Region Segmentation

Prof. Ioannis Pitas
Aristotle University of Thessaloniki
pitas@csd.auth.gr
www.aiia.csd.auth.gr
Version 3.4

Artificial Intelligence &
Information Analysis Lab

Region Segmentation

- **Introduction**
- Image Thresholding
- Region Growing
- Split/Merge Techniques
- Relaxation Algorithms in Region Analysis
- Connected Component Labeling
- Texture Description.

Region Segmentation

Object shape can be described in terms of:

- ***Its boundary***:
  - It requires image edge detection and following.
- ***The region*** (set of pixels) it occupies:
  - It requires image segmentation in homogeneous regions.
  - Image regions are expected to have homogeneous characteristics (e.g. intensity, texture).
  - These characteristics can form a feature vector that can be used to discriminate region from one another.

Region Segmentation

- An image domain $\mathcal{X}$ must be segmented in N different regions $\mathcal{R}_1, \dots, \mathcal{R}_N$.

- The segmentation rule is a logical predicate of the form $P(\mathcal{R})$.

Region Segmentation

- Image segmentation partitions the set $\mathcal{X}$ into the subsets $\mathcal{R}_i$, $i = 1, \ldots, N$, having the following properties:

$$\mathcal{X} = \bigcup_{i=1}^{N} \mathcal{R}_i,$$

$$\mathcal{R}_i \cap \mathcal{R}_j = \emptyset, \quad \text{for } i \neq j,$$

$$P(\mathcal{R}_i) = \text{TRUE}, \quad \text{for } i = 1,2, \ldots, N,$$

$$P(\mathcal{R}_i \cup \mathcal{R}_j) = \text{FALSE}, \quad \text{for } i \neq j.$$

Region Segmentation

- Region segmentation can employ a logical predicate of the form $P(\mathcal{R}, \mathbf{x}, \mathbf{t})$.
  - $\mathbf{x}$ is a feature vector associated with an image pixel or pixel set.
  - $\mathbf{t}$ is a parameter vector (usually thresholds).

- A simple segmentation rule has the form:

$$P(\mathcal{R}): f(k, l) < T.$$

Region Segmentation

- In RGB colour images, the feature vector $\mathbf{x}$ can be the three RGB image components:

$$\mathbf{x} = [f_R(k,l), f_G(k,l), f_B(k,l)]^T.$$

- A simple RGB image segmentation rule having $\mathbf{t} = [T_R, T_G, T_B]^T$ may have the form:

$$P(\mathcal{R}, \mathbf{x}, \mathbf{t}): (f_R(k,l) < T_R) \; \&\& \; (f_G(k,l) < T_G) \; \&\& \; (f_B(k,l) < T_B).$$

Region Segmentation

- ***Geometrical proximity*** plays an important role in image segmentation.

- Segmentation algorithms must incorporate both ***pixel proximity*** and ***pixel homogeneity***.

- A simple approach to geometrical proximity is through image neighrborhood definition.

Region Segmentation

We can define two types of image neighbourhoods on $\mathbb{Z}^2$:

- The ***4-neighbourhood*** $\mathcal{N}_4(\mathbf{x})$ of a pixel $\mathbf{x} = [x, y]^T$ is the set that includes its horizontal and vertical neighbours:

$$\mathcal{N}_4(\mathbf{x}) = \{[x-1, y]^T, [x+1, y]^T, [x, y-1]^T, [x, y+1]^T\}.$$

- The ***8-neighbourhood*** $\mathcal{N}_8(\mathbf{x})$ of pixel $\mathbf{x} = [x, y]^T$ is a superset of the 4-neighbourhood and contains the horizontal, vertical and diagonal neighbours:

$$\mathcal{N}_8(\mathbf{x}) = \mathcal{N}_4(\mathbf{x}) \cup \{[x-1, y-1]^T, [x-1, y+1]^T, [x+1, y-1]^T, [x+1, y+1]^T\}.$$

Region Segmentation

- The paths defined by using the 4-neighbourhood consist of horizontal and vertical streaks of length $\Delta x = \Delta y = 1$.

- The paths using the 8-neighbourhood consist of horizontal and vertical streaks of length 1 and of diagonal streaks having length $\sqrt{2}$.

Region Segmentation

- A region $\mathcal{R}$ is called ***connected region*** if:

  - any two pixels $\mathbf{x}_A, \mathbf{x}_B$ belonging to $\mathcal{R}$ can be connected by a path $\mathbf{x}_A, \ldots, \mathbf{x}_{i-1}, \mathbf{x}_i, \mathbf{x}_{i+1}, \mathbf{x}_B$, whose pixels $\mathbf{x}_i$ belong to $\mathcal{R}$;

and

  - any pixel $\mathbf{x}_i$ is adjacent to both the previous pixel $\mathbf{x}_{i-1}$ and the next one $\mathbf{x}_{i+1}$ in the path.

- A pixel $\mathbf{x}_k$ is said to be adjacent to pixel $\mathbf{x}_l$, if it belongs to its immediate neighbourhood.

Region Segmentation

Region segmentation techniques can be grouped in three different classes:

- ***Local region segmentation techniques*** are based on the local properties of the pixels and their neighbourhoods.
- ***Global region segmentation techniques*** segment an image on the basis of information obtained globally (e.g., by using the image histogram).

- ***Split, merge and growing techniques*** use both the notions of homogeneity and geometrical proximity.

Region Segmentation

- Introduction
- **Image Thresholding**
- Region Growing
- Split/Merge Techniques
- Relaxation Algorithms in Region Analysis
- Connected Component Labeling
- Texture Description.

Image Thresholding

- The simplest image segmentation problem occurs when an image contains:
  - an ***object*** having homogeneous intensity.
  - a ***background*** with a different intensity level.
- Such an image can be segmented in two regions by simple ***thresholding***:

$$g(x,y) = \begin{cases} 1, & \text{if } f(x,y) > T, \\ 0, & \text{otherwise.} \end{cases}$$

Image Thresholding

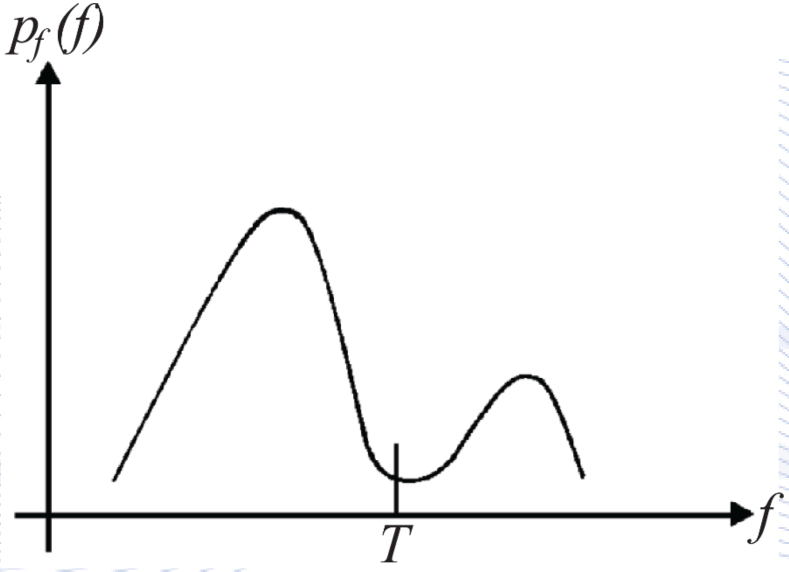
- The choice of threshold T can be based on ***image histogram*** measuring intensity level frequencies in an image having $N_1 \times N_2$ pixels:

$$h(i) = \frac{1}{N_1 N_2} \sum_{k=0}^{N_1-1} \sum_{l=0}^{N_2-1} \delta(f(k,l) - i).$$

Artificial Intelligence & Information Analysis Lab

Image Thresholding

- If the image contains one object and a background having homogeneous intensity, it usually possesses a ***bimodal image histogram.***



Bimodal image histogram and histogram choice.

