

DigPro2 Summary

J.Rast, G.Köppel, R.Nestler, H.Badertscher *et. al*

September 8, 2014

This summary follows the lectures of Prof. Dr. G. Schuster and the book "Digital Image Processing" by Rafael C. Gonzalez and Richard E. Woods. Most figures and formulas are taken from there.

Contents

9 Morphological Image Processing	Ch. 9 S627-688	3
9.1 Set Theory		3
9.2 Erosion and Dilation		3
9.3 Opening and Closing		3
9.4 The hit-or-miss transform	S640	4
9.5 Some basic morphological algorithms	S642	4
9.6 Gray-scale morphology	S665	8
10 Image Segmentation	Ch. 10 S689-794	10
10.1 Fundamentals	S690	10
10.2 Point, line and edge detection	S692	10
10.3 Thresholding	S738	16
10.4 Region based segmentation	S763	20
10.5 Segmentation using morphological watersheds	S769	20
10.6 The use of motion in segmentation	S778	21
11 Representation and Description	Ch. 11 S796-860	22
11.1 Representation	S796	22
11.2 Boundary Descriptors	S815	26
11.3 Regional descriptors	S822	28
11.4 Use of Principal Components for Description	S842	31
11.5 Relational Descriptors	S852	32
12 Object Recognition	Ch. 12 S861-907	33
12.1 Patterns and Pattern Classes	Ch. 12.1	33
12.2 Recognition Based on Decision Theory	Ch. 12.2	34
12.3 Structural Methods	Ch. 12.3	39

9 Morphological Image Processing Ch. 9 S627-688

9.1 Set Theory

9.1.1 Reflection S628

$$\hat{B} = \{w | w = -b, \text{ for } b \in B\}$$

9.1.2 Translation S629

$$(B)_z = \{c | c = b + z, \text{ for } b \in B\}$$

9.2 Erosion and Dilation

9.2.1 Erosion S631

With A and B as sets in Z^2 , the erosion of A by B is defined as

$$A \ominus B = \{z | (B)_z \subseteq A\}$$

This is the set of all translations by z such that the set B is still a subset of A.

Equivalently, if B must be contained in A, it is not allowed to intersect with the background:

$$A \ominus B = \{z | (B)_z \cap A^c = \emptyset\}$$

9.3 Opening and Closing

9.3.1 Opening and closing S635

The *opening* of A by B is defined: (first erosion then dilation)

$$A \circ B = (A \ominus B) \oplus B$$

The *closing* of A by B is defined: (first dilation then erosion)

$$A \bullet B = (A \oplus B) \ominus B$$

Opening and closing are dual operations:

$$(A \bullet B)^c = (A^c \circ \hat{B})$$

$$(A \circ B)^c = (A^c \bullet \hat{B})$$

The **opening** operation satisfies the following properties:

- (a) $A \circ B$ is a subset (subimage) of A.
- (b) If C is a subset of D, then $C \circ B$ is a subset of $D \circ B$.
- (c) $(A \circ B) \circ B = A \circ B$

The **closing** operation satisfies the following properties:

- (a) A is a subset (subimage) of $A \bullet B$.
- (b) If C is a subset of D, then $C \bullet B$ is a subset of $D \bullet B$.
- (c) $(A \bullet B) \bullet B = A \bullet B$

9.2.2 Dilation S633

With A and B as sets in Z^2 , the dilation of A by B is defined as

$$A \oplus B = \{z | (\hat{B})_z \cap A \neq \emptyset\}$$

This is the set of all translations by z such that the set B reflected about its origin has at least one overlap with the set A.

Equivalently, since the intersection is not allowed to be empty and all elements in it must be from A:

$$A \oplus B = \{z | [(\hat{B})_z \cap A] \subseteq A\}$$

9.2.3 Duality erosion/dilation S635

Erosion and dilation are dual:

$$(A \ominus B)^c = A^c \oplus \hat{B}$$

$$(A \oplus B)^c = A^c \ominus \hat{B}$$

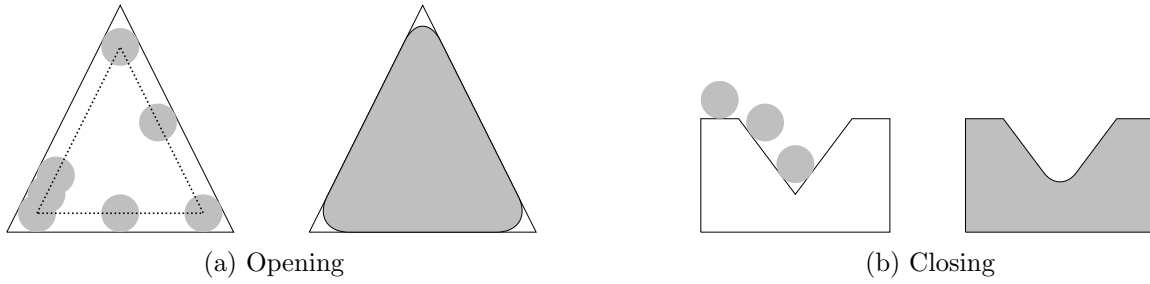


Figure 1: Geometrical interpretation of the opening and closing transformations

9.4 The hit-or-miss transform S640

The hit-or-miss transform is a basic tool for shape detection. Basic idea:

- Through erosion of the image by the object, possible locations for the object in the image are left, since elements smaller than the objects disappear
- Through erosion of the background of the image with the local background of the object, possible locations for the object in the image are left
- The intersection of these possible locations result in a reliable detection of the object location

$$A \circledast B = (A \ominus D) \cap [A^c \ominus (W - D)]$$

- In general, if B1 is the object and B2 is the corresponding background this can be written

$$A \circledast B = (A \ominus B_1) \cap (A^c \ominus B_2)$$

9.5 Some basic morphological algorithms S642

9.5.1 Boundary Extraction S642

$$\beta(A) = A - (A \ominus B)$$

The mask



leads to a 1 pixel thick boundary

9.5.2 Hole Filling S643

To fill a hole, a seed inside the connected border is needed, i.e., in an empty array X_0 , the seed point is set to 1.

Usual mask:



The dilation is forced to stay inside the connected border with $\cap A^c$. The iteration can stop, when $X_k = X_{k-1}$.

$$X_k = (X_{k-1} \oplus B) \cap A^c \quad k = 1, 2, 3, \dots$$

9.5.3 Extraction of Connected Components S645

Initial array (X_0) contains a 1 in the region of the connected component.

The iteration can stop, when $X_k = X_{k-1}$.

$$X_k = (X_{k-1} \oplus B) \cap A \quad k = 1, 2, 3, \dots$$

The mask



leads to 8-connected regions.

