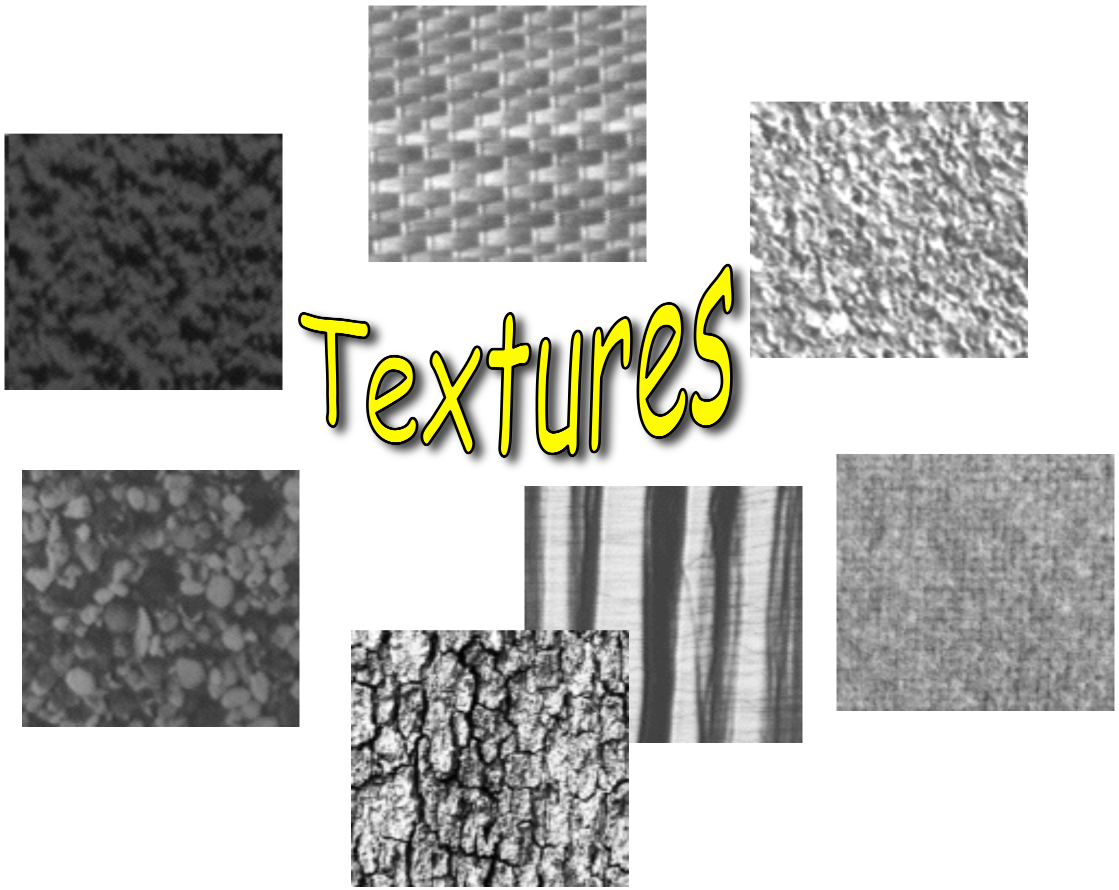
Wavelet Applications

Texture analysis&synthesis

Wavelet based IP

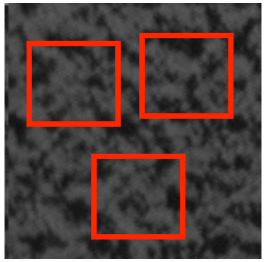
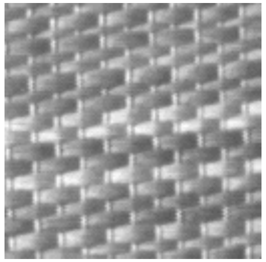
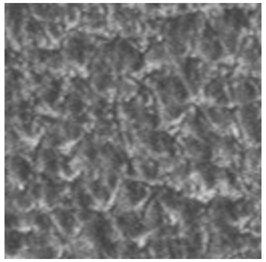
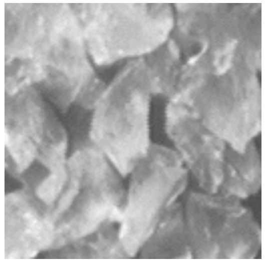
- Compression and Coding
  - The good approximation properties of wavelets allow to represent reasonably smooth signals with few non-zero coefficients → efficient wavelet based coding systems
  - DWT (critically sampled)
  - Among the most famous are
    - Embedded Zerotree Wavelet (EZW)
    - Layered Zero (LZ) Coding
    - Embedded Block Coding (EBCOT)
- Image denoising
- Image quality assessment

- Signal analysis
  - The good spatial and frequency domain localization properties make wavelet a powerful tool for characterizing signals
  - DWF (overcomplete)
  - Feature extraction
- ❑ Pattern recognition
  - Identification of structures in natural images
    - Curvelets, ridgelets
  - Identification of textures
    - Classification
    - Segmentation


Textures

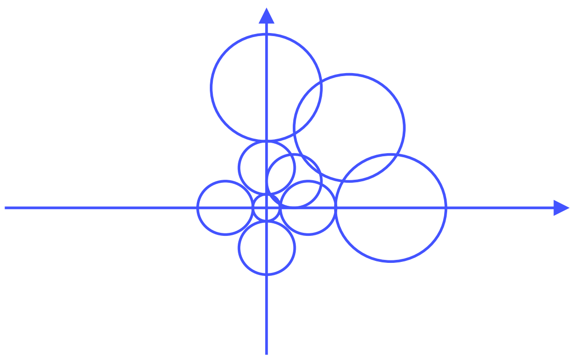
What is texture?

- No agreed reference definition
  - Texture is property of areas
  - Involves spatial distributions of grey levels
  - A region is perceived as a texture if the number of primitives in the field of view is sufficiently high
  - Invariance to translations
  - Macroscopic visual attributes
    - uniformity, roughness, coarseness, regularity, directionality, frequency [Rao-96]
  - Sliding window paradigm


Feature extraction for texture analysis

- Statistical methods
  - Textures as realizations of an underlying stochastic process
    - Spatial distributions of grey levels
  - Statistical descriptors
    - Subband histograms, co-occurrence matrices, autocorrelation, n-th order moments, MRFs...
  - A-priori assumptions
    - locality, *stationarity*, spatial *ergodicity*, parametric form for the pdf (Gaussian)

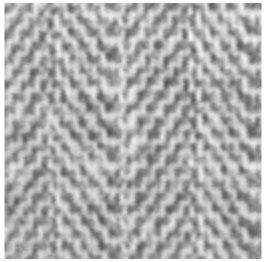


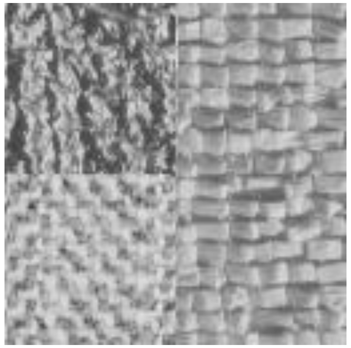
- Structural methods
  - Texture as sets of geometric structures
  - Descriptors
    - primitives+placement rules
  - Suited for highly regular textures

- Multi-scale methods
  - Combined with statistical methods
  - Models of early visual processes
  - Multi-resolution analysis (wavelet based)
    - Gabor wavelets are *optimal* as they have maximum resolution in space *and* frequency