Image Thresholding

Image thresholding.

Image Thresholding

- If the histogram is noisy:
  - The calculation of the local histogram minimum is difficult.
  - ***Histogram smoothing*** or ***image smoothing*** (e.g., by using one-dimensional low-pass filtering) is recommended.
- If the object and/or background intensity varies:
  - Image histogram may not contain two clearly distinguished lobes.
  - Threshold can be calculated so that only $a\%$ of image prixels belong to object.
  - A ***spatially varying threshold*** can be applied.

Image Thresholding

Multiple thresholding can be used for segmenting images containing N objects, provided that each object $\mathcal{R}_i$ occupies a distinct intensity range, defined by two thresholds T_{i-1}, T_i.

- The thresholding operation takes the following form:

$$g(x,y) = \mathcal{R}_i, \qquad \text{if } T_{i-1} \leq f(x,y) \leq T_i, \qquad i = 1, \ldots, N.$$

- Thresholds can be obtained from the image histogram.
- In many cases, the various histogram lobes are not clearly distinguished.

Image Thresholding

Multiple thresholding can be used for segmenting images containing N objects, provided that each object $\mathcal{R}_i$ occupies a distinct intensity range, defined by two thresholds T_{i-1}, T_i.

- The thresholding operation takes the following form:

$$g(x,y) = \mathcal{R}_i, \qquad \text{if } T_{i-1} \le f(x,y) \le T_i, \qquad i = 1, \ldots, N.$$

- Thresholds can be obtained from the image histogram.

Image Thresholding

- In many cases, the various histogram lobes are not clearly distinguished.

- Image thresholding in N different ***equirange*** regions:

$$g(x,y) = \begin{cases} \mathcal{R}_i & \text{if } i[L/N] \leq f(k,l) < (i+1)[L/N], i = 0,1,\dots,N-2, \\ \mathcal{R}_{N-1} & \text{if } (N-1)[L/N] \leq f(k,l) < L. \end{cases}$$

Artificial Intelligence & Information Analysis Lab

Image Thresholding

(a)

(b)

a) Original image; b) Image segmentation in four equirange regions.

Image Thresholding

- ***Histogram modification***: Perform edge detection and exclude all pixels belonging to edges, from histogram calculation.
- Another approach is to define a modified histogram:

$$h(i) = \sum_{k=0}^{N_1-1} \sum_{l=0}^{N_2-1} t(e(k,l))\delta(f(k,l) - i).$$

$e(k,l)$ is an edge detector output,
$\delta(i)$ is the delta function.

Image Thresholding

- A monotonically decreasing function t can be chosen for histogram modification:

$$t\big(e(k,l)\big) = \frac{1}{1 + |e(k,l)|}.$$

Image Thresholding

- If the image histogram is concentrated in a small intensity range:
  - Uniform thresholding does not give good results.
  - Non-uniform thresholding creates much better results in this case.
- ***Non-uniform thresholding*** can be based on ***histogram equalization*** described by $G\big(f(k,l)\big)$:

$$g(k,l) = \begin{cases} \mathcal{R}_i, & \text{if } i[L/N] \le G(f(k,l)) < (i+1)[L/N], \\ \mathcal{R}_{N-1,} & \text{if } (N-1)[L/N] \le G(f(k,l)) < L. \end{cases}$$

Region Segmentation

- Introduction
- Image Thresholding
- **Region Growing**
- Split/Merge Techniques
- Relaxation Algorithms in Region Analysis
- Connected Component Labeling
- Texture Description.

Region growing

- Image segmentation can start from some pixels (seeds) representing distinct image regions.
- ***Pixel seeds*** can be chosen in a ***supervised or unsupervised mode***.
- At least one ***seed*** s_i, $i = 1, \dots, N$ is chosen per image region $\mathcal{R}_i$.
- Seeds are ***grown***, until they cover the entire image.
- We need:
  - a rule describing a growth mechanism and
  - a rule checking region homogeneity after each growth step.

Region growing

- ***Growth mechanism***: at each stage (k) and for each region $\mathcal{R}_I^{(k)}, i = 1, \ldots, N,$ we check if there are unclassified pixels in the 8-neighbourhood of each pixel of the region border.

- Before assigning such a pixel $\mathbf{x}$ to a region $\mathcal{R}_I^{(k)}$, we check if the region homogeneity:

$$P(\mathcal{R}_i^{(k)} \cup \{\mathbf{x}\}) = TRUE$$

is still valid.

Region growing

Region merging can be incorporated in the growing mechanism:

- If we are currently at the pixel $\mathbf{x} = [k, l]^T$:
  - First, we try to merge this pixel with one of its adjacent regions $\mathcal{R}_i$.
  - If this merge fails, or if no adjacent region exists, this pixel is assigned to a new region.

- The merging rule can be based on the region mean and standard deviation described by m_i and σ_i.

Region growing

- The arithmetic mean m_i and standard deviation σ_i of a class $\mathcal{R}_i$ having n pixels are given by:

$$m_i = \frac{1}{n} \sum_{(k,l)\in\mathcal{R}_i} f(k,l),$$

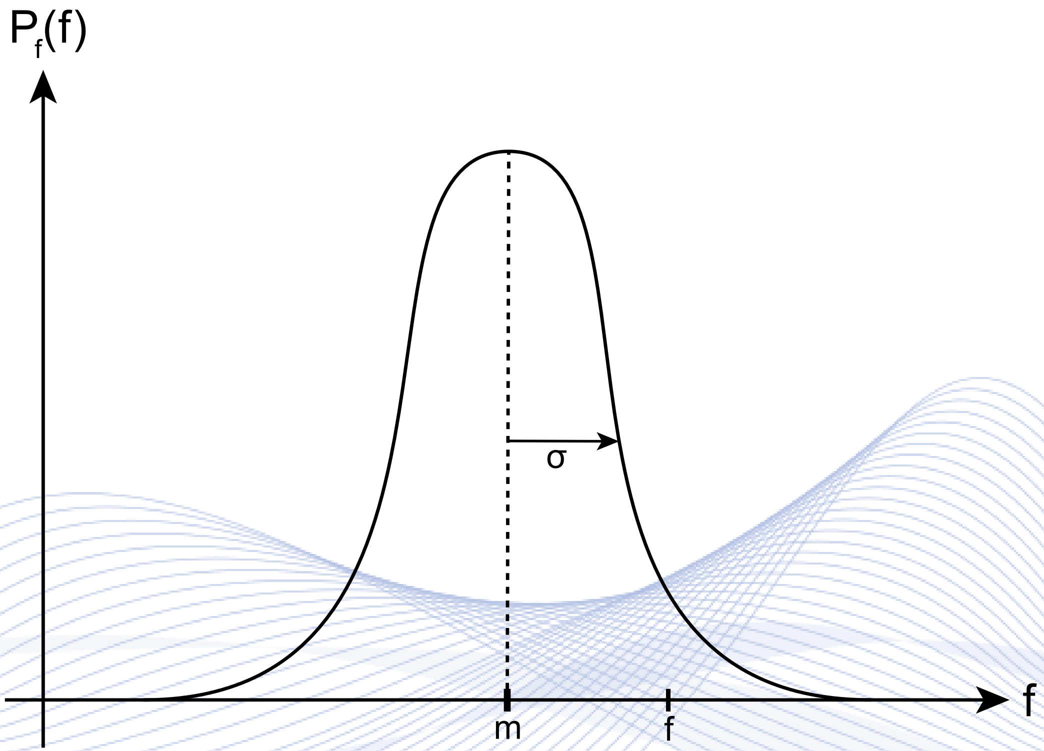
$$\sigma_i = \sqrt{\frac{1}{n}\sum_{(k,l)\in\mathcal{R}_i}[f(k,l) - m_i]^2}.$$

- Merging is allowed, if the pixel intensity is close to the region mean value:

$$|f(k,l) - m_i| \leq T_i(k,l).$$

Artificial Intelligence & Information Analysis Lab

Region growing



Decision on merging a pixel with a region.

