Segmentation
Region-based processing

Feature extraction

Gonzalez-Woods Chapt. 10

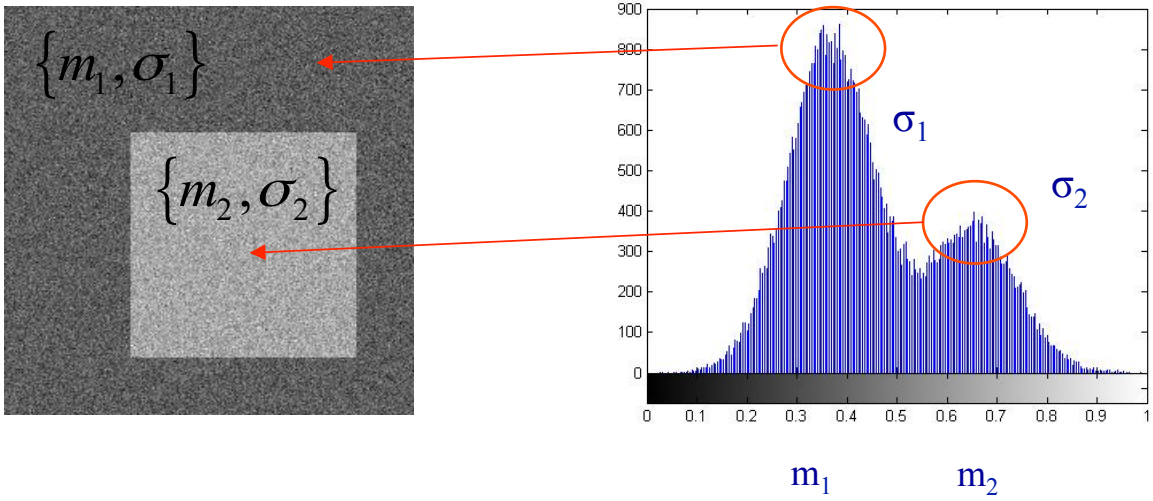
Pratt Chapt. 16 and 17

Topics

- Feature extraction
- Image segmentation
- Clustering and classification

Region based processing

- Complementary to edge detection
- Based on neighborhood characteristics
- Local descriptors represent properties of sets of pixels. Typically these are representative of the pdf (histogram) of the gray values in each region



Applications

- Image segmentation
  - Group similar components (such as, pixels in an image, image frames in a video) to obtain a compact representation.
  - Clustering, classification
    - Methods: Thresholding, K-means clustering, etc.
- Pattern recognition
  - Classification
- Scenarios: Finding tumors, veins, etc. in medical images, finding targets in satellite/aerial images, finding people in surveillance images, summarizing video, etc.

Segmentation strategy

Edge-based

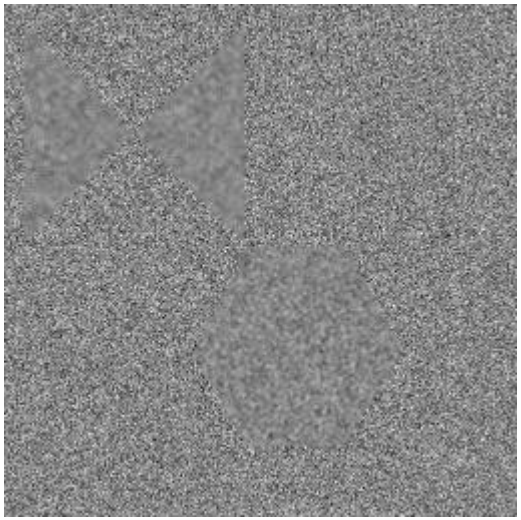
- Assumption: different objects are separated by edges (grey level discontinuities)
- The segmentation is performed by identifying the grey level gradients
- The same approach can be extended to color channels

Region-based

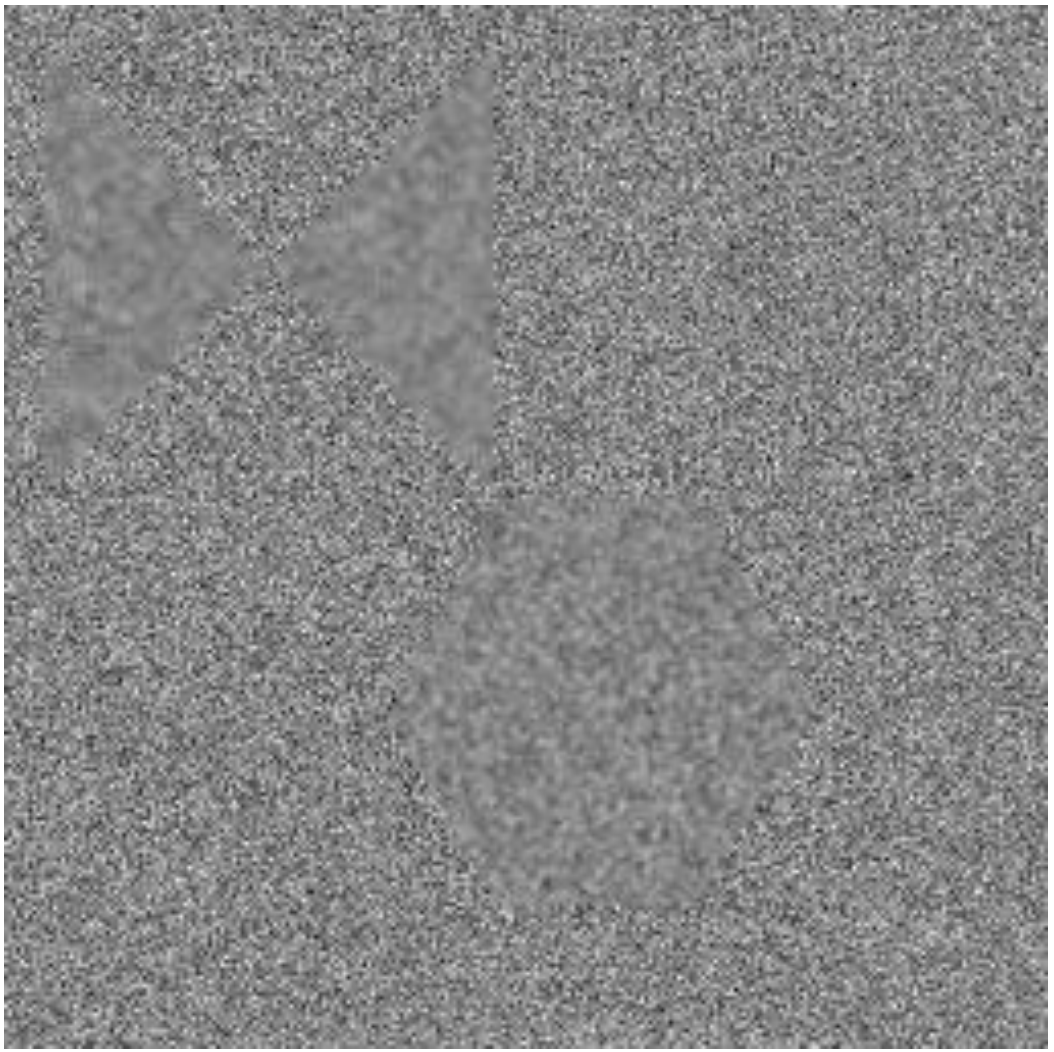
- Assumption: different objects are separated by other kind of perceptual boundaries
  - neighborhood features
- Most often texture-based
  - Textures are considered as instantiations of underlying stochastic processes and analyzed under the assumptions that stationarity and ergodicity hold
- Method
  - Region-based features are extracted and used to define “classes”

Examples

original

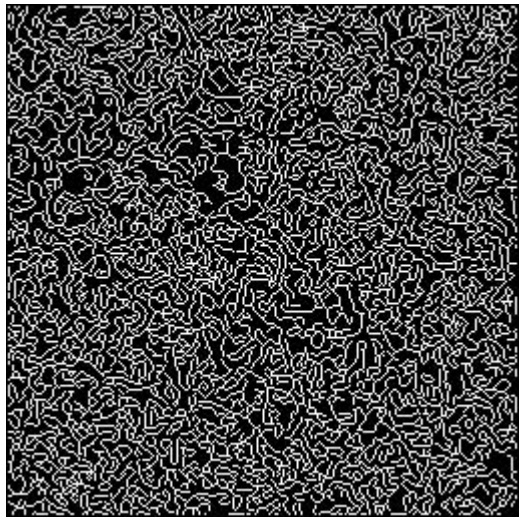


zoomed

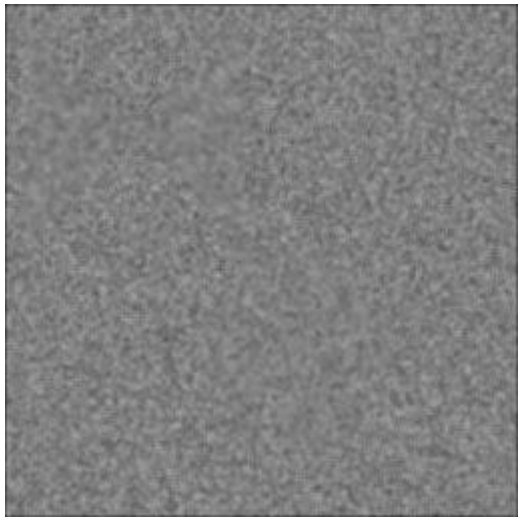


Examples

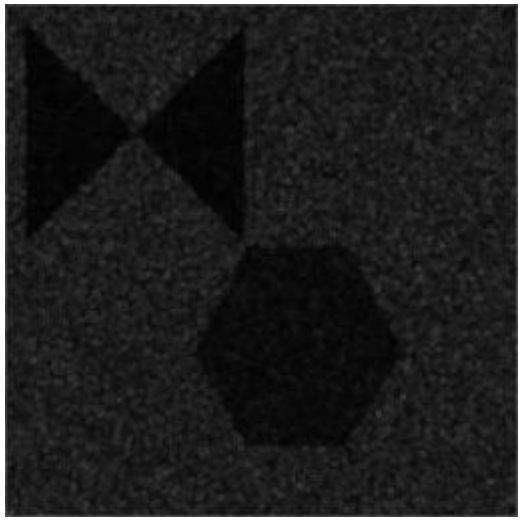
Canny



block mean



block std



Feature extraction

Types of features

- An *image feature* is a distinguishing primitive characteristic or attribute of an image.
- Amplitude features: image domain
- Transformed coefficients features: transformed domain
  - Fourier domain
  - Principal components (PCA)
  - Wavelet domain

Amplitude features

- Image variables such as luminance or tristimulus values may be utilized directly, or alternatively, some linear, nonlinear, or perhaps noninvertible transformation can be performed to generate variables in a new amplitude space.
- Amplitude measurements may be made at specific image points f[i,j], [e.g., the amplitude at pixel coordinate] , or over a neighborhood centered at [i,j].
  - An advantage of a neighborhood, as opposed to a point measurement, is a *diminishing of noise effects* because of the averaging process.
  - A disadvantage is that object edges falling within the neighborhood can lead to erroneous measurements.

Amplitude features

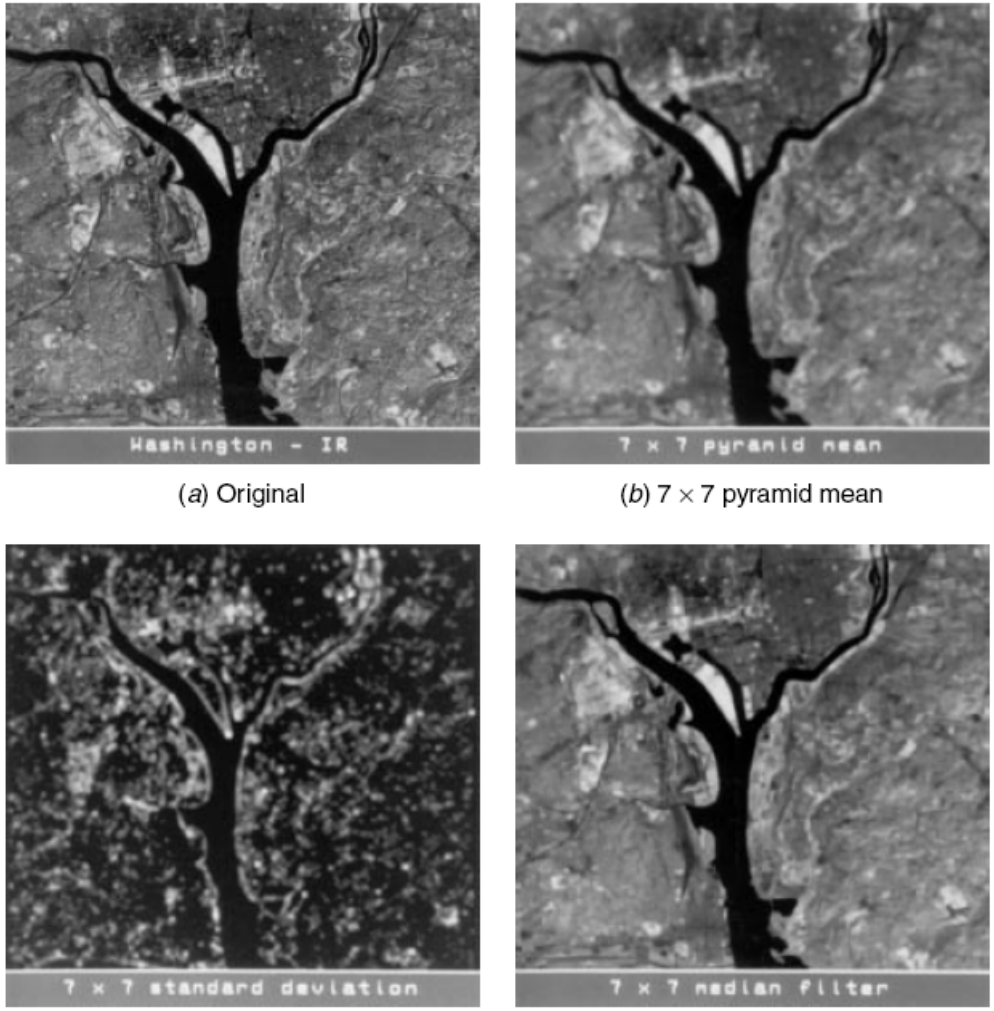
- Mean over a window W=2w+1

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

- Median over a window W=2w+1
  - The *median* is defined to be that pixel amplitude in the window for which one-half of the pixels are equal or smaller in amplitude, and one-half are equal or greater in amplitude.
- Variance over a window W=2w+1

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

Example



(*a*) Original (*b*) 7 × 7 pyramid mean

(*c*) 7 × 7 standard deviation (*d*) 7 × 7 plus median

FIGURE 16.2-1. Image amplitude features of the `washington_ir` image.

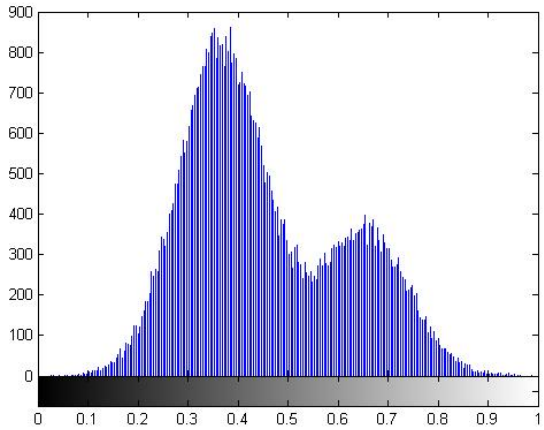
Histogram features

- Mean and standard deviation can be calculated based on the histogram, as well as other features representing the distribution of gray level (or tristimulus) values
- First order histogram

$$h[g] = \frac{N(g)}{N}$$

$N(g)$: number of pixels with graylevel=g

N: total number of pixels



Quantitative histogram shape descriptors

- First order descriptors

Mean:

$$S_M \equiv \bar{b} = \sum_{b=0}^{L-1} bP(b)$$

Standard deviation:

$$S_D \equiv \sigma_b = \left[\sum_{b=0}^{L-1} (b - \bar{b})^2 P(b) \right]^{1/2}$$

Skewness:

$$S_S = \frac{1}{\sigma_b^3} \sum_{b=0}^{L-1} (b - \bar{b})^3 P(b)$$

Quantitative histogram shape descriptors

- First order descriptors

Kurtosis:

$$S_K = \frac{1}{\sigma_b^4} \sum_{b=0}^{L-1} (b - \bar{b})^4 P(b) - 3$$

Energy:

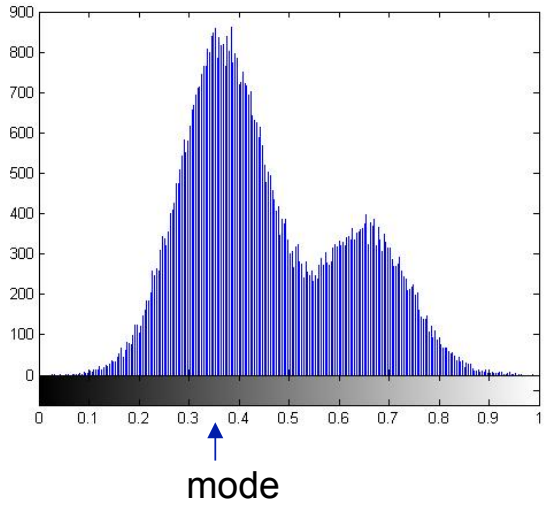
$$S_N = \sum_{b=0}^{L-1} [P(b)]^2$$

Entropy:

$$S_E = -\sum_{b=0}^{L-1} P(b) \log_2 \{P(b)\}$$

Quantitative histogram shape descriptors

- *histogram mode*: the pixel amplitude corresponding to the histogram peak (i.e., the most commonly occurring pixel amplitude in the window).
- If the histogram peak is not unique, the pixel at the peak closest to the mean is usually chosen as the histogram shape descriptor.



bi-modal histogram

Quantitative histogram shape descriptors

- Second-order histogram features are based on the definition of the joint probability distribution of pairs of pixels.

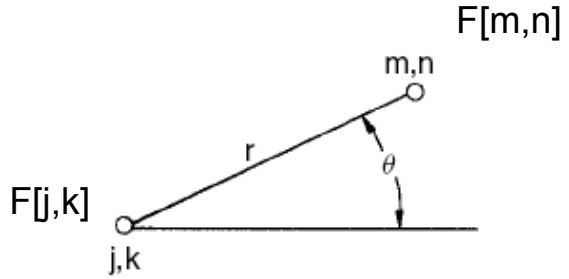


FIGURE 16.2-2. Relationship of pixel pairs.

The joint probability depends on the pixel relative position!

$$p(a,b) = p\{F(j,k) = a, F(n,m) = b\}$$

- Histogram estimate of the second-order distribution

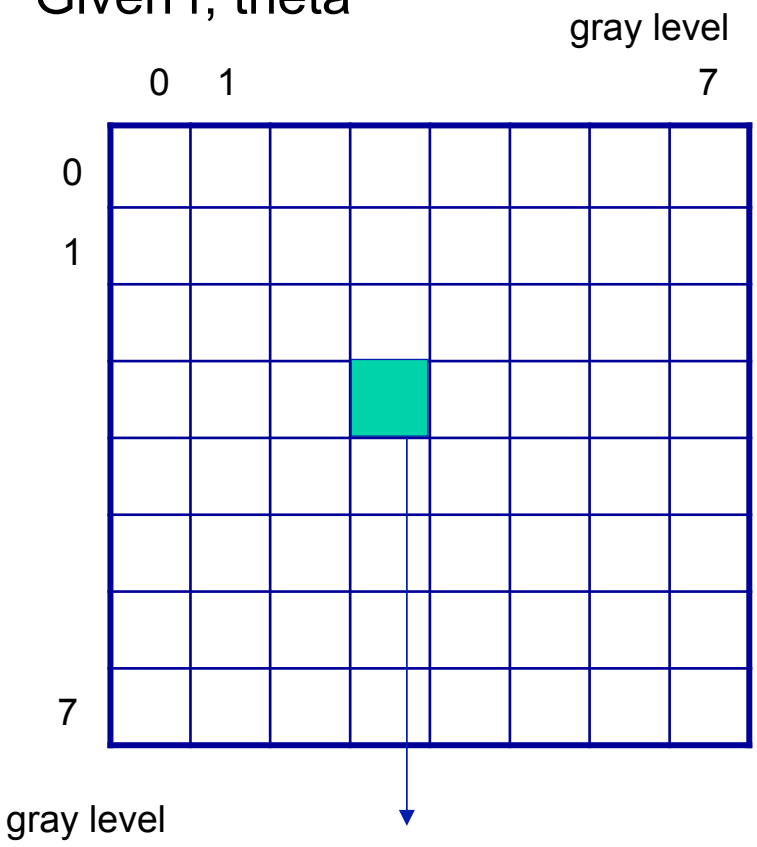
$$p(a,b); \quad \frac{N(a,b)}{N}$$

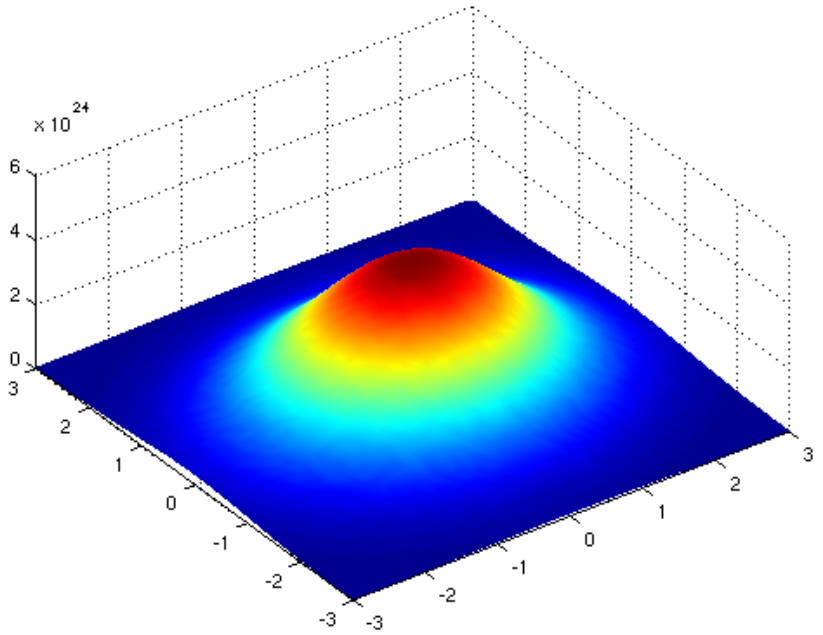
$N(a,b)$ → number of occurrences for which F[j,k]=a AND F[n,m]=b