Texture analysis

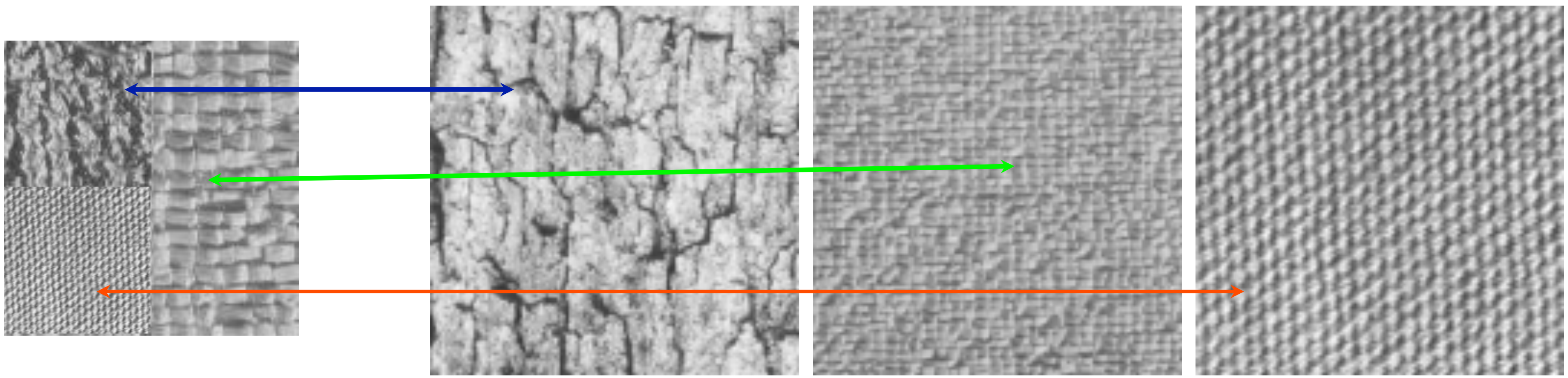
- Texture segmentation
  - Spatial localization of the different textures that are present in an image
  - Does not imply texture recognition (classification)
  - The textures do not need to be *structurally* different
  - *Apparent* edges
    - Do not correspond to a discontinuity in the luminance function
    - Texture segmentation → Texture segregation
  - *Complex* or *higher-order* texture channels





Texture analysis

- Texture classification (recognition)
  - Hypothesis: textures pertaining to the same class have the same visual appearance → the same *perceptual features*
  - Identification of the class the considered texture belongs to within a given set of classes
  - Implies texture recognition
  - The classification of different textures within a composite image results in a segmentation map



Texture Classification

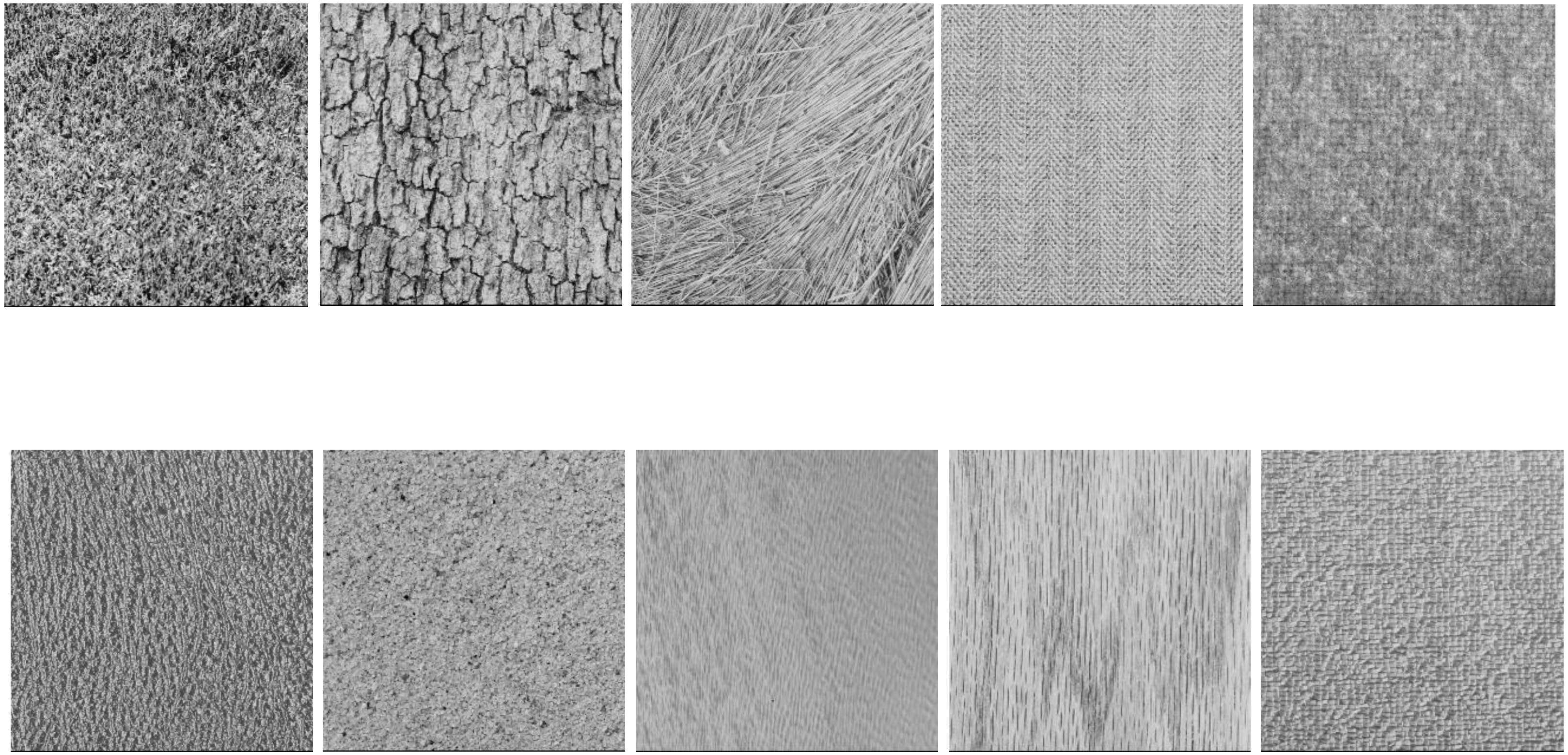
- Problem statement
  – Given a set of classes $\{\omega_i, i=1,...N\}$ and a set of observations $\{\boldsymbol{x}_{i,k}, k=1,...M\}$ determine the most probable class, given the observations. This is the class that maximizes the conditional probability:

$$\omega_{winner} = \max_k P(\omega_i | x_k)$$

- Method
  – Describe the texture by some *features* which are related to its *appearance*
    - Texture → class → ω_k
    - Subband statistics → Feature Vectors (FV) → $\boldsymbol{x}_{i,k}$
  – Define a distance measure for FV
    - Should reflect the *perceived similarity/dissimilarity* among textures (**unsolved**)
  – Choose a *classification rule*
    - Recipe for comparing FV and choose 'the winner class'
  – Assign the considered texture sample to the class which is the *closest* in the feature space