9.5.4 Convex hull S647

- The convex hull H of a set S is the smallest convex set containing S .
- $H-S$ is called the convex deficiency of S

The convex hull is constructed with a convex hull with four structuring elements. (x = don't care)



Figure 2: Structuring elements for convex hull

$$\begin{aligned}
 X_0^i &= A \\
 X_k^i &= (X_{k-1}^i \circledast B^i) \cup A \\
 \text{when } X_k^i &= X_{k-1}^i \longrightarrow D^i = X_k^i \\
 C(A) &= \bigcup_{i=1}^4 D_i
 \end{aligned}
 \quad i = 1, 2, 3, 4 \text{ and } k = 1, 2, 3, \dots$$

Note that the hit-or-miss transform $\circledast$ here is applied without background.

This algorithm does not guarantee that the found convex set is the convex hull, which has to be the smallest convex set containing the original object

The following additional constraints can be used:

- easiest is, that the convex hull is not allowed to grow beyond the original vertical and horizontal dimension
- More complex constraints are possible, but at a higher computational cost

9.5.5 Thinning S649

Thinning is the operation of making the foreground thinner.

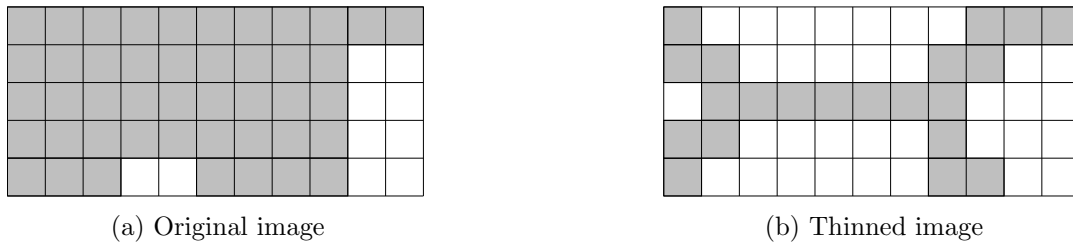


Figure 3: Thinning

$$A \otimes B = A - (A \circledast B)$$

Hit-or-miss transformation with rotating structuring elements. Note that the hit-or-miss is applied without background.

$$A \otimes B = ((\dots((A \otimes B^1) \otimes B^2) \dots) \otimes B^n)$$

This is repeated until nothing changes anymore.

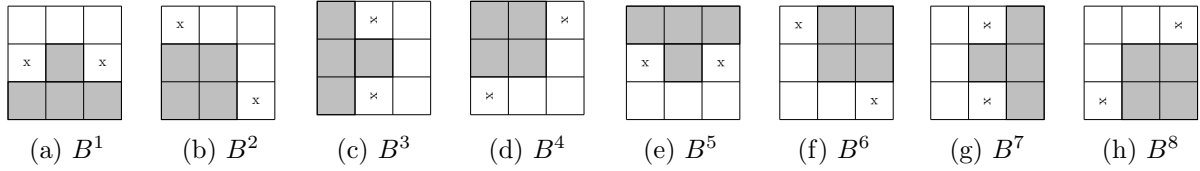


Figure 4: Thinning masks

9.5.6 Thickening S650

This is the morphological dual of thinning. It is implemented as thinning the background and then taking the complement. Thickening can lead to disconnected points. These have to be removed by post-processing.

9.5.7 Skeleton S(A) S651



Figure 5: Skeleton algorithm: basic idea

Finding a disk D that:

- centers at z (z being a point of $S(A)$)
- there is no larger disk containing D and included in A
- touches the boundary of A at two or more places

This is realized by

$$S(A) = \bigcup_{k=0}^K S_k(A)$$

with

$$S_k(A) = (A \ominus kB) - (A \ominus kB) \circ B$$

$$B = \begin{bmatrix} & & \\ & \cdot & \\ & & \end{bmatrix}$$

and

$$K = \max \{k | (A \ominus kB \neq \emptyset)\}$$

K is the last iterative step before A is eroded to an empty step.

Reconstruction can be achieved by

$$A = \bigcup_{k=0}^K (S_k(A) \oplus kB)$$

Problems:

- The resulting skeleton is not connected
- The skeleton is a bit thicker than it needs to be.

9.5.8 Pruning S654

Pruning is a procedure that can remove parasitic components of a length below some threshold.

1. Find endpoints with a hit or miss transform with the following structuring elements. Remove the endpoints.
2. Repeat this. If you want to remove all spurs of length 3 or less, repeat for 3 times

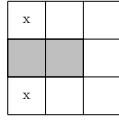
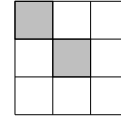
(a) B^1, B^2, B^3, B^4 (rotated 90°)(b) B^5, B^6, B^7, B^8 (rotated 90°)

Figure 6: Pruning structuring elements

3. Find the endpoints using the same structuring elements
4. Dilate the endpoints with a 3x3 structuring element and make a logical and with the original set

9.5.9 Morphological Reconstruction S656

Geodesic dilation and erosion S656 This is a transformation which uses two images and a structuring element

- F is the marker image
- G is the mask image
- Both are binary and F is a subset of G

Note that morphological reconstruction will always converge after a finite number of steps.

Geodesic dilation

$$D_G^{(1)}(F) = (F \oplus B) \cap G$$

is the dilation of F by B , where the result must stay inside G . This can be repeated:

$$\begin{aligned} D_G^{(0)}(F) &= F \\ D_G^{(1)}(F) &= (F \oplus B) \cap G \\ D_G^{(n)}(F) &= D_G^{(1)}[D_G^{(n-1)}(F)] \end{aligned}$$

notice the recursive definition

Geodesic erosion

$$E_G^{(0)}(F) = F$$

is the erosion of F by B , where the result contains at least G .

$$\begin{aligned} E_G^{(1)}(F) &= (F \ominus B) \cup G \\ E_G^{(n)}(F) &= E_G^{(1)}[E_G^{(n-1)}(F)] \end{aligned}$$

Morphological reconstruction by dilation and by erosion S658 This is geodesic dilation/erosion until nothing changes.

$$\begin{aligned} \text{Reconstruction by dilation: } R_G^D(F) &= D_G^{(k)} \quad \text{with} \quad D_G^{(k)}(F) = D_G^{(k+1)}(F) \\ \text{Reconstruction by erosion: } R_G^E(F) &= E_G^{(k)} \quad \text{with} \quad E_G^{(k)}(F) = E_G^{(k+1)}(F) \end{aligned}$$