Region growing

- If more than one merge are possible, the region with the closest mean value is chosen.

- Threshold T_i varies, depending on the region $\mathcal{R}_i$ and the intensity of the pixel $f(k,l)$. It can be chosen this way:

$$T_i(k,l) = \left(1 - \frac{\sigma_i}{m_i}\right)T.$$

Region growing

- If merging $P(\mathcal{R}_i \cup \{\mathbf{x}\})$ was allowed, the updated mean and standard deviation of region $\mathcal{R}_i$ are given by:

$$m_i' = \frac{1}{n+1}[f(k,l) + nm_i],$$

$$\sigma_i' = \sqrt{\frac{1}{n+1}\left(n\sigma_i^2 + \frac{n}{n+1}[f(k,l) - m_i]^2\right)}.$$

Region growing

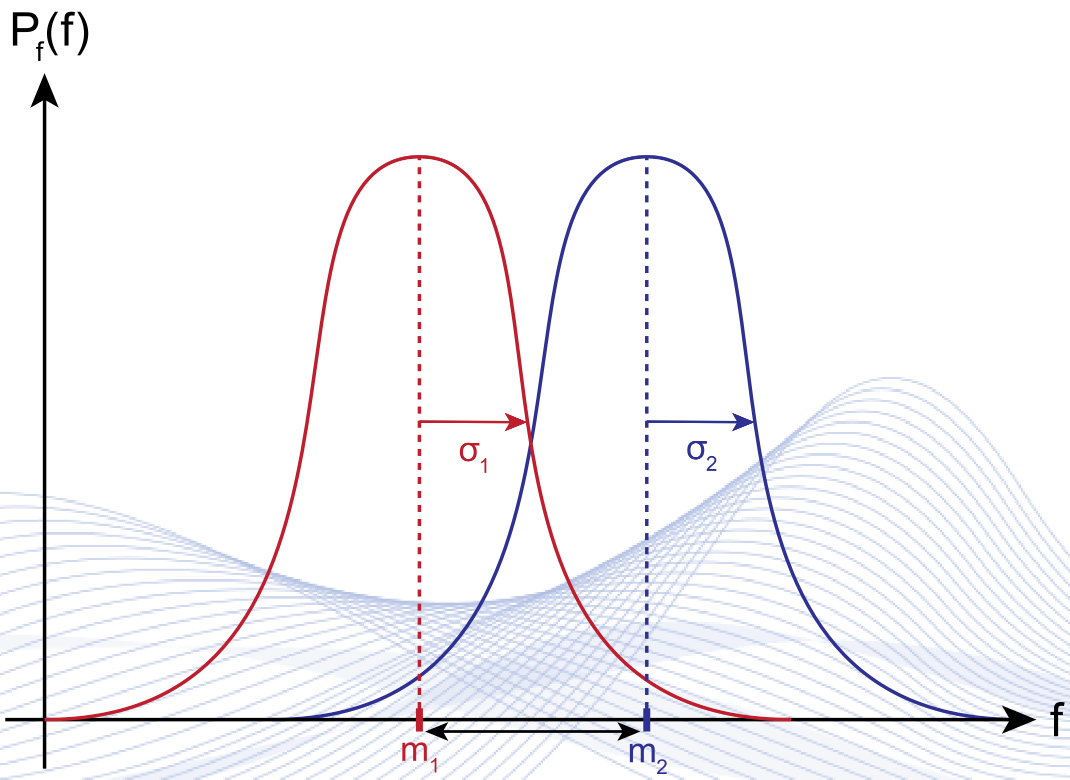
- The region statistics can be used to decide if the merging of two regions $\mathcal{R}_1, \mathcal{R}_2$ is allowed.
- If arithmetic means m_1, m_2 are close to each other:

$$|m_1 - m_2| < k\sigma_i, \qquad i = 1,2,$$

the two regions are merged.

- If no a priori information is available about the image, the image can be scanned in a row-wise manner.

Region growing



Decision on merging two regions.

Region Segmentation

- Introduction
- Image Thresholding
- Region Growing
- **Split/Merge Techniques**
- Relaxation Algorithms in Region Analysis
- Connected Component Labeling
- Texture Description.

Region Splitting/Merging

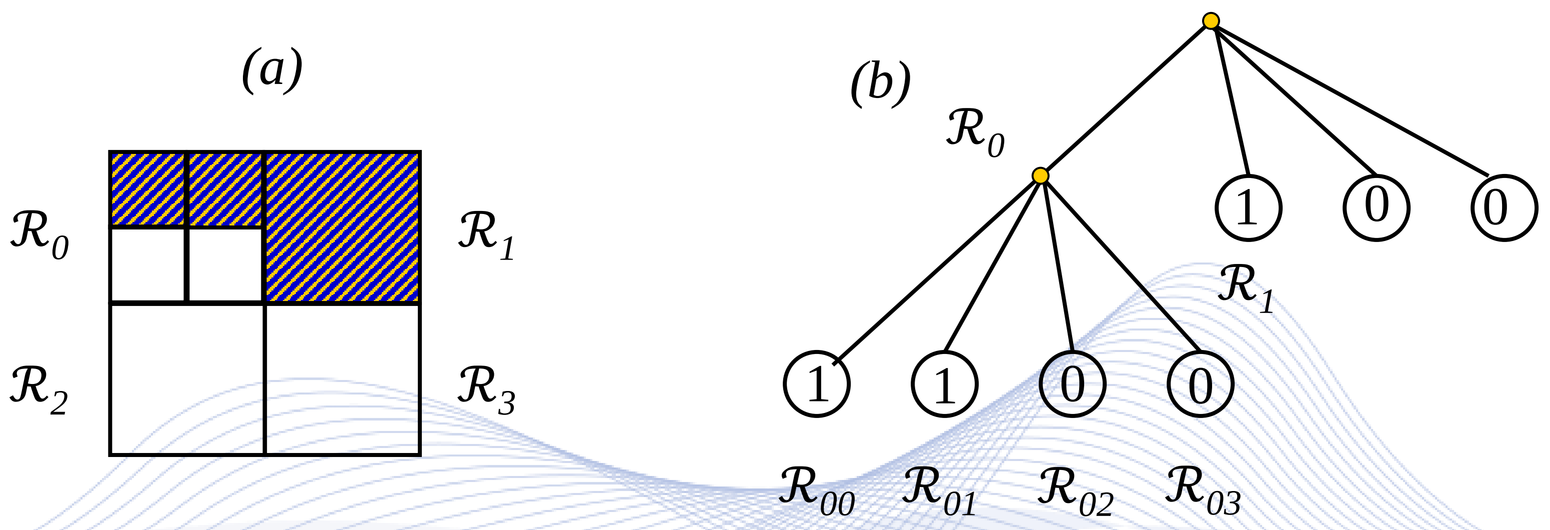
- The opposite approach to ***region merging*** is ***region splitting***:
  - It is a top-down approach.
  - It starts with the assumption that the entire image is homogeneous.
  - If this is not true, the image is split into four sub-images.