N → total number of pixels in the observation window

Second order histogram estimates

Given r, theta





co-occurrence matrix

Descriptors

Autocorrelation:

$$S_A = \sum_{a=0}^{L-1} \sum_{b=0}^{L-1} abP(a,b)$$

Covariance:

$$S_C = \sum_{a=0}^{L-1} \sum_{b=0}^{L-1} (a-\bar{a})(b-\bar{b})P(a,b)$$

where

$$\bar{a} = \sum_{a=0}^{L-1} \sum_{b=0}^{L-1} aP(a,b)$$

$$\bar{b} = \sum_{a=0}^{L-1} \sum_{b=0}^{L-1} bP(a,b)$$

Descriptors

Inertia:

$$S_I = \sum_{a=0}^{L-1} \sum_{b=0}^{L-1} (a-b)^2 P(a,b)$$

Absolute value:

$$S_V = \sum_{a=0}^{L-1} \sum_{b=0}^{L-1} |a-b| P(a,b)$$

Inverse difference:

$$S_F = \sum_{a=0}^{L-1} \sum_{b=0}^{L-1} \frac{P(a,b)}{1+(a-b)^2}$$

Energy:

$$S_G = \sum_{a=0}^{L-1} \sum_{b=0}^{L-1} [P(a,b)]^2$$

Entropy:

$$S_T = -\sum_{a=0}^{L-1} \sum_{b=0}^{L-1} P(a,b) \log_2 \{P(a,b)\}$$

Transform features: Fourier

- The distribution (histogram) of the **transformed coefficients** is characterized either considering the whole frequency domain or partitioning it according to different criteria
- In each frequency region, the most common choice consists in using first order descriptors
  - mean
  - variance (as indicator of the energy)

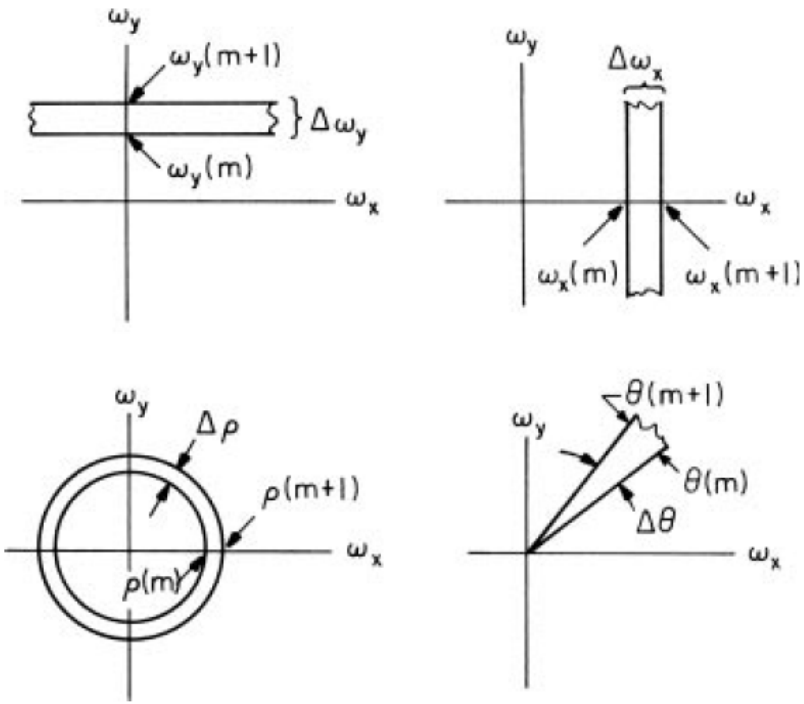


FIGURE 16.3-1. Fourier transform feature masks.

Texture definition

- Many portions of images of natural scenes are devoid of sharp edges over large areas.
- In these areas, the scene can often be characterized as exhibiting a consistent structure analogous to the texture of cloth.
- Image texture measurements can be used to segment an image and classify its segments.
- The notion of texture appears to depend upon three ingredients:
  - (1) some local 'order' is repeated over a region which is large in comparison to the order's size, (2) the order consists in the nonrandom arrangement of elementary parts and (3) the parts are roughly uniform entities having approximately the samedimensions everywhere within the textured region.”

Texture definition

- Although these descriptions of texture seem perceptually reasonably, they do not immediately lead to simple quantitative textural measures in the sense that the description of an edge discontinuity leads to a quantitative description of an edge in terms of its location, slope angle, and height.
- Textures are often described in terms of “coarseness”
  - The coarseness index is related to the spatial repetition period of the local structure.
  - A large period implies a coarse texture; a small period implies a fine texture.

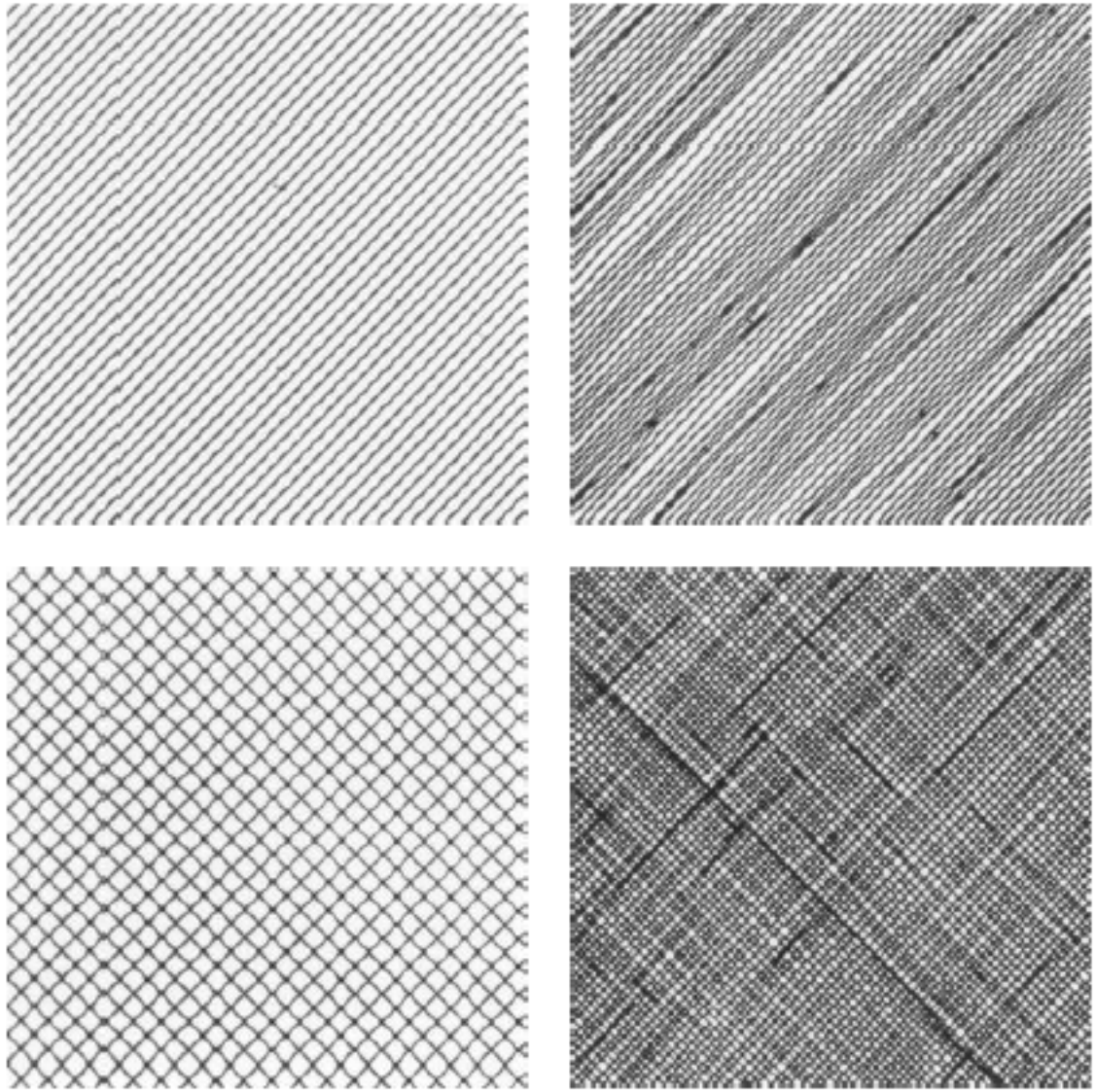
Texture definition

- Texture is a neighborhood property of an image point.
  - Therefore, texture measures are inherently dependent on the size of the observation neighborhood.
- Because texture is a spatial property, measurements should be restricted to regions of relative uniformity.
- Hence it is necessary to establish the boundary of a uniform textural region by some form of image segmentation before attempting texture measurements.

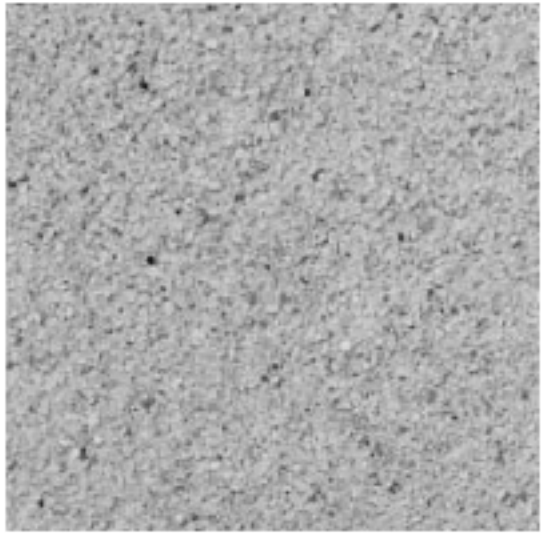
Artificial vs natural textures

- Texture may be classified as being artificial or natural.
  - Artificial textures consist of arrangements of symbols, such as line segments, dots, and stars placed against a neutral background. Several examples of artificial texture are presented in Figure16.4-1 (9).
  - As the name implies, natural textures are images of natural scenes containing semirepetitive arrangements of pixels. Examples include photographs of brick walls, terrazzo tile, sand, and grass.
- Brodatz (11) has published an album of photographs of naturally occurring textures that is used as the reference texture database for texture processing.

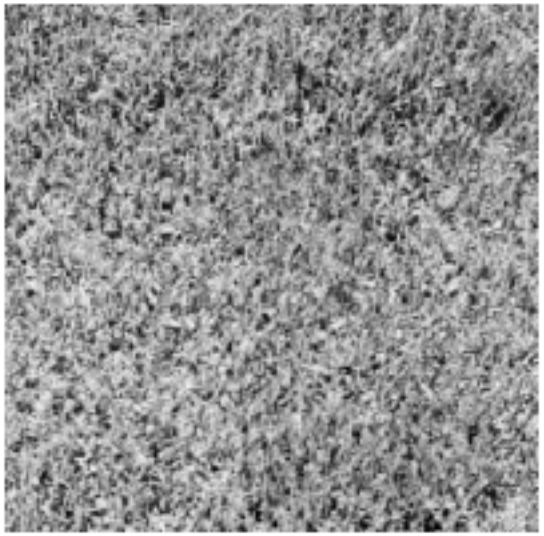
Artificial textures



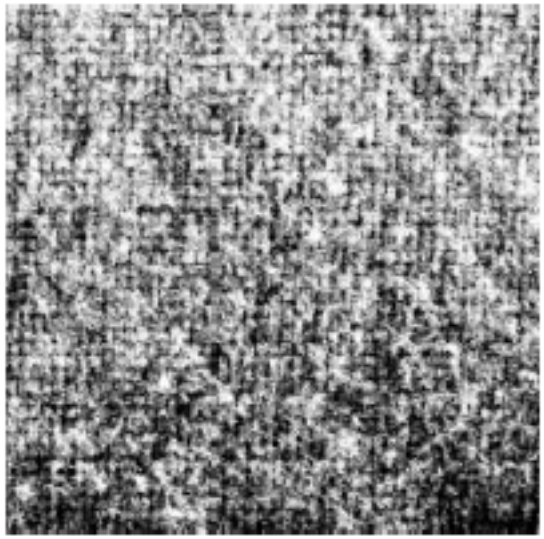
Natural textures



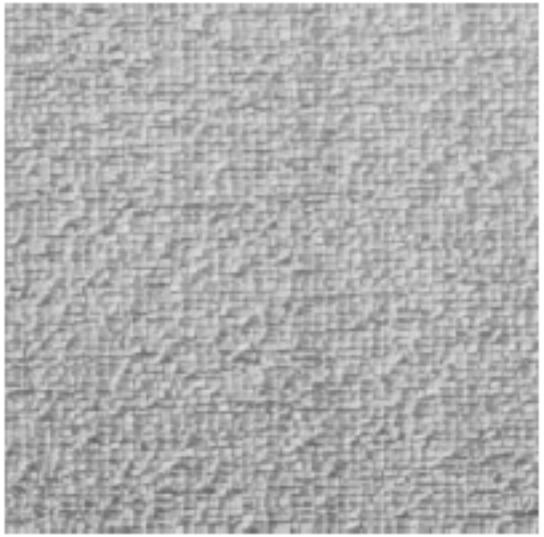
(*a*) Sand



(*b*) Grass



(*c*) Wool



(*d*) Raffia

FIGURE 16.4-2. Brodatz texture fields.

Visual texture discrimination

- A discrete stochastic field is an array of numbers that are randomly distributed in amplitude and governed by some joint probability density. When converted to light intensities, such fields can be made to approximate natural textures surprisingly well by control of the generating probability density.
  - This technique is useful for generating realistic appearing artificial scenes for applications such as airplane flight simulators.
- Stochastic texture fields are also an extremely useful tool for investigating human perception of texture as a guide to the development of texture feature extraction methods
- Julesz: identification of the parameters of stochastic fields that are relevant from a perceptual point of view
  - Namely that are used by the visual system for distinguishing different textures

Julesz's main moments

$$\eta = E\{x_0\}$$ First order (16.5-5a)

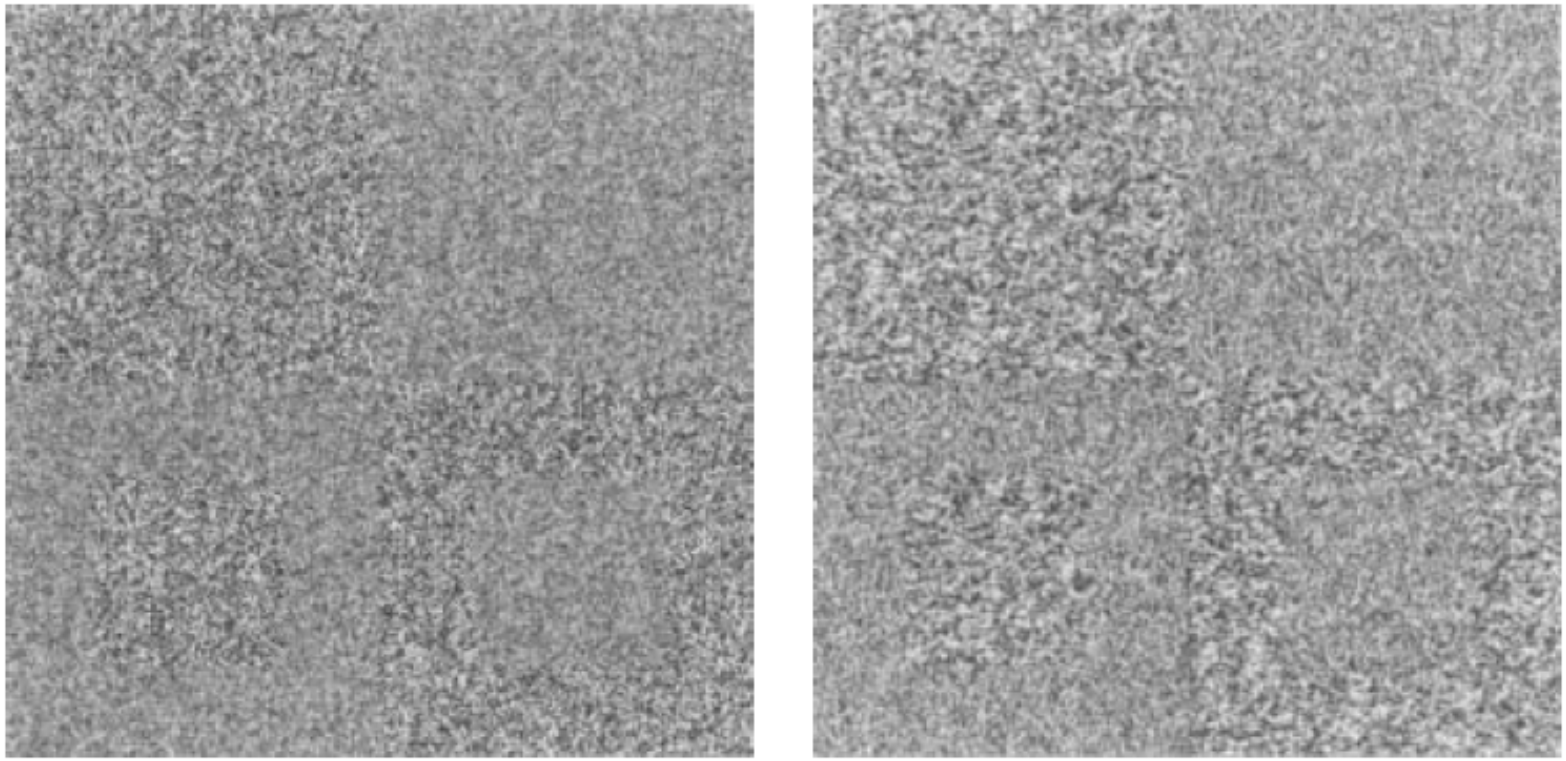
$$\sigma^2 = E\{[x_0 - \eta]^2\}$$ Second order (16.5-5b)

$$\alpha = \frac{E\{[x_0 - \eta][x_1 - \eta]\}}{\sigma^2}$$ Third order (16.5-5c)

$$\theta = \frac{E\{[x_0 - \eta][x_1 - \eta][x_2 - \eta]\}}{\sigma^3}$$ (16.5-5d)

Julesz's moments perceptual significance

$$\eta_A = \eta_B = 0.500$$

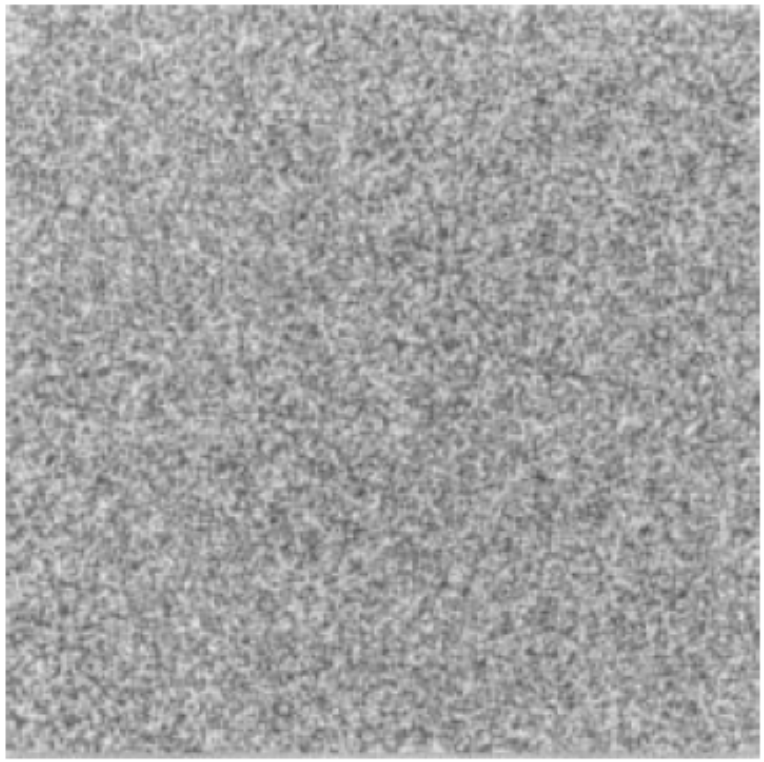


(*a*) Different first order
$\sigma_A = 0.289$, $\sigma_B = 0.204$

(*b*) Different second order
$\sigma_A = 0.289$, $\sigma_B = 0.289$
$\alpha_A = 0.250$, $\alpha_B = -0.250$

Julesz's moments perceptual significance

$\eta_A = \eta_B = 0.500$



(*c*) Different third order
$\sigma_A = 0.289$, $\sigma_B = 0.289$
$\alpha_A = 0.000$, $\alpha_B = 0.000$
$\theta_A = 0.058$, $\theta_B = -0.058$

Julesz's moments perceptual significance

- The examples of indicate that
  - texture field pairs differing in their first- and second-order distributions can be discriminated and
  - support the conjecture, attributed to Julesz, that differences in third order,and presumably, higher-order distribution texture fields cannot be perceived provided that their first-order and second- distributions are pairwise identical.