Sample applications S659 Opening and closing by reconstruction

Opening by reconstruction removes all small object, all others stay unchanged. Closing by reconstruction closes gaps, the rest is unchanged. This is in contrast to morphological opening and closing, where the erosion removes the small objects and the dilation "sort of" reconstructs the other objects.

$$\begin{aligned} O_R^{(n)}(F) &= R_F^D[(F \ominus nB)] \\ C_R^{(n)}(F) &= R_F^E[(F \oplus nB)] \end{aligned}$$

Filling holes in a binary image I

Without needing a seed!

$$H = [R_F^D(F)]^C \quad \text{with} \quad F(x, y) = \begin{cases} 1 - I(x, y) & \text{if } (x, y) \text{ is on the border of } I \\ 0 & \text{otherwise} \end{cases}$$

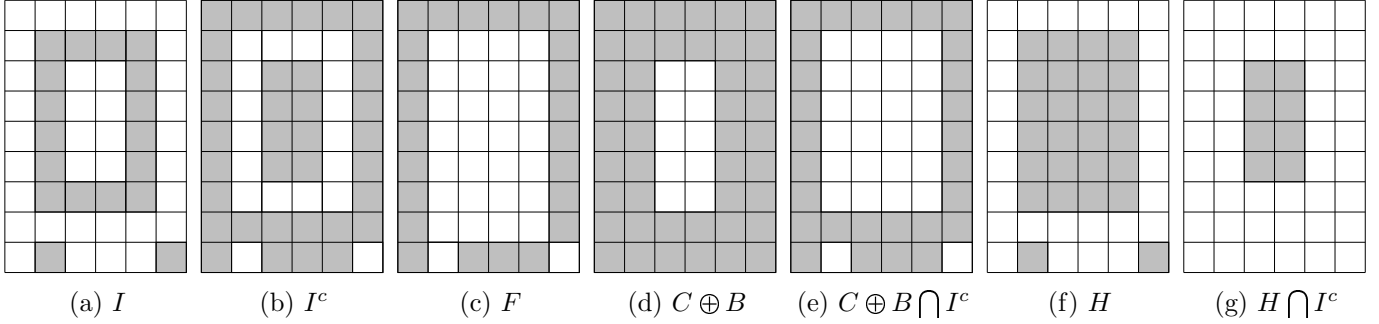


Figure 7: Hole Filling on a simple image

Border clearing S661 Remove object that touch the border.

$$X = I - R_I^D(F) \quad \text{with} \quad F(x, y) = \begin{cases} I(x, y) & \text{if } (x, y) \text{ is on the border of } I \\ 0 & \text{otherwise} \end{cases}$$

9.6 Gray-scale morphology S665

The fundamental operations dilation, erosion, opening and closing can be extended to gray scale. Hence the more advanced operations which are based on the above operations can be extended too. The structuring elements are also gray-scale. In practice, flat structuring elements are preferred.



Figure 8: Nonflat and flat structuring elements and corresponding horizontal intensity profiles through their center

9.6.1 Erosion and Dilation S666

Erosion

$$[f \ominus b](x, y) = \min_{(s, t) \in b} \{f(x + s, y + t)\} \quad \text{Flat SE}$$

$$[f \ominus b_N](x, y) = \min_{(s, t) \in b_N} \{f(x + s, y + t) - b_N(s, t)\} \quad \text{Non-Flat SE}$$

Dilation

$$\begin{aligned} [f \oplus b](x, y) &= \max_{(s,t) \in b} \{f(x-s, y-t)\} && \text{Flat SE} \\ [f \oplus b_N](x, y) &= \max_{(s,t) \in b_N} \{f(x-s, y-t) + b_N(s, t)\} && \text{Non-Flat SE} \end{aligned}$$

9.6.2 Opening and Closing S668

Opening and closing are defined exactly the same as in the binary case:

$$f \circ b = (f \ominus b) \oplus b \qquad f \bullet b = (f \oplus b) \ominus b$$

9.6.3 Some Basic Gray-Scale Morphological Algorithms S670

Morphological smoothing S670 Opening suppresses bright details and closing suppresses dark details. They can be used in combination for image smoothing and/or noise suppression.

Morphological gradient S671 This operation results in an image where the edges are enhanced, not unlike a classical magnitude gradient image.

$$g = (f \oplus b) - (f \ominus b)$$

Top-hat transformation S672

$$T_{hat}(f) = f - (f \circ b)$$

The resulting image contains mostly bright details that are smaller than the structuring element.

Bottom-hat transformation S672

$$B_{hat}(f) = (f \bullet b) - f$$

The resulting image contains mostly dark details, that are smaller than the structuring element.

Granulometry S674 The goal of granulometry is to find the particle size distribution in an image. Opening with a structuring element similar in shape as the particles but the size scaled from small to large. The biggest effect in the resulting opened image is, when the size of the structuring element changes from just too small to just too large. This effect can be measured by summing up the entire opened image, the surface area. Jumps in the change of surface area show the dominant particle sizes.

Textural segmentation S675 Figure 9.43

The goal of textural segmentation is, to segment the image into two textural regions. The basic idea is to remove the small circles with a structuring element larger than the small circle but smaller than the large circle. With a following opening fills the white patches between the large elements.

Gray-scale morphological reconstructions S676 Follows the definition for binary images.

Geodesic dilation

$$\begin{aligned} D_g^{(1)}(f) &= (f \oplus b) \wedge g \\ D_g^{(n)}(f) &= D_g^{(1)}[D_g^{(n-1)}(f)]g \end{aligned}$$

$\wedge$ denotes the point-wise minimum operator.

Geodesic erosion

$$\begin{aligned} E_g^{(1)}(f) &= (f \ominus b) \vee g \\ E_g^{(n)}(f) &= E_g^{(1)}[E_g^{(n-1)}(f)] \end{aligned}$$

$\vee$ denotes the point-wise maximum operator.

Opening by reconstruction

$$O_R^{(n)}(f) = R_f^D [(f \ominus nb)]$$

Closing by reconstruction

$$C_R^{(n)}(f) = R_f^E [(f \oplus nb)]$$

10 Image Segmentation Ch. 10 S689-794

Preliminaries

- Separate the image into different parts, i.e. regions or objects
- Often a scene can be designed that the segmentation task is simplified
- Segmentation is based on some property of the scene
 - Intensity discontinuity
 - Intensity similarity
 - More Complex properties: Motion, Texture, Shape, Color, etc.