  - This splitting procedure is repeated recursively until we split the image into homogeneous regions.

Region Splitting/Merging

- If the original image is square $N \times N$, having dimensions that are powers of 2 ($N = 2^n$):
  - All regions produced by the splitting algorithm are squares having dimensions $M \times M$, where M is a power of 2 as well ($M = 2^m, m \leq n$).
  - Since the procedure is recursive, it produces an image representation that can be described by a ***quadtree*** whose nodes have four sons each.
  - A quadtree is a very convenient region representation.

Region Splitting/Merging



a) Image segmentation by region splitting; b) Quadtree.

Region Splitting/Merging

Disadvantages of region splitting techniques:

- ***Oversegmentation.*** Regions are created that may be adjacent and homogeneous, but not merged.
- Oblique lines create many small regions of size 2×2 pixels.
  - Solution: ***region split and merge algorithm***.
- Sensitivity to geometrical transformations.
- As this is a ***recursive algorithm***, ***stack overflow*** may occur.

Region Splitting/Merging

Region split and merge algorithm.

- It is an iterative algorithm that includes both splitting and merging at each iteration:
  - If a region $\mathcal{R}$ is inhomogeneous $(P(\mathcal{R}) = FALSE)$, it is split into four subregions.
  - Two adjacent regions $\mathcal{R}_i, \mathcal{R}_j$ are merged if they are homogeneous: $P(\mathcal{R}_i \cup \mathcal{R}_j) = TRUE$.

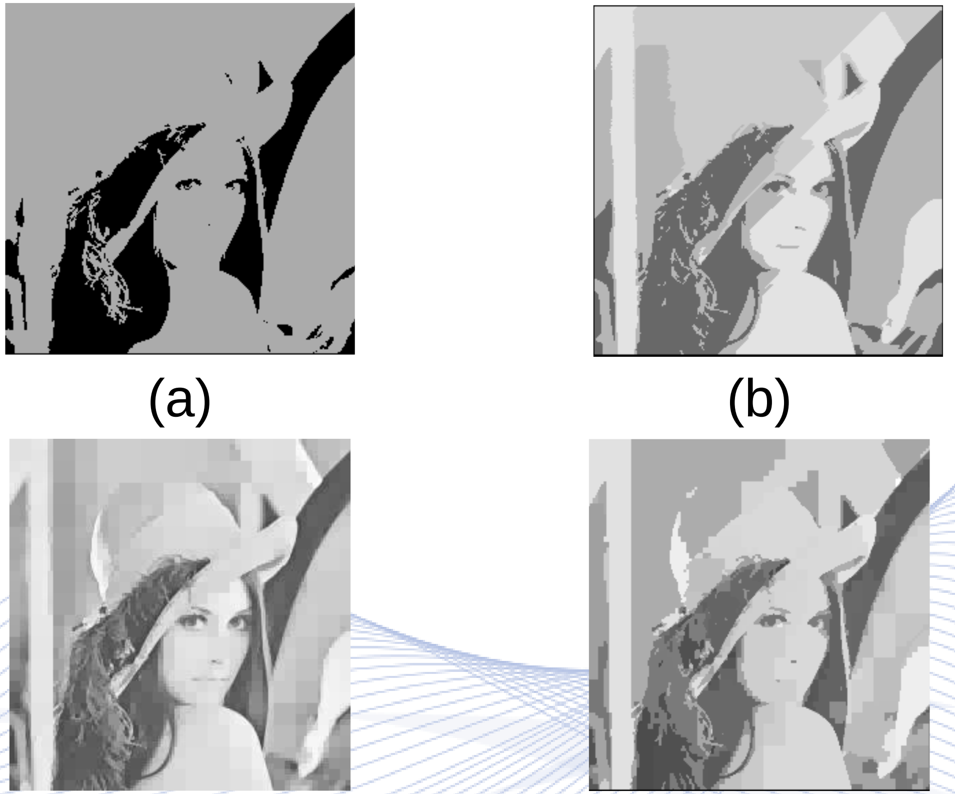
  - The algorithm stops when no further splitting or merging is possible.

Region Splitting/Merging

- The split and merge algorithm produces more compact regions than the pure splitting algorithm.
- Its major disadvantage is that it does not produce quadtree region descriptions.

- Several modifications of the basic split and merge algorithm have been proposed to solve this problem.
  - The most straightforward procedure is to use the splitting algorithm and to postpone merging until no further splitting is possible.

Region Splitting/Merging



(a) (b)

(c) (d)

Output of: a) region thresholding; b) region growing; c) region splitting; d) region split and merge algorithm.

Region Segmentation

- Introduction
- Image Thresholding
- Region Growing
- Split/Merge Techniques
- **Relaxation Algorithms in Region Analysis**
- Connected Component Labeling
- Texture Description.

Relaxation Labeling

- All previous region segmentation methods are deterministic:
  - they assign each image pixel to just one region.

- Such a segmentation is desirable, but not always useful, because they treat ambiguous cases in a rather inflexible way.

Relaxation Labeling

- It is more useful to produce confidence vectors $\mathbf{p}_k$ for each pixel $\mathbf{x}_k$ that contain the probabilities $p_k(i)$ that a pixel $\mathbf{x}_k$ belongs to a class $\mathcal{R}_i, i = 1, \dots, N$:

$$\mathbf{p}_k = [p_k(1), \dots, p_k(N)]^T.$$

- Probabilities $p_k(l)$, called ***confidence weights***, must satisfy the following relations:

$$0 \leq p_k(i) \leq 1, \qquad \sum_{i=1}^{N} p_k(i) = 1.$$

- Pixel $\mathbf{x}_k$ is assigned to the region $\mathcal{R}_l$ having the maximal probability $p_k(l)$.

Relaxation Labeling

- Let $m_i, i = 1, \dots, N$, be the arithmetic means of the intensity of each region that usually correspond to histogram peaks and $f(\mathbf{x}_k) = f(n, l)$ the pixel intensity at location $\mathbf{x}_k = [n, l]^T$.

