

## Computer Vision Week 9

Feature Extraction:  
Texture features, shape features,  
and boundary features

### What we would like to be able to do ...

- Visual recognition and scene understanding
- What is in the image and where



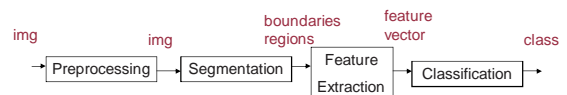
- scene type: outdoor, city ...
- object classes
- material properties
- actions

### Various tasks

- Image classification:  
"the image contains an airplane"
- Object detection/localization:  
"the object is here, in a bounding box"
- Object pixel-level segmentation:  
"the object 'owns' these pixels"



### Feature Extraction



- Note that this order is not always necessary: Segmentation may be omitted
- Regions: may be represented as binary masks
- Boundaries: border between regions
- Textures

### Region Features

- Regions may be represented as binary masks.
- Therefore, binary image features may be used for region representation

### Shape descriptors

- Simple geometric features
- Projections
- Moments
- How good is a feature? It should be:
  - Translation invariant
  - Scale invariant
  - Rotation invariant

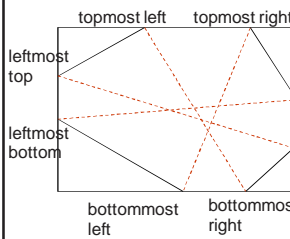
### Simple Geometric Features

- Area:  $A = \sum \sum B[i,j]$
- Center of mass:  $\bar{x} = \frac{\sum \sum jB[i,j]}{A}$      $\bar{y} = \frac{\sum \sum iB[i,j]}{A}$
- Perimeter length
- Circularity:  $C = \frac{|P|^2}{A}$
- Better measure for circularity (Haralick):

$$C_2 = \frac{\text{mean radial distance}}{\text{std. dev. of mean radial distance}}$$

### Simple Geometric Features

- Bounding box and extremal points



4 extremal axes  
 maximal axis  
 minimal axis  
 elongation:  $\frac{|\text{max axis}|}{|\text{min axis}|}$

### Projections

- Horizontal projection:  $H[i] = \sum_{j=1}^m B[i,j]$
- Vertical projection  $V[i] = \sum_{i=1}^m B[i,j]$

|   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 1 | 1 | 0 | 3 |
| 0 | 1 | 1 | 1 | 1 | 0 | 4 |
| 0 | 0 | 0 | 1 | 1 | 0 | 2 |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 | 0 | 0 | 2 |
| 0 | 1 | 1 | 0 | 0 | 0 | 2 |
| 0 | 3 | 4 | 4 | 3 | 0 |   |

Projections at arbitrary directions may also be defined

### Moments

- Central moments:  $\mu_{jk} = \sum \sum (x-\bar{x})^j (y-\bar{y})^k B[x,y]$   
 $\mu_{10} = \mu_{01} = 0$
- Central moments are translation invariant; but not rotation and scale invariant
- Normalized central moments:  $\eta_{jk} = \frac{\mu_{jk}}{\mu_{00}^{\frac{j+k}{2}}}$      $\gamma = \frac{j+k}{2} + 1$
- Normalized central moments are scale and translation invariant

### Interpretation of central moments

| Central moment | interpretation   |
|----------------|--|
| $\mu_{20}$     | horizontal centralness                                       |
| $\mu_{02}$     | vertical centralness   |
| $\mu_{11}$     | diagonality  |
| $\mu_{12}$     | horizontal divergence<br>(relative extent of right wrt left) |
| $\mu_{21}$     | vertical divergence<br>(relative extent of bottom wrt top)   |
| $\mu_{30}$     | horizontal imbalance   |
| $\mu_{03}$     | Vertical imbalance   |

### Moment Invariants

$$\begin{aligned} \phi_1 &= \eta_{20} + \eta_{02} \\ \phi_2 &= (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2 \\ \phi_3 &= (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2 \\ \phi_4 &= (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2 \\ \phi_5 &= (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] \\ &\quad + (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \\ \phi_6 &= (\eta_{20} - \eta_{02})(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2 \\ &\quad + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03}) \\ \phi_7 &= (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] \\ &\quad - (\eta_{30} - 3\eta_{12})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \end{aligned}$$