10.1 Fundamentals S690

- | | |
|--|--|
| <p>a) $\bigcup_{i=1}^n R_i = R$</p> <p>b) R_i is a connected set, $i = 1, 2, \dots, n$</p> <p>c) $R_i \cap R_j = \emptyset$ for all i and $j, i \neq j$</p> <p>d) $Q(R_i) = \text{TRUE}$ for $i = 1, 2, \dots, n$</p> <p>e) $Q(R_i \cup R_j) = \text{FALSE}$ for any adjacent Regions R_i and R_j</p> | <ul style="list-style-type: none"> • The entire spatial region that is occupied by an image is called R • Image segmentation is the task of partitioning R into sub-regions R_i • $Q()$ is a measure of similarity of a particular property • d) must be true, since this defines the region • e) must be false, since otherwise, the regions should have been merged |
|--|--|

Focus on the intensity values

- there are discontinuities, then a new region starts.
- If the intensity does not vary much, then this is one region.

10.2 Point, line and edge detection S692

When the intensity changes abruptly in a local neighborhood, this indicates an edge. It can be a simple point or a line.

10.2.1 Background S692

Abrupt local changes can be detected using derivatives.

First order derivative

$$\frac{\partial f}{\partial x} = f'(x) = f(x+1) - f(x)$$

The second order derivative

$$\frac{\partial^2 f}{\partial x^2} = \frac{\partial f'(x)}{\partial x} = f''(x) = f(x+1) + f(x-1) - 2f(x)$$

This can be calculated using a spatial filter.

$$\begin{bmatrix} w_1, w_2, w_3 \\ w_4, w_5, w_6 \\ w_7, w_8, w_9 \end{bmatrix}$$

- First order in y direction $w_6 = 1, w_5 = -1$
- First order in x direction $w_8 = 1, w_5 = -1$
- Second order in y direction $w_6 = 1, w_4 = 1, w_5 = -2$
- Second order in x direction $w_8 = 1, w_2 = 1, w_5 = -2$

10.2.2 Detection of Isolated Points S696

The Laplacian combines the second derivatives in both spatial directions. This results in

$$\nabla^2 f(x, y) = f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1) - 4f(x, y)$$

It can be extended to also include the diagonal terms which results in the following mask

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

The Laplacian is isotropic with respect to $0^\circ, 45^\circ, 90^\circ$ and 135° .

Point detection is now achieved by filtering the image with the mask and thresholding this result image R .

$$g(x, y) = \begin{cases} 1 & \text{if } |R(x, y)| \geq T \\ 0 & \text{otherwise} \end{cases}$$

10.2.3 Line Detection S697

Often, a line of a known direction should be detected. We use special masks for that particular direction

-1	-1	-1
2	2	2
-1	-1	-1

(a) Horizontal

2	-1	-1
-1	2	-1
-1	-1	2

(b) $+45^\circ$

-1	2	-1
-1	2	-1
-1	2	-1

(c) Vertical

-1	-1	2
-1	2	-1
2	-1	-1

(d) -45°

Figure 9: Line detection masks

10.2.4 Edge Models

The typical intensity profile of an edge is shown in figure 10. Note that the derivatives are very sensitive to noise. Usually the image is smoothed before edge detection.

10.2.5 Edge localization S706

The previously shown method generates edge points. The goal is to keep the ones which truly belong to an edge. Since first order derivatives are helpful, a nice way of combining the two partial derivatives into one value is the magnitude of the gradient

$$\nabla f = \nabla f = \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

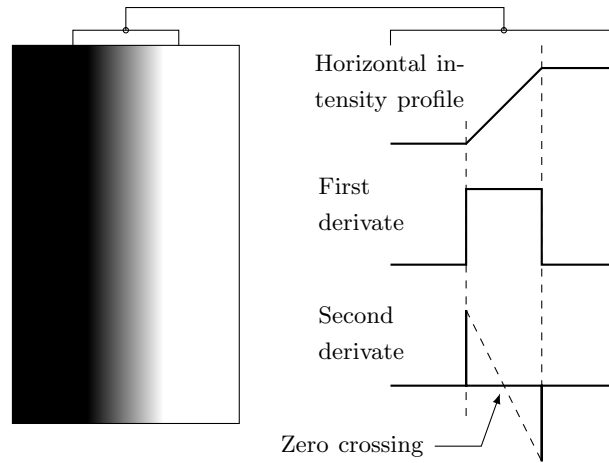


Figure 10: Ramp Edge with intensity profile and derivatives.

-1	-1	-1	-1	0	1	-1	-2	-1	-1	0	1
0	0	0	-1	0	1	0	0	0	-2	0	2
1	1	1	-1	0	1	1	2	1	-1	0	1
(a) Prewitt			(b) Prewitt			(c) Sobel			(d) Sobel		

Figure 11: Gradient operators: Prewitt and Sobel masks

The gradient (which is a vector) points into the direction of the greatest rate of change of f at the location (x,y) . The magnitude of the gradient vector is the value of the rate of change in that direction.

After the gradient images have been calculated, often the magnitude of the gradient is required.

$$M(x,y) = \text{mag}(\nabla f) \sqrt{g_x^2 + g_y^2} \approx |g_x| + |g_y| \quad \alpha(x,y) = \tan^{-1} \left[\frac{g_y}{g_x} \right]$$

If the goal is to find the dominant edges, then smoothing the image before calculating the magnitude gradient image works well. Alternatively, the magnitude gradient image can also be thresholded such that only values above this value are considered edges (set to one). Everything else is set to 0.

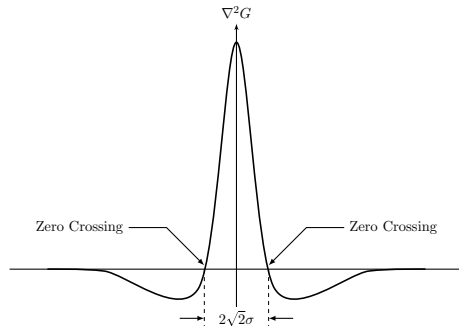
10.2.6 More Advanced Techniques for Edge Detection S714

Marr-Hildreth edge detector S714 New Idea: Edges exist on different scales. The edge detector needs to be tuned to the scale of the desired edge. Use a Gaussian filter (with space constant σ) and then take the Laplacian. The Gaussian filter is a low pass, which blurs the image for large σ . This results in a Laplacian of Gaussian (LoG) filter.

$$\nabla^2 G(x,y) = \left[\frac{x^2 + y^2 - 2\sigma^2}{\sigma^4} \right] e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Summary of the Marr-Hildreth edge detection:

1. Filter the input image with an $n \times n$ Gaussian lowpass filter $G(x,y) = e^{-\frac{x^2+y^2}{2\sigma^2}}$
2. Compute the Laplacian of the image resulting from Step 1. $\nabla^2[G(x,y) \star f(x,y)]$
3. Find the zero crossing of the image from Step 2.



