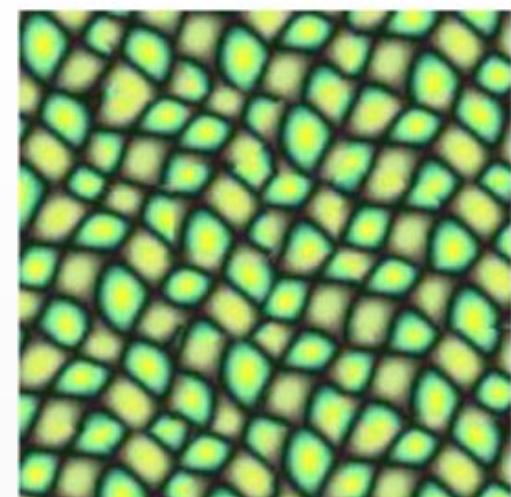
Random placement  
of blocks



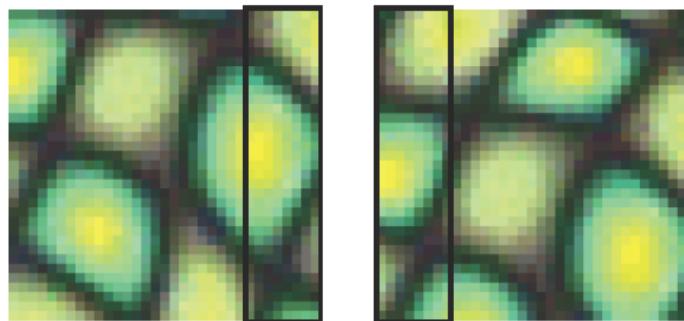
Neighboring blocks  
constrained by overlap



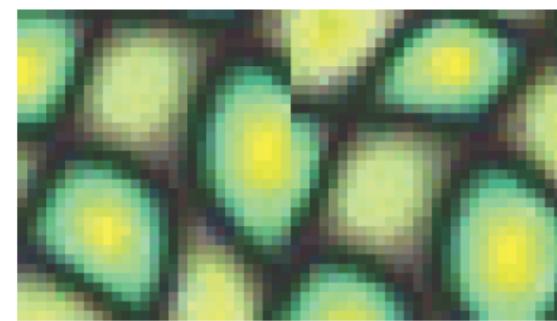
Minimal error  
boundary cut



**overlapping blocks**



**vertical boundary**

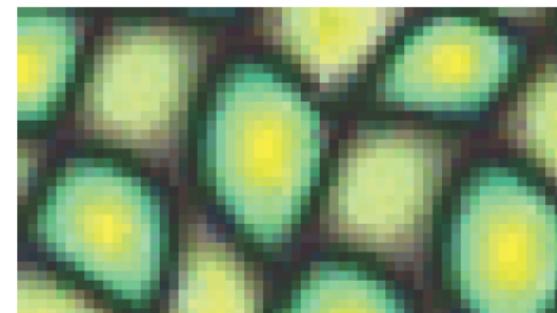


$$\left( \begin{array}{c|c} \text{block 1} & \text{block 2} \\ \hline \end{array} \right) - \left( \begin{array}{c|c} \text{block 1} & \text{block 2} \\ \hline \end{array} \right)^2 = \text{overlap error}$$

A diagram showing the calculation of overlap error. It consists of two 8x8 blocks labeled "block 1" and "block 2". The difference between them is calculated, and then squared. The result is a 1x8 pixel vector labeled "overlap error", which is mostly black with a red jagged line along its center representing the vertical boundary.

**overlap error**

**min. error boundary**



# Texture Transfer



Constraint



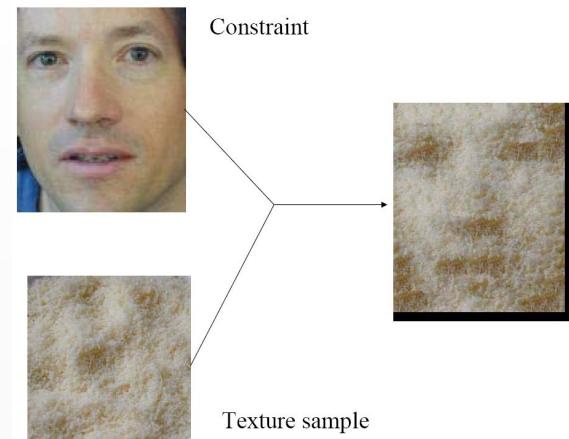
Texture sample



# Texture Transfer

- Each patch satisfy a desired correspondence map  $C$  as well as satisfy the texture synthesis requirements.
- $C$ : a spatial map of some corresponding quantity over both the texture source image and a controlling target image.
  - E.g. image intensity, blurred image intensity, local image orientation angles, or other derived quantities.