Texture analysis in F-domain

- Several studies have considered textural analysis based on the Fourier spectrum of an image region
  - Because the degree of texture coarseness is proportional to its spatial period, a region of coarse texture should have its Fourier spectral energy concentrated at low spatial frequencies. Conversely, regions of fine texture should exhibit a concentration of spectral energy at high spatial frequencies.
  - Although this correspondence exists to some degree, difficulties often arise because of *spatial changes in the period and phase* of texture pattern repetitions.
  - Experiments have shown that there is considerable spectral overlap of regions of distinctly different natural texture, such as urban, rural, and woodland regions extracted from aerial photographs.
- Solution: pave the F-domain and extract band-wise features

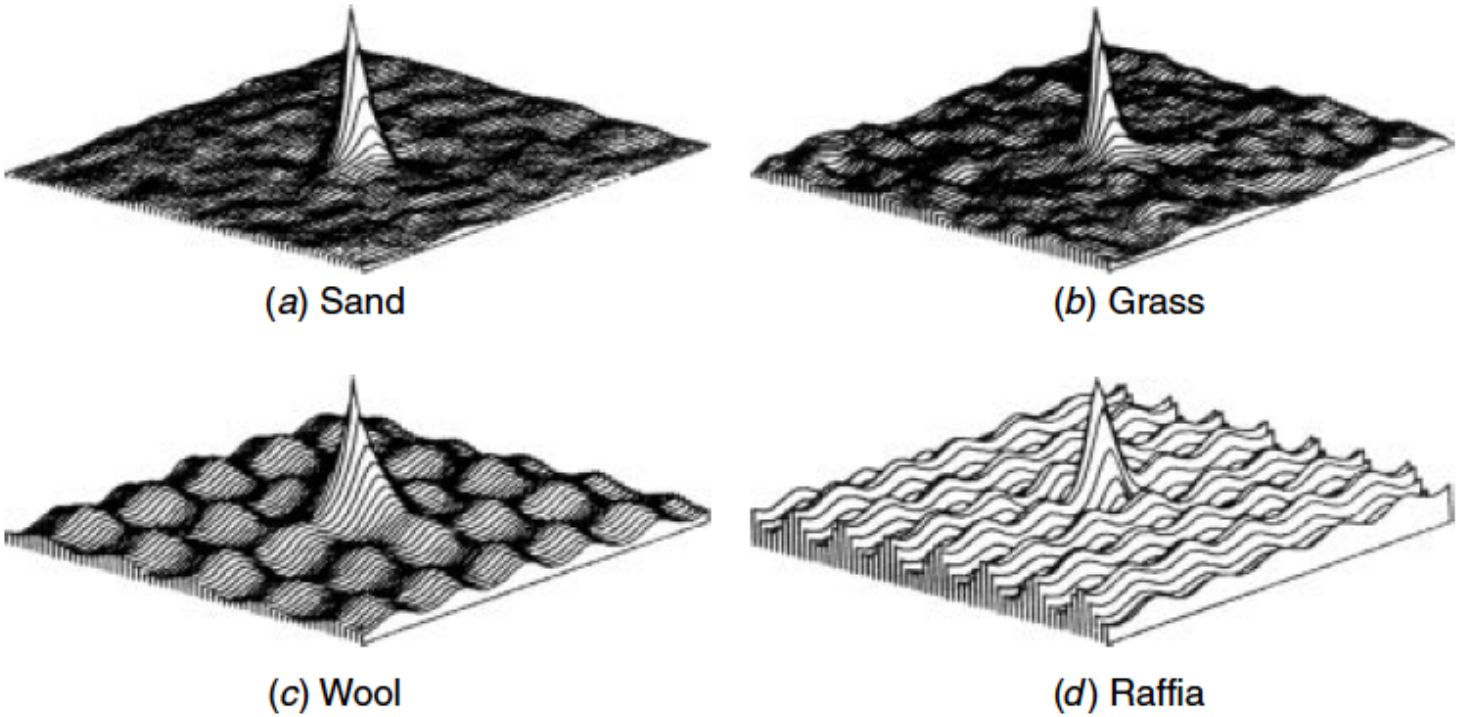
Texture analysis by Autocorrelation

- Autocorrelation function

$$A_F(m,n) = \sum_j \sum_k F(j,k)F(j-m,k-n)$$

- Rationale: presumably, a region of coarse texture will exhibit a higher correlation for a fixed shift than will a region of fine texture.
- Thus, texture coarseness should be proportional to the spread of the autocorrelation function.

Texture analysis by Autocorrelation



(*a*) Sand (*b*) Grass

(*c*) Wool (*d*) Raffia

FIGURE 16.6-1. Perspective views of autocorrelation functions of Brodatz texture fields.

TA by 2D Histograms

- Two-dimensional histogram estimation, also said gray-scale dependency matrix

$$P(a, b; j, k, r, \theta) \approx P_R[F(j, k) = a, F(m, n) = b]$$

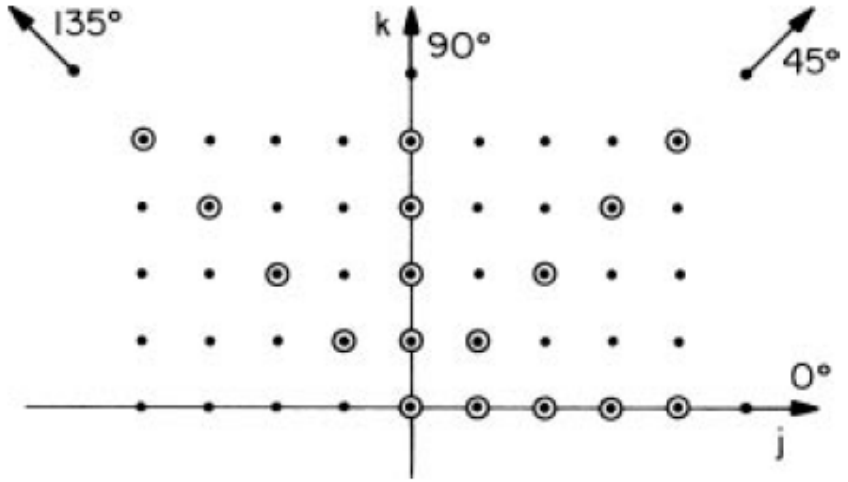
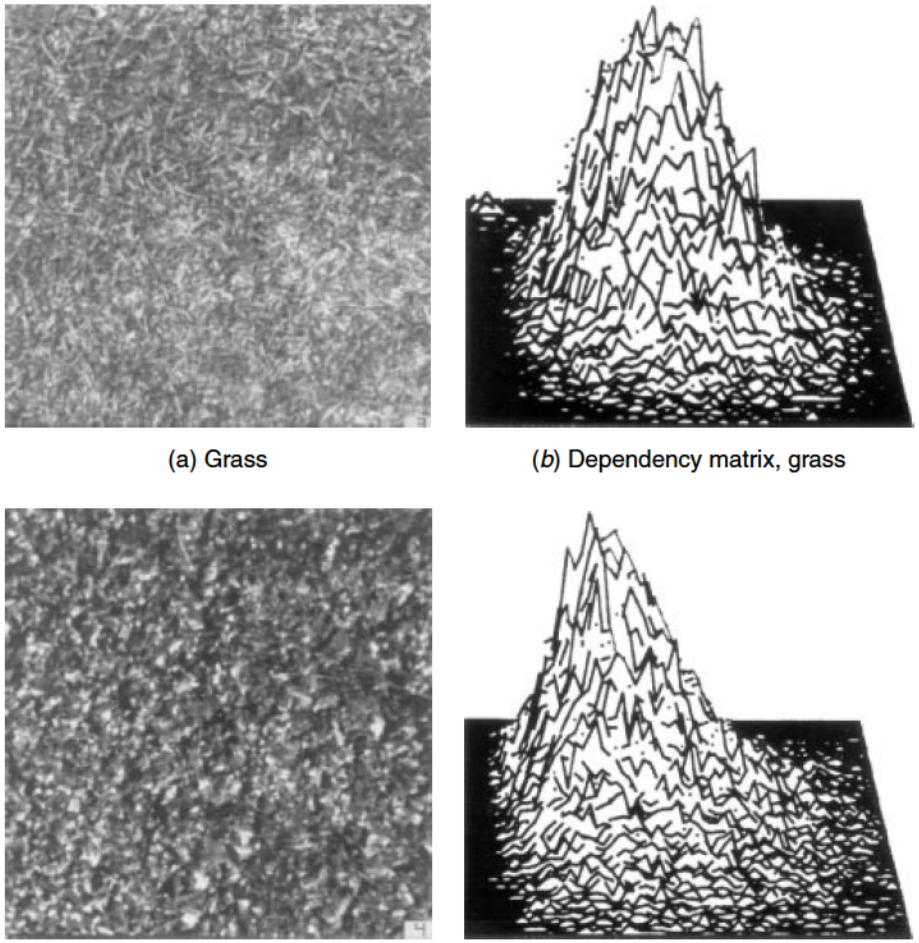


FIGURE 16.6-6. Geometry for measurement of gray scale dependency matrix.

TA by 2D Histograms

- For each member of the parameter set , the two-dimensional histogram may be regarded as a array of numbers relating the measured statistical dependency of pixel pairs
- For a given separation set , the histogram obtained for fine texture tends to be more uniformly dispersed than the histogram for coarse texture.
- Texture coarseness can be measured in terms of the relative spread of histogram occupancy cells about the main diagonal of the histogram

TA by 2D Histograms: example



(a) Grass (*b*) Dependency matrix, grass

(c) Ivy (*d*) Dependency matrix, ivy

FIGURE 16.6-7. Perspective views of gray scale dependency matrices for $r = 4$, $\theta = 0$.

TA by filter banks

$$M_i(j,k) = F(j,k) \circledast H_i(j,k)$$

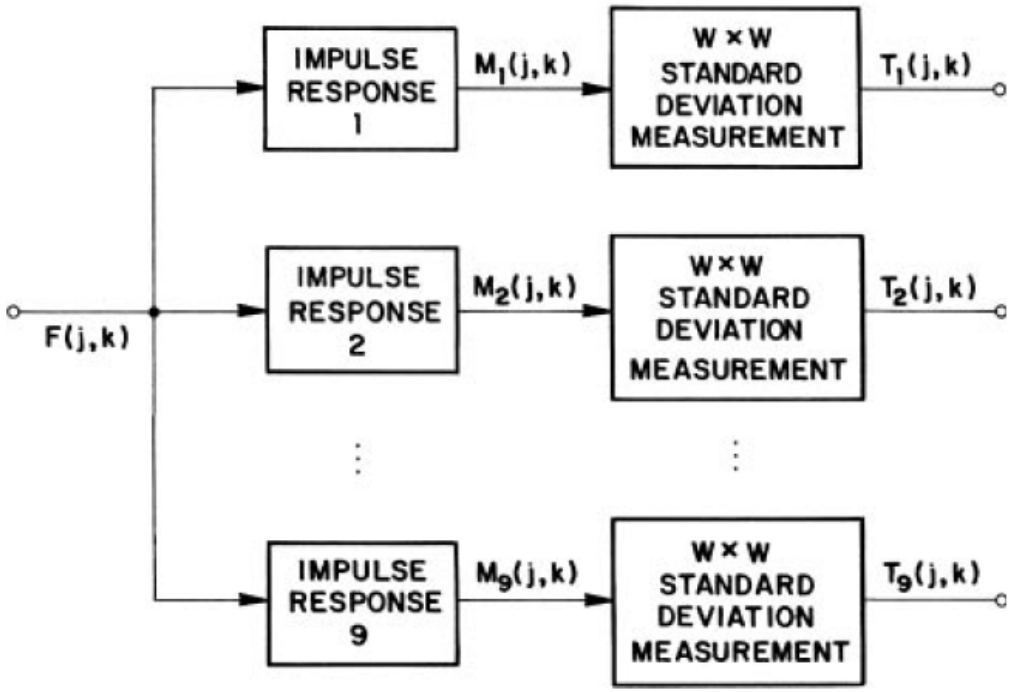


FIGURE 16.6-8. Laws microstructure texture feature extraction method.

TA by filter banks

- Advantage: spatial features in different directions are extracted
- Disadvantage: NOT scalable (filters of fixed size in F-domain)

$$\frac{1}{36}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \quad \frac{1}{12}\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} \quad \frac{1}{12}\begin{bmatrix} -1 & 2 & -1 \\ -2 & 4 & -2 \\ -1 & 2 & -1 \end{bmatrix}$$

Laws 1 Laws 2 Laws 3

$$\frac{1}{12}\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \quad \frac{1}{4}\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix} \quad \frac{1}{4}\begin{bmatrix} -1 & 2 & -1 \\ 0 & 0 & 0 \\ 1 & -2 & 1 \end{bmatrix}$$

Laws 4 Laws 5 Laws 6

$$\frac{1}{12}\begin{bmatrix} -1 & -2 & -1 \\ 2 & 4 & 2 \\ -1 & -2 & -1 \end{bmatrix} \quad \frac{1}{4}\begin{bmatrix} -1 & 0 & 1 \\ 2 & 0 & -2 \\ -1 & 0 & 1 \end{bmatrix} \quad \frac{1}{4}\begin{bmatrix} 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{bmatrix}$$

Laws 7 Laws 8 Laws 9

FIGURE 16.6-9. Laws microstructure impulse response arrays.

TA by Gabor filters

- A two-dimensional Gabor filter is a complex field sinusoidal grating that is modulated by a two-dimensional Gaussian function in the spatial domain
- Gabor filters have tunable orientation and radial frequency passbands and tunable center frequencies.

$$H(x,y) = G(x',y')\exp\{2\pi iFx'\}$$

F: parameter ruling frequency displacement

$$(x',y') = (x\cos\phi + y\sin\phi, -x\sin\phi + y\cos\phi)$$

Rotated reference frame by angle Φ

$$G(x,y) = \frac{1}{2\pi\lambda\sigma^2}\exp\left\{-\frac{(x/\lambda)^2 + y^2}{2\sigma^2}\right\}$$

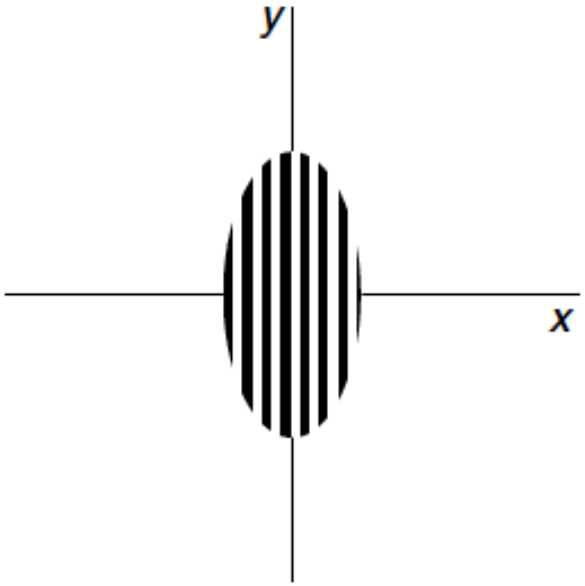
Gaussian window

$$\mathcal{H}(u,v) = \exp\{-2\pi^2\sigma^2[(u'-F)^2 + (v')^2]\}$$

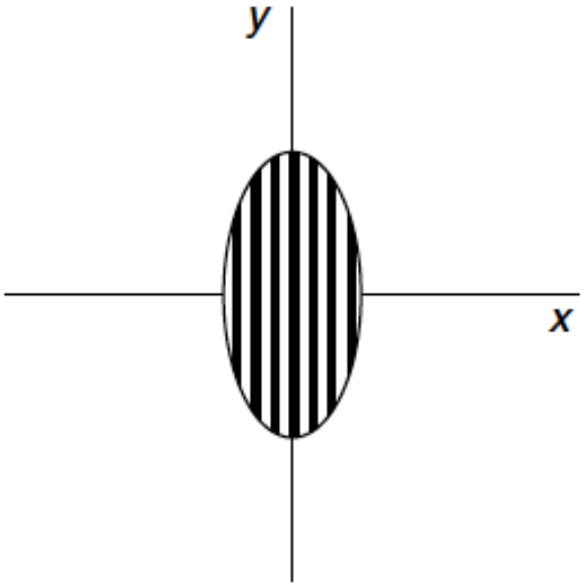
Transfer function

Gabor filters

- Impulse response (image domain)



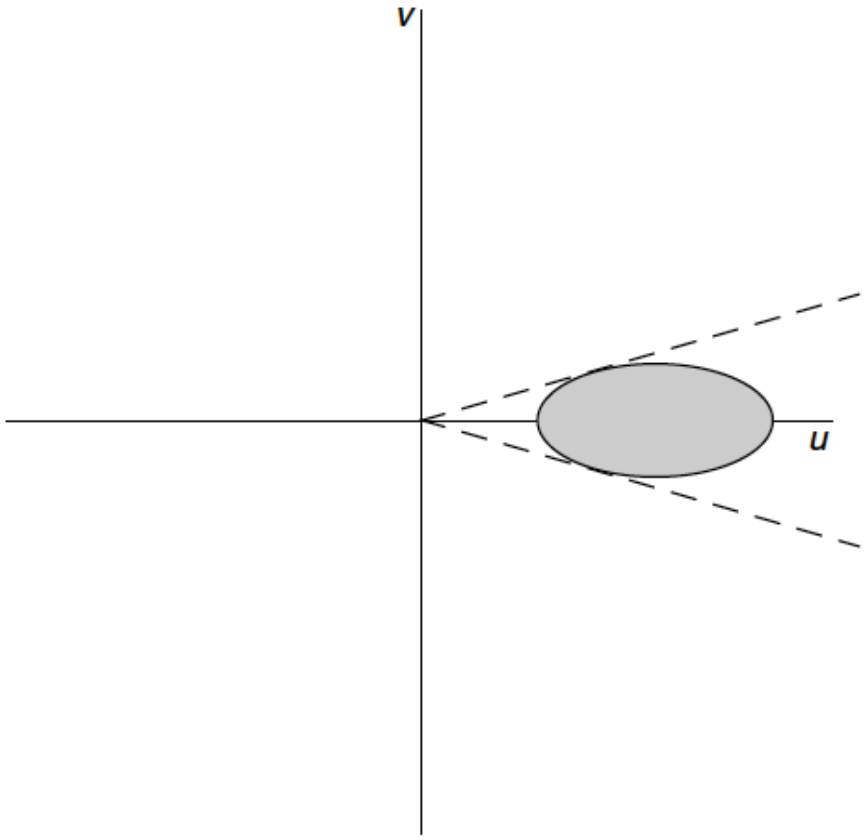
(*a*) Real part of $H(x, y)$



(*b*) Imaginary part of $H(x, y)$

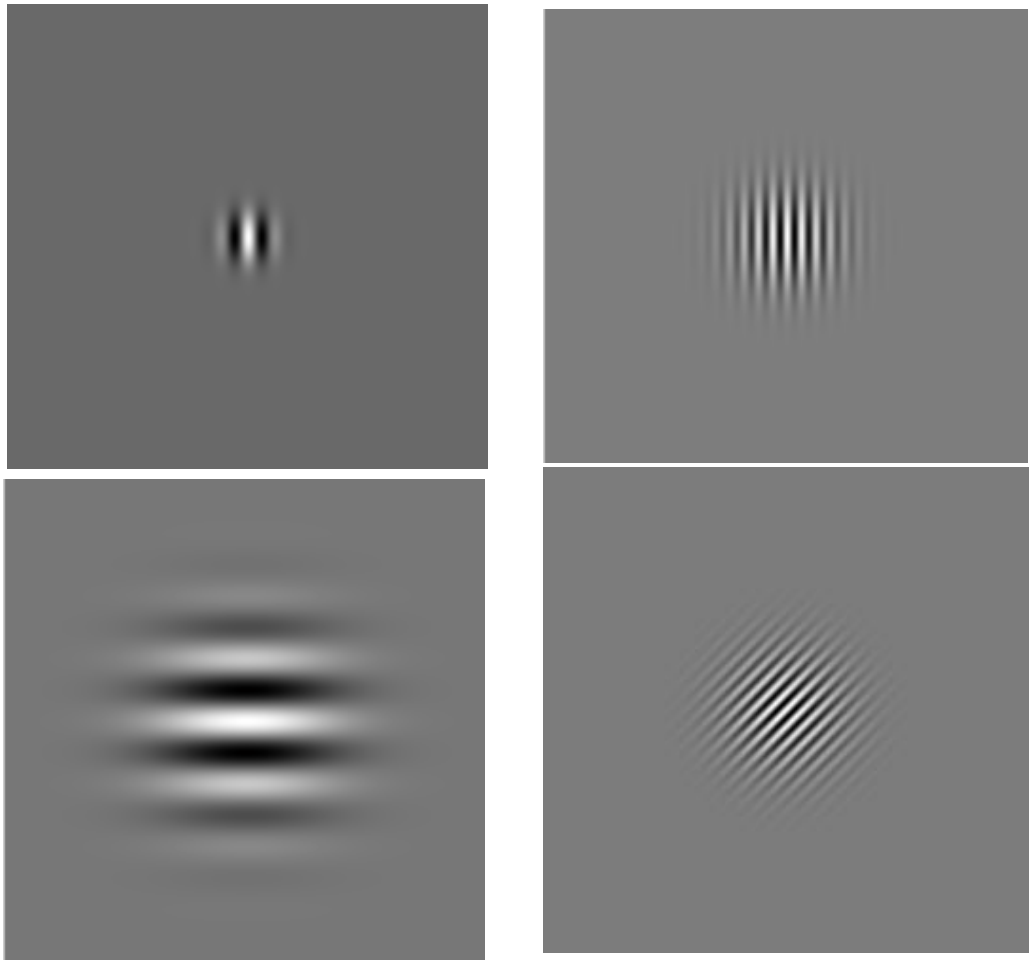
Gabor filters

- Transfer funcion (F-domain) for a given direction

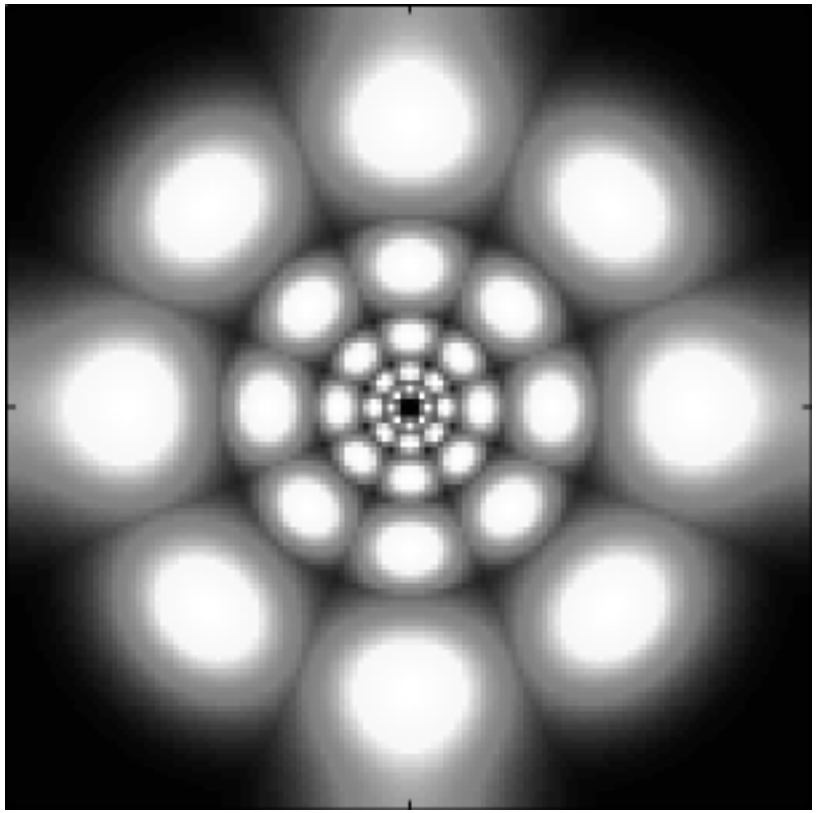


(c) $\mathcal{H}(u, v)$

Gabor filters



Dyadic Gabor filter (F-domain)



Segmentation

Region-based processing

Image Segmentation

Contour-based

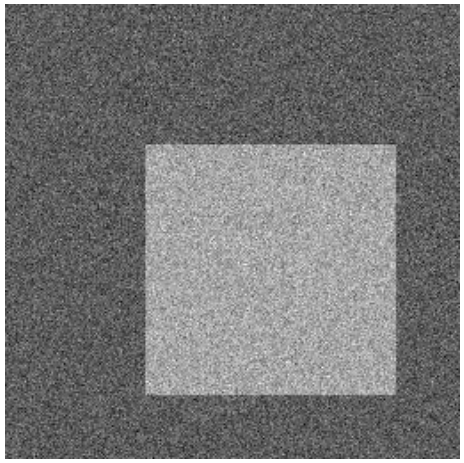
- Discontinuity
  - The approach is to partition an image based on *abrupt changes* in gray-scale levels.
  - The principal areas of interest within this category are detection of isolated points, lines, and edges in an image.