0	0	-1	0	0
0	-1	-2	-1	0
-1	-2	16	-2	-1
0	-1	-2	-1	0
0	0	-1	0	0

(a) Cross Section of the negative of the LoG, showing zero crossings (b) 5x5 mask approximation. The negative of this mask would be used in practice

Figure 12: Marr-Hildreth

Size of the discrete LoG filter Rule of thumb: $n \times n$ LoG filter should have the smallest odd integer n greater than or equal to 6σ .

How to find the zero crossings?

A zero-crossing exists, if at least two opposing pixelpairs (a-a, b-b, c-c, d-d) have a sign difference.

a	b	c
d	0	d
c	b	a

Figure 13: Zero-Crossing: opposing pixel pairs

LoG DoG The Laplacian of Gaussian (LoG) can be approximated using a difference of Gaussian (DoG) filter

$$DoG(x, y) = \frac{1}{2\pi\sigma_1^2} e^{-\frac{x^2+y^2}{2\sigma_1^2}} - \frac{1}{2\pi\sigma_2^2} e^{-\frac{x^2+y^2}{2\sigma_2^2}}$$

$$\sigma^2 = \frac{\sigma_1^2\sigma_2^2}{\sigma_1^2 - \sigma_2^2} \ln \left[\frac{\sigma_1^2}{\sigma_2^2} \right] \quad (1)$$

- To compare LoG and DoG filters, they must have the same zero crossing
- This can be achieved by setting σ of the LoG in equation (1)
- amplitude will scale differently, both can be normalized, such that the amplitude at 0 is 1
- The negative functions are usually shown
- The LoG and the DoG filters can be implemented by 1 D convolutions (they are separable)

Canny edge detector S719

1. Smooth the image with a Gaussian filter.

$$G(x, y) = e^{-\frac{x^2+y^2}{2\sigma^2}}$$

$$f_s(x, y) = G(x, y) \star f(x, y)$$

2. Compute the gradient magnitude and angle images (Sobel operators).

$$M(x, y) = \sqrt{g_x^2 + g_y^2} \approx |g_x| + |g_y|$$

$$\alpha(x, y) = \tan^{-1} \left[\frac{g_y}{g_x} \right]$$

3. Apply nonmaxima suppression to the gradient image: Let d_1, d_2, d_3 and d_4 denote the four basic edge directions for a 3x3 region: horizontal, -45° , vertical and 45° . We can formulate the following nonmaxima suppression scheme for a 3x3 region centered at every point (x, y) in $\alpha(x, y)$:

- a) Find the direction d_k that is closest to $\alpha(x, y)$
- b) If the value of $M(x, y)$ is less than at least one of its two neighbors along d_k , let $g_N(x, y) = 0$ (suppression); otherwise, let $g_N(x, y) = M(x, y)$

Now g_N is the nonmaxima-suppressed gradient magnitude image.

4. Use double thresholding and connectivity analysis to detect and link edges:

- T_H a high threshold. If the values are above T_H , they are almost certainly a true edges

$$g_{NH}(x, y) = g_N(x, y) \geq T_H$$

- T_L a low threshold. High enough that most of the noise is filtered out, but much lower (2-3 times lower) than T_H resulting in edge points that are likely to be true edges, but might also contain erroneous edges

$$g_{NL}(x, y) = g_N(x, y) \geq T_L \quad g_{NL} = g_{NL}(x, y) - g_{NH}(x, y)$$

- Now in $g_{NH}(x, y)$ are the strong edge pixels, while in $g_{NL}(x, y)$ are the weak ones.
- A large T_H leads to gaps in the edges, which now can be filled using weak pixels in $g_{NL}(x, y)$ that are 8-connected to valid edge pixels in $g_{NH}(x, y)$:
 - a) Locate the next unvisited edge pixel, p , in $g_{NH}(x, y)$
 - b) Mark as valid edge pixels all the weak pixels in $g_{NL}(x, y)$ that are connected to p using 8-connectivity.
 - c) If all nonzero pixels in $g_{NH}(x, y)$ have been visited go to Step d. Else, return to step a.
 - d) Set to zero all pixels in $g_{NL}(x, y)$ that were not marked as valid edge pixels.

10.2.7 Edge linking and boundary detection S725

Three schemes to link edges:

Local processing S726 For every edge-point at a location the gradient vector magnitude and angle are compared with the magnitude and angle of edge-points in a neighborhood. If these are similar, they belong to the same boundary. This can be indicated by giving them the same color and/or gray value.

$$|M(s, t) - M(x, y)| \leq E \quad |\alpha(s, t) - \alpha(x, y)| \leq A$$

A much simpler scheme can be used if the linking follows certain directions and the maximum gap size is roughly known.

1. Compute the gradient magnitude and angle arrays, $M(x, y)$ and $\alpha(x, y)$ of the input image, $f(x, y)$

- Form a binary image, g , whose value at any pair of coordinates (x, y) is given by:

$$g(x, y) = \begin{cases} 1 & \text{if } M(x, y) > T_M \text{ and } \alpha(x, y) = A \pm T_A \\ 0 & \text{otherwise} \end{cases}$$

Where T_M is a threshold, A is a specified angle direction, and $\pm T_A$ defines a "band" of acceptable directions about A .

- Scan the rows of g and fill (set to 1) all gaps (sets of 0s) in each row that do not exceed a specified length, K . Note that, by definition, a gap is bounded at both ends by one or more 1s. The rows are precessed individually, with no memory between them.
- To detect gaps in any other direction, θ , rotate g by this angle and apply the horizontal scanning process procedure in Step 3. Rotate the result back by $-\theta$. Note that rotating is computationally expensive, so this method is best for horizontal or vertical edges.

Regional processing S728 If the location of a boundary is roughly known, it is possible to use the detected edge points as indicators for the suspected boundary.

- Let P be a sequence of ordered, distinct, 1-valued points of a binary image. Specify two starting points, A and B . these are two starting vertices of the polygon.
- Specify a threshold T , and two empty stacks, OPEN and CLOSED.
- If the points in P correspond to a closed curve, put A into OPEN and put B into OPEN and CLOSED.
- Compute the parameters of the line passing from the last vertex in CLOSED to the last vertex in OPEN.
- Compute the distances from the line in Step 4 to all the points in P whose sequence places them between the vertices from Step 4. Select the point, V_{max} , with the maximum distance D_{max} (ties are resolved arbitrarily)
- If D_{max} , place V_{max} at the end of the OPEN stack as a new vertex. Go to Step 4.
- Else, remove the last vertex from Open and insert it as the last vertex of CLOSED.
- If OPEN is not empty, go to Step 4.
- Else, exit. The vertices in CLOSED are the vertices of the polygonal fit to the points in P .