## Distance measures

- Euclidean  $d_E(P_1, P_2) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$
- City block  $d_{city}(P_1, P_2) = |x_1 - x_2| + |y_1 - y_2|$
- Chessboard  $d_{chess}(P_1, P_2) = \max(|x_1 - x_2|, |y_1 - y_2|)$

## Distance Transforms

- Minimum distance between a pixel of the object and the background
- Iterative algorithm to compute the distance transform:

$$f^0[i, j] = f[i, j]$$

$$f^m[i, j] = f^0[i, j] + \min(f^{m-1}[u, v])$$

$$\forall [u, v] \text{ which are 4-neighbors of } [i, j]$$

## Boundary Features

1. Simple descriptors
2. Chain code
3. Polygonal approximations
4. Boundary segments
5. Skeleton
6. Signatures
7. Fourier Descriptors

## Simple Descriptors

1. Length of boundary
2. Diameter of boundary
3. Curvature
4. Minor axis
5. Eccentricity: Major / Minor
6. Basic rectangle
7. Extremal points

## Chain code



Representation contains two components

- Coordinates of the start pixel  
(2,3)
- Vector denoting directions between linked edges

direction code (3 bits):

|   |   |   |
|---|---|---|
| 1 | 2 | 3 |
| 0 | . | 4 |
| 7 | 6 | 5 |

Chain code vector for above curve:

|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 4 | 4 | 4 | 4 | 4 | 4 | 4 | 3 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 5 | 4 | 4 | 4 | 4 | 4 | 5 | 7 |
| 0 | 0 | 0 | 0 | 7 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 2 | 2 |

## Shape Number

- Chain code not starting point or scaling invariant: Shape number is starting pt, translation and rotation invariant.
- To get shape number:
- Obtain chain code
- Find 1<sup>st</sup> difference mod 8 (or 4)
- Rotate this until it is minimized

## Polygonal Approximations

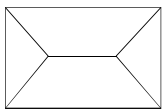
- We already talked about polygonal approximation
- Coordinates of vertices saved

## Boundary Segments

- Find the convex hull
- Convex deficiency= convex hull-object
- Boundary segments: transitions in and out of convex deficiency

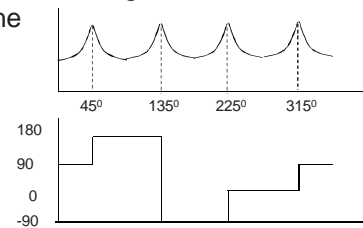
## Skeleton

- Skeleton: Loci of the center of maximally inscribed discs.



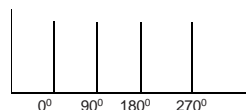
## Signatures

- 1D function that represents the boundary
- Distance vs angle to center
- Or: angle between tangent line and reference line



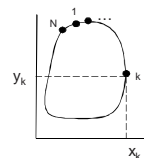
## Shape Density Function

- Histogram of tangent-angle values



## Fourier Descriptors

- Each point on the boundary is treated as a complex number



$$s(k) = x_k + jy_k$$

$$a(k) = \frac{1}{N} \sum_{k=0}^{N-1} s(k) e^{-j\frac{2\pi ik}{N}}$$

- A few Fourier Descriptors will represent the shape in smoothed form

## Invariance of Fourier Descriptors

Rotation

$$s(k)e^{j\theta} \Leftrightarrow a(u)e^{j\theta} \quad \text{affects all coefficients by some constant}$$

Translation

$$s(k + \Delta) \Leftrightarrow a(u) + \Delta\delta(u) \quad \text{affects 0th coefficient only}$$

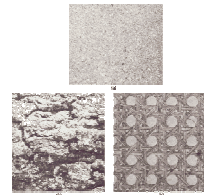
Scaling

$$\alpha s(k) \Leftrightarrow \alpha a(u) \quad \text{affects all coefficients by the same constant}$$

Starting Point

$$s(k - k_0) \Leftrightarrow a(u)e^{-j2\pi \frac{k_0 u}{N}} \quad \text{affects phase only}$$