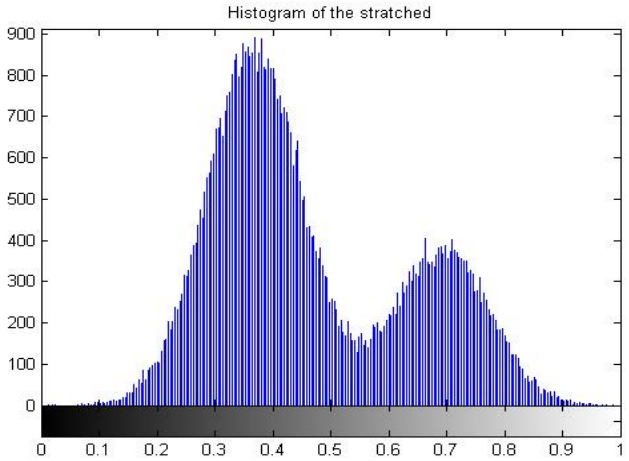
Region-based

- Similarity, homogeneity
- The principal approaches in this category are based on
  - thresholding,
  - region growing
  - region splitting/merging
  - clustering in feature space

Thresholding

- Image model
  - The objects in the image differ in the graylevel distribution
    - Simplest: object(s)+background
  - The spatial (image domain) parameters (i.e. mean, variance) are sufficient to characterize each object category
    - rests on the ergodicity assumption
  - Easily generalized to multi-spectral images (i.e. color images)





Thresholding

- Individual pixels in an image are marked as “object” pixels if their value is greater than some threshold value and as “background” pixels otherwise → *threshold above*
  - assuming an object to be brighter than the background
  - Variants
    - *threshold below*, which is opposite of threshold above;
    - *threshold inside*, where a pixel is labeled "object" if its value is between two thresholds
    - *threshold outside*, which is the opposite of threshold inside
  - Typically, an object pixel is given a value of “1” while a background pixel is given a value of “0.”
  - Finally, a binary image is created by coloring each pixel white or black, depending on a pixel's label.

Thresholding types

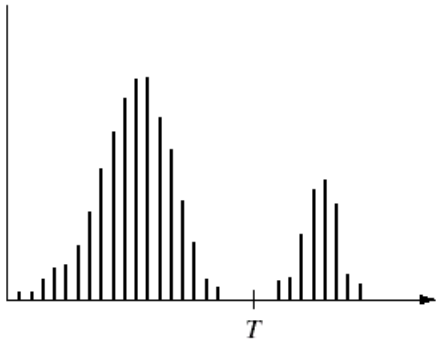
- Histogram shape-based methods
  - Peaks, valleys and curvatures of the smoothed histogram are analyzed
- Clustering-based methods
  - gray-level samples are *clustered* in two parts as background and foreground (object), or alternately are modeled as a mixture of two Gaussians
- Entropy-based methods
  - Entropy of the foreground and background regions, cross-entropy between the original and segmented image, etc.
- Object attribute-based methods
  - Based on a measure of similarity between the gray-level and the binarized images, such as fuzzy shape similarity, edge coincidence, etc.

Thresholding types

- Stochastic methods: use higher-order probability distributions and/or correlation between pixels
- *Local* or *adaptive* methods: adapt the threshold value on each pixel to the local image characteristics

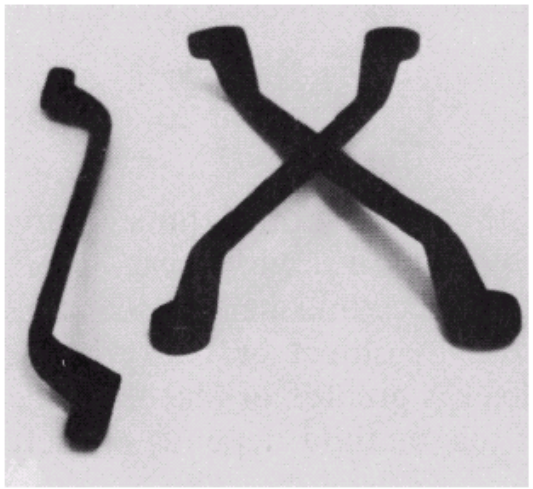
Histogram thresholding

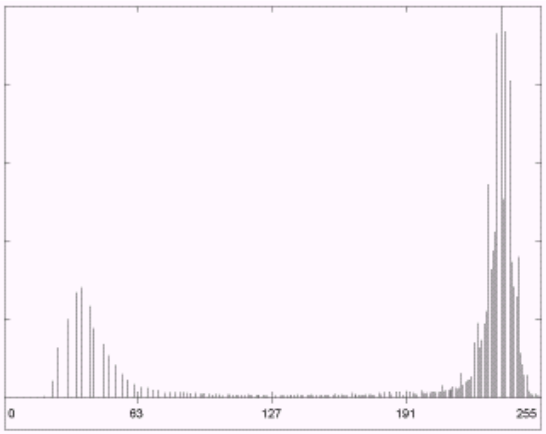
- Suppose that an image, f(x,y), is composed of light objects on a dark background, and the following figure is the histogram of the image.

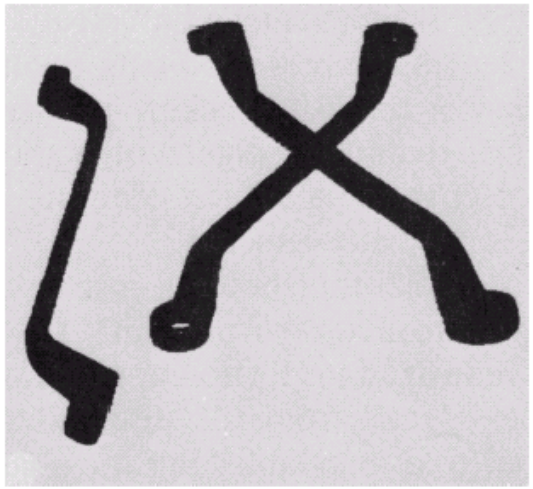


- Then, the objects can be extracted by comparing pixel values with a threshold T.

Thresholding



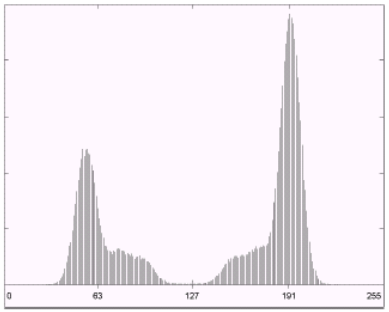


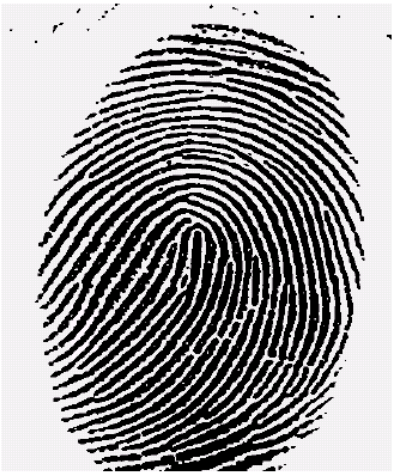


a
b c

FIGURE 10.28
(a) Original image. (b) Image histogram. (c) Result of global thresholding with T midway between the maximum and minimum gray levels.

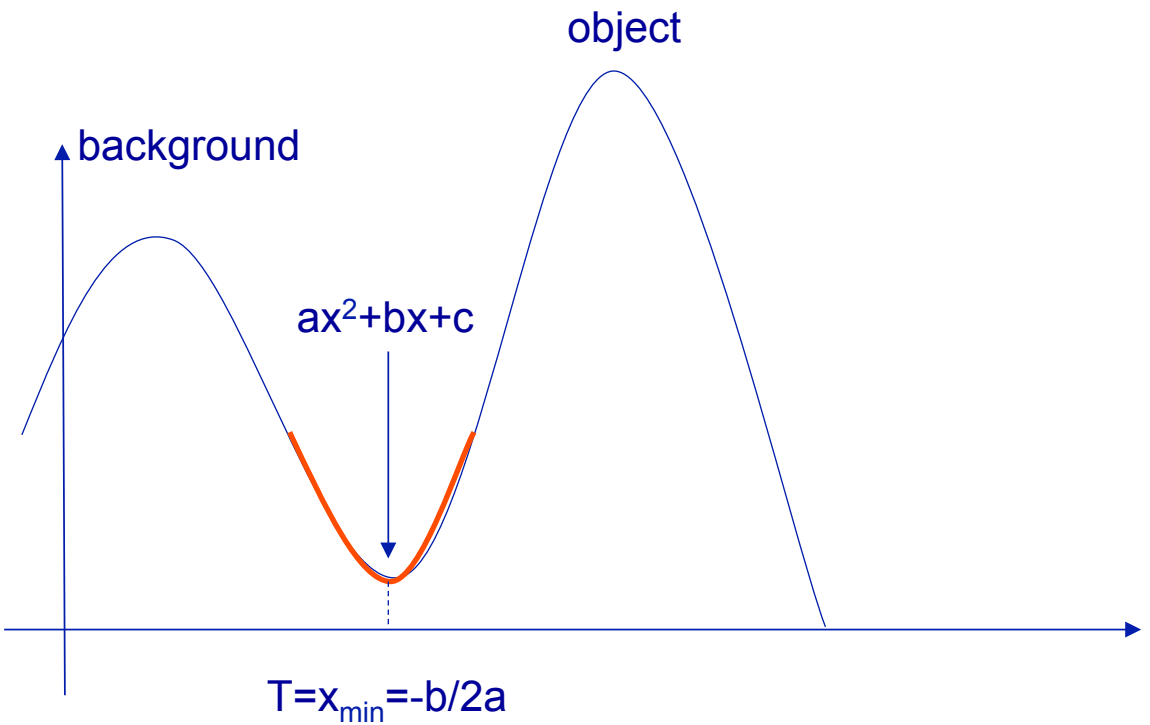
Thresholding





Histogram thresholding

- Analytical models can be fit to the valleys of the histogram and then used to find local minima



Choice of the T value

- Empirical, by inspection
- Automatic
  1. Choose an initial value T
  2. Segment the image accordingly
     - This will produce twos sets of pixels, G1 and G2

$$G_1 : g > T$$

$$G_2 : g \le T$$

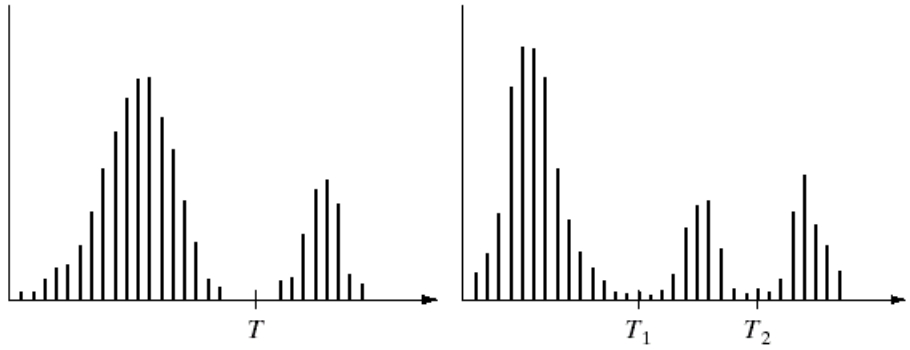
  3. Update the threshold

$$T = \frac{1}{2}(\mu_1 + \mu_2), \qquad \mu_1 = \frac{1}{|G_1|}\sum_{i,j \in G_1} g[i,j], \qquad \mu_2 = \frac{1}{|G_2|}\sum_{i,j \in G_2} g[i,j]$$

  4. Go back to 2 until the change due to the update of T reaches a lower bound ΔT_0

Multilevel luminance thresholding

- It is also possible to extract objects that have a specific intensity range using multiple thresholds.



a b

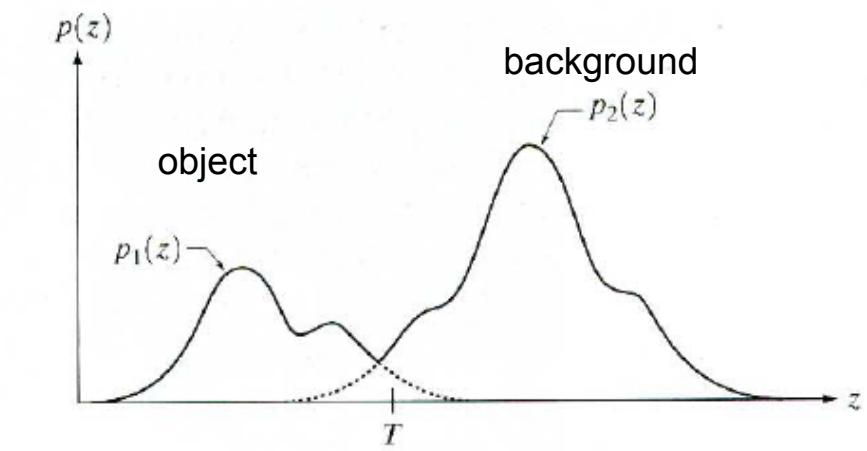
FIGURE 10.26 (a) Gray-level histograms that can be partitioned by (a) a single threshold, and (b) multiple thresholds.

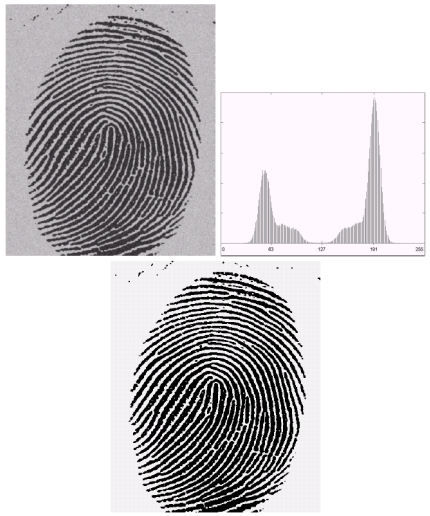
Extension to color images is straightforward: There are three color channels, in each one specifies the intensity range of the object… Even if objects are not separated in a single channel, they might be with all the channels… Application example: Detecting/Tracking faces based on skin color…

Optimal global and adaptive thresholding

- Assumptions:
  - The image contains only two principal gray level regions
  - The histogram is bimodal
  - It can be considered as a good estimate of the pdf
- Model:
  - The global histogram is given by the mixture (sum) of the two pdfs
  - The weights are proportional to the relative areas of the dark and light regions
    - And thus are given by the areas under the two, respectively

Optimal global and adaptive thresholding





Mixture pdf describing the global gray level variations in the image

$$p(z) = P_1 p_1(z) + P_2 p_2(z)$$
$$P_1 + P_2 = 1$$

Probability of errors

- Misclassification of a background point as object

$$E_1(T) = \int_{-\infty}^{T} p_2(z)\,dz$$

- Misclassification of an object point as background

$$E_2(T) = \int_{T}^{+\infty} p_1(z)\,dz$$

- Total error probability

$$E(T) = P_2 E_1(T) + P_1 E_2(T)$$

  - E_1 is weighted by P_2 because if the probability of background points is zero than the contribution to such points to the error is zero too

Finding the threshold

- Take the derivative of E with respect to T and set it to zero

$$E(T) = P_2 E_1(T) + P_1 E_2(T)$$

$$\frac{dE}{dt} = P_2 \frac{dE_1}{dT} + P_1 \frac{dE_2}{dT} = p_2 P_2 - p_1 P_1 = 0$$

$$p_1 P_1 = p_2 P_2$$

- Notes
  - If P_1=P_2 then the optimal value for T corresponds to the intersect of the curves
  - The explicit calculation of T requires the knowledge of the pdf, which is not always the case
  - In general, it is assumed that the two pdfs are Gaussian

Finding the threshold

- For Gaussian mixtures

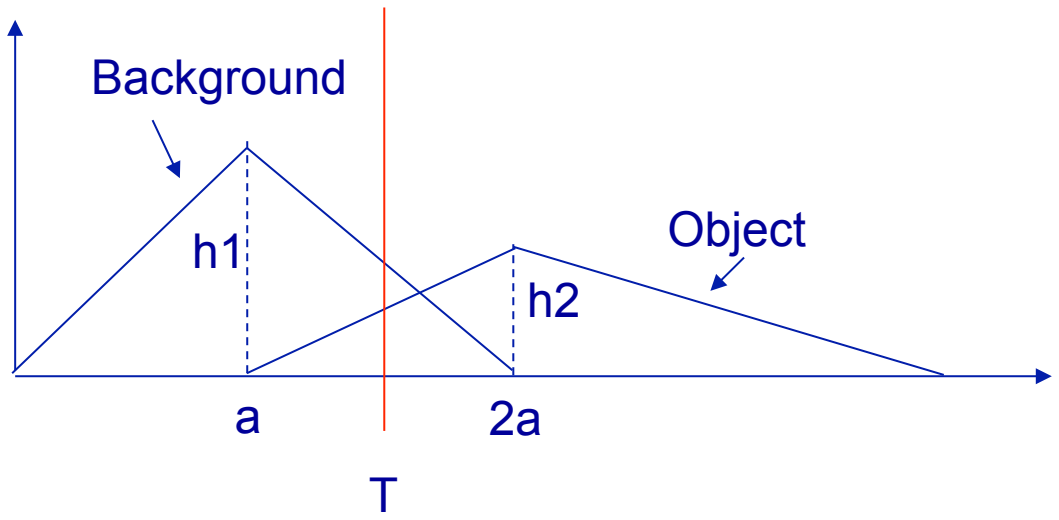
$$p(z) = \frac{P_1}{\sqrt{2\pi}\sigma_1} e^{-\frac{(z-\mu_1)^2}{2\sigma_1^2}} + \frac{P_2}{\sqrt{2\pi}\sigma_2} e^{-\frac{(z-\mu_2)^2}{2\sigma_2^2}}$$

$$\text{if} \quad \sigma = \sigma_1 = \sigma_2$$

$$\rightarrow T = \frac{\mu_1 + \mu_2}{2} \frac{\sigma^2}{\mu_1 - \mu_2} \ln\left(\frac{P_2}{P_1}\right)$$

Clustering based thresholding

- Definition and optimization of a *cost (or objective) function*
  - Cost of classifying a background pixel as an object pixel is Cb.
  - Cost of classifying an object pixel as a background pixel is Co.
  - Find the threshold, T, that minimizes the total cost.