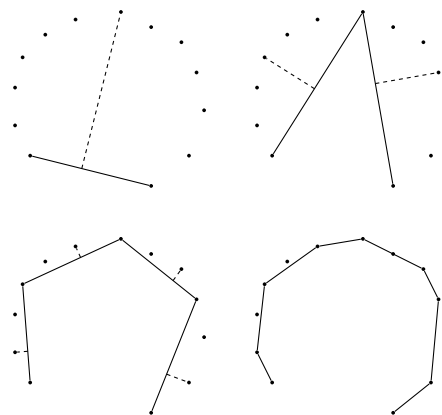
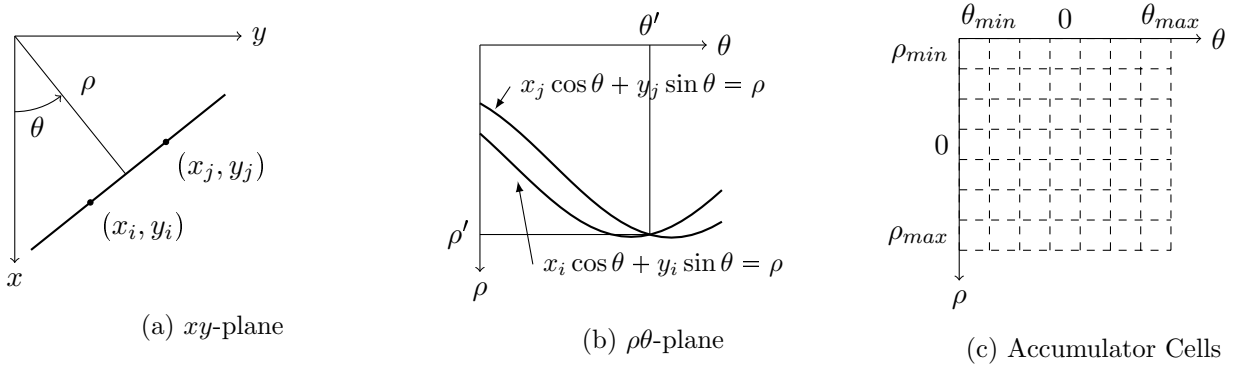


Figure 14: Regional Processing

Figure 15: Hough-Transform: From xy -plane to $\rho\theta$ -plane**Global processing using the Hough transform S733**

- Each point in the xy plane belongs to infinite many lines $y_i = ax_i + b$
- If we interpret this line equation differently, then $b = -x_i a + y_i$ represent for each point in the xy plane a line in the ab plane
- For each point in the xy plane, a line can be drawn in the ab plane and the principal lines in the xy plane can be found by the points with the most intersections in the ab plane
- Vertical lines require an infinite a , this does not work. Therefore the line representation $x \cos \theta + y \sin \theta = \rho$ is used. A point in the xy plane does not result in a simple straight line in the parameter space ($\theta\rho$). The curve $\rho = f(\theta)$ is now a sinusoidal
- Practical scheme
 - For every θ_q in the $\rho\theta$ plane
 - * Calculate $\rho = f(\theta)$ and quantize it into ρ_q
 - * Increase the cell in the $\rho\theta$ plane at location (ρ_q, θ_q) by one
 - The dominant line is now found by selecting the point in the $\rho\theta$ plane with the highest value.

This theory can be generalized to find e.g. circles etc.

10.3 Thresholding S738**10.3.1 Foundation S738**

To be able to partition an image into regions based on intensity values, the histograms have to be able to be separated.

$$\text{single thresholding} \quad g(x, y) = \begin{cases} 1 & \text{if } f(x, y) > T \\ 0 & \text{if } f(x, y) \leq T \end{cases} \quad (2)$$

$$\text{multiple thresholding} \quad g(x, y) = \begin{cases} a & \text{if } f(x, y) > T_2 \\ b & \text{if } T_1 < f(x, y) \leq T_2 \\ c & \text{if } f(x, y) \leq T_1 \end{cases} \quad (3)$$

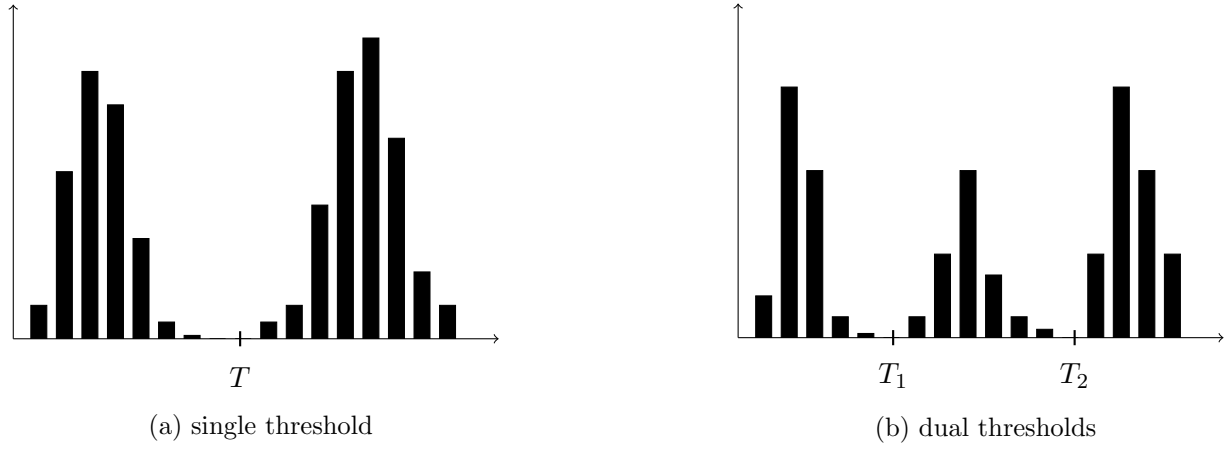


Figure 16: Intensity histograms that can be partitioned

10.3.2 Basic global thresholding S741

1. Select an initial estimate for the global threshold, T
2. Segment the image using T in equation (2). This will produce two groups of pixels: G_1 consisting of all pixels with intensity values $> T$, and G_2 consisting of pixels with values $\leq T$
3. Compute the average (mean) intensity values m_1 and m_2 for the pixels in G_1 and G_2 , respectively
4. Compute a new threshold value:

$$T = \frac{1}{2}(m_1 + m_2)$$

5. Repeat Steps 2 through 4 until the difference between values of T in successive iterations is smaller than a predefined parameter ΔT