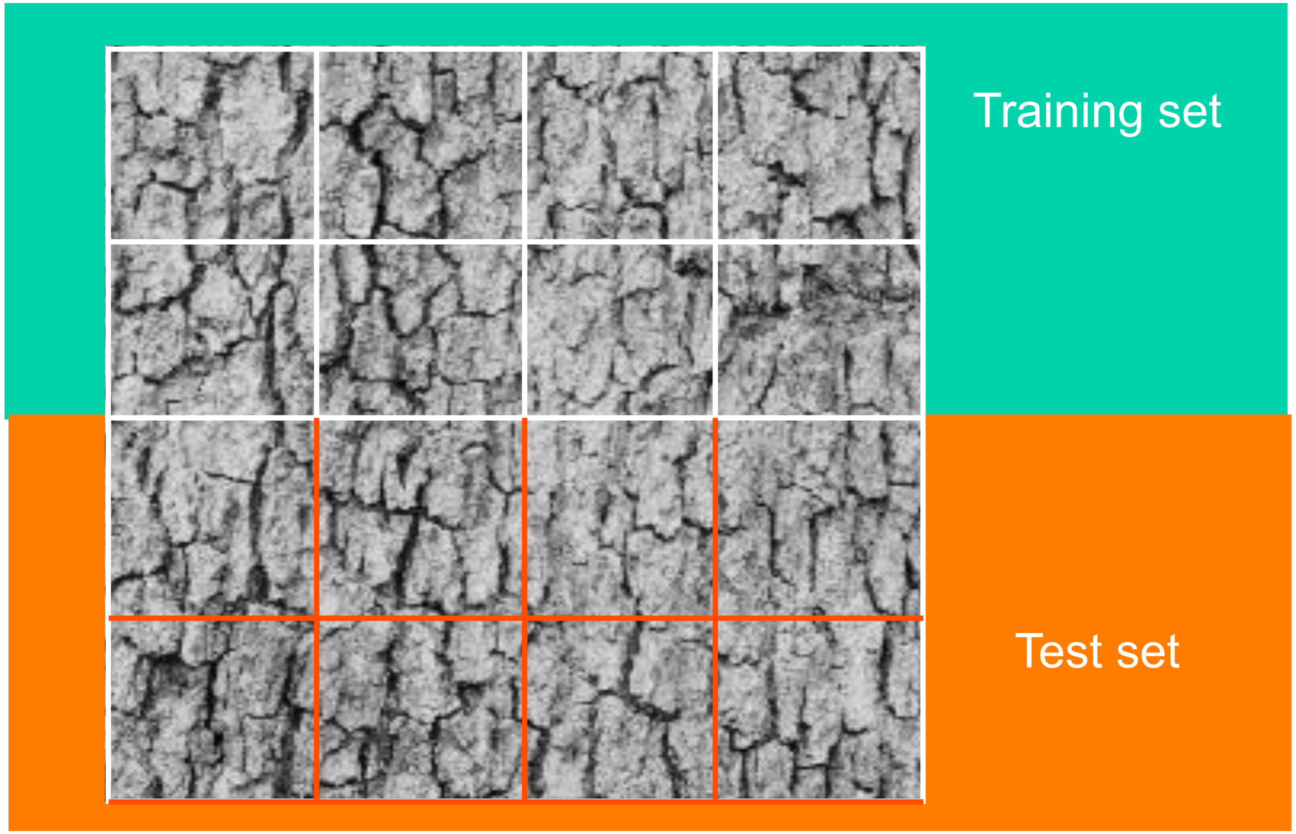
Exemple: texture classes

ω_1 ω_2 ω_3 ω_4



FV extraction

- Step 1: create independent texture instances



Feature extraction

- Step 2: extract features to form *feature vectors*

Intensity image

↓

DWT/DWF

↓

For each sub-image

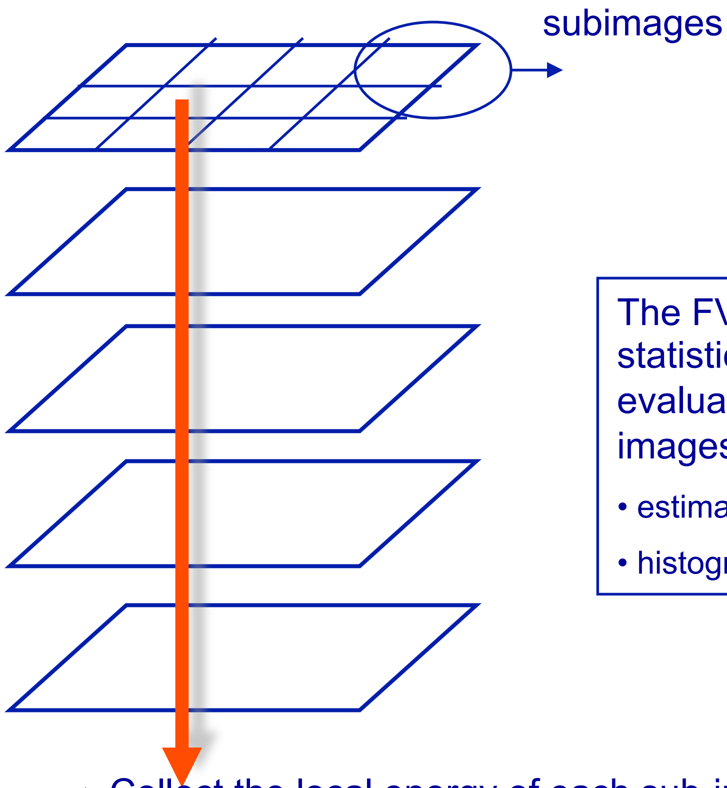
For each subband

Calculate the local energy (variance)