Clustering based thresholding

- Idea 1: pick a threshold such that each pixel on each side of the threshold is **closer** in intensity to the mean of all pixels on that side of the threshold than the mean of all pixels on the other side of the threshold. Let
  - $\mu_B(T)$ = the mean of all pixels less than the threshold (background)
  - $\mu_O(T)$ = the mean of all pixels greater than the threshold (object)
- We want to find a threshold such that the grey levels for the object are closest to the average of the object and the grey levels for the background are closest to the average of the background:

$$\forall g \geq T \rightarrow |g - \mu_o(T)| < |g - \mu_B(T)|$$
$$\forall g < T \rightarrow |g - \mu_o(T)| \geq |g - \mu_B(T)|$$

Clustering based thresholding

- Idea 2: select T to minimize the *within-class* variance—the weighted sum of the variances of each cluster:

$$\sigma^2_{within}(T) = n_B(T)\sigma^2_B(T) + n_o(T)\sigma^2_o(T)$$

$$n_B(T) = \sum_{g=0}^{T-1} p(g)$$

mixture weights: normalized number of pixels in the two regions

$$n_o(T) = \sum_{g=T}^{N-1} p(g)$$

$\sigma^2_B(T)$: variance of the pixels in the background (g<T)

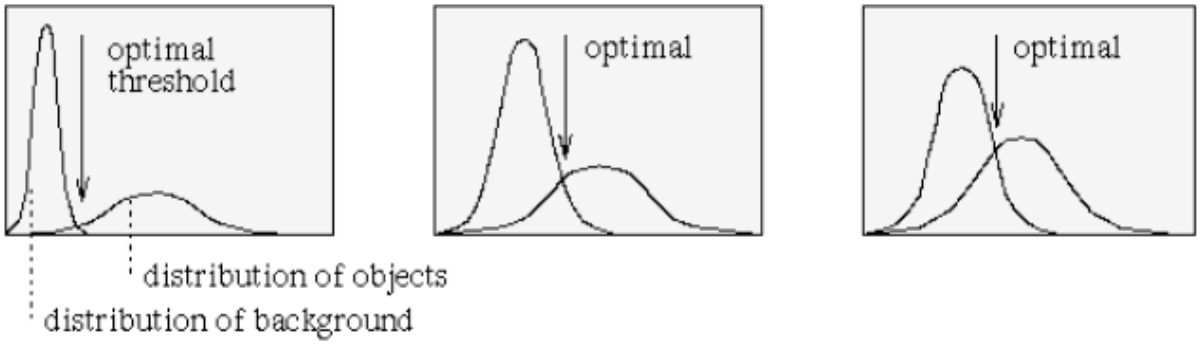
$\sigma^2_0(T)$: variance of the pixels in the object (g ≥ T)

$0,\ldots,N-1$: range of intensity levels

Clustering based thresholding

- Idea 3: Modeling the pdf as the superposition of two Gaussians and take the overlapping point as the threshold

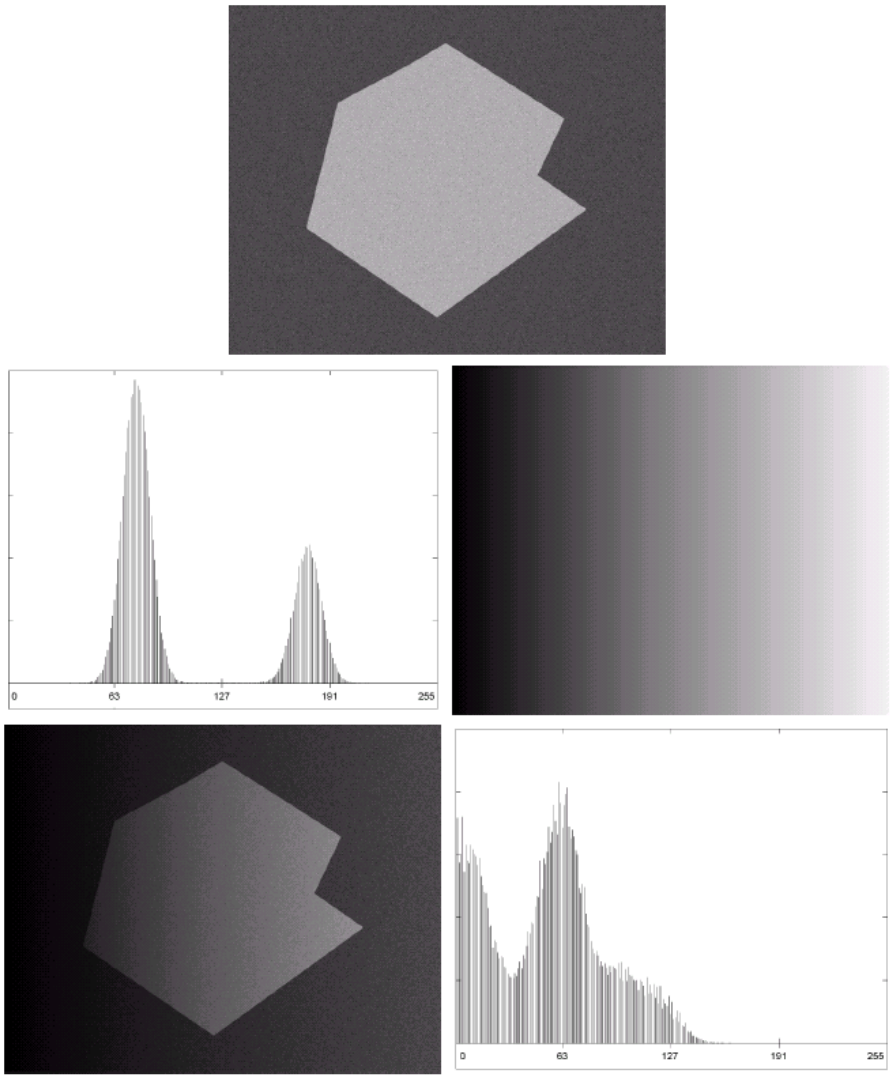
$$h(x) = P_1 p_1(x) + P_2 p_2(x) = \frac{P_1}{\sqrt{2\pi\sigma_1^2}} e^{-\frac{1}{2}\left(\frac{x-\mu_1}{\sigma_1}\right)^2} + \frac{P_2}{\sqrt{2\pi\sigma_2^2}} e^{-\frac{1}{2}\left(\frac{x-\mu_2}{\sigma_2}\right)^2}$$



Bottlenecks

- Non-uniform illumination may change the histogram in a way that it becomes impossible to segment the image using a single global threshold.
- Choosing *local threshold* values may help
- Guideline: *partition the image in blocks* of almost uniform luminance and perform the segmentation locally
  - In alternative, one can apply chromatic adaptation transforms which compensate for differences in the scene illumination, such as retinex

Examples



a
b c
d e

FIGURE 10.27
(a) Computer generated reflectance function.
(b) Histogram of reflectance function.
(c) Computer generated illumination function.
(d) Product of (a) and (c).
(e) Histogram of product image.

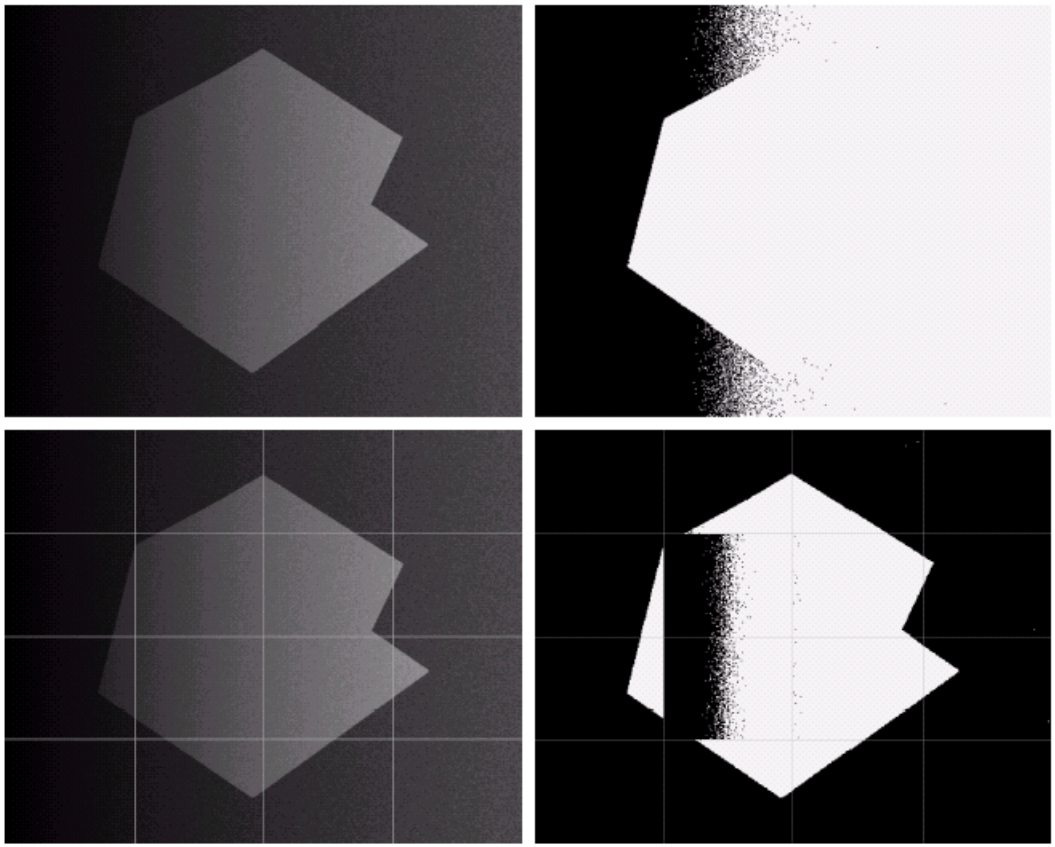
Examples

- Adaptive thresholding

Global thresholding

a b
c d

FIGURE 10.30
(a) Original image. (b) Result of global thresholding. (c) Image subdivided into individual subimages. (d) Result of adaptive thresholding.



Adaptive thresholding

Region based segmentation qui

- Formulation

$$\bigcup_{i=1}^{n} R_i = R$$ the segmentation must be complete

R_i is a connected region $i = 1,\ldots,n$ the points in a region must be connected according to a predefined criterion

$R_i \cap R_j = \varnothing \qquad \forall i \neq j$ the regions must be disjoint

$PR(R_i) = TRUE \qquad i = 1,\ldots,n$ condition that is satisfied by all points in R_i

$PR(R_i \cup R_j) = FALSE \qquad \forall i \neq j$ regions Ri and Rj are different with respect to predicate PR

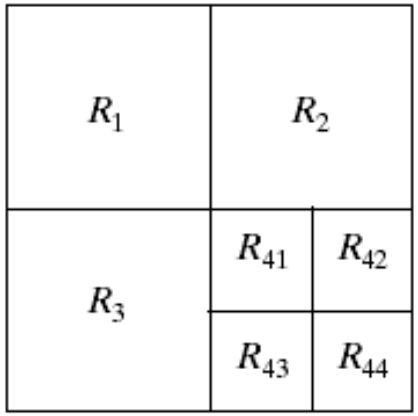
PR: logical predicate defined over the region. Ex: all points in the region have the same gray level

Region-Oriented Segmentation

- Region Growing
  - Region growing is a procedure that groups pixels or subregions into larger regions.
  - The simplest of these approaches is pixel aggregation, which starts with a set of “seed” points and from these grows regions by appending to each seed points those neighboring pixels that have similar properties (such as gray level, texture, color, shape).
  - Region growing based techniques are better than the edge-based techniques in noisy images where edges are difficult to detect.

Region-Oriented Segmentation

- Region Splitting
  - Region growing starts from a set of seed points.
  - An alternative is to start with the whole image as a single region and subdivide the regions that do not satisfy a condition of homogeneity.
- Region Merging
  - Region merging is the opposite of region splitting.
  - Start with small regions (e.g. 2x2 or 4x4 regions) and merge the regions that have similar characteristics (such as gray level, variance).
- Typically, splitting and merging approaches are used iteratively.



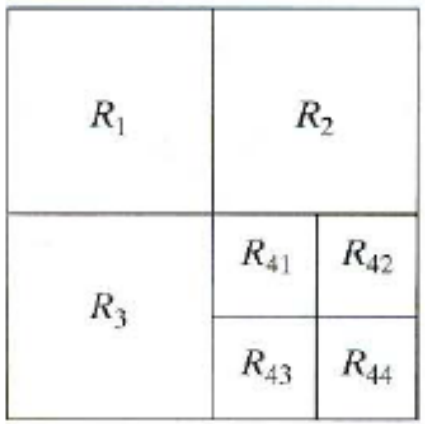
Region-Oriented Segmentation

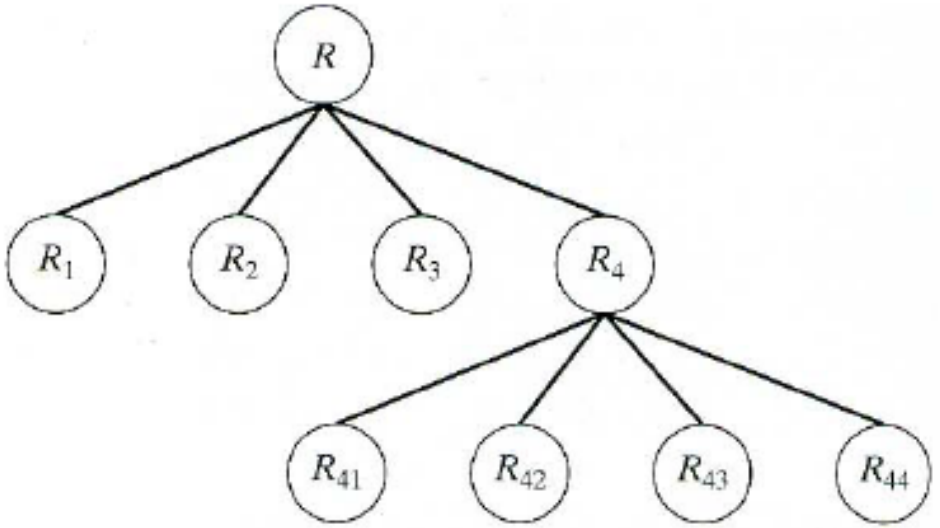
- Region splitting and merging
  - The image is initially splitted into regions arbitrarily. These are subsequently merged and/or further splitted to satisfy the predefined homogeneity criterion
  - Let R be a region and PR a predicate. The approach consists in taking initially R=entire image and splitting it in subregions such that at the end of the process $PR(R_i)$=TRUE in every region.
  - Recipe:

1. Evaluate PR over R: if it is FALSE then split R in, let's say, 4 subregions
2. Repeat the procedure for each resulting region
3. For each couple i,j evaluate $PR(R_i \cup R_j)$. If this is TRUE then merge R_i and R_j
4. Stop when no further splitting or merging is possible

Region splitting and merging

- Image *quadtree* resulting for the considered type of splitting

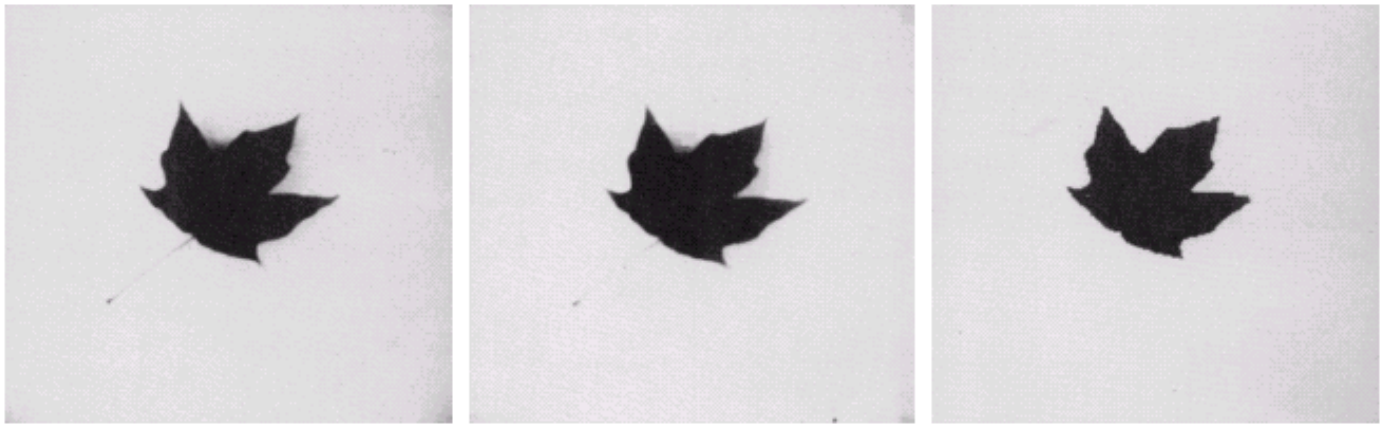




Region-Oriented Segmentation

a b c

FIGURE 10.43
(a) Original image. (b) Result of split and merge procedure. (c) Result of thresholding (a).



Towards texture segmentation

- All the methods using means and variances to characterize regions basically characterize the *texture* of the region
- The concept of texture segmentation consists in using *texture features* as predicates

Example

Suppose that we have the image given below.

(a) Use the region growing idea to segment the object. The seed for the object is the center of the image. Region is grown in horizontal and vertical directions, and when the difference between two pixel values is less than or equal to 5.

Table 1: Show the result of Part (a) on this figure.

10	10	10	10	10	10	10
10	10	10	69	70	10	10
59	10	60	64	59	56	60
10	59	10	60	70	10	62
10	60	59	65	67	10	65
10	10	10	10	10	10	10
10	10	10	10	10	10	10

Example

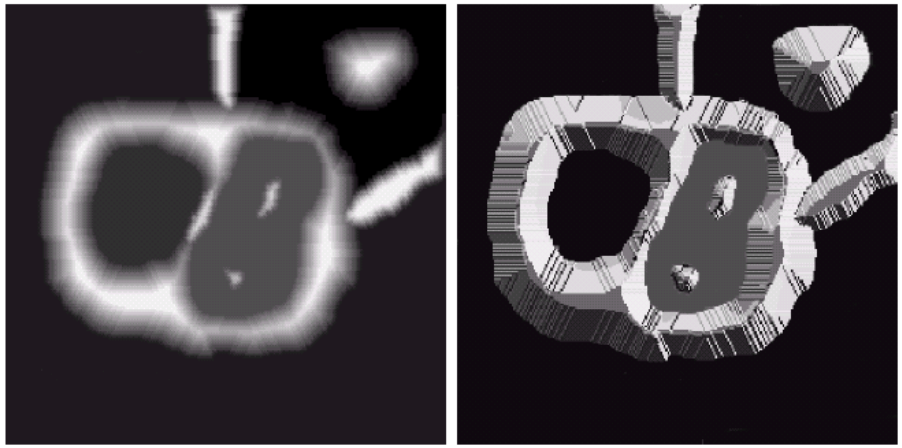
(b) What will be the segmentation if region is grown in horizontal, vertical, and diagonal directions?

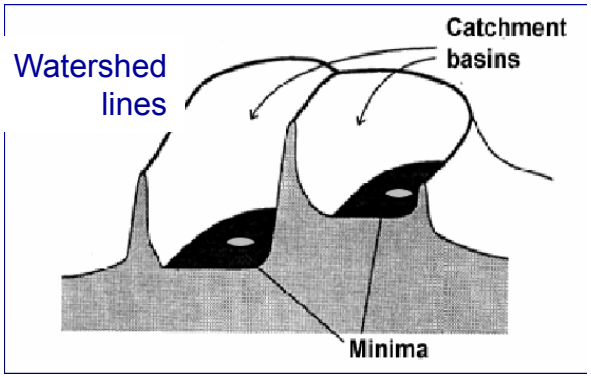
Table 2: Show the result of Part (b) on this figure.

10	10	10	10	10	10	10
10	10	10	69	70	10	10
59	10	60	64	59	56	60
10	59	10	60	70	10	62
10	60	59	65	67	10	65
10	10	10	10	10	10	10
10	10	10	10	10	10	10

Watershed Segmentation Algorithm

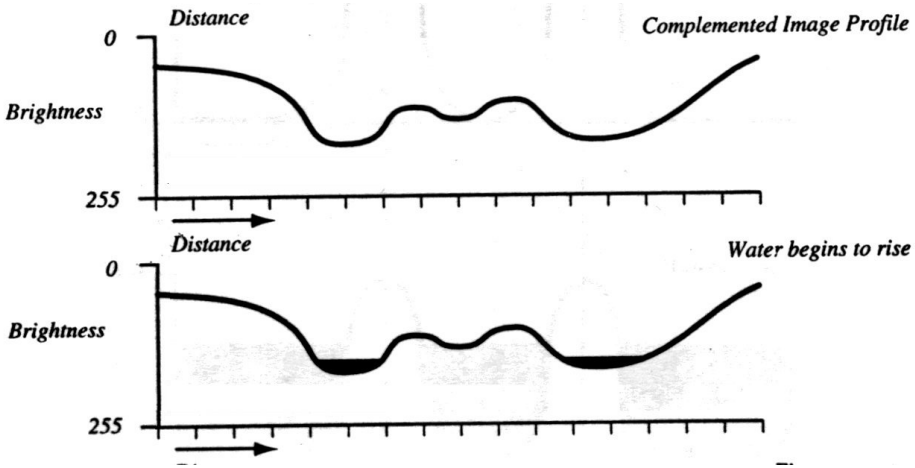
- Visualize an image in 3D: spatial coordinates and gray levels.
- In such a topographic interpretation, there are 3 types of points:
  - Points belonging to a **regional minimum**
  - Points at which a drop of water would fall to a single minimum. (→The *catchment basin* or ***watershed*** of that minimum.)
  - Points at which a drop of water would be equally likely to fall to more than one minimum. (→The *divide lines* or ***watershed lines***.)



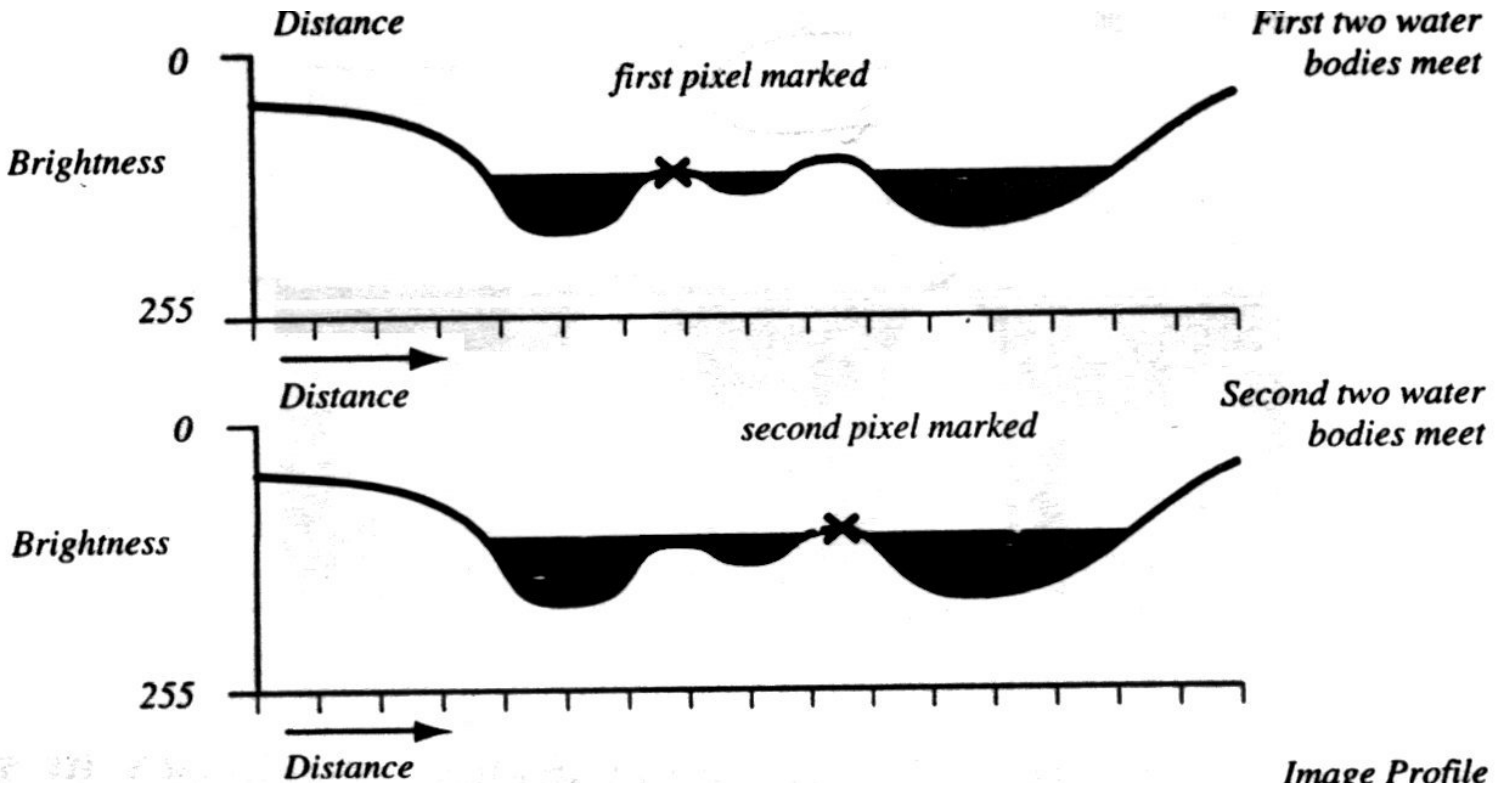


Watershed Segmentation Algorithm

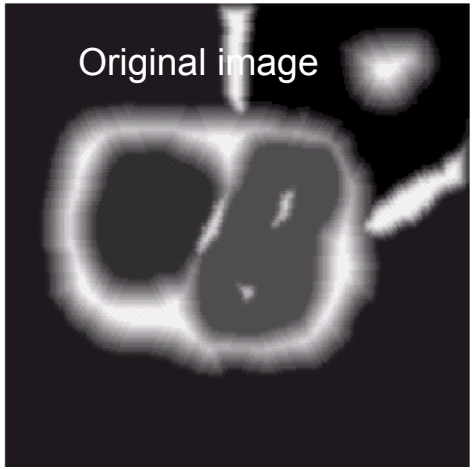
- *The objective is to find watershed lines.*
- The idea is simple:
  - Suppose that a hole is punched in each regional minimum and that the entire topography is flooded from below by letting water rise through the holes at a uniform rate.
  - When rising water in distinct catchment basins is about to merge, a dam (*diga*) is built to prevent merging.
  - Dam boundaries correspond to the watershed lines.