10.3.3 Optimum global thresholding using Otsu's Method S742

Two different variances are important

- Within-class variance $\sigma_W^2 = P_1\sigma_1^2 + P_2\sigma_2^2$
- Between-class variance $\sigma_B^2 = P_1(m_1 - m_G)^2 + P_2(m_2 - m_G)^2$

The target is to maximize the ratio between the between-class variance and the within-class variance. This is the same as maximizing the between-class variance.

$$\sum_{i=0}^{L-1} p_i = 1 \quad \text{normalized histogram}$$

$$m_G = \sum_{i=0}^{L-1} i p_i \quad m_G : \text{global mean} \quad (4)$$

$$\sigma_G^2 = \sum_{i=0}^{L-1} (i - m_G)^2 p_i \quad \sigma_G^2 : \text{global variance}$$

$$P_1(k) = \sum_{i=0}^k p_i \quad P_1 : \text{probability of the background} \quad (5)$$

$$P_2(k) = \sum_{i=k+1}^{L-1} p_i = 1 - P_1(k) \quad P_2 : \text{probability of the object}$$

$$m_1(k) = \frac{1}{P_1(k)} \sum_{i=0}^k i p_i \quad m_1 : \text{mean intensity of the background}$$

$$m_2(k) = \frac{1}{P_2(k)} \sum_{i=k+1}^{L-1} i p_i \quad m_2 : \text{mean intensity of the object}$$

$$m(k) = \sum_{i=0}^k i p_i \quad m : \text{cummulative mean up to level } k \quad (6)$$

$$\begin{aligned} \sigma_B^2 &= P_1(m_1 - m_G)^2 + P_2(m_2 - m_G)^2 \\ &= P_1 P_2 (m_1 - m_2)^2 \end{aligned} \quad \begin{aligned} \sigma_B^2 &: \text{between-class variance} \\ \text{Using algebra to change the form} \end{aligned}$$

$$\sigma_B^2(k) = \frac{(m_G P_1(k) - m(k))^2}{P_1(k)(1 - P_1(k))} \quad \sigma_B^2 \text{ in dependence of } k \quad (7)$$

$$\sigma_B^2(k^*) = \max_{0 \leq k \leq L-1} \sigma_B^2(k) \quad \text{Optimum of } k \text{ for maximizing is } k^*$$

$$\eta(k) = \frac{\sigma_B^2(k)}{\sigma_G^2} \quad 0 \leq \eta(k^*) \leq 1 \quad (8)$$

What to do now?

1. Compute the normalized histogram of the input image. Denote the components of the histogram by p_i , $i = 0, 1, 2, \dots, L - 1$
2. Compute the cumulative sums, $P_1(k)$, for $k = 0, 1, 2, \dots, L - 1$, using Eq. (5) (10.3-4)
3. Compute the cumulative means, $m(k)$, for $k = 0, 1, 2, \dots, L - 1$, using Eq. (6) (10.3-8)
4. Compute the global intensity mean, m_G , using Eq. (4) (10.3-9)
5. Compute the between-class variance, $\sigma_B^2(k)$, for $k = 0, 1, 2, \dots, L - 1$, using Eq. (7) (10.3-17)
6. Obtain the Otsu threshold, k^* , as the value k for which $\sigma_B^2(k)$ is maximum. If the maximum is not unique, obtain k^* by averaging the values of k corresponding to the various maxima detected.
7. Obtain the separability measure, η^* , by evaluating Eq. (8) (10.3-16) at $k = k^*$.

10.3.4 Using Image Smoothing to Improve Global Thresholding S747

The obvious approach to improve the situation is to low pass filter the image, i.e., to smooth the image before the global thresholding method is applied

10.3.5 Using Edges to Improve Global Thresholding S749

Useful when trying to find small objects. Only the strongest edges are usually kept, say the top 10%, to calculate the threshold.

1. Compute an edge image using any method (e.g. magnitude of gradient, absolute value of laplacian)
2. Threshold the edge image to produce a binary image $g_T(x, y)$ use this as a *mask image* to select pixels from the original image
3. Compute the histogram of the selected pixels
4. Use this histogram to calculate the global segmentation threshold using, for example, Otsu's method

10.3.6 Multiple Thresholds S752

$$\sigma_B^2 = P_1(m_1 - m_G)^2 + P_2(m_2 - m_G)^2 + P_3(m_3 - m_G)^2$$

$$P_1(k) = \sum_{i=0}^{k_1} p_i$$

$$P_2(k) = \sum_{i=k_1+1}^{k_2} p_i$$

$$P_3(k) = \sum_{i=k_2+1}^{L-1} p_i$$

$$m_G = P_1 m_1 + P_2 m_2 + P_3 m_3$$

$$\sigma_B^2(k_1^*, k_2^*) = \max_{0 \leq k_1 \leq k_2 \leq L-1} \sigma_B^2(k_1, k_2)$$

$$m_1(k) = \frac{1}{P_1(k)} \sum_{i=0}^{k_1} i p_i$$

$$m_2(k) = \frac{1}{P_2(k)} \sum_{i=k_1+1}^{k_2} i p_i$$

$$m_3(k) = \frac{1}{P_3(k)} \sum_{i=k_2+1}^{L-1} i p_i$$

$$P_1 + P_2 + P_3 = 1$$

$$\eta(k_1^*, k_2^*) = \frac{\sigma_B^2(k_1^*, k_2^*)}{\sigma_G^2}$$

10.3.7 Variable Thresholding S756

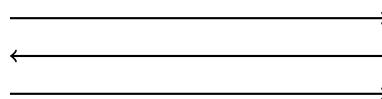
Image Partitioning S756 The simplest approach is to partition a given image into non overlapping regions and then use Otsu's method on smaller regions. This works well if the illumination is changing slowly.

Based on local image properties S758 A threshold is computed for each point. The threshold depends on some property of a local neighborhood. Common properties are the mean and standard deviation for each pixel. An even more powerful scheme can be built, if the 1 or 0 decision is based on some logical condition.

Using moving averages S759 Useful to reduce illumination effects. Very fast to calculate, since only subtracting the oldest sample z_{k-n} and adding the new sample z_{k+1} to the current average are needed.

$$\begin{aligned} m(k+1) &= \frac{1}{n} \sum_{i=k+2-n}^{k+1} z_i \\ &= m(k) + \frac{1}{n} (z_{k+1} - z_{k-n}) \end{aligned}$$

This is achieved by moving along zig-zag scan lines:



10.3.8 Multivariable Thresholding S761

10.4 Region based segmentation S763

The goal is to segment the image into regions directly, not using abrupt changes or thresholding.