Fill the corresponding position in the FV

↓

One FV for each sub-image ⇨ Classification algorithm

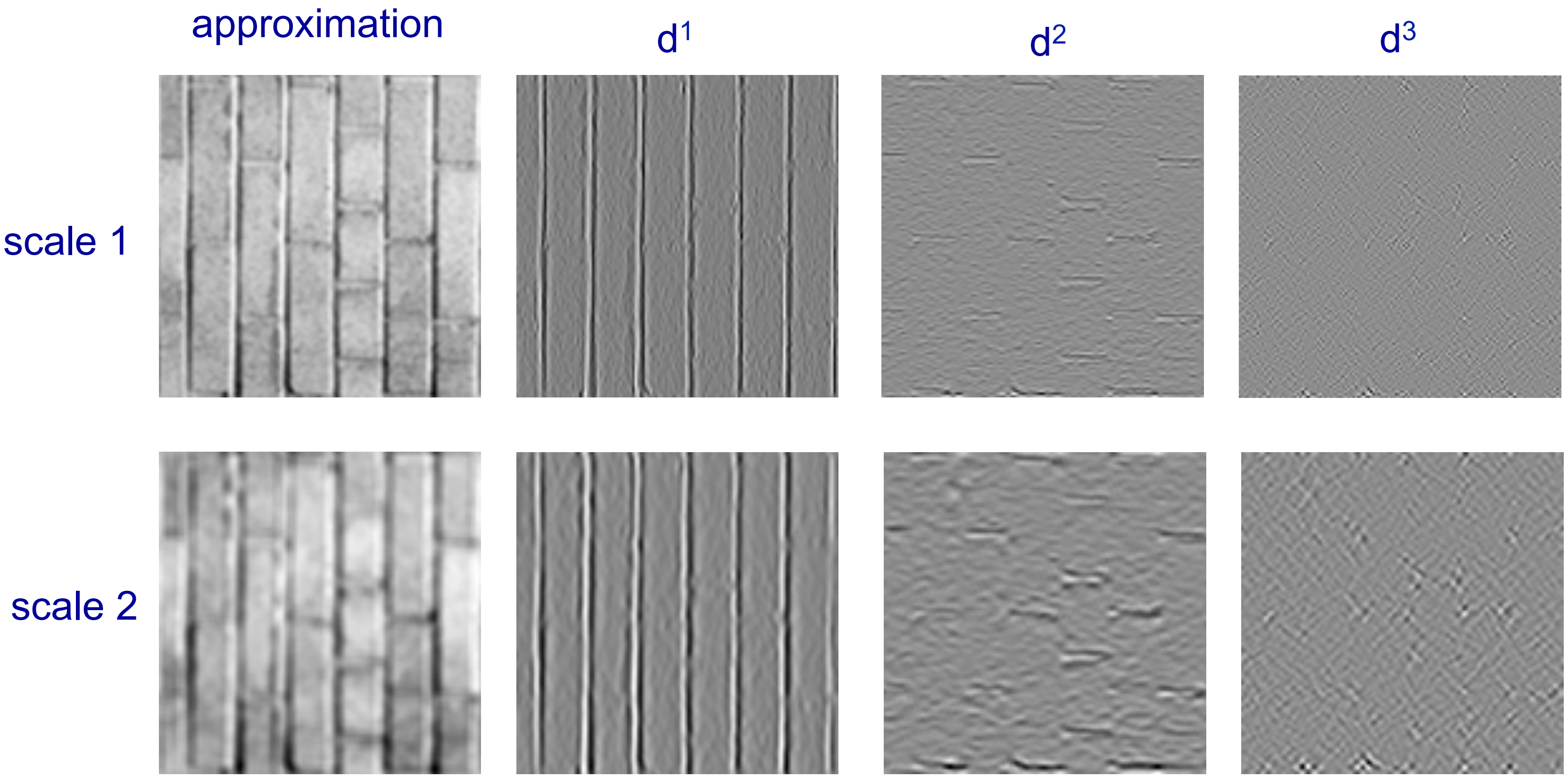


The FVs contain some statistical parameters evaluated on the subband images

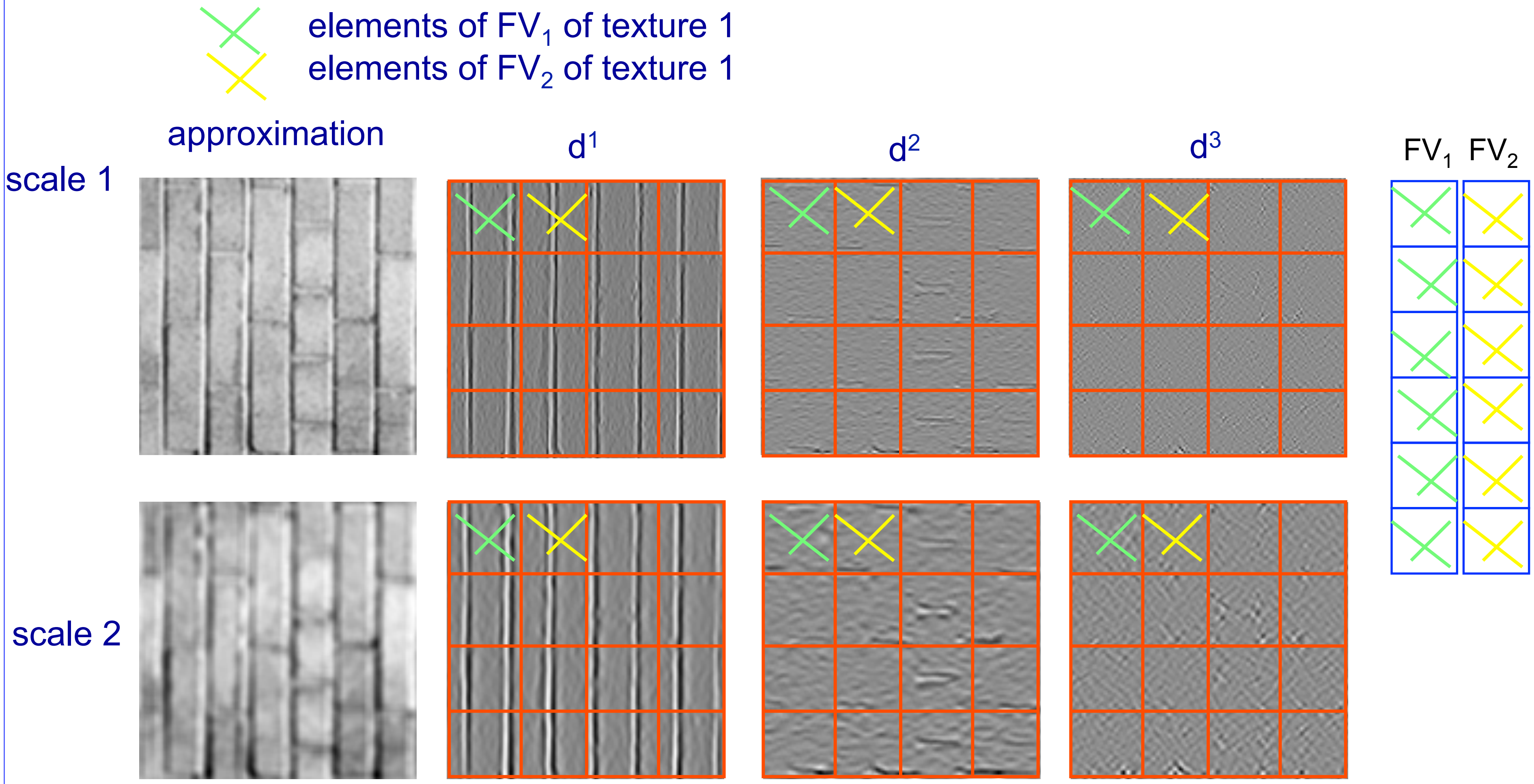
- estimates of local variances
- histograms

⇒ Collect the local energy of each sub-image in the different subbands in a vector

Building the FV

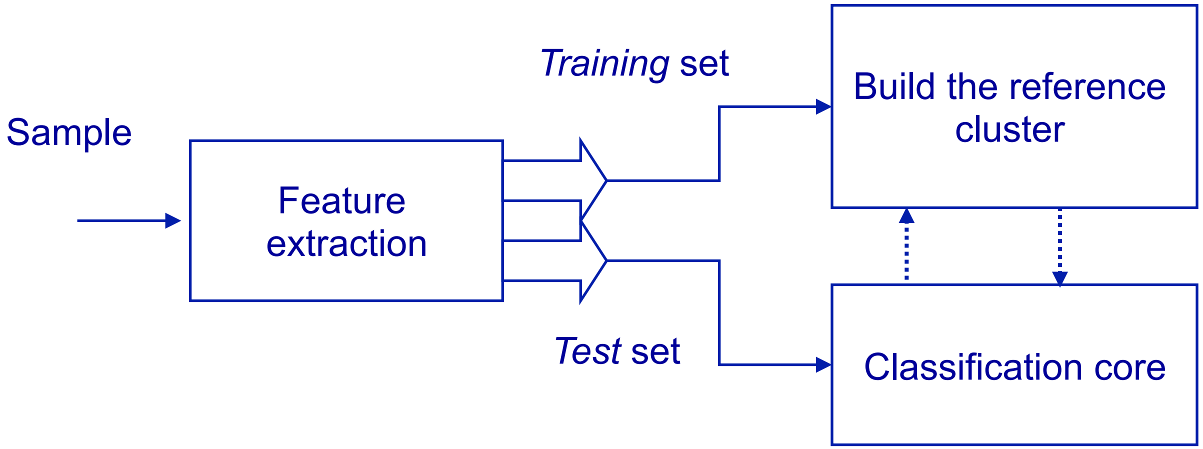


Building the FV



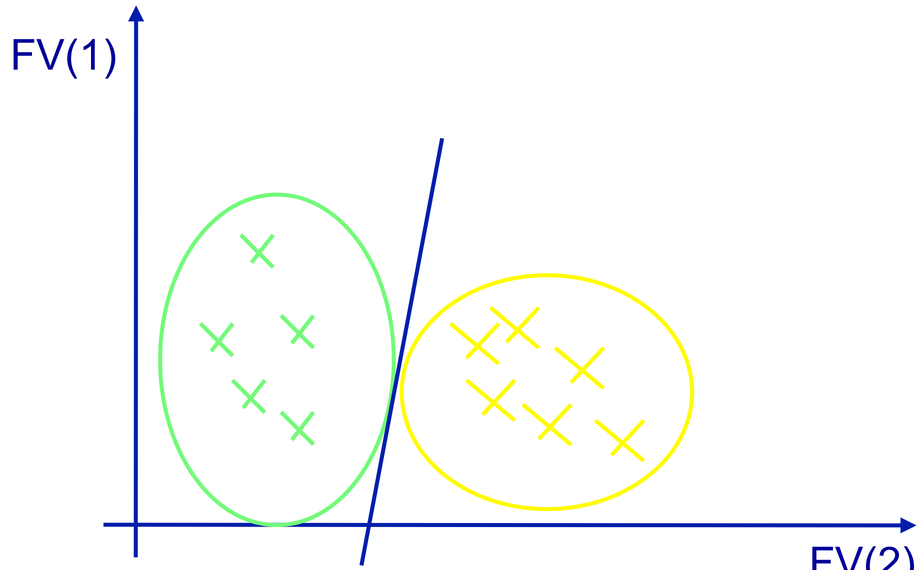
Implementation

- Step 1: Training
  - The classification algorithm is provided with many examples of each texture class in order to build clusters in the feature space which are representative of each class
    - Examples are sets of FV for each texture class
    - Clusters are formed by aggregating vectors according to their "distance"
- Step 2: Test
  - The algorithm is fed with an example of texture ω_i (vector $\boldsymbol{x}_{i,k}$) and determines which class it belongs as the one which is "closest"

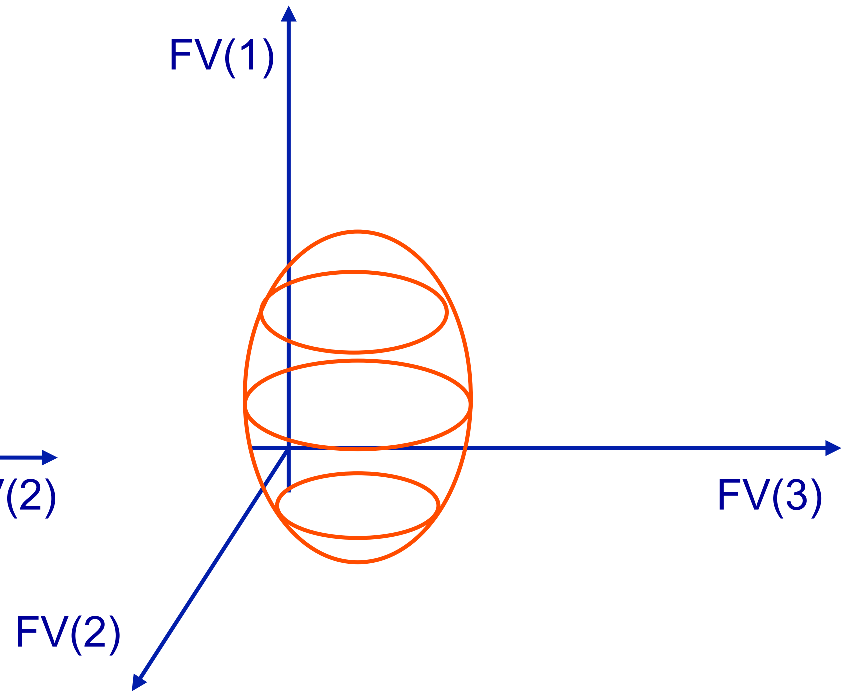


Clustering in the Feature Space

Bi-dimensional feature space (FV of size 2)