- The initial estimate of confidence weights is given by:

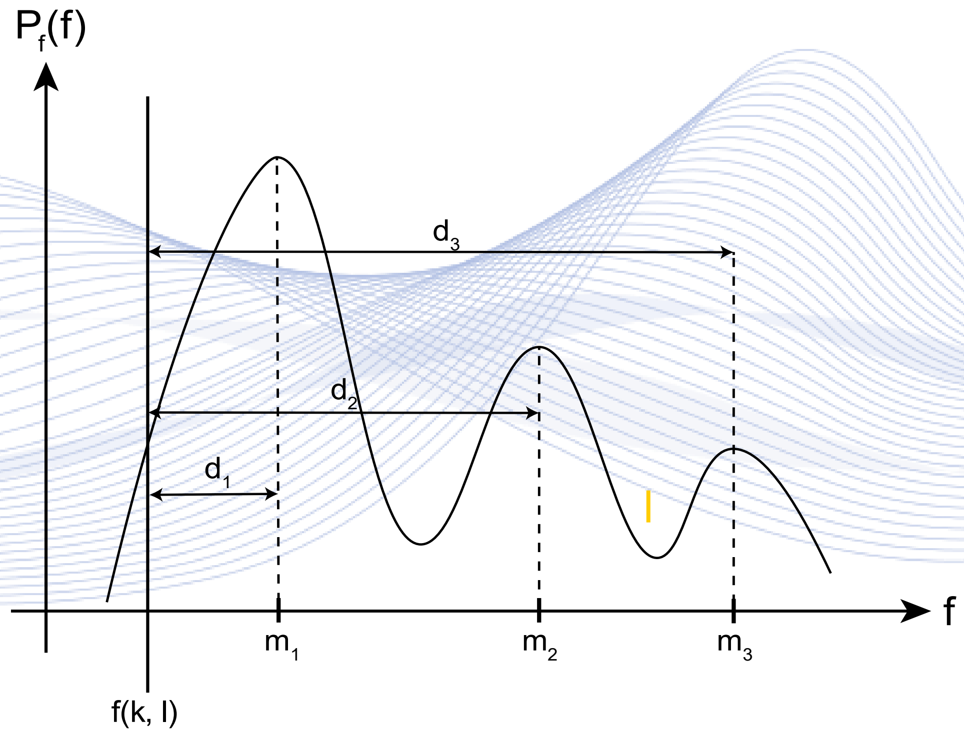
$$p_k^{(0)}(i) = \frac{\frac{1}{|f(n,l) - m_i|}}{\sum_{i=1}^{N} \frac{1}{|f(n,,l) - m_i|}}, \qquad i = 1, \dots, N.$$

Relaxation Labeling

- It is inversely proportional to the distance:

$$d_i = |f(n,l) - m_i|$$

of the pixel intensity $f(n,l)$ from the region arithmetic mean m_i.



Relaxation Labeling

- In many cases, it is highly probable that two adjacent pixels belong to two specific ***compatible*** classes $\mathcal{R}_i, \mathcal{R}_j$, e.g.,:
  - Pixels of classes ‘Road’ and ‘Pavement’.

- ***Incompatible*** regions are those that are not expected to be found in adjacent image locations, e.g.,:
  - Pixels of classes ‘Road’ and ‘Sea’.

Relaxation Labeling

- The compatibility between two regions $\mathcal{R}_{i,}\mathcal{R}_{j,}$ is described in terms of a ***compatibility function*** $r(i,j)$, whose range is:

$$-1 \le r(i,j) \le 1.$$

- Its values have the following meaning:

$$r(i,j) = \begin{cases} < 0, & \text{Regions } \mathcal{R}_i, \mathcal{R}_j \text{ are incompatible.} \\ = 0, & \text{Regions } \mathcal{R}_i, \mathcal{R}_j \text{ are independent.} \\ > 0, & \text{Regions } \mathcal{R}_i, \mathcal{R}_j \text{ are compatible.} \end{cases}$$

Relaxation Labeling

- Compatibility functions are known a priori or can be estimated from an initial image segmentation.
- Incompatible regions tend to compete in adjacent image pixels, whereas compatible regions tend to cooperate.

- ***Competition and cooperation*** can continue in an iterative way until a steady state is reached.
  - Each pixel $\mathbf{x}_k$ receives confidence contributions from any pixel $\mathbf{x}_l$ lying in its 4- or 8-neighbourhood.

Relaxation Labeling

The resulting change in confidence weight $p_k(i)$ of the pixel $\mathbf{x}_k$ at step (n) is the following:

$$\Delta p_k^{(n)} = \sum_l d_{kl} \left[\sum_{j=1}^{N} r_{kl}(i,j) p_l^{(n)}(j) \right].$$

- The sum of the parameters d_{kl} is chosen to be equal to 1:

$$\sum_l d_{kl} = 1.$$

Relaxation Labeling

- The updated probabilities for the pixel $\mathbf{x}_k$ are given by:

$$p_k^{(n+1)}(i) = \frac{p_k^{(n)}(i)\left[1 + \Delta p_k^{(n)}(i)\right]}{\sum_{i=1}^{N} p_k^{(n)}(i)\left[1 + \Delta p_k^{(n)}(i)\right]}.$$

- The iterations stop when convergence is achieved.
- The iterative equations form ***relaxation labelling***.
- It is expected to produce relatively large connected homogeneous image regions, by removing small spurious noisy regions within larger regions.

NN region segmentation

Street scene segmentation [APOLLO].

Region Boundary Following

- In certain cases, the region boundary is desired.
- If the segmented image $g(x,y)$ is available, the boundary obtained by finding ***region transition pixels*** $b(x,y)$:

$$b(x,y)=\begin{cases}1, & \text{if } \{\left(g(x,y)\in\mathcal{R}_i \text{ and } g(x,y-1)\in\mathcal{R}_j, i\neq j\right) \\ & \text{or } (g(x,y)\in\mathcal{R}_i \text{ and } g(x-1,y)\in\mathcal{R}_j . i\neq j)\}, \\ 0, & \text{otherwise.}\end{cases}$$

Region Segmentation

- Introduction
- Image Thresholding
- Region Growing
- Split/Merge Techniques
- Relaxation Algorithms in Region Analysis
- **Connected Component Labeling**
- Texture Description.

Connected Component Labeling

- Digital image segmentation produces either a binary or a multivalued image output $g(k,l)$.
  - Each image region is labelled by a region number.
  - Typically, background has label 0.
  - Each region may consist of several disconnected subregions.

- ***Connected component labeling*** assigns a unique number to each ***pixel blob*** of $1s$.

Connected Component Labeling

- Connected component labeling algorithms can be divided into two large classes:
  - ***Local neighborhood algorithms*** (performing local operations, typically in a recursive manner).
  - ***Divide-and-conquer*** algorithms.

- If each blob corresponds to a single object, connected component labeling performs ***object counting*** in a binary image.

Connected Component Labeling

Fire propagation algorithm:

- The image is scanned in a row-wise manner, until the first pixel at an object boundary is hit.
- A ‘fire’ is set at this pixel that propagates to all pixels belonging to the 8-neighbourhood of the current pixel.
- Then the curent pixel is burned out (e.g., takes value 0).
- This recursive operation continues, until all image pixels of the image object are ‘burnt out’ and the fire is extinguished.

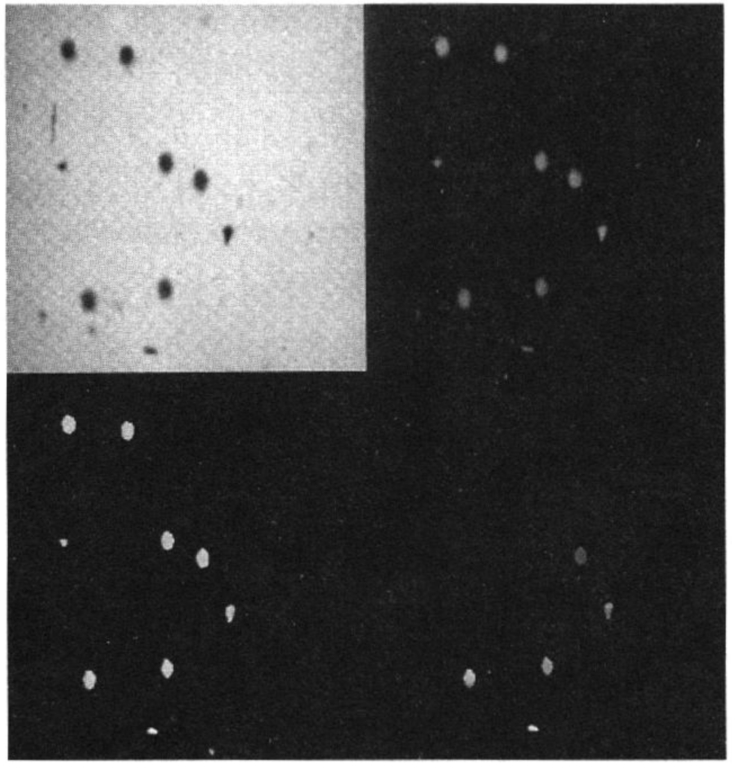
Connected Component Labeling

- When an object is burned out, all its pixels have value 0 and cannot be distinguished from the background.