## Texture



Fine textures vs coarse textures  
 Contrast                      directionality  
 Roughness – regularity  
 Microtexture vs. macrotexture

## Problems in Texture Analysis

1. Given a textured region, classify the region
2. Given a textured region, find a model (use parameters of the model as features)
3. Given an image having many textured areas, determine the boundaries between the differently textured areas (segmentation)

## Texture Features

Statistical Approach:

- Gray-level co-occurrence matrices
- Discrete Markov Random Fields
- Laws Texture Measures

Spectral Approach:

- Autocorrelation and PSD

Structural Approach

- Voronoi polygons

## Gray-level co-occurrence matrices

- GxG square matrix with elements corresponding to the relative frequency of occurrence of pairs of gray levels
  - d=(dx,dy) is the displacement
  - G: # gray levels
  - I: intensity function
  - $P(i,j)=\{I(r,s),I(r+dx,s+dy)\}/\{I(r,s)=i;I(r+dx,s+dy)=j\}$

|   |   |   |   |
|---|---|---|---|
| 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 1 |
| 0 | 2 | 2 | 2 |
| 2 | 2 | 3 | 3 |

image

|   |   |   |   |   |
|---|---|---|---|---|
|   | 0 | 1 | 2 | 3 |
| 0 | 2 | 2 | 1 | 0 |
| 1 | 0 | 2 | 0 | 0 |
| 2 | 0 | 0 | 3 | 1 |
| 3 | 0 | 0 | 0 | 1 |

P(1,0)

$$P(-1,0)=P(1,0)^T$$

|   |   |   |   |   |
|---|---|---|---|---|
|   | 0 | 1 | 2 | 3 |
| 0 | 4 | 2 | 1 | 0 |
| 1 | 2 | 4 | 0 | 0 |
| 2 | 1 | 0 | 6 | 1 |
| 3 | 0 | 0 | 1 | 2 |

$$P_H=P(1,0)+P(-1,0)$$

## Co-occurrence features

$$\text{Uniformity} : \sum_{i,j} P_{ij}^2$$

$$\text{Entropy} : - \sum_{i,j} P_{ij} \log P_{ij}$$

$$\text{Max probability} : \max_{i,j} P_{ij}$$

$$\text{Contrast} : \sum_{i,j} |i-j|^k (P_{ij})^l \quad \text{commonly, } k=2, l=1$$

$$\text{Homogeneity} : \sum_{i,j} \frac{P_{ij}}{1+|i-j|}$$

$$\text{Cluster tendency} : \sum_{i,j} (i+j-2\mu)^k P_{ij}$$

$$\text{Inverse Difference Moment} \sum_{i,j} \frac{P_{ij}^2}{|i-j|^k}$$

$$\text{Correlation} : \sum_{i,j} \frac{(i-\mu)(j-\mu)P_{ij}}{\sigma^2} \quad \text{where } \mu = \sum_{i,j} iP_{ij}$$

$$\text{Pr of a run length of } n \text{ (of gray level } i) : \sum_i \frac{(P_i - P_{ij})^2 (P_i)^{n-1}}{P_i^n}$$

### Co-occurrence, continued.

- How to choose d?

$$\kappa^2(d) = \sum_{i,j} \frac{P_{ij}^2}{P_i P_j} - 1 \quad \text{where } P_i = \sum_j P_{ij}$$

- Subband domain co-occurrence (Ertüzün)
  - Use wavelet decomposition into LL, LH,HL, HH bands
  - Keep only bands whose energy is at least 35 % of LL band
  - Find co-occurrence matrix features in those bands

### Discrete Markov Random Fields

- Let N(r,c) be a neighborhood of (r,c)
- Assume that the conditional probability of f(r,c) given the image is equal to the conditional prob. given only the pixels in the neighborhood

$$P(f(r,c) / f(i,j) : (i,j) \text{ in } I) = P(f(r,c) / f(i,j) : (i,j) \text{ in } N)$$