Multi-dimensional feature space



FV classification: identification of the cluster which best represents the vector according to the chosen distance measure

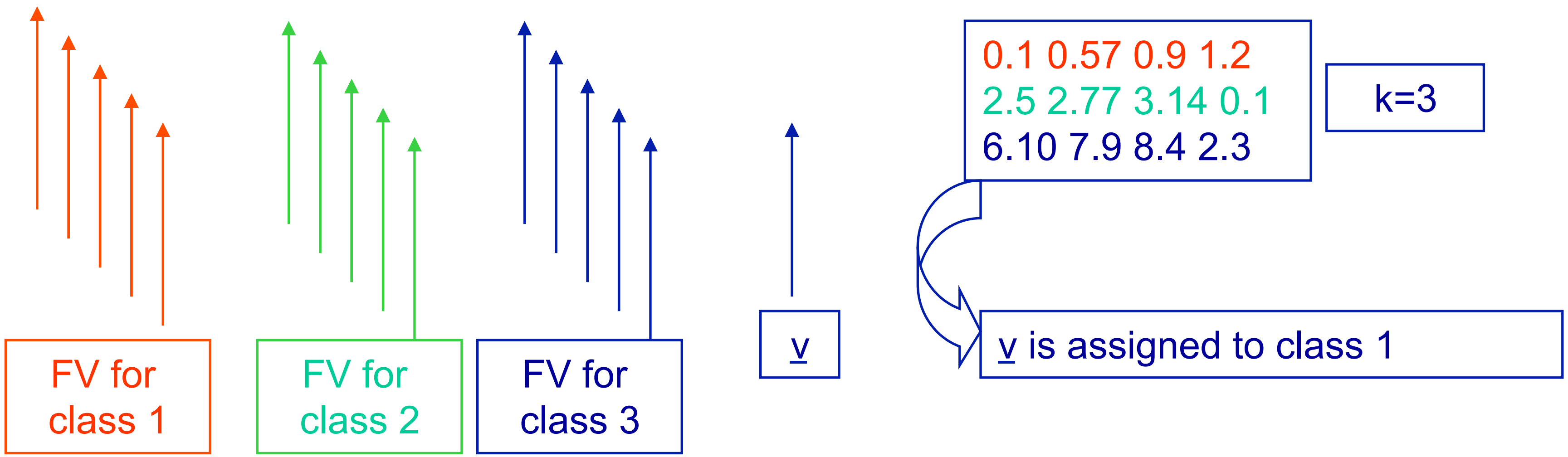
Classification algorithms

- Measuring the distance among a *class* and a *vector*
  - Each class (set of vectors) is represented by the mean (m) vector and the vector of the variances (s) of its components ⇒ the training set is used to build m and s
  - The distance is taken between the test vector and the m vector of each class
  - The test vector is assigned to the class to which it is closest
    - Euclidean classifier
    - Weighted Euclidean classifier

- Measuring the distance among *every couple* of vectors
  - kNN classifier

kNN classifier

- Given a vector v of the test set
  - Take the distance between the vector v and ALL the vectors of the training set
  - (while calculating) keep the k smallest distances and keep track of the class they correspond to
  - Assign v to the class which is most represented in the set of the k smallest distances



Confusion matrix

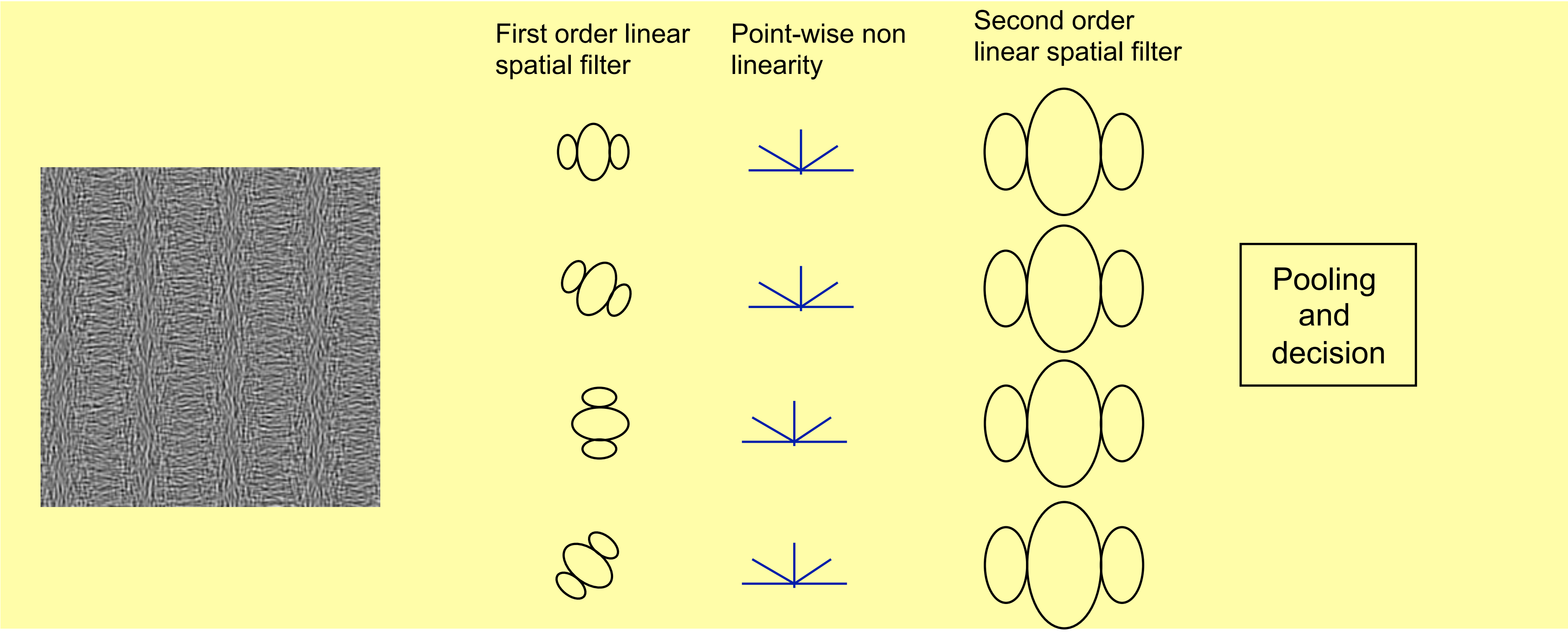
textures	1	2	3	4	5	6	7	8	9	10	% correct
1	841	0	0	0	0	0	0	0	0	0	100.00%
2	0	840	1	0	0	0	0	0	0	0	99.88%
3	2	0	839	0	0	0	0	0	0	0	99.76%
4	0	0	0	841	0	0	0	0	0	0	100.00%
5	0	0	88	0	753	0	0	0	0	0	89.54%
6	0	0	134	0	0	707	0	0	0	0	84.07%
7	0	66	284	0	0	0	491	0	0	0	58.38%
8	0	0	58	0	0	0	0	783	0	0	93.10%
9	0	0	71	0	0	0	0	0	770	0	91.56%
10	0	4	4	0	0	0	0	0	0	833	99.05%
				Average recognition rate							91.53%