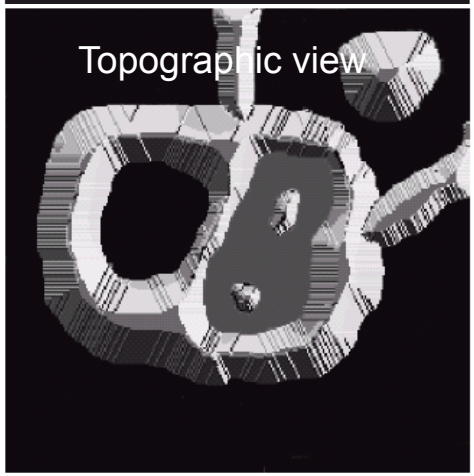


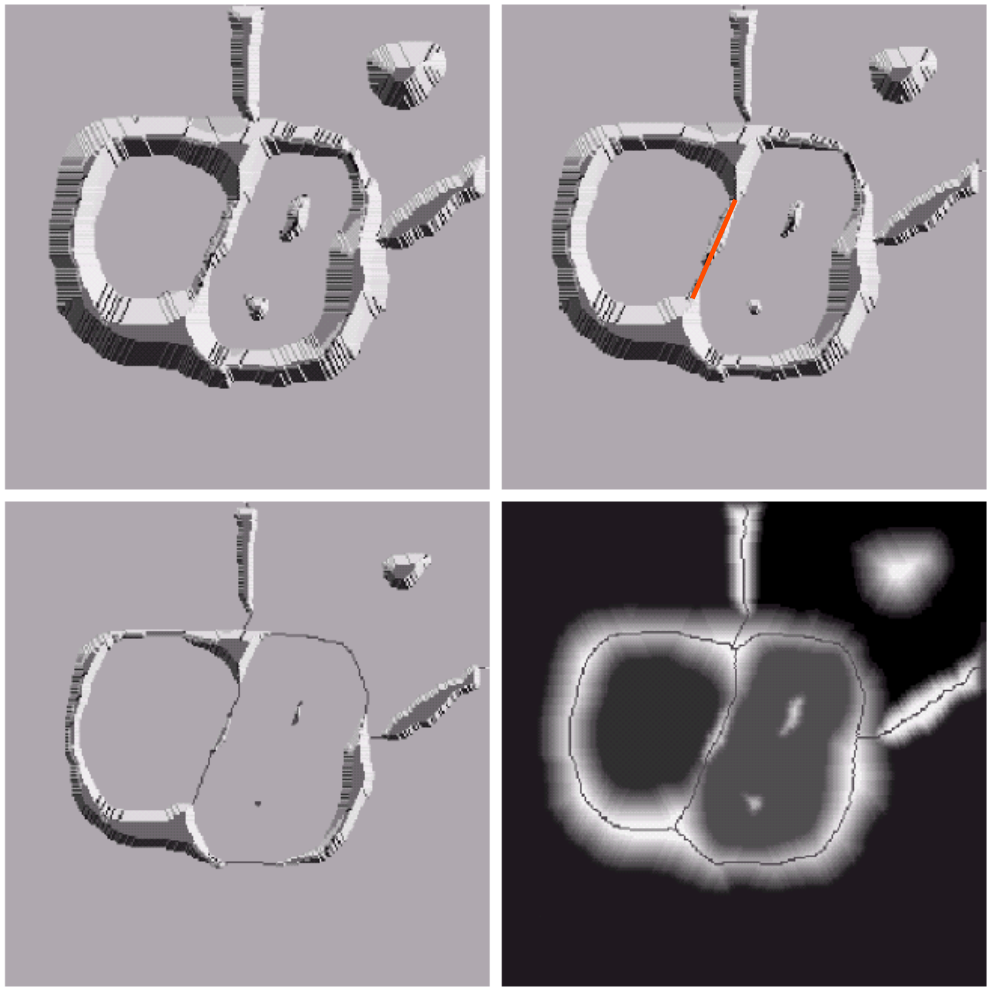
Watershed Segmentation Algorithm



Watershed Segmentation Algorithm







e f
g h

FIGURE 10.44 *(Continued)* (e) Result of further flooding. (f) Beginning of merging of water from two catchment basins (a short dam was built between them). (g) Longer dams. (h) Final watershed (segmentation) lines. (Courtesy of Dr. S. Beucher, CMM/Ecole des Mines de Paris.)

Watershed Segmentation Algorithm

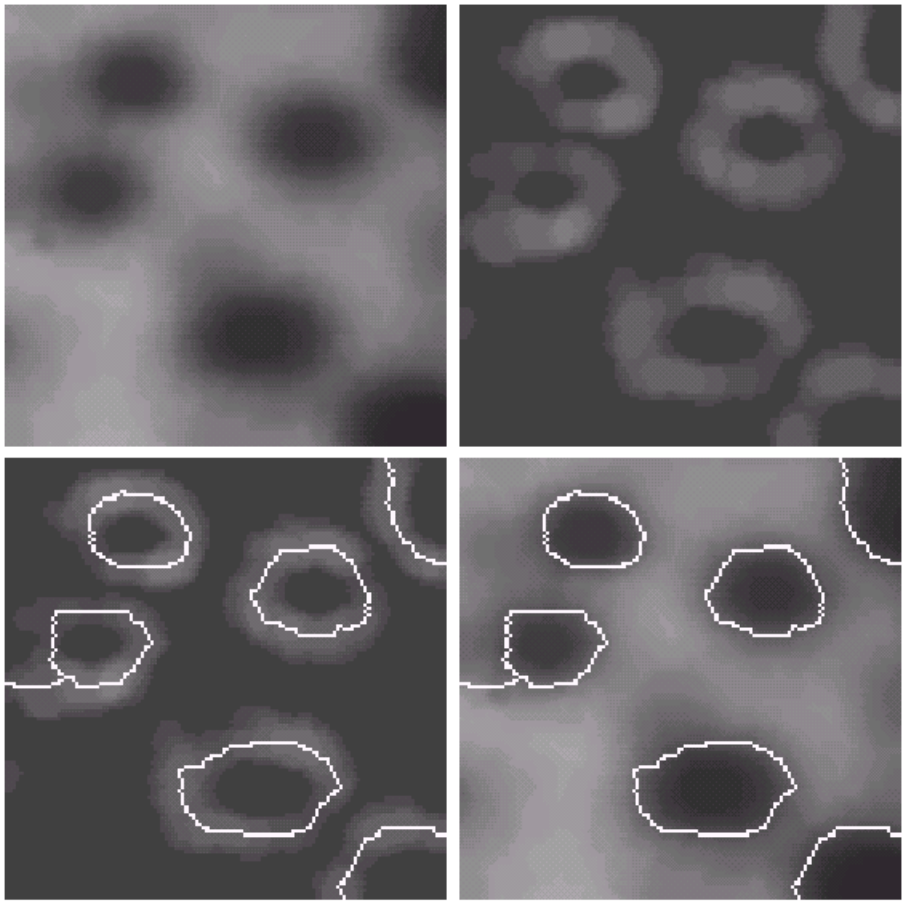
- Start with all pixels with the lowest possible value.
  - These form the basis for initial watersheds
- *For each intensity level k*:
  - For each group of pixels of intensity k
    - If adjacent to exactly one existing region, add these pixels to that region
    - Else if adjacent to more than one existing regions, mark as boundary
    - Else start a new region

Watershed Segmentation Algorithm

Watershed algorithm might be used on the gradient image instead of the original image.

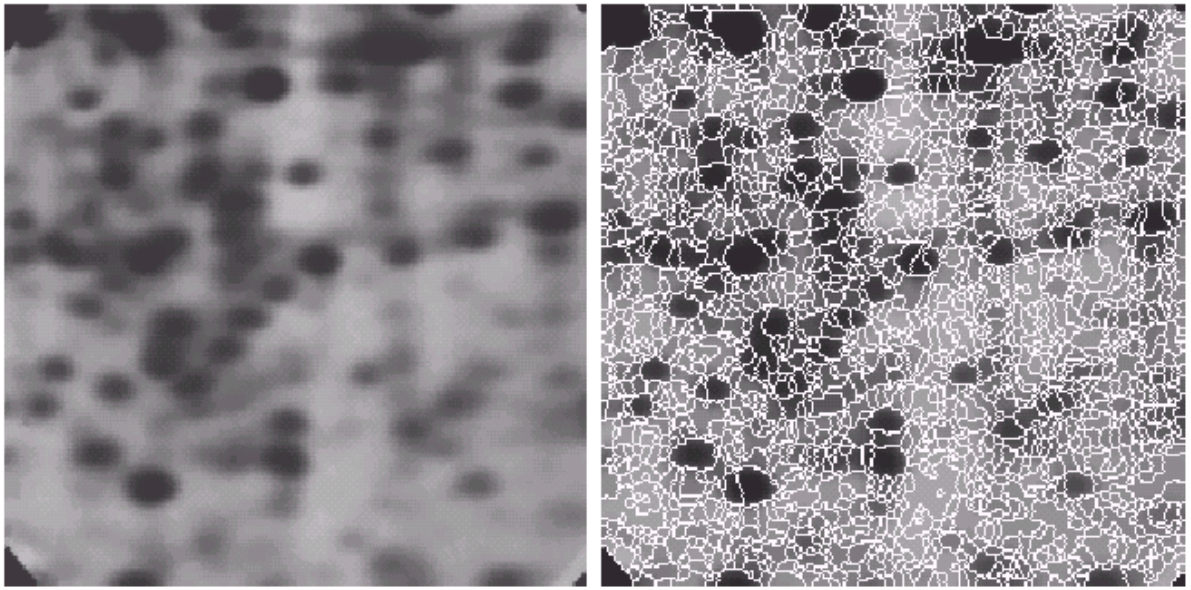
a b
c d

FIGURE 10.46
(a) Image of blobs. (b) Image gradient. (c) Watershed lines. (d) Watershed lines superimposed on original image. (Courtesy of Dr. S. Beucher, CMM/Ecole des Mines de Paris.)



Watershed Segmentation Algorithm

Due to noise and other local irregularities of the gradient, over-segmentation might occur.

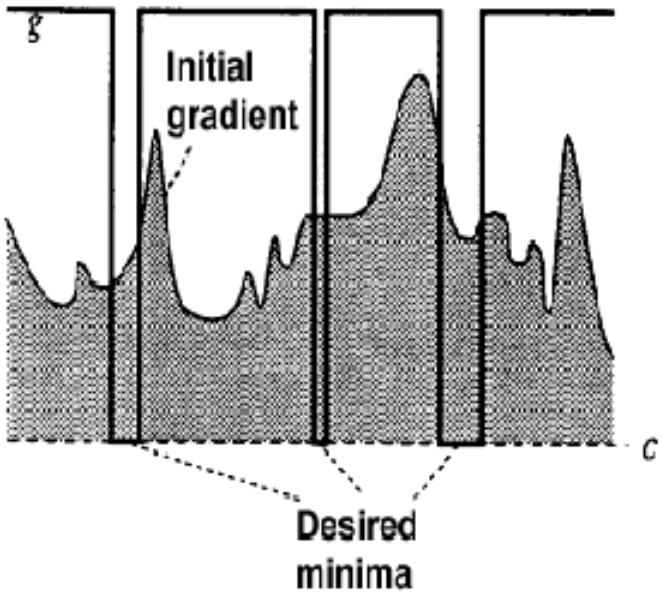


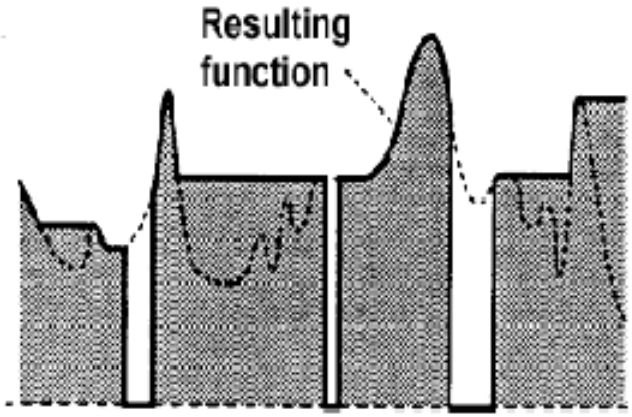
a b

FIGURE 10.47
(a) Electrophoresis image. (b) Result of applying the watershed segmentation algorithm to the gradient image. Oversegmentation is evident. (Courtesy of Dr. S. Beucher, CMM/Ecole des Mines de Paris.)

Supervised Watershed Segmentation

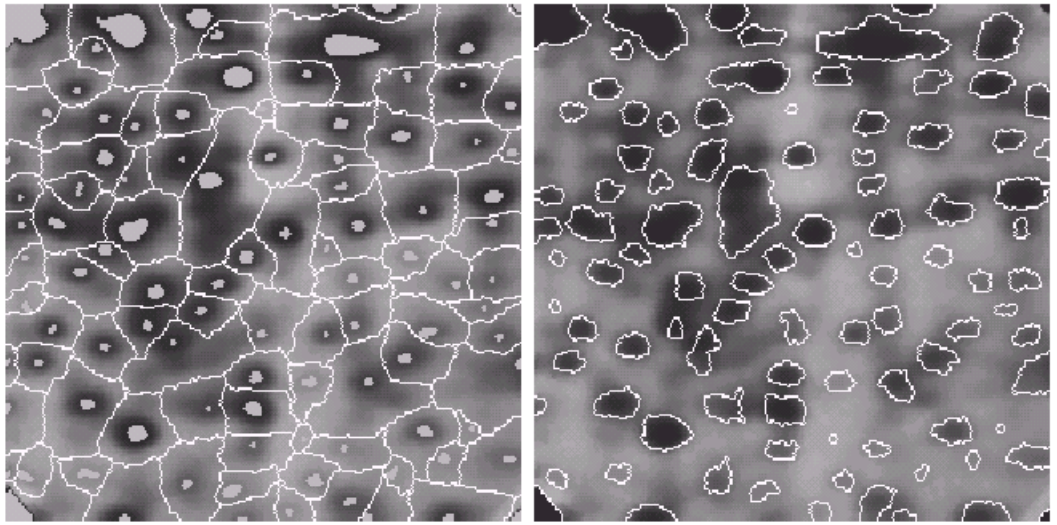
A solution is to limit the number of regional minima. Use markers to specify the only allowed regional minima.





Watershed Segmentation Algorithm

A solution is to limit the number of regional minima. Use markers to specify the only allowed regional minima. (For example, gray-level values might be used as a marker.)



a b

FIGURE 10.48
(a) Image showing internal markers (light gray regions) and external markers (watershed lines). (b) Result of segmentation. Note the improvement over Fig. 10.47(b). (Courtesy of Dr. S. Beucher, CMM/Ecole des Mines de Paris.)

A detailed description of the algorithm can be found in Gonzalez, Chapt. 10.

Use of Motion In Segmentation

Take the difference between a reference image and a subsequent image to determine the still elements image components.

a b c

FIGURE 10.50 Building a static reference image. (a) and (b) Two frames in a sequence. (c) Eastbound automobile subtracted from (a) and the background restored from the corresponding area in (b). (Jain and Jain.)

Motion detection and estimation

Alan Bovik

Handbook of image and Video Processing

Chapter 3.10

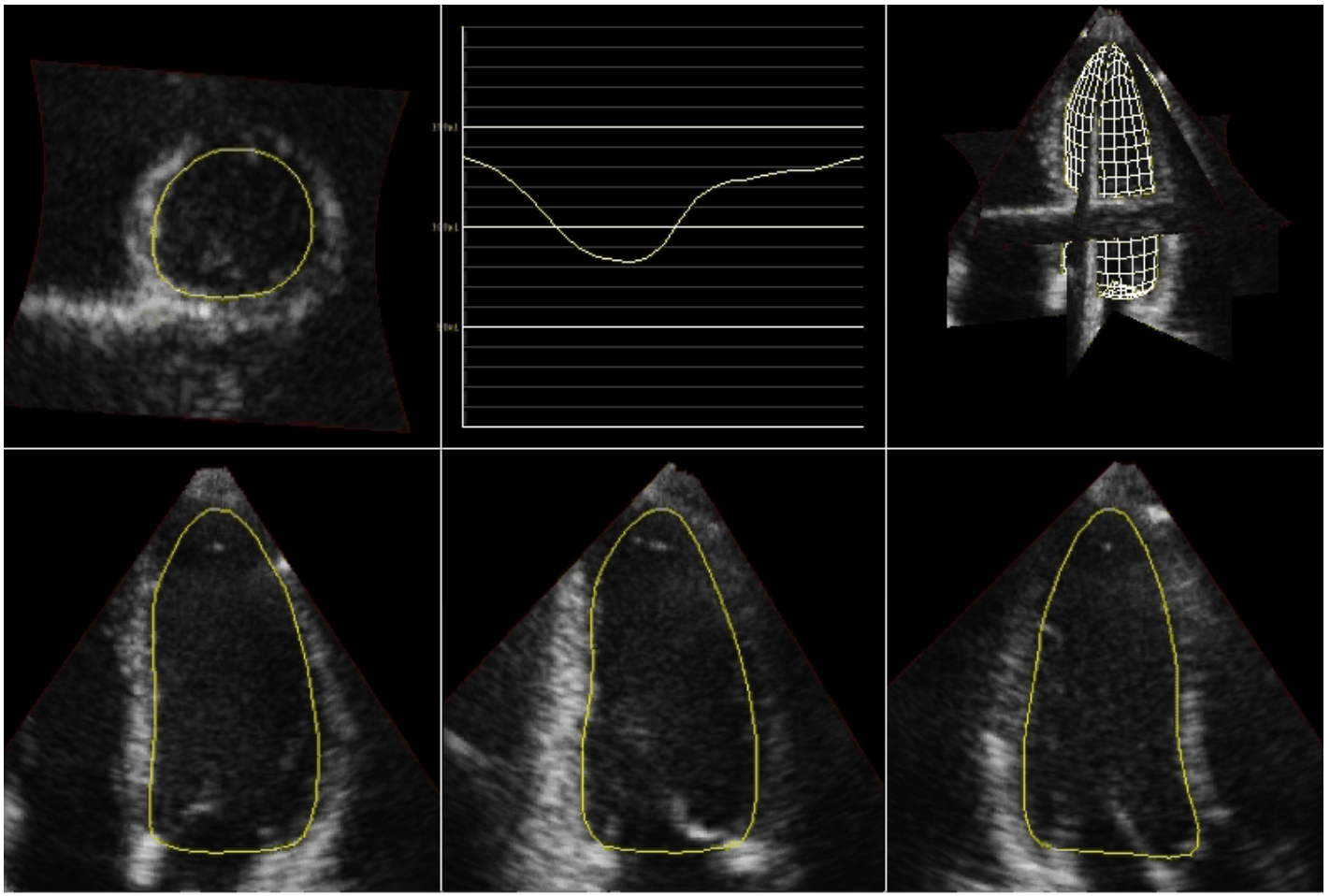
Motion detection and estimation

- A video sequence is a much richer source of visual information than a still image.
  - This is primarily due to the capture of motion; while a single image provides a snapshot of a scene, a sequence of images registers the dynamics in it.
  - The registered motion is a very strong cue for human vision; we can easily recognize objects as soon as they move even if they are inconspicuous when still.
- Main applications
  - Video analysis (through feature extraction)
  - Video compression and coding (MPEG4)
  - Investigation of the dynamics of human organs

Why is it important?