If it is assumed that the joint pdf of pixels is Gaussian,

$$f(r,c) = \underbrace{\sum_{i,j \in N} f(r-i,c-j)h(i,j)}_{\text{Linear combination of pixels in the neighborhood}} + \underbrace{u(r,c)}_{\text{Noise term}}$$

### Discrete Markov Random Fields

- To find the coefficients h(i,j) use least squares (define error and differentiate wrt. h(i,j)'s)

$$\mathcal{E}^2 = \sum_{r,c} [f(r,c) - \sum_{(i,j) \in N} f(r-i,c-j)h(i,j)]^2$$

$$\frac{\partial \mathcal{E}^2}{\partial h_{ij}} = 0 \quad \text{gives: } \sigma(m,n) = \sum_{i,j} h(i,j)\sigma(m-i,n-j)$$

where  $\sigma(m,n) = \sum_{r,c} f(r,c)r(r-m,c-n)$  are the covariance terms

- Solve and use h(i,j)'s as features

### Laws Texture Energy Features

- Use local masks to detect various types of texture

L5 (Level) = [1 4 6 4 1]

E5 (Edge) = [-1 -2 0 2 1]

S5 (Spot) = [-1 0 2 0 -1]

R5 (Ripple) = [1 -4 6 -4 1]

2D masks are obtained by convolutions (outer products)

E5L5:

|    |   |   |   |   |   |   |    |    |     |    |    |
|----|---|---|---|---|---|---|----|----|-----|----|----|
| -1 | 1 | 4 | 6 | 4 | 1 | = | -1 | -4 | -6  | -4 | -1 |
| -2 |   |   |   |   |   |   | -2 | -8 | -12 | -8 | -2 |
| 0  |   |   |   |   |   |   | 0  | 0  | 0   | 0  | 0  |
| 2  |   |   |   |   |   |   | 2  | 8  | 12  | 8  | 2  |
| 1  |   |   |   |   |   |   | 1  | 4  | 6   | 4  | 1  |

### Laws Texture Energy Features, cont.

1. Remove effects of illumination by moving a small window and subtract local average
2. Apply each of the 5x5 masks to the image
 

|           |   |  |
|-----------|---|--|
| E5E5      | } | Each pixel is replaced by a 9-dimensional vector of energies |
| S5S5      |   |  |
| R5R5      |   |  |
| E5L5+L5E5 |   |  |
| S5L5+L5S5 |   |  |
| R5L5+L5R5 |   |  |
| S5E5+E5S5 |   |  |
| R5E5+E5R5 |   |  |
| R5S5+S5E5 |   |  |
3. Cluster the image into regions of uniform texture

### Autocorrelation and Power Spectral Density Functions

- Consider two image transparencies that are copies of each other. Overlay them on top of each other and translate. Measure the amount of light that passes through.

$$p(x,y) = \frac{\sum_{r,c} f(r,c)f(r+x,c+y)}{\sum_{r,c} f^2(r,c)}$$

- For periodic textures, p(x,y) will be periodic
- PSD is the Fourier transform of the Autocorrelation function

## Structural Approach

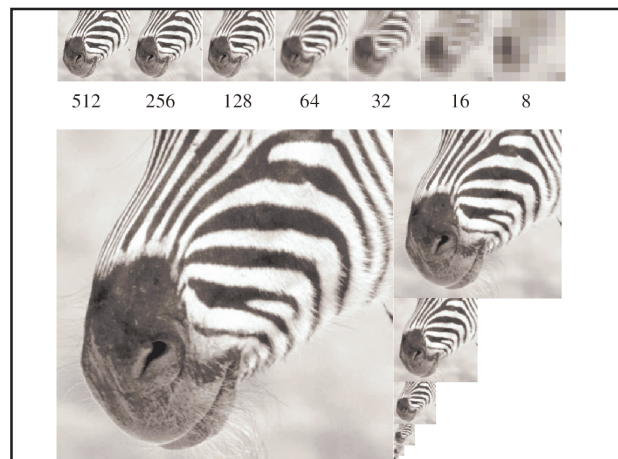
- Texels: texture primitives that are repeated in some regular relationship
- Approach: segment texels and describe relationship
- Suppose texels have been extracted; Let  $S$  be the set of centroids
- Find the Voronoi polygons of centroids  $S$
- Use shape features of Voronoi polygons