Texture Segmentation

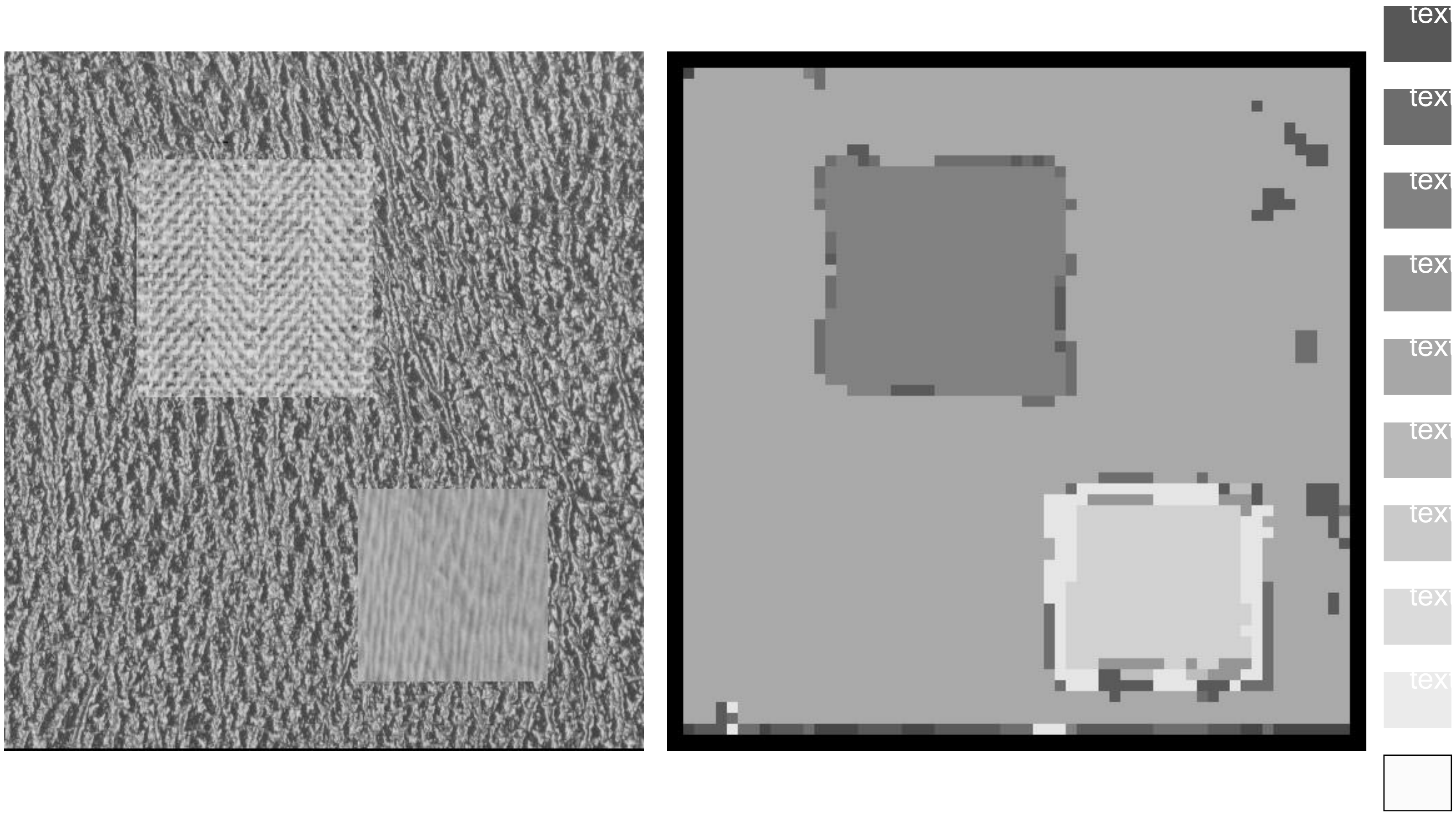
- Problem statement
  - Given an image, identify the *regions* characterized by different features
- How?
  - Same approach used for classification
  - Key difference: focus on *feature gradients*, namely local discontinuities in feature space represented by *differences* in feature vectors
    - If feature vectors are collections of local variances, it is the difference in such a parameter that is assumed to reveal the presence of an apparent edge
- Noteworthy
  - More in general, segmentation is based on image interpretation, which is very difficult to model
  - Often “supervised”
  - Tailored on the application: no golden rule for segmentation!
  - Key point: image interpretation and semantics

Relation to complex texture channels

- Model for pre-attentive texture segregation
  - LNL (linear-non linear-linear) model
    - The idea is to detect low spatial frequency features of high spatial frequency first-stage responses [Landi&Oruc 2002]



Example of a segmentation map



Texture synthesis

- Define a generative model to create new textures having the same *visual appearance* of the original one

- Stochastic methods
  - Reproduce statistical descriptors
    - Co-occurrence matrices, autocorrelation features, MRF
    - Very natural
    - Could require parameter estimation
    - Usually high computational cost

- Structural methods
  - Crystal growth
    - Highly structured and regular textures

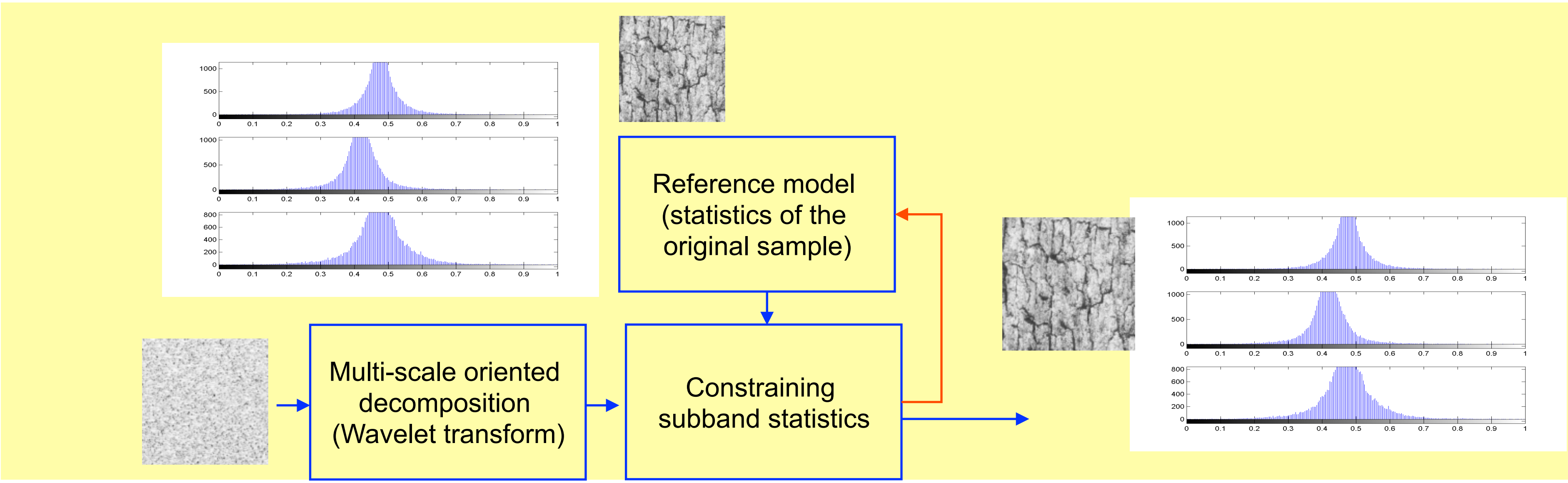
- Multi-scale methods
  - Reproduce *Intra-band* and *Inter-band* relationships among subband coefficients
    - pixel statistics,subband marginals and covariance, subband joint distributions
  - *Explicit* or *Implicit*
  - Suitable for both natural and artificial structured textures