- This procedure is repeated until all objects in the image are counted.

- A by-product of this algorithm is the ***area*** of each object (number of its pixel).

Connected Component Labeling



a) Microscopy image; b) Negative image; c) Thresholded negative image; d) Labelled connected regions (some of them are not visible).

Connected Component Labeling

Local CCL algorithm:

- Each pixel $f(n, l)$ having value 1 is labeled by the concatenation of its (n, l) coordinates.
- We scan the labeled image.
- We assign to each pixel the minimum of the labels in its 4-connected or 8-connected neighborhood.

- This process is repeated until no more label changes are made.

Connected Component Labeling

***Blob coloring algorithm*.**

- It has two passes:
  - In the first pass, colors are assigned to image pixels by using a three-pixel L-shaped mask, while color equivalencies are established and stored, when needed.
  - In the second pass, the pixels of each connected region are labeled with a unique color by using the color equivalences obtained in the first pass.

Connected Component Labeling

Shrinking algorithm.

- If a pixel $f(n,l)$ has value 1, it retains this value after local shrinking, if and only if at least one of its East, South or South-East neighbors has value 1.
- This local operation is described by the following recursive relation:

$$f(n,l) = h[h[f(n,l-1) + f(n,1) + f(n+1,l) - 1] + h[f(n,l) + f(n+1,l-1) - 1]].$$

Connected Component Labeling

- Function $h(t)$ is given by:

$$h(t) = \begin{cases} 0, & \text{for } t \leq 0, \\ 1, & \text{for } t > 0. \end{cases}$$

- After repeated binary image scanning by this shrinking operation, each connected component shrinks to the North-West corner of its bounding box, before it vanishes at the next shrinking operation.

Connected Component Labeling

Divide-and-conquer CCL algorithm.

- It uses the split and merge algorithm:
  - Inhomogeneous regions consisting of 0s and 1s are split recursively, until we reach homogeneous regions consisting only of 1s.
  - These regions are assigned a unique label (split step).
  - Label equivalences can be established, by checking the borders of all homogeneous regions.
  - Those regions having equivalent labels are merged to a single connected component.

Region Segmentation

- Introduction
- Image Thresholding
- Region Growing
- Split/Merge Techniques
- Relaxation Algorithms in Region Analysis
- Connected Component Labeling
- **Texture Description.**

Texture description

Image texture is a measure of image coarseness, smoothness and regularity.

- Texture description methods:

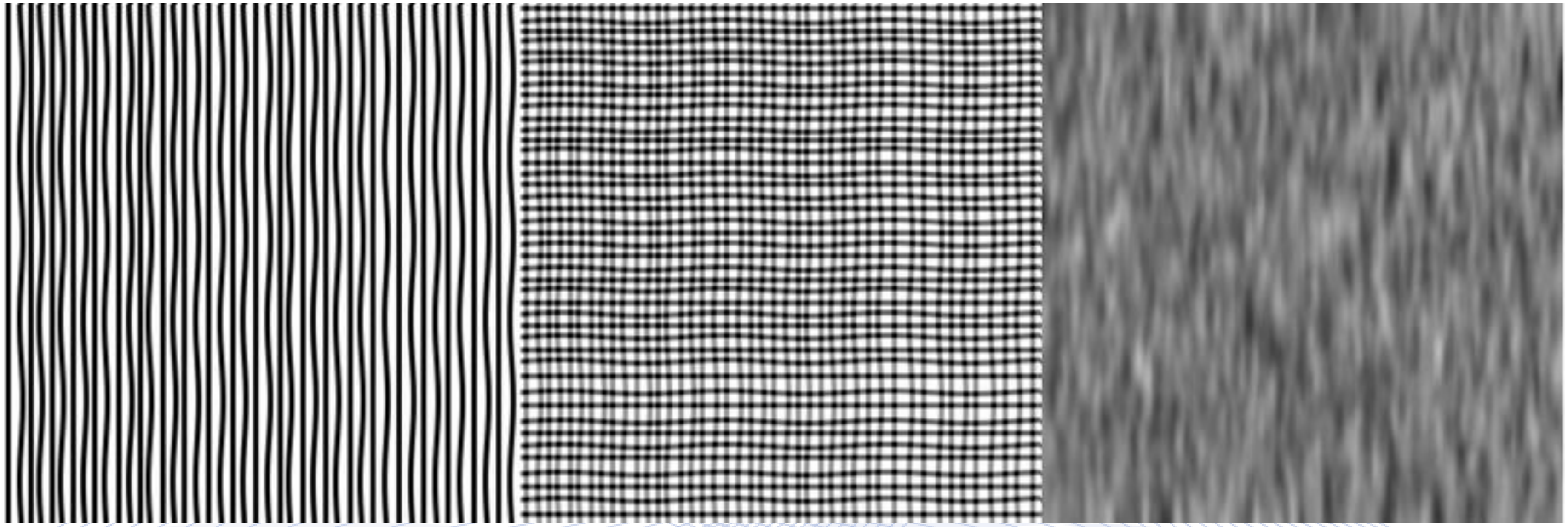
  - ***Statistical techniques***:
    - They are based on region histograms.
    - They measure contrast, granularity, and coarseness.

Texture description

a) Coarse image texture;

b) Fine image texture.

Texture description



Directional image texture [RES].

Texture description

- ***Spectral methods***:
  - They are based on:
    - ***autocorrelation function*** of an image region or
    - ***image periodogram*** (Fourier transform power distribution),
  - in order to exploit ***texture periodicity***.
- ***Structural methods***:
  - They describe the texture by using pattern primitives accompanied by certain geometrical placement rules.

Texture description

The simplest texture descriptors are based on ***image pixel probability distribution*** (***pdf***) $p_f(f)$.

- Image histogram is an estimation of pixel pdf, when assuming ***image signal stationarity***.
- Let $f_k, k = 1, \ldots, N$ be the various image intensity levels.
- The first four histogram central moments are given by:
  - ***Image Mean***:

$$\mu = \sum_{k=1}^{N} f_k p_f(f_k).$$

Texture description

- ***Image Skewness***:

$$\mu_3 = \frac{1}{\sigma^3} \sum_{k=1}^{N} (f_k - \mu)^3 p_f(f_k) \,.$$

- ***Image Variance***:

$$\sigma^2 = \sum_{k=1}^{n} (f_k - \mu)^2 p_f(f_k) \,.$$

Texture description

- ***Image Kurtosis***:

$$\mu_4 = \frac{1}{4}\sum_{k=1}^{n}(f_k - \mu)^4 p_f(f_k) - 3\,.$$

- ***Image entropy*** is defined in terms of the histogram as well:

$$H = -\sum_{k=1}^{N} p_f(f_k)\ln p_f(f_k)$$

and can be used for feature description.