## Multiscale approach

- Use a multiscale representation first; then apply statistical or other approaches
- Wavelets; pyramids

## The Gaussian pyramid

- Smooth with gaussians, because
  - a gaussian\*gaussian=another gaussian
- Synthesis
  - smooth and sample
- Analysis
  - take the top image
- Gaussians are low pass filters, so representation is redundant



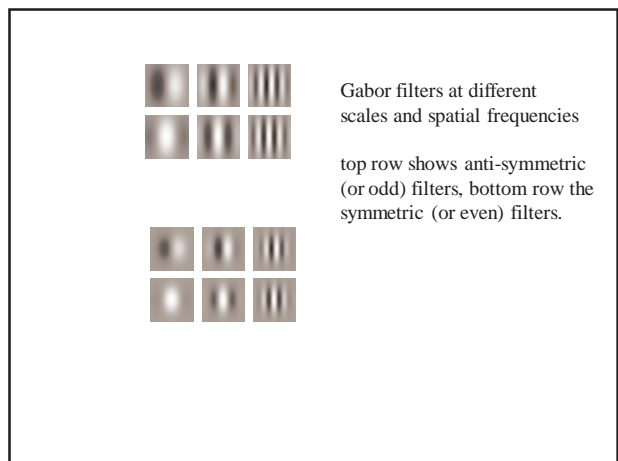
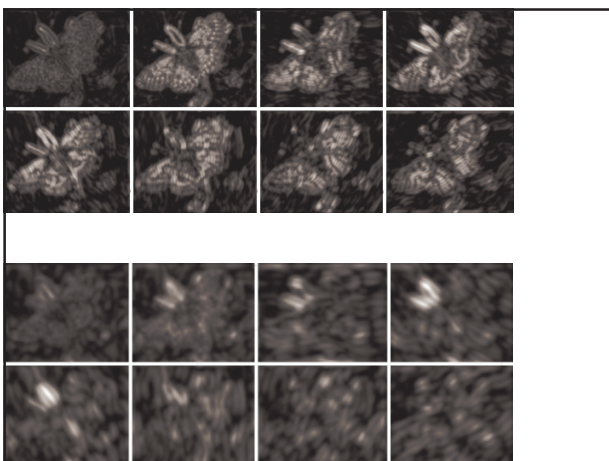
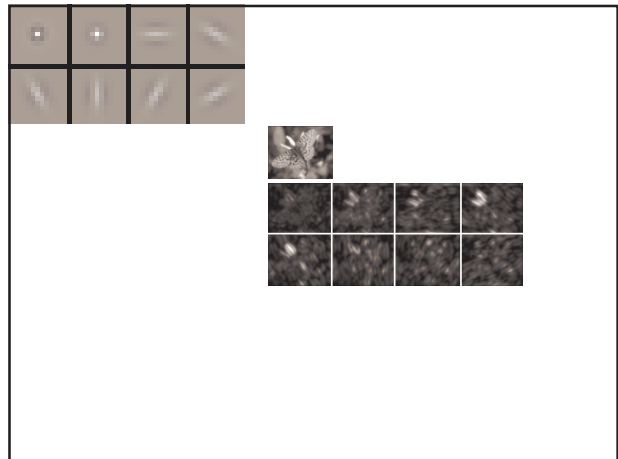
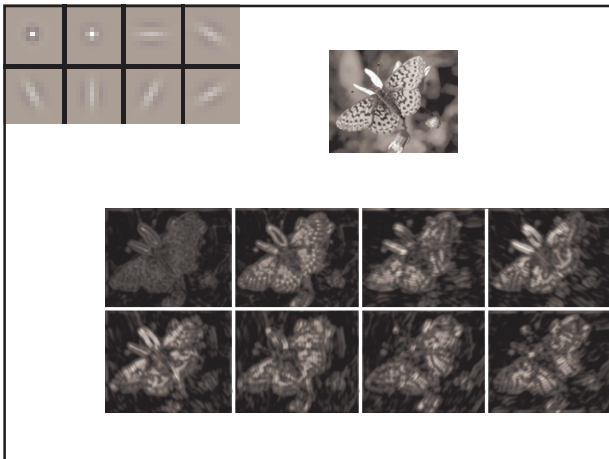
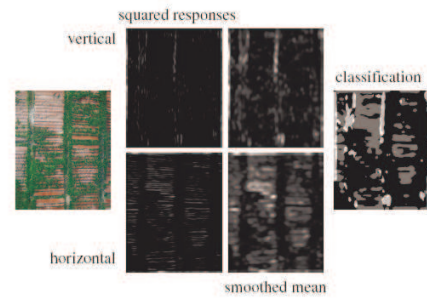
## Texture