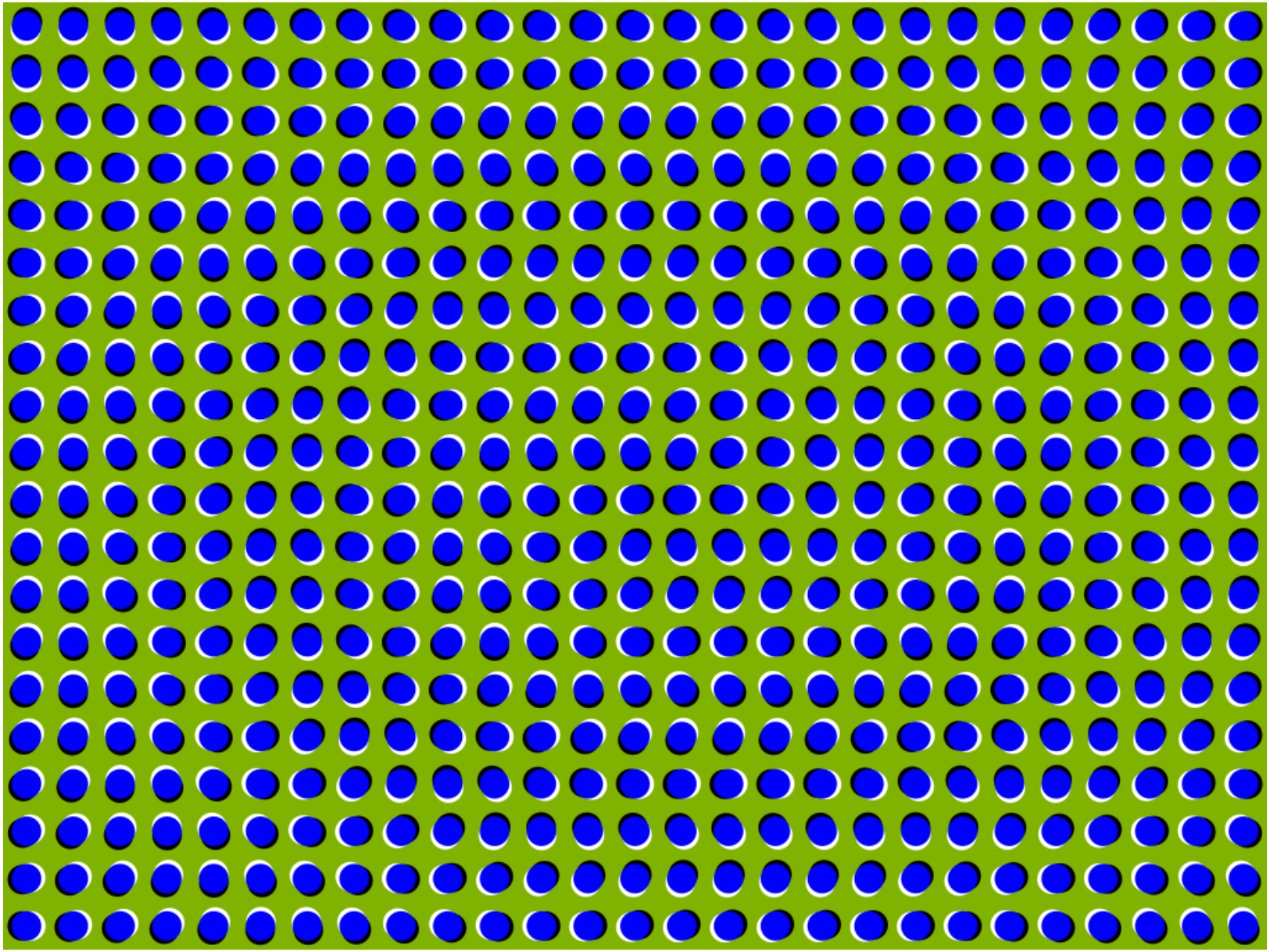
- First, motion carries a lot of information about *spatiotemporal relationships* between image objects. This information can be used in such applications as traffic monitoring or security surveillance, for example to identify objects entering or leaving the scene or objects that just moved.
- Secondly, image properties, such as intensity or color, have a very high correlation in the direction of motion, i.e., they do not change significantly when tracked in the image (the color of a car does not change as the car moves across the image). This can be used for the removal of temporal video redundancy;

Example: Left ventricle dynamics from US



Basic operations

- Motion detection: do the points (or objects) move?
- Motion estimation: how do they move?
- Special case: apparent motion
  - One object displayed at different positions in different time instants is perceived as a moving object



Motion based segmentation

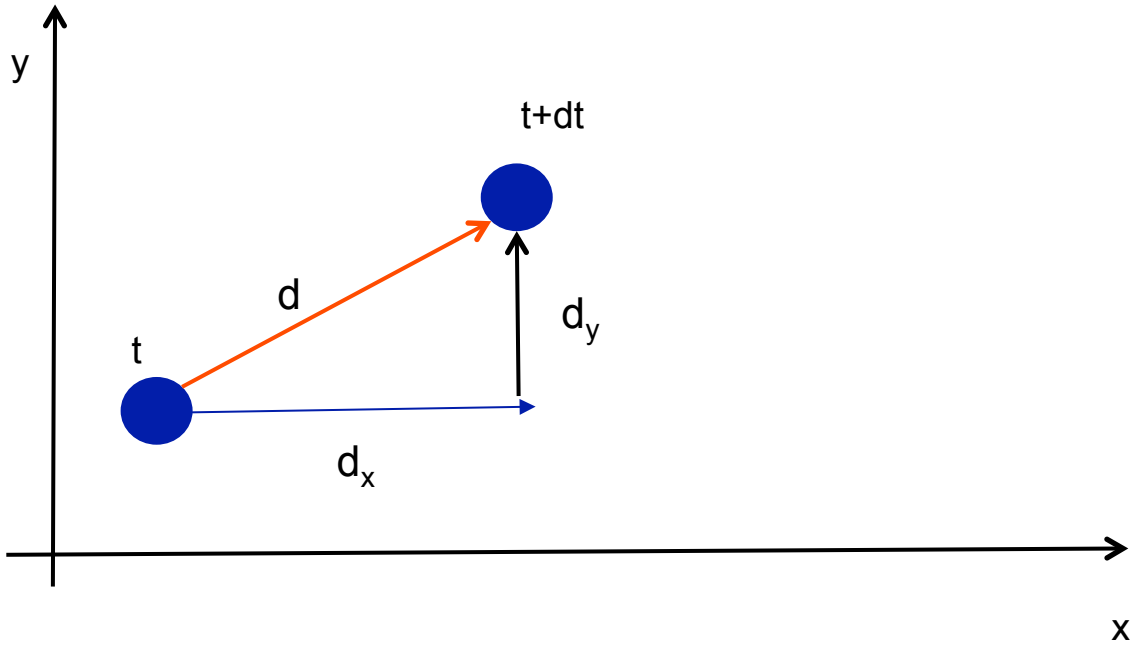
- Motion segmentation, i.e., the identification of groups of image points moving similarly
- Concept: detect the changes from one image to the next
- Possible approaches
  - Taking image differences
  - Block matching
  - Optical flow

Notions and preliminaries

- Let $\mathbf{x} = (x, y)^T$ be a spatial position of a pixel in continuous coordinates, i.e., x is in R^2 within image limits, and let I_t denote image intesity at time t
- After sampling, x is discrete and lives in Z^2
- Let v(x,y) be the velocity at the spatial position (x,y). Then, v_t will denote a velocity field or motion field, i.e., the set of all velocity vectors within the image, at time t
- For discrete images, the notion of velocity is replaced by displacement d, but the meaning is unchanged since d represents the displacement of pixel (x,y) between two time instants t1 and t2 thus it is representative of its “velocity”

Displacement / velocity

$$P\left(x+d_x, y+d_y\right)_{t+\Delta_t} = P\left(x,y\right)_t$$



Motion estimation

- Pixel differences
- Optical flow
- Block matching

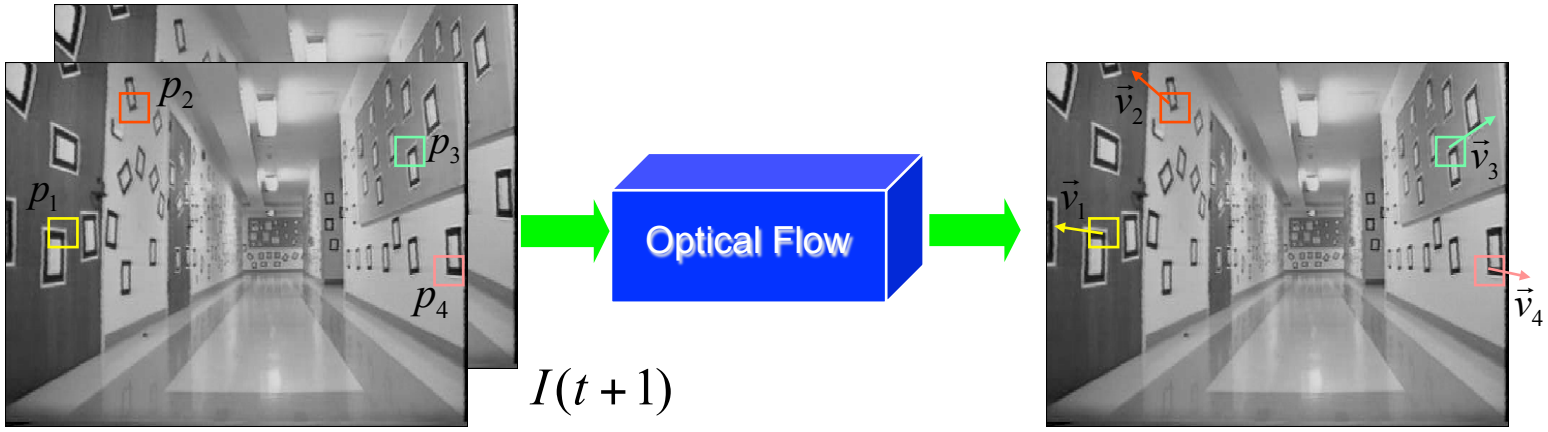
Difference images

- Difference image between two images taken at time points ti and tj

$$d_{ij}(x, y) = \begin{cases} 1 & \text{if } |f(x, y, t_i) - f(x, y, t_j)| > T \\ 0 & \text{otherwise} \end{cases}$$

  - d_{ij}=1 only if the difference between the pixel values in the two images are above a given threshold T
  - d_{ij} has the same size as the two images
- Drawbacks
  - Sensitivity to noise
    - *Accumulation* strategies can be devised
  - Only allows to detect motion but not to characterize it
    - This would require establishing correspondences among pixels to calculate *motion vectors*

What is Optical Flow?



$I(t), \{p_i\}$ $I(t+1)$ **Velocity vectors** $\{\vec{v}_i\}$

Optical flow: *2D projection of the physical movement of points relative to the observer* to 2D displacement of pixel patches on the image plane.

Common assumption:

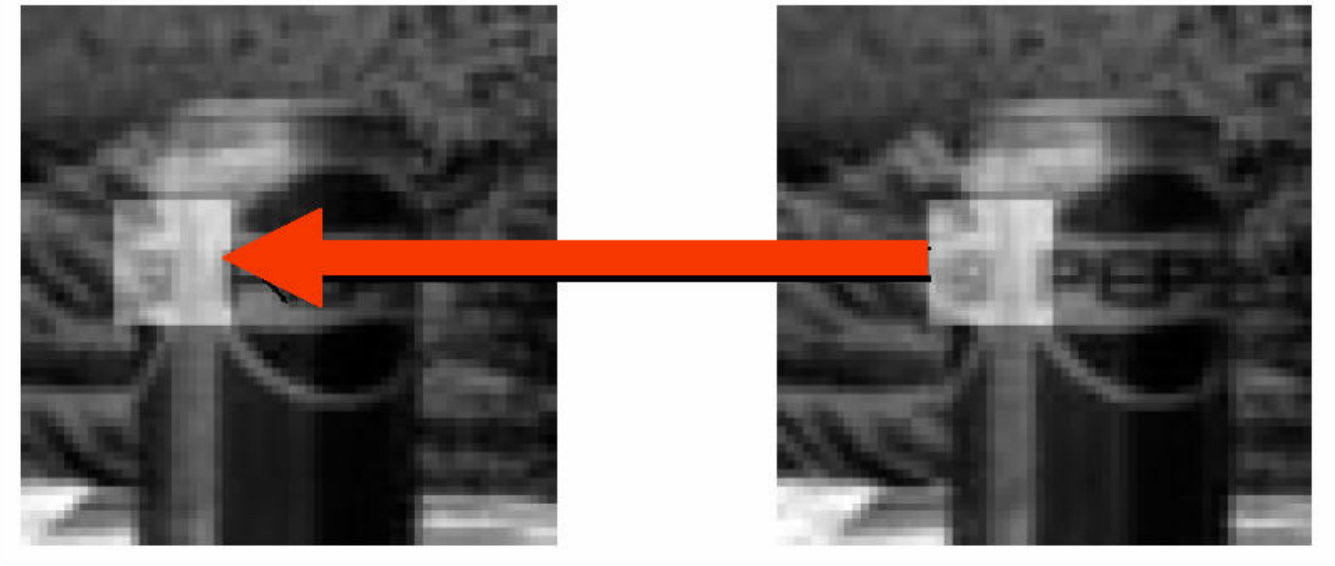
The appearance of the image patches do not change (brightness constancy)

$$I(p_i, t) = I(p_i + \vec{v}_i, t+1)$$

When does it fail?

- Illusory motion: the set is stationary yet things seem to move
- A uniform rotating sphere: nothing seems to move, yet it is rotating
- Changing directions or intensities of lighting can make things seem to move
  - – for example, if the specular highlight on a rotating sphere moves.
- Muscle movement can make some spots on a cheetah move opposite direction of motion.
- And infinitely more break downs of optical flow.

OF Assumptions: Brightness Constancy

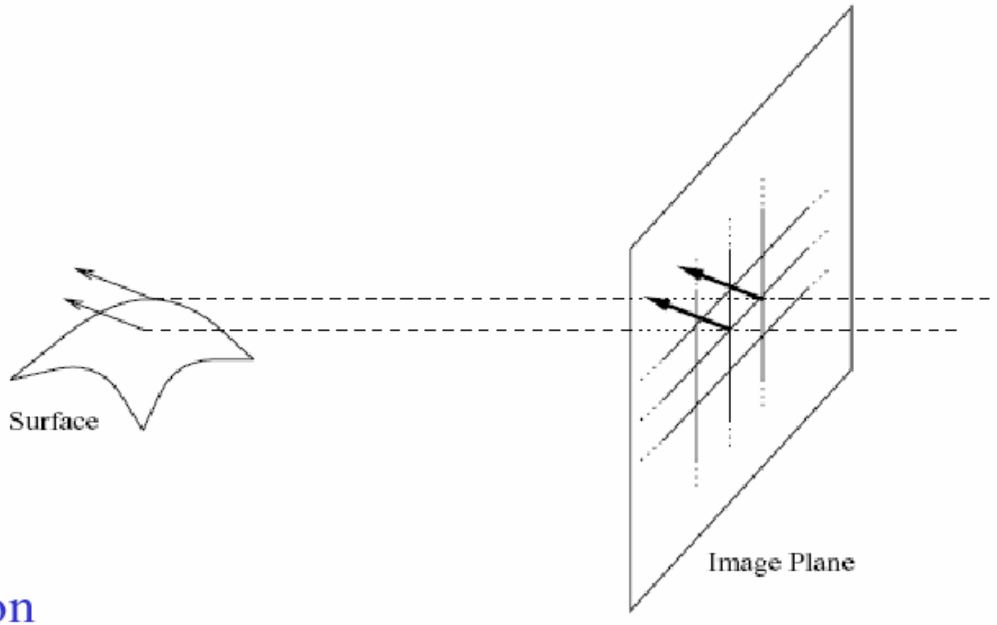


Assumption

Image measurements (e.g. brightness) in a small region remain the same although their location may change.

OF Assumptions

Spatial Coherence



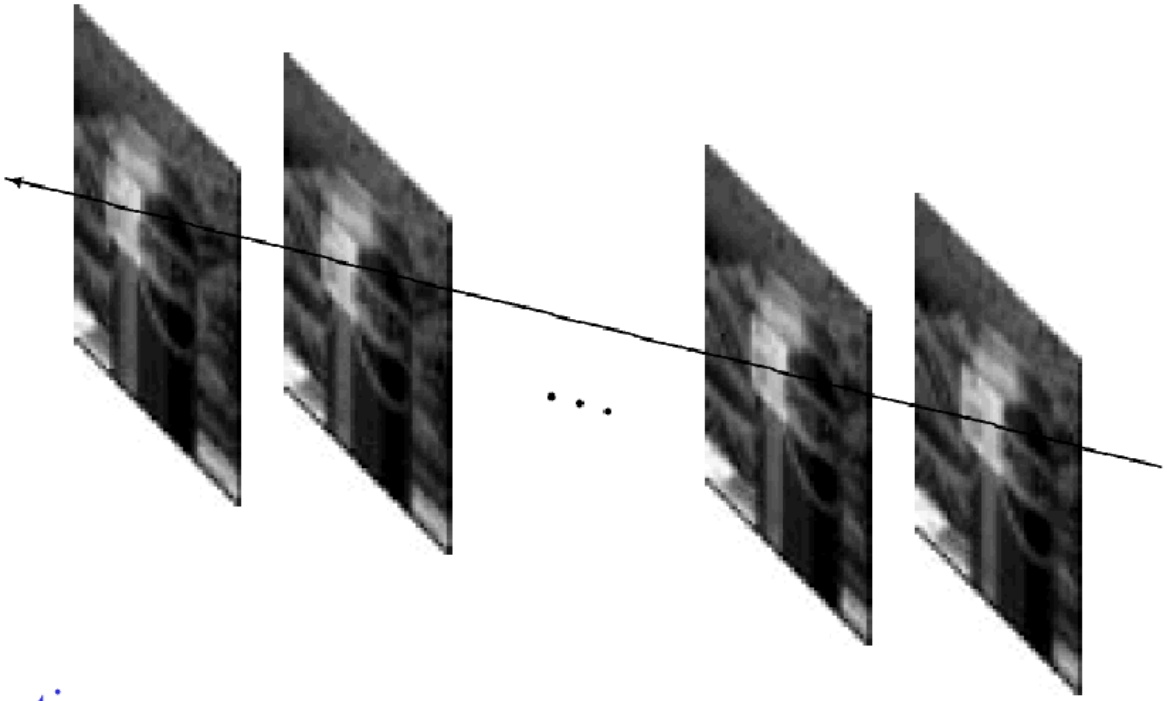
Assumption

* Neighboring points in the scene typically belong to the same surface and hence typically have similar motions.
* Since they also project to nearby points in the image, we expect spatial coherence in image flow.

* Slide from Michael Black, CS143 2003

OF Assumptions

Temporal Persistence



Assumption:

The image motion of a surface patch changes gradually over time.

1D case

Brightness Constancy Assumption:

$$f(t) \equiv I(x(t),t) = I(x(t+dt),t+dt)$$

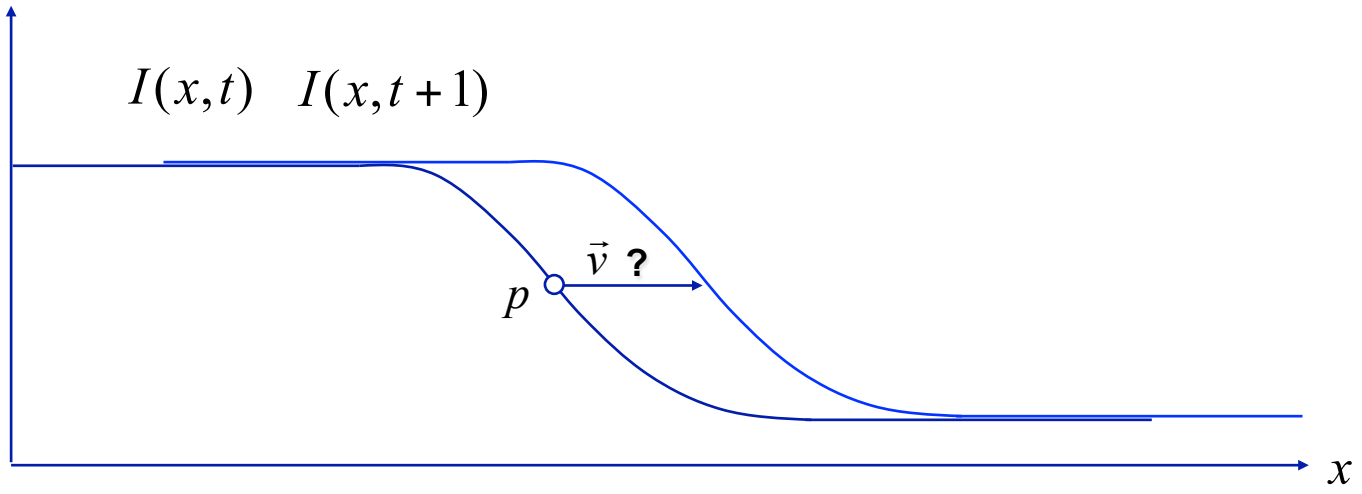
$$\frac{\partial f(x)}{\partial t} = 0$$

Because no change in brightness with time

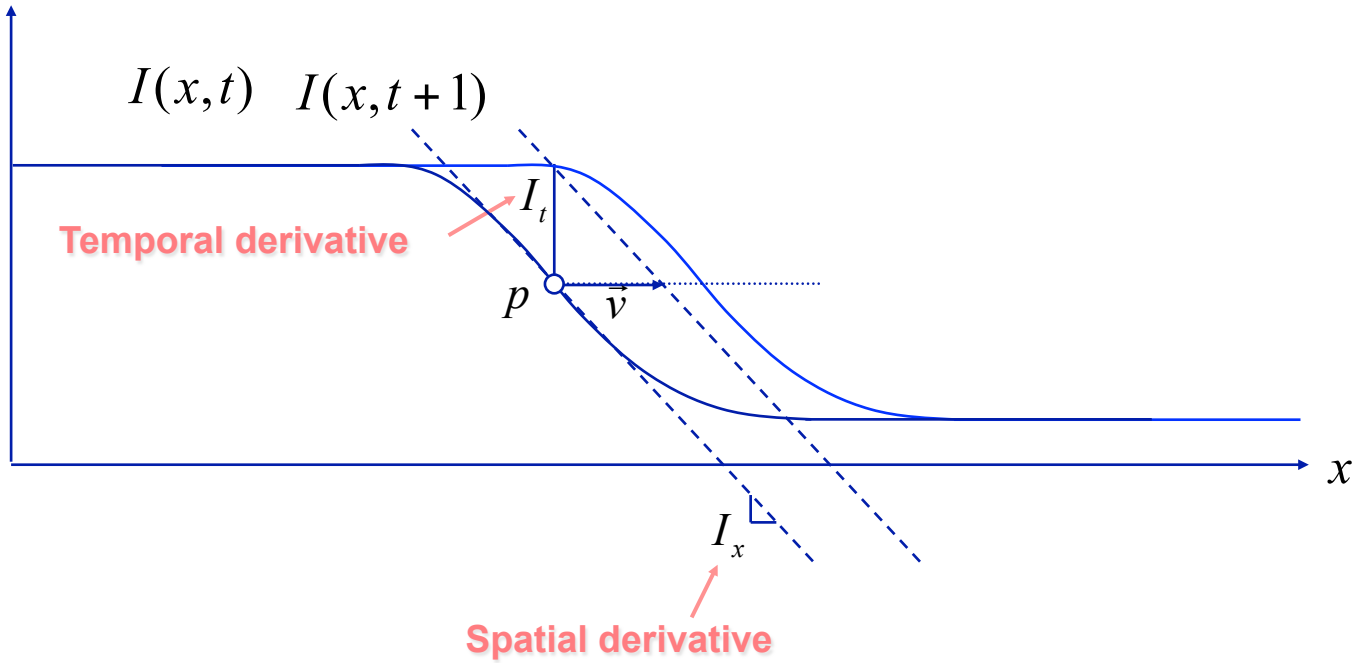
$$\underbrace{\left.\frac{\partial I}{\partial x}\right|_t}_{I_x} \underbrace{\left(\frac{\partial x}{\partial t}\right)}_{v} + \underbrace{\left.\frac{\partial I}{\partial t}\right|_{x(t)}}_{I_t} = 0$$

$$\Rightarrow v = \frac{I_t}{I_x}$$

Tracking in the 1D case:



Tracking in the 1D case:



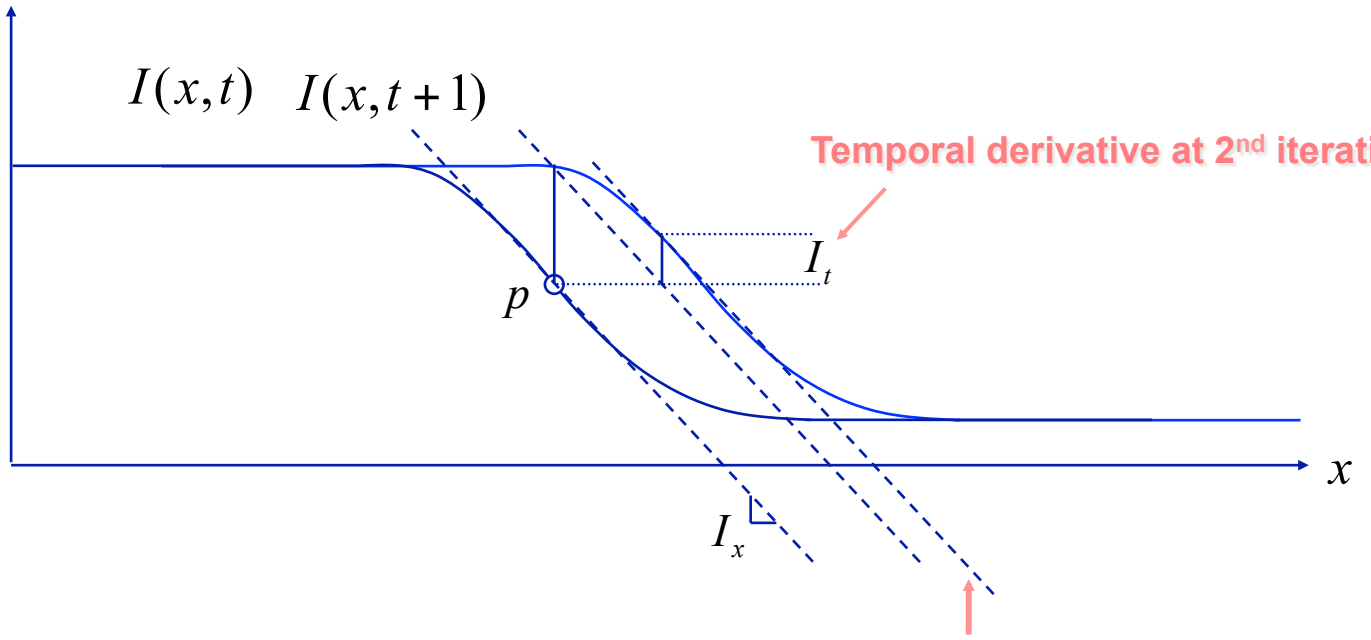
$$I_x = \left.\frac{\partial I}{\partial x}\right|_t \qquad I_t = \left.\frac{\partial I}{\partial t}\right|_{x=p} \qquad \Rightarrow \qquad \vec{v} \approx -\frac{I_t}{I_x}$$

Assumptions:

- **Brightness constancy**
- **Small motion**

Tracking in the 1D case

Iterating helps refining the velocity vector



$$\vec{v} \leftarrow \vec{v}_{previous} - \frac{I_t}{I_x}$$

Converges in about 5 iterations

Algorithm

For all pixel of interest *p*:

- **Compute local image derivative at *p*:** I_x
- **Initialize velocity vector:** $\vec{v} \leftarrow 0$
- **Repeat until convergence:**
  - **Compensate for current velocity vector:** $I'(x,t+1) = I(x+\vec{v},t+1)$
  - **Compute temporal derivative:** $I_t = I'(p,t+1) - I(p,t)$
  - **Update velocity vector:** $\vec{v} \leftarrow \vec{v} - I_t / I_x$

Requirements:

- Need access to neighborhood pixels round p to compute I_x
- Need access to the second image patch, for velocity compensation:
  - The pixel data to be accessed in next image depends on current velocity estimate (bad?)
  - Compensation stage requires a bilinear interpolation (because *v* is not integer)
- The image derivative I_x needs to be kept in memory throughout the iteration process

From 1D to 2D tracking

1D: $$\left.\frac{\partial I}{\partial x}\right|_t \left(\frac{\partial x}{\partial t}\right) + \left.\frac{\partial I}{\partial t}\right|_{x(t)} = 0$$

2D: $$\left.\frac{\partial I}{\partial x}\right|_t \left(\frac{\partial x}{\partial t}\right) + \left.\frac{\partial I}{\partial y}\right|_t \left(\frac{\partial y}{\partial t}\right) + \left.\frac{\partial I}{\partial t}\right|_{x(t)} = 0$$

$$\left.\frac{\partial I}{\partial x}\right|_t u + \left.\frac{\partial I}{\partial y}\right|_t v + \left.\frac{\partial I}{\partial t}\right|_{x(t)} = 0$$

Shoot! One equation, two velocity *(u,v)* unknowns…

From 1D to 2D tracking

Notation

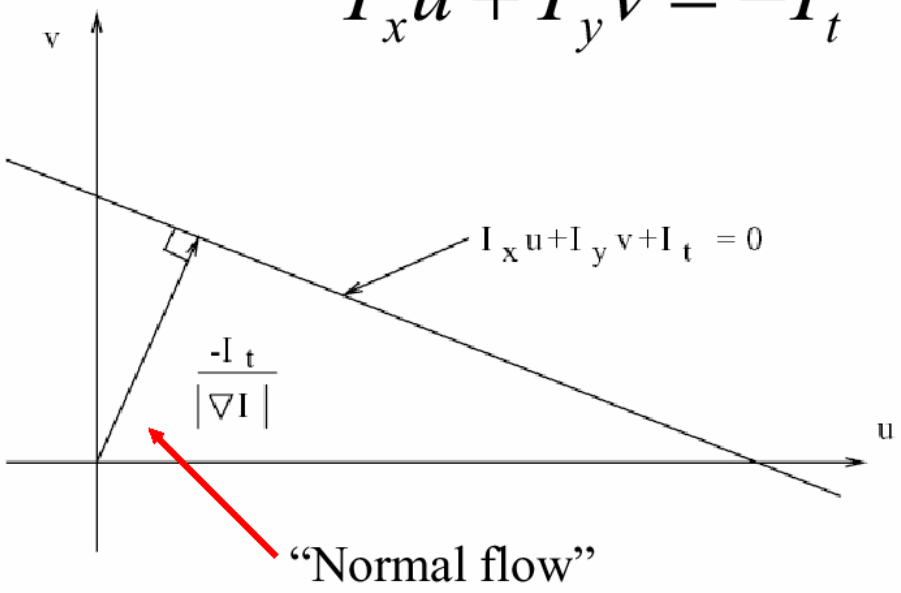
$$I_x u + I_y v + I_t = 0$$

$$\nabla I^T \mathbf{u} = -I_t$$

$$\mathbf{u} = \begin{bmatrix} u \\ v \end{bmatrix} \quad \nabla I = \begin{bmatrix} I_x \\ I_y \end{bmatrix}$$

At a single image pixel, we get a line:

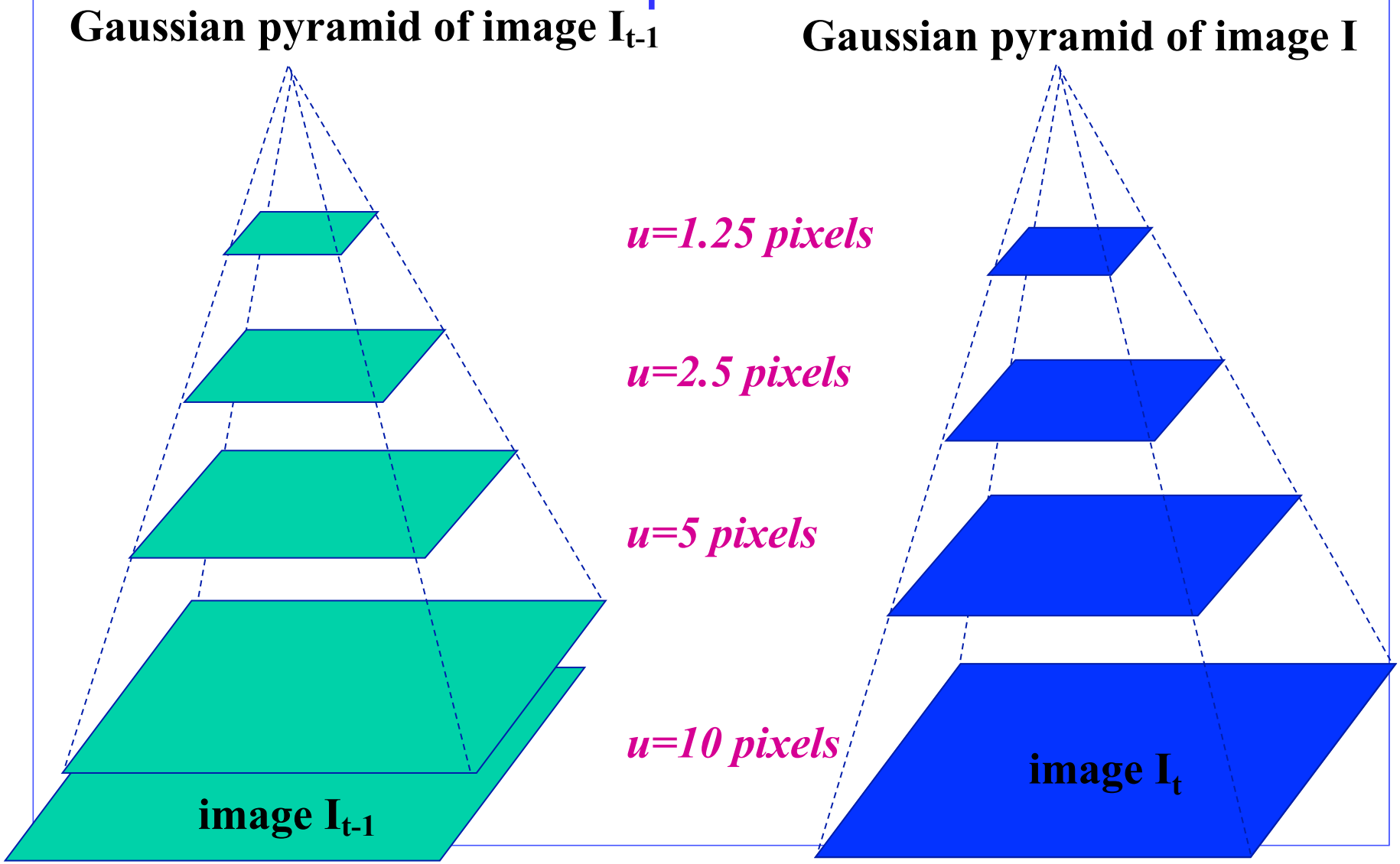
$$I_x u + I_y v = -I_t$$



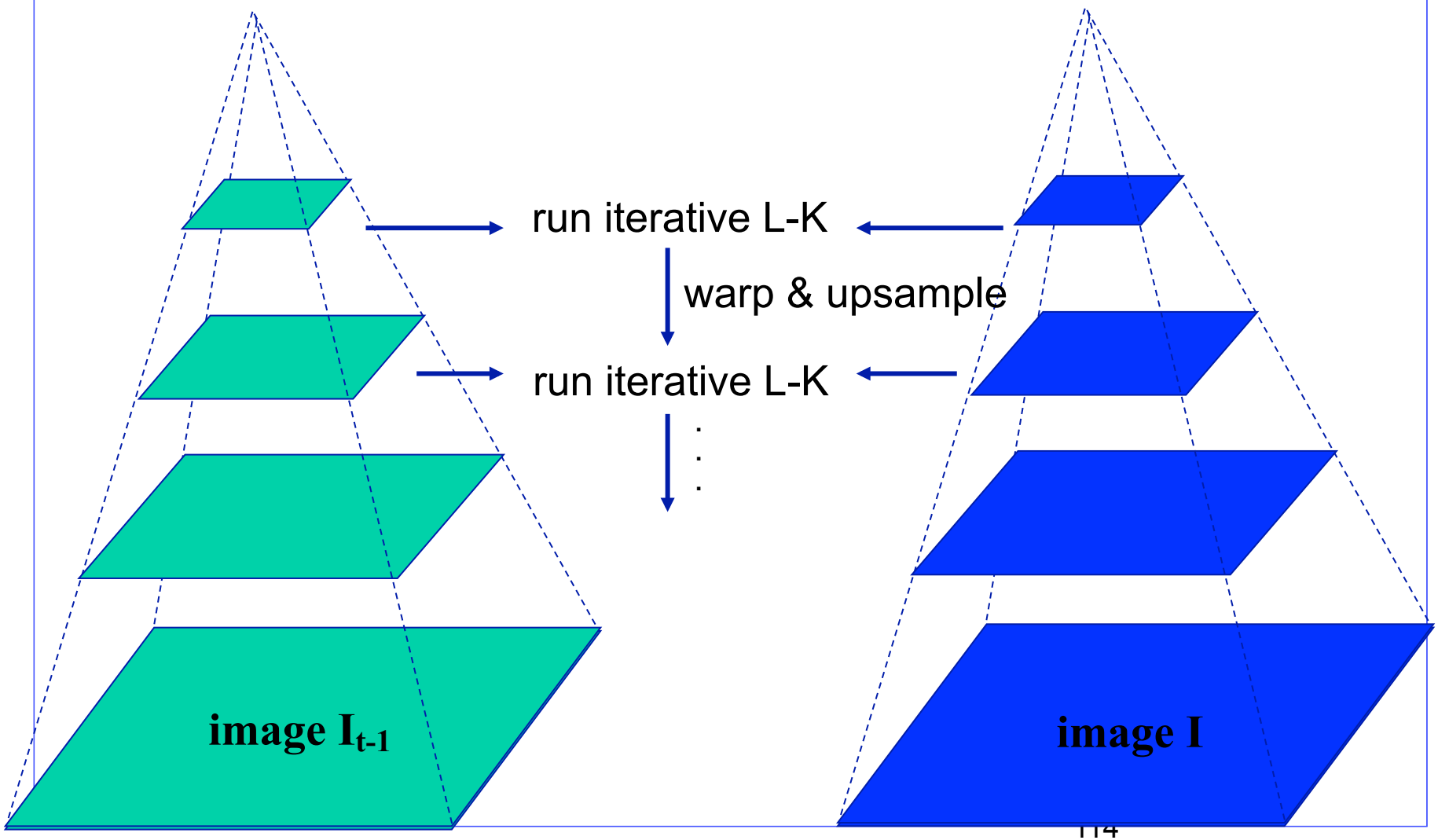
We get at most "Normal Flow" – with one point we can only detect movement perpendicular to the brightness gradient. Solution is to take a patch of pixels Around the pixel of interest.

* Slide from Michael Black, CS143 2003

Coarse-to-fine optical flow estimation

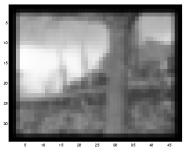


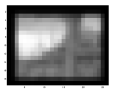
Coarse-to-fine optical flow estimation



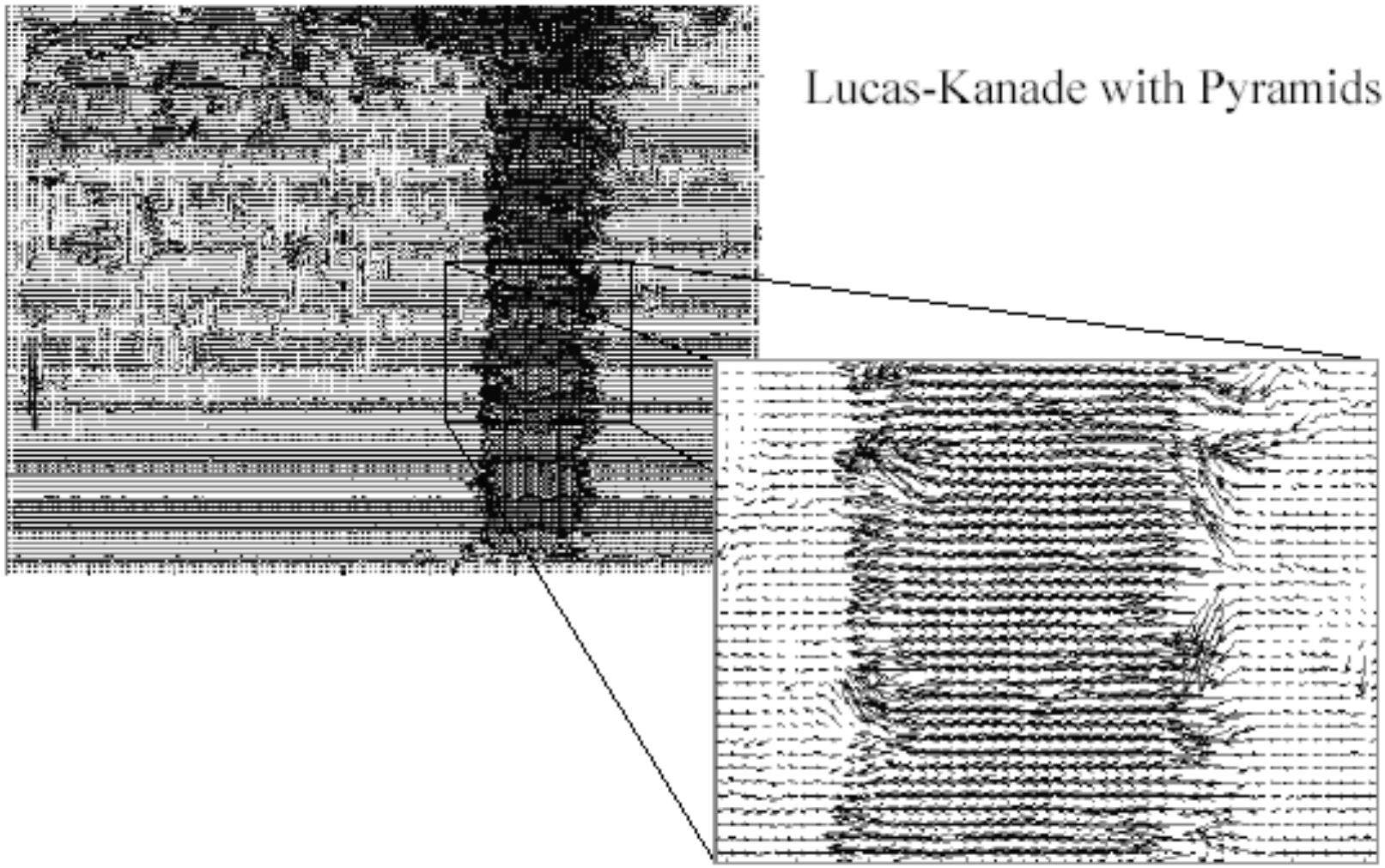
Gaussian pyramid of image I_{t-1} **Gaussian pyramid of image I**

Example





Example: Results



Block matching

Block-matching

16x16 pixels/block

Search window: ±16 pixels from the original position

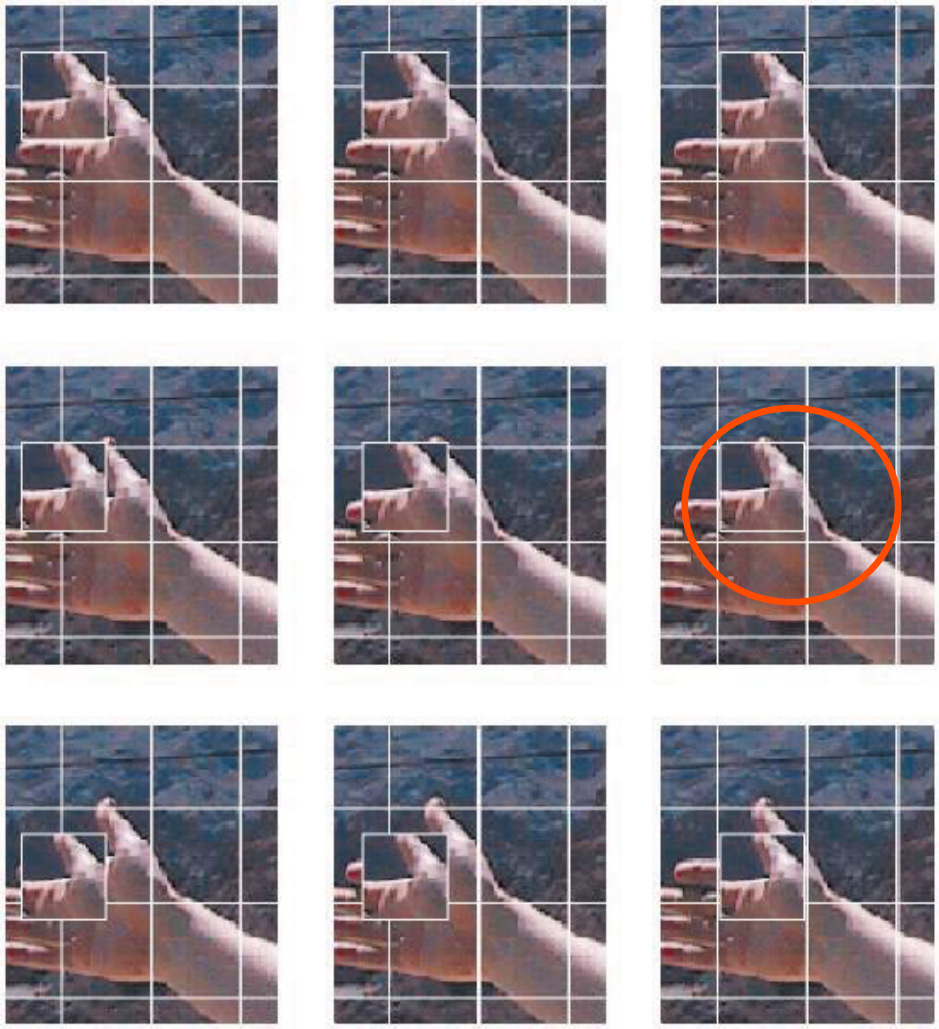
Computationally heavy!

To reduce the complexity

Sub-optimal algorithms

Hardware assisted

Block matching



Motion estimation & Motion compensation