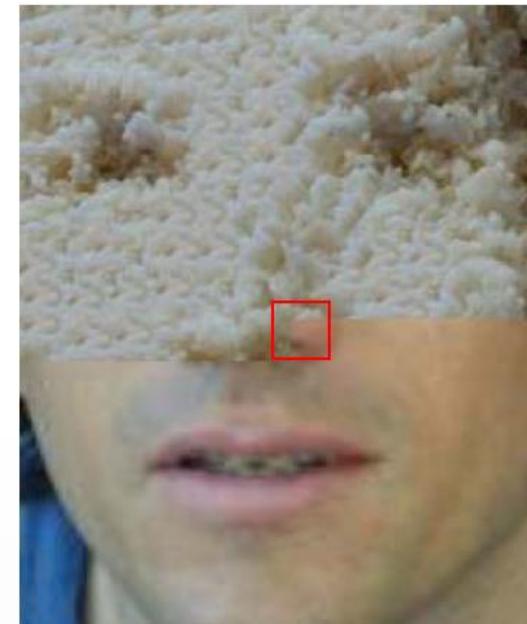
Here: bright patches of face and bright patches of rice are defined to have a low correspondence error.



# Texture Transfer

---

- Take the texture from one image and “paint” it onto another object

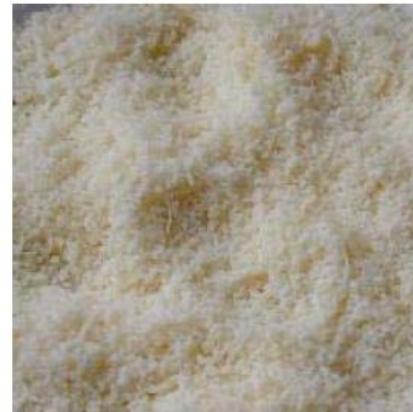


- Same algorithm as before with additional term
  - do texture synthesis on image1, create new image (size of image2)
  - add term to match intensity of image2

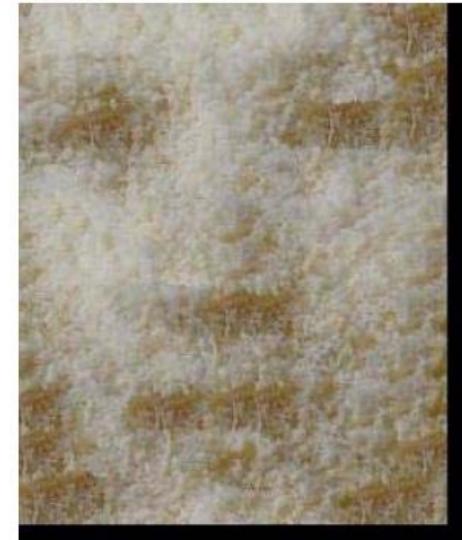
parmesan



+



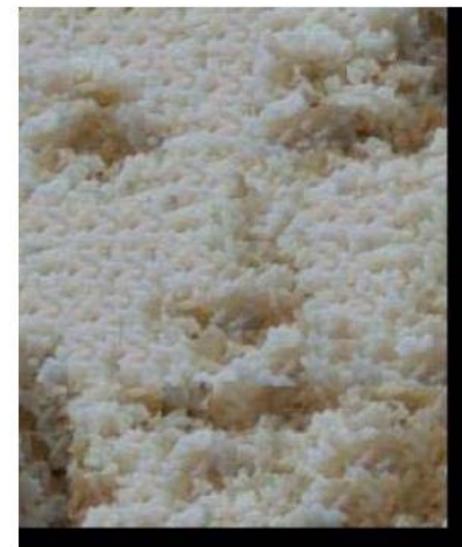
=



+



=



Target image



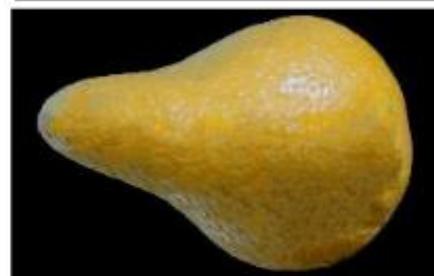
Source texture



Texture transfer result



# Texture transfer



# Texture Databases

---

- ◉ Categorize them in to four areas

- Texture databases in *medical* imaging
- *Natural* texture image database
- Texture of *materials* database
- *Dynamic* texture database

# Databases: Various Properties

---

- Image size
- No. of classes
- No. of images
- Gray-scale vs. color image
- Image rotation
- Illumination – static/varied; indoor/outdoor
- Camera/sensors
- Image depth/distance from the camera

# Some Key Databases

---

- Brodatz texture database
- MRI brain database
- USF Digital database for screening mammography
- Vision texture database (VisTex)
- USC-SIPI texture database
- PhoTex database
- ALOT database
- UMD dataset
- CUReT database
- UIUC database
- KTH-TIPS database
- UCLA dynamic database
- DynTex database
- MIT Szummer database

# Natural textures - Brodatz

---

- Phil Brodatz,

- "Textures: A Photographic Album for Artists and Designers",
- „Land Sea and Skies: A Photographic Album for Artists and Designers",
- „Wood and Wood Grains: A Photographic Album for Artists and Designers",
- „The Elements of Landscape: A Photographic Handbook for Artists"