- Key issue: representing texture
  - Texture based matching
    - little is known
  - Texture segmentation
    - key issue: representing texture
  - Texture synthesis
    - useful; also gives some insight into quality of representation
  - Shape from texture
    - cover superficially



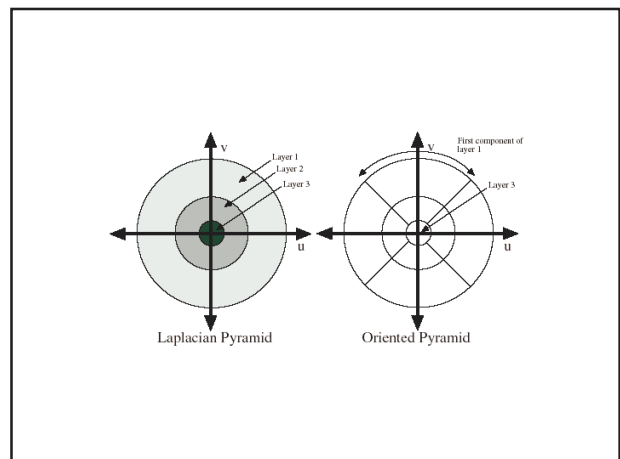
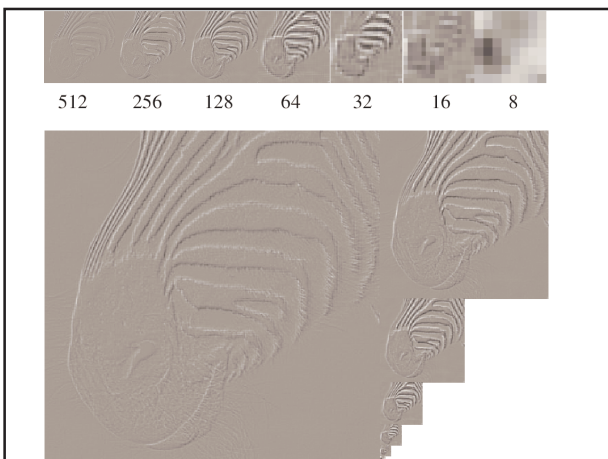
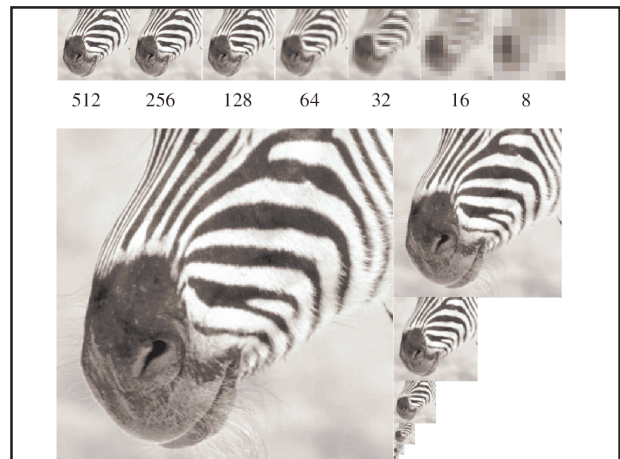
## Representing textures

- Textures are made up of quite stylised subelements, repeated in meaningful ways
- Representation:
  - find the subelements, and represent their statistics
- But what are the subelements, and how do we find them?
  - recall normalized correlation
  - find subelements by applying filters, looking at the magnitude of the response
- What filters?
  - experience suggests spots and oriented bars at a variety of different scales
  - details probably don't matter
- What statistics?
  - within reason, the more the merrier.
  - At least, mean and standard deviation
  - better, various conditional histograms.