Texture description

Spatial information can be described by using the ***histogram of grey-level differences***:

- Let $\mathbf{d} = [d_1, d_2]^T$ be the ***displacement vector*** between two image pixels and $g(\mathbf{d})$ the grey-level difference at a displacement $\mathbf{d}$:

$$g(\mathbf{d}) = |f(k, l) - f(k + d_1, l + d_2)|.$$

- $p_g(g, \mathbf{d})$ denotes the ***grey-level difference histogram*** at a displacement $\mathbf{d}$.

Texture description

- If an image region has ***coarse texture***, the histogram $p_g(g, \mathbf{d})$ tends to concentrate around $g = 0$ for small displacements $\mathbf{d}$.

- If the region has ***fine texture***, it tends to spread, when is larger than the ***texture grain*** size.

Texture description

Several texture measures can be extracted from the histogram of grey-level differences:

- ***Mean:***

$$\mu_{\mathbf{d}} = \sum_{k=1}^{N} g_k p_g(g_k, \mathbf{d}).$$

- ***Variance***:

$$\sigma_{\mathbf{d}}^2 = \sum_{k=1}^{N} (g_k - \mu_{\mathbf{d}})^2 p_g(g_k, \mathbf{d}).$$

Texture description

- ***Contrast***:

$$c_{\mathbf{d}} = \sum_{k=1}^{N} g_k^2 p_g(g_k, \mathbf{d}).$$

- ***Entropy***:

$$H_{\mathbf{d}} = -\sum_{k=1}^{N} p_g(g_k, \mathbf{d}) \ln p_g(g_k, \mathbf{d}).$$

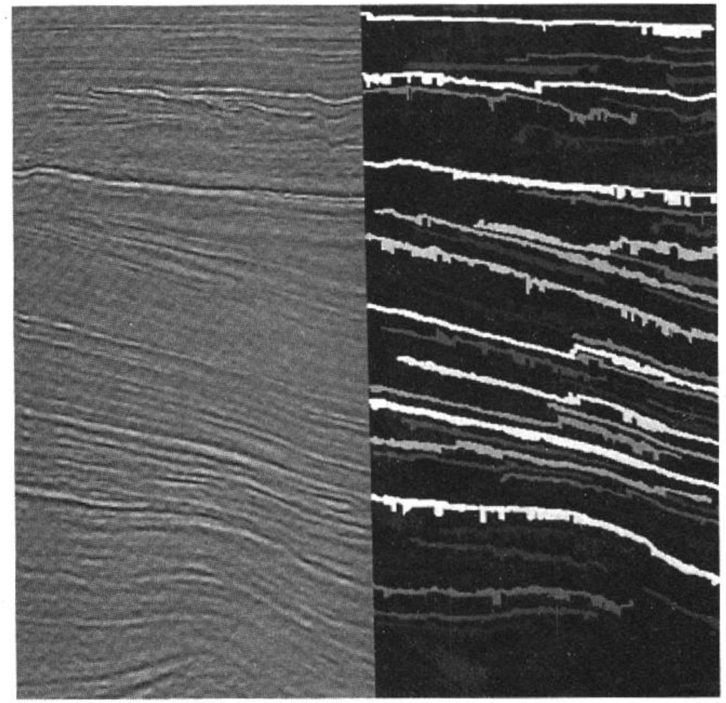
- Advantages: computational simplicity and capability to give information about the spatial texture organization.

Texture description

- A ***run length*** l of pixels having equal intensity f in a direction θ is an event denoted by (l, f, θ).

  - Run lengths reveal both ***texture directionality*** and ***texture coarseness***.
  - Coarse textures tend to produce long grey-level runs.
  - Directional texture tends to produce long runs at specific directions θ.

Texture description



a) Original image; b) Run-length image.

Texture description

Let $N(l, f, \theta)$ denote the number of events (l, f, θ) in an image having dimensions $N_1 \times N_2$ and N_R denote the total number of existing runs:

$$T_R = \sum_{k=1}^{N} \sum_{l=1}^{N_R} N(l, f_k, \theta).$$

- The ratio $N(l, f, \theta)/T_R$ is the ***grey-level run histogram*** at a specific direction θ.

Texture description

The following texture features can be calculated from the grey-level run lengths:

- ***Short-run emphasis***:

$$A_1 = \frac{1}{T_R} \sum_{k=1}^{N} \sum_{l=1}^{N_R} \frac{1}{k^2} N(l, f_k, \theta) .$$

- ***Long-run emphasis***:

$$A_2 = \frac{1}{T_R} \sum_{k=1}^{N} \sum_{l=1}^{N_R} k^2 N(l, f_k, \theta) .$$

Texture description

- ***Grey-level distribution***:

$$A_3 = \frac{1}{T_R} \sum_{k=1}^{N} \left[\sum_{l=1}^{N_R} \frac{1}{k^2} N(l, f_k, \theta) \right]^2 .$$

- ***Run-length distribution***:

$$A_4 = \frac{1}{T_R} \sum_{l=1}^{N_R} \left[\sum_{k=1}^{N} \frac{1}{k^2} N(l, f_k, \theta) \right]^2 .$$

Texture description

- ***Run percentages***:

$$A_5 = \frac{1}{N_1 N_2} \sum_{k=1}^{N} \sum_{l=1}^{N_R} N(l, f_k, \theta).$$

Texture description

Grey-level co-occurrence matrix elements $p(f_k, f_l, \mathbf{d})$ denote the joint probability of two pixels f_k, f_l that are displaced by $\mathbf{d}$.

- It is estimated from an image by counting the number n_{kl} of occurrences of the pixel values f_k, f_l distanced by displacement $\mathbf{d}$ in the image.
- If n be the total number of any possible joint pairs, co-occurrence matrix elements $C_{\mathbf{d}}(k, l)$ are given by:

$$C_{\mathbf{d}}(k, l) = \hat{p}(f_k, f_l, \mathbf{d}) = \frac{n_{kl}}{n}.$$

Texture description

- Co-occurrence matrix $\mathbf{C_d}$ has dimension $N \times N$, where N is the number of grey levels in the image.
- Co-occurrence matrices carry very useful information about spatial texture organization.
  - If the texture is coarse, their mass tends to be concentrated around the main diagonal of $\mathbf{C_d}$.
  - If the texture is fine, co-occurrence matrix values are much more spread.
  - If texture carries strong directional information along direction $\mathbf{d}$, co-occurrence matrix entries tends to have their mass in the main diagonal of $\mathbf{C_d}$.

Texture description

Several texture descriptors have been proposed to characterize the cooccurrence matrix content:

- ***Maximum probability***:

$$p_{\mathbf{d}} = max_{k,l} C_{\mathbf{d}}(k,l).$$

- ***Entropy***:

$$H_{\mathbf{d}} = -\sum_{k=1}^{N}\sum_{l=1}^{N} C_{\mathbf{d}}(k,l) \ln C_{\mathbf{d}}(k,l) .$$

- ***Moment of order*** m:

$$I_{\mathbf{d}} = \sum_{k=1}^{N}\sum_{l=1}^{N} |k-l|^m C_{\mathbf{d}}(k,l).$$

Texture description

Spectral texture characterization is based on:

- ***image power spectrum***, e.g., ***periodogram*** $|F(u,v)|^2$:

$$F(u,v)=\sum_{n=0}^{N-1}\sum_{m=0}^{M-1}f(n,m)exp\left[-i\left(\frac{2\pi nu}{N}+\frac{2\pi mv}{M}\right)\right].$$

- ***autocorrelation function*** $R_{ff}(k,l)$ of an image $f(i,j)$:

$$R_{ff}(k,l)=\frac{1}{(2N_1+1)(2N_2+1)}\sum_{i=-N_1}^{N_1}\sum_{i=-N_2}^{N_2}f(i,j)f(i+k,j+l).$$

Texture description

- It can be calculated both for positive and negative lags (k, l).
- It usually attains a maximum for zero lag $(0,0)$.
- It drops exponentially with (k, l) (positive or negative).
- Direct definition-based computation of the autocorrelation function is preferred for a small number of lags (k, l).