Recipe for perceptual texture synthesis

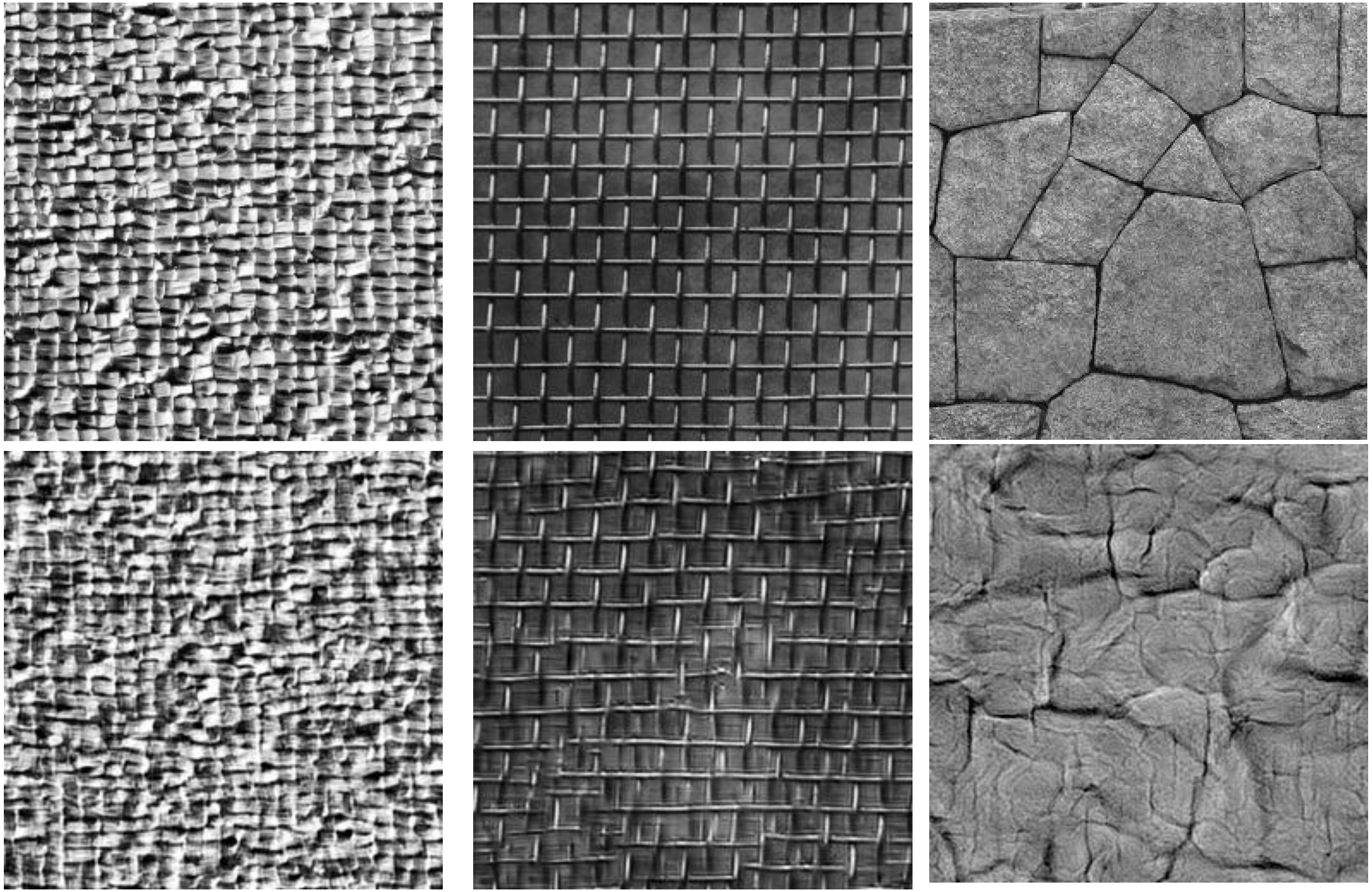
- Consider the image as a realization of an underlying stochastic process
- Define a stochastic model for the stimulus as well as criterion for sampling from the corresponding distribution and generating a new realization
- Possible approaches
  - Parametric techniques: explicit constraining of statistical parameters
    - Filters Random fields And Maximum Entropy (FRAME) model [Zhu&Mumford-05]
    - Constraining Joint statistics of subband coefficients [Portilla&Simoncelli-00]
  - Non parametric techniques
    - Multi-resolution probabilistic texture modeling [De Bonet-97]
    - DWT based non parametric texture synthesis [Menegaz-01]

Portilla&Simoncelli

- Statistical parameters
  - Marginal and joint subband statistics
    - Variance and other 2^{nd} order moments
  - Auto and mutual correlations between subbands
  - Magnitude correlations → non-linearity
  - Self and mutual correlations between *phase* images

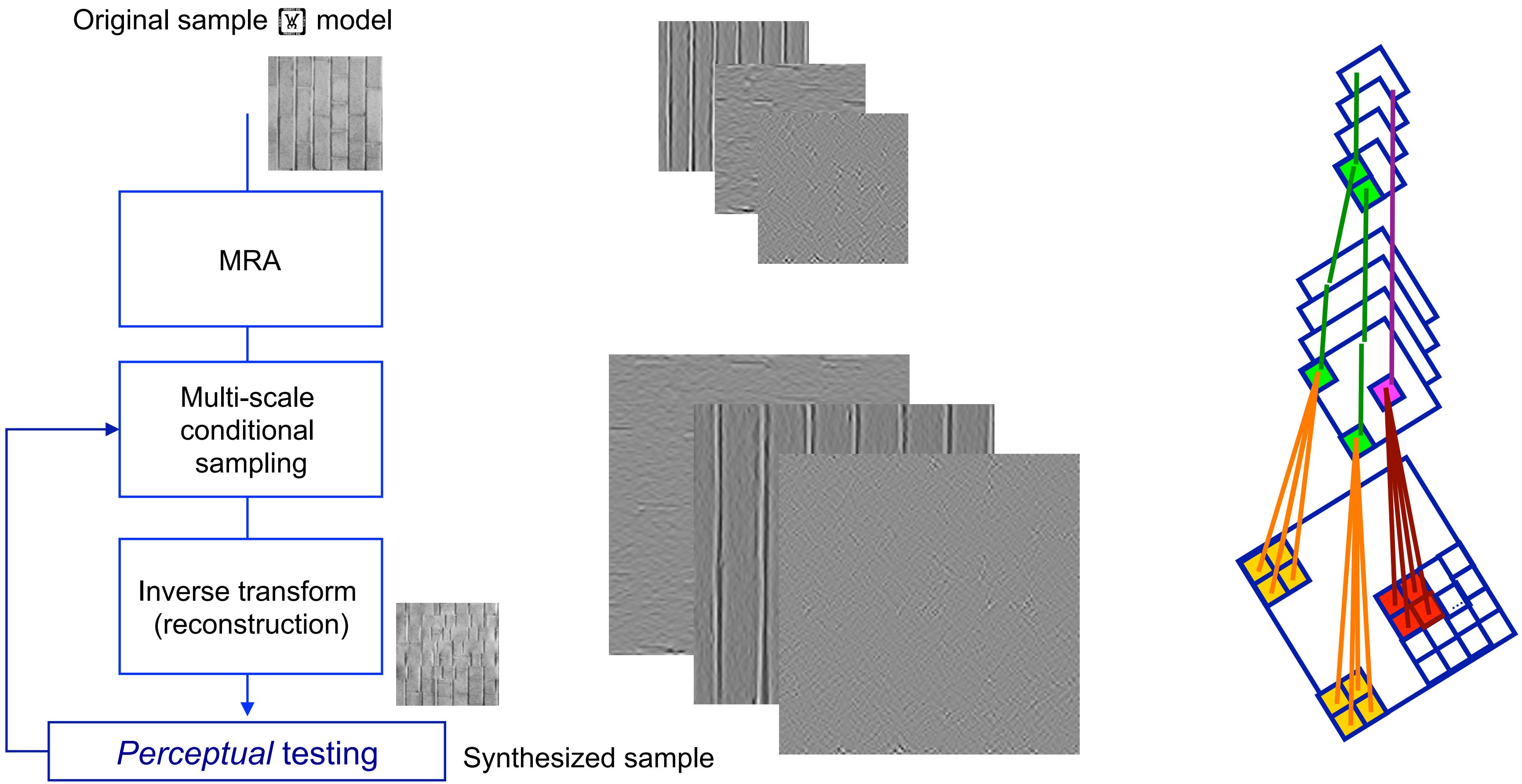


Portilla&Simoncelli

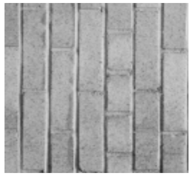


DWT based texture synthesis

- Controlled shuffling of hierarchies of wavelet coefficients



Multiresolution Probabilistic TM



original sample

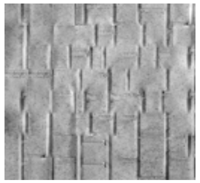
Perceptual decomposition

Statistical modeling

Sampling

Reconstruction

synthesized sample



Perceptual testing

Multiscale orientation-selective decomposition mimicking the neural responses to the visual stimulus

Non-parametric constraining of the joint distributions of subband coefficients. Inter-band dependencies among subbands with same orientation at different scales are preserved by multiscale conditional sampling

The appearance of a subband coefficient at a given scale and orientation is conditioned to the appearance of its ancestors at all coarser scales → *parent vector*