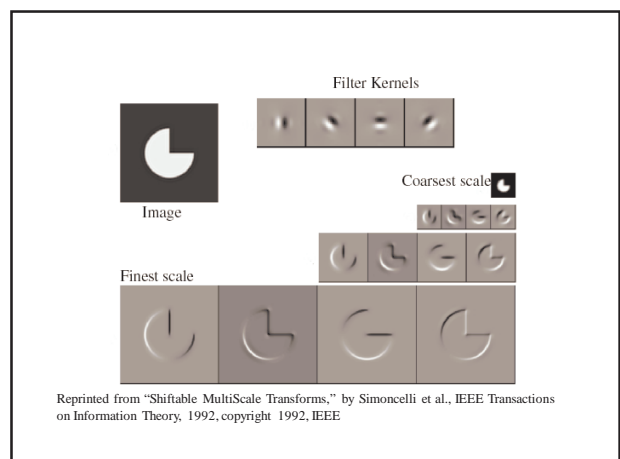
## The Laplacian Pyramid

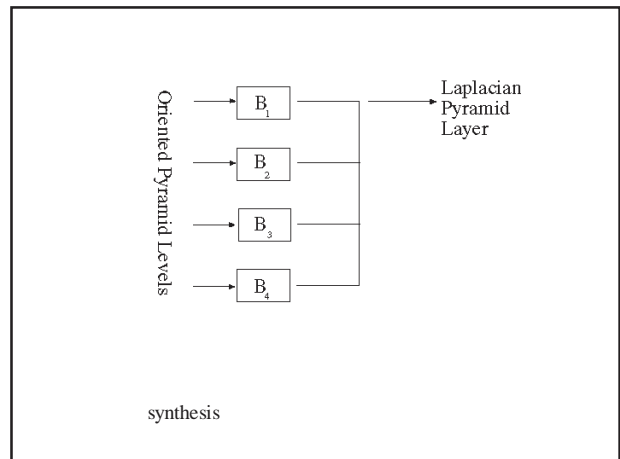
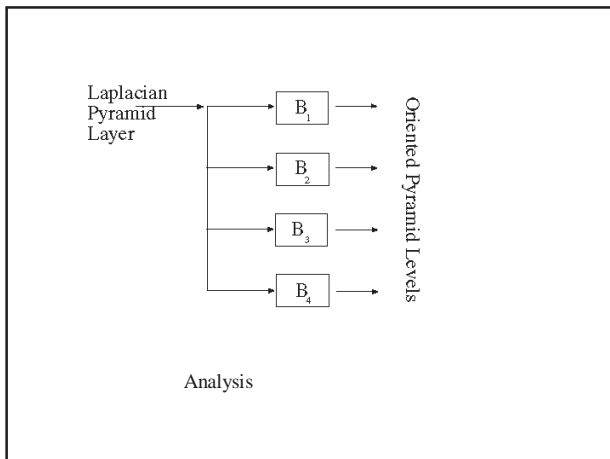
- Synthesis
  - preserve difference between upsampled Gaussian pyramid level and Gaussian pyramid level
  - band pass filter - each level represents spatial frequencies (largely) unrepresented at other levels
- Analysis
  - reconstruct Gaussian pyramid, take top layer



## Oriented pyramids

- Laplacian pyramid is orientation independent
- Apply an oriented filter to determine orientations at each layer
  - by clever filter design, we can simplify synthesis
  - this represents image information at a particular scale and orientation





- ### Final texture representation
- Form an oriented pyramid (or equivalent set of responses to filters at different scales and orientations).
  - Square the output
  - Take statistics of responses
    - e.g. mean of each filter output (are there lots of spots)
    - std of each filter output
    - mean of one scale conditioned on other scale having a particular range of values (e.g. are the spots in straight rows?)