- The calculation of $R_{ff}(k, l)$ for a large number of lags is performed using 2D FFT.

Texture description

- Autocorrelation function $R_{ff}(k,l)$ is given by the inverse 2D DFT:

$$R_{ff}(k,l) = \frac{1}{NM}\sum_{u=0}^{N-1}\sum_{u=0}^{M-1} F(u,v)F^*(u,v)exp\left[i\left(\frac{2\pi ku}{N}+\frac{2\pi lv}{M}\right)\right].$$

- Autocorrelation function $R_{ff}(k,l)$ is the inverse 2D DFT of Periodogram:

$$|F(u,v)|^2 = F(u,v)F^*(u,v).$$

Texture description

- Pre-multiplication of the image $f(m,n)$ by a two-dimensional window $w(m,n)$ produces a relatively smooth power spectrum estimate.

- Both 2D DFT and inverse 2D DFT can be calculated via 2D Fast Fourier Transform algorithms.

Texture description

- If polar coordinates are used for power spectrum $R_{ff}(r, \varphi)$ description:

$$r = \sqrt{\omega_1^2 + \omega_2^2}.$$

$$\phi = arc\tan\left(\frac{\omega_2}{\omega_1}\right).$$

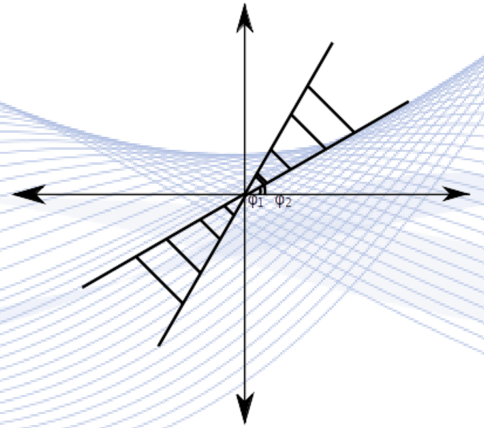
- ***Angular power spectrum distribution*** $P_\phi(\phi)$ is a very good descriptor of texture directionality:

$$P_\phi(\phi) = \int_0^{r_{max}} P_{ff}(r, \phi) dr.$$

Texture description

- This integral can be approximated by a summation within a wedge $\phi_1 \leq \phi < \phi_2$ in the spectral domain:

$$P_\phi(\phi) \approx \sum_{\sqrt{\omega_1^2+\omega_2^2}<r_{max},\ \phi_1\leq\phi<\phi_2} |F(\omega_1,\omega_2)|^2 .$$



Integration wedge for the calculation of $P_\phi(\phi)$.

Artificial Intelligence & Information Analysis Lab

Texture description

- ***Radial power spectrum distribution***:

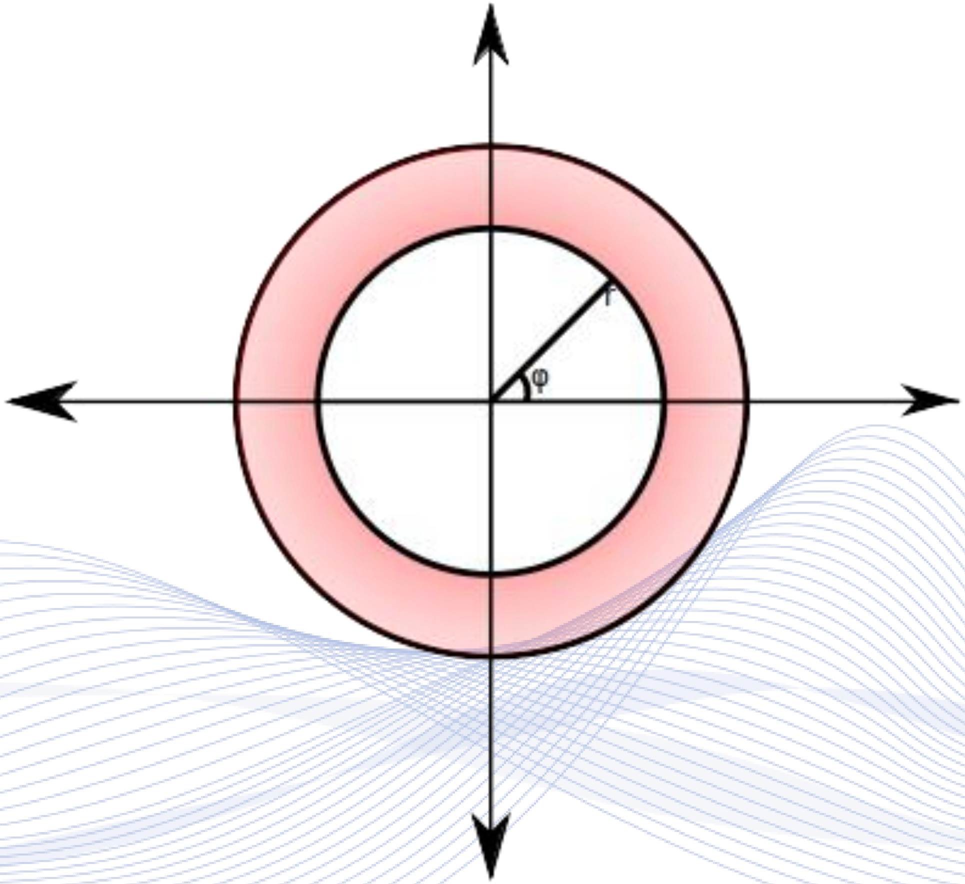
$$P_r(r) = \int_0^{2\pi} P_{ff}(r, \phi) d\phi$$

can describe texture coarseness.

- It can be approximated, by splitting the spectral domain into concentric rings:

$$P_r(r) \approx \sum_{r_1^2 \le \sqrt{\omega_1^2 + \omega_2^2} < r_2^2} |F(\omega_1, \omega_2)|^2, \qquad r_1 \le r < r_2 .$$

Texture description



Integration ring for the calculation of $P_r(r)$.

Bibliography

[PIT2019] I. Pitas, “Computer vision”, Createspace/Amazon, in press.

[SZE2011] R.Szelinski, “ Computer Vision ” , Springer 2011

[PIT2017] I. Pitas, “Digital video processing and analysis ” , China Machine Press, 2017 (in Chinese).

[PIT2013] I. Pitas, “Digital Video and Television ” , Createspace/Amazon, 2013.

[PIT2000] I. Pitas, Digital Image Processing Algorithms and Applications, J. Wiley, 2000.

[NIK2000] N. Nikolaidis and I. Pitas, 3D Image Processing Algorithms, J. Wiley, 2000.

[APOLLO] http://apolloscape.auto/

[RES] https://www.researchgate.net/figure/1-directional-2-directional-and-the-lic-like-texture-patterns-as-in-KHSI03b_fig8_228697126

Q & A

Thank you very much for your attention!

More material in
http://icarus.csd.auth.gr/cvml-web-lecture-series/

Contact: Prof. I. Pitas
pitas@csd.auth.gr