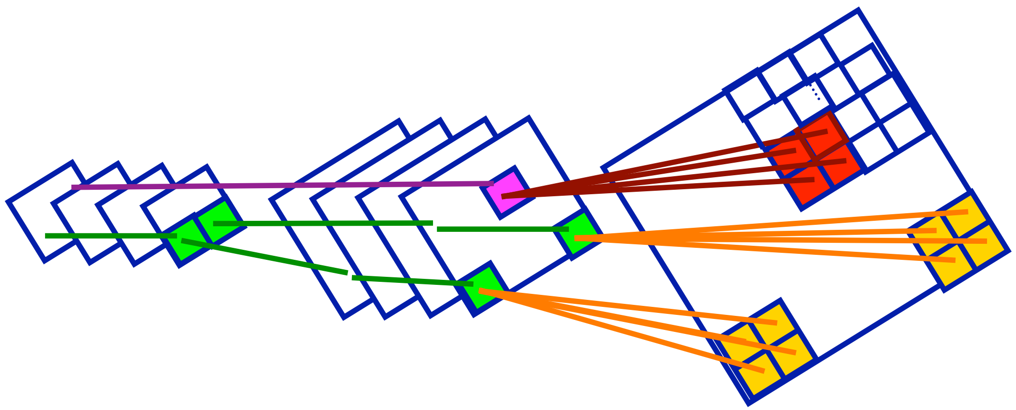
The synthesis pyramid is filled by sampling from the analysis pyramid, and is then collapsed to get the synthetic image

If the resulting texture is not satisfying, the procedure is repeated with different model parameters

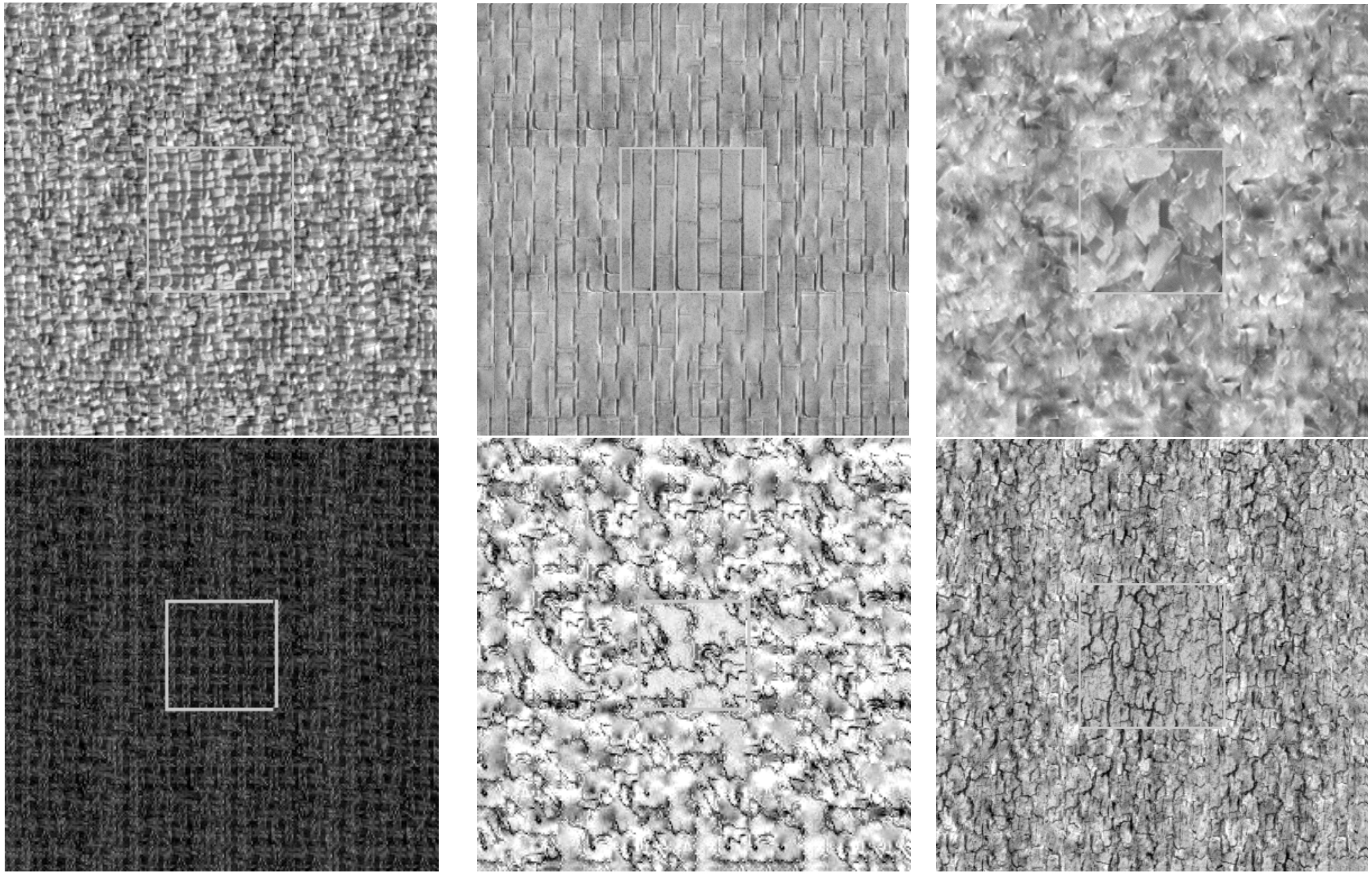
Formally

Feature Vector (M: #feature images, N:# of levels):

$$
\begin{aligned}
\vec{V}(x,y) = [&F_0^0(x,y), F_0^1(x,y), \ldots, F_0^M(x,y),\\
&F_1^0\left(\frac{x}{2},\frac{y}{2}\right), F_1^1\left(\frac{x}{2},\frac{y}{2}\right), \ldots, F_1^M\left(\frac{x}{2},\frac{y}{2}\right), \ldots,\\
&F_N^0\left(\frac{x}{2^N},\frac{y}{2^N}\right), F_N^1\left(\frac{x}{2^N},\frac{y}{2^N}\right), \ldots, F_N^M\left(\frac{x}{2^N},\frac{y}{2^N}\right)]
\end{aligned}
$$



MPTM results

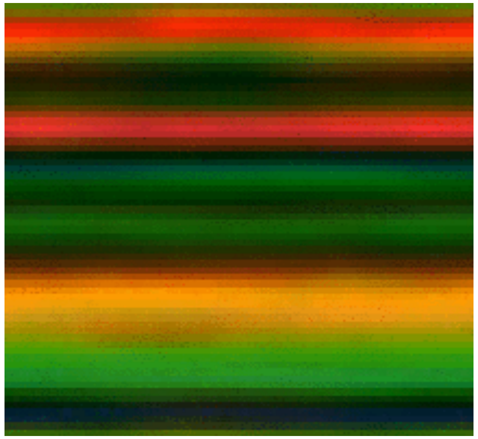


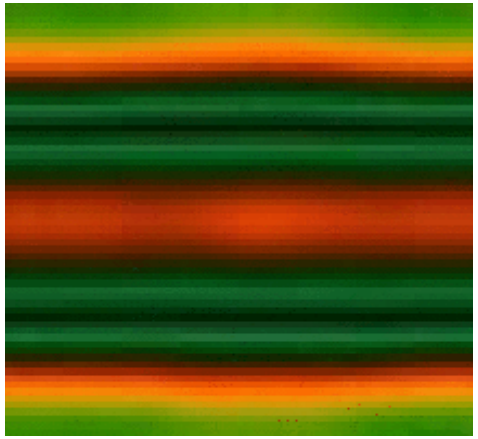
Generalization to 2D+1 Textures

- 2D+1 textures are meant as the result of the observation of a realization of a stochastic 2D process by a moving observer
  - Temporal features are due to the change of the observation point of view
  - Key point: preserve the temporal relation between successive images in the sequence
  - Major issue: define a *growing rule* for subband regions simulating any displacement in image space
- Hypothesis
  - The motion is given
  - The trajectory is piece-wise linear
- Guideline
  - Integrate the motion information within the DWT-based Multiresolution Probabilistic Texture Modeling (MPTM) algorithm [Menegaz-00]
- Advantages
  - Suitable for the integration in a coding system
  - Low complexity → running in real time

Color Textures

Textures ⇔ Color distributions with a *spatial* structure





- When different color distributions are perceptually equivalent?
- How do *texture* and *color* interact?

On going research