- ### Texture synthesis
- Use image as a source of probability model
  - Choose pixel values by matching neighbourhood, then filling in
  - Matching process
    - look at pixel differences
    - count only synthesized pixels

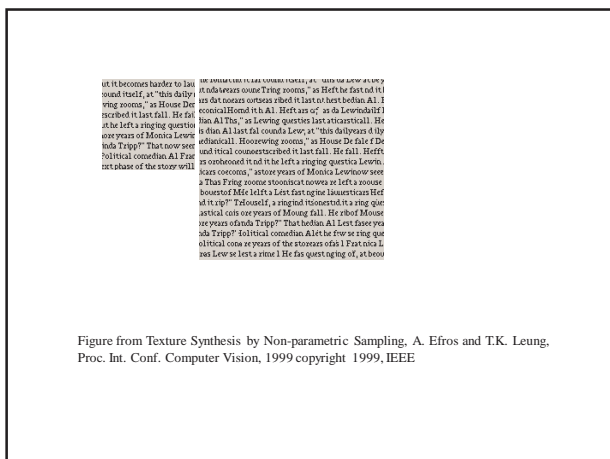


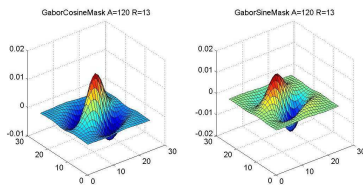
Figure from Texture Synthesis by Non-parametric Sampling, A. Efros and T.K. Leung, Proc. Int. Conf. Computer Vision, 1999 copyright 1999, IEEE

- ### 2D Gabor Wavelets I
- Why Gabor wavelets?
    - Physiological studies found cells in V1 (simple cells) that are selectively tuned to orientation as well as to spatial frequency
    - Response profile of a simple cell could be approximated by 2D Gabor filters
  - Features of 2D Gabor wavelets
    - They are oriented and have characteristic frequencies

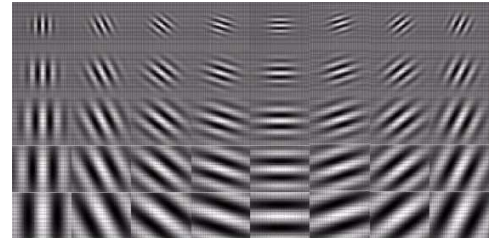
## 2D Gabor Wavelets II

- Mathematical formulation

$$\psi_j(\vec{x}) = \frac{k_j^2}{\sigma^2} e^{-\frac{k_j^2 x^2}{2\sigma^2}} [e^{ik_j \vec{x}} - e^{-\frac{\sigma^2}{2}}]$$

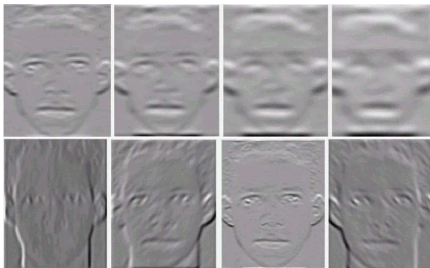


## 2D Gabor Wavelets II



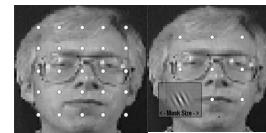
## 2D Gabor Wavelets III

- Gabor convolution of images



## Face representation using Gabors

- Place a grid on the face region of image
- At each grid point convolve image with 2D Gabor wavelets
  - 5 different frequencies
  - 8 different orientations (with an interval of 22.5 degree, from 0 to 157.5)
  - A jet holds the responses (magnitude) of Gabor convolutions at each grid point.
  - Form a feature vector by combining all the jets.
  - $3 \times 3 \times (5 \times 8) = 9 \times 40 = 360$  (feature vector size)



## Current approaches

- Problem based
- WT works for texture
- Subspace: PCA, LDA, KPCA, ICA, NMF
- Registration!
- Later approaches based on Machine learning: Guest lecture on mining MR image data
- Bag of visual words lecture

## More challenging

- Analyse actions



Car exit      telephone talk      smoke      shake hands      drink from cup

- Need to track objects or features over several frames
- Analyse movement using different methods
- Next week: Tracking, analysis of actions