10.4.1 Region growing S763

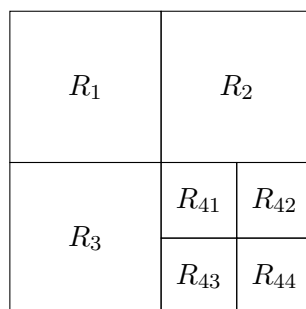
1. Find all connected components in $S(x, y)$ and erode each connected component to one pixel; label all such pixels found as 1. All other pixels in S are labeled 0.
2. Form an image f_Q such that, at a pair of coordinates (x, y) , let $f_Q(x, y) = 1$ if the input image satisfies the given predicate, Q , at those coordinates; otherwise, let $f_Q(x, y) = 0$
3. Let g be an image formed by appending to each seed point in S all the 1-valued points in f_Q that are 8-connected to that seed point
4. Label each connected component in g with a different region label (e.g. 1,2,3,...). This is the segmented image obtained by Region growing.

10.4.2 Region splitting and merging S766

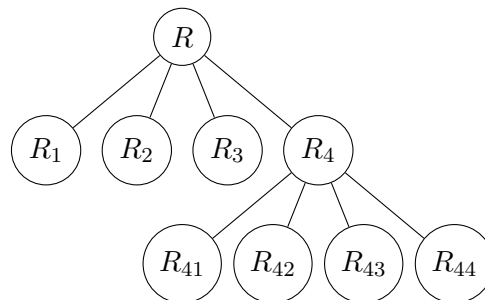
The image is subdivided initially into a set of arbitrary disjoint regions and then the regions are merge and or split in an attempt to improve the segmentation.

A common scheme to split images is a quadtree.

1. Split into four disjoint quadrants any region R_i for which $Q(R) = FALSE$.
2. When no further splitting is possible, merge any adjacent regions R_i and R_k for which $Q(R_j \cup R_k) = TRUE$.
3. Stop when no further merging is possible



(a) Partioned image



(b) Corresponding quadtree. R represents the entire image region.

Figure 17: Quadtree example

10.5 Segmentation using morphological watersheds S769

10.5.1 Background S769

The basic idea is to think of images as 3D landscapes. There are three kinds of points:

- Points belonging to a regional minimum.
- Points, where water would flow to a single minimum. These are called catchment basin.

- Points, where water would be equally likely to fall to more than one such minimum. These are called watershed lines.

Basic idea:

- In each basin a hole is punched
- The topological image is pushed into a full bathtub. The water rises uniformly in the different basins
- As soon as the water in different basins want to merge, a dam is build
- At the end only the dams are left. These are the watershed lines which are connected boundaries segmenting the image into different regions.

10.5.2 Dam Construction S772

- The easiest way to construct such a dam is using dilation
- When basins merge, then a dam needs to be constructed

10.5.3 Watershed segmentation algorithm S774

- $T[n]$ is the set of coordinates flooded $T[n] = \{(s, t) | g(s, t) < n\}$
- The flooding goes in integers from $n = \min + 1$ to $n = \max + 1$
- The algorithm starts with $C[\min + 1] = T[\min + 1]$. This leads to $C_{\min + 1}(M_i)$ for every minimum.

$$C[n] = \bigcup_{i=1}^R C_n(M_i)$$
- For every n until $n = \max + 1$, $C[n]$ is created from $C[n - 1]$.
 Let $Q[n]$ be the set of connected components in $T[n]$. For each connected component q , there are three possibilities:
 1. $q \cap C[n - 1]$ is empty.
 i.e. a new basin was detected (was not there in $C[n - 1]$) $\Rightarrow$ simply incorporate into $C[n]$.
 2. $q \cap C[n - 1]$ contains one connected component of $C[n - 1]$.
 i.e. a basin was extended $\Rightarrow$ simply incorporate into $C[n]$.
 3. $q \cap C[n - 1]$ contains more than one connected component of $C[n - 1]$.
 i.e. basins want to merge $\Rightarrow$ build a dam (or dams) by dilating $q \cap C[n - 1]$ with a 3×3 structuring element of 1s and constraining the dilation to q .

10.6 The use of motion in segmentation S778

If an object moves in a scene, it can easily be segmented from the background using its motion information.

10.6.1 Spatial Techniques S778

For a sequence of images $f(x, y, t_k)$, a reference image $R(x, y)$ is needed. Usually the first image is chosen: $R(x, y) = f(x, y, 1)$. One can also use a IIR lowpass filtered image of all frames: $R_k(x, y) = (1 - \alpha)R_{k-1}(x, y) + \alpha f(x, y, k)$. (using a small α)

The *accumulative difference image* (ADI) is generated by comparing this reference image with every subsequent image in the scene.

$$A_k(x, y) = \begin{cases} A_{k-1}(x, y) + 1 & \text{if } |R(x, y) - f(x, y, k)| > T \\ A_{k-1}(x, y) & \text{otherwise} \end{cases}$$

$$P_k(x, y) = \begin{cases} P_{k-1}(x, y) + 1 & \text{if } (R(x, y) - f(x, y, k)) > T \\ P_{k-1}(x, y) & \text{otherwise} \end{cases}$$

$$N_k(x, y) = \begin{cases} N_{k-1}(x, y) + 1 & \text{if } (R(x, y) - f(x, y, k)) < -T \\ N_{k-1}(x, y) & \text{otherwise} \end{cases}$$

where $A_k(x, y)$, $P_k(x, y)$, $N_k(x, y)$ are the absolute, positive and negative ADIs, after the k th image in the sequence is encountered and T is threshold, e.g. 5% of the brightness.

10.6.2 Frequency Domain Techniques S782

11 Representation and Description Ch. 11 S796-860

11.1 Representation S796

After the image segmentation, the boundaries need to be represented in a form, that is suitable for further processing.

11.1.1 Boundary following S796

The following algorithm is the moore boundary tracking algorithm.

1. Let the starting point, b_0 be the uppermost, leftmost point in the image that is labeled 1. Denote by c_0 the west neighbor of b_0 clearly, c_0 is a background point. Examine the 8-neighbors of b_0 , starting at c_0 and proceeding in a clockwise direction. Let b_1 denote the first neighbor encountered whose value is 1, and let c_1 be the (background) point immediately preceding b_1 in the sequence. Store the localtions of b_0 and b_1 for use in Step 5.
2. Let $b = b_1$ and $c = c_1$
3. Let the 8-neighbors of b , starting at c and proceeding in a clockwise direction, be denoted by $n_1, n_2, \dots, n_8$. Find the first n_k labeled 1.
4. Let $b = n_k$ and $c = n_{k-1}$
5. Repeat Steps 3 and 4 until $b = b_0$ and the next boundary point found is b_1 . The sequence of b points found when the algorithm stops constitutes the set of ordered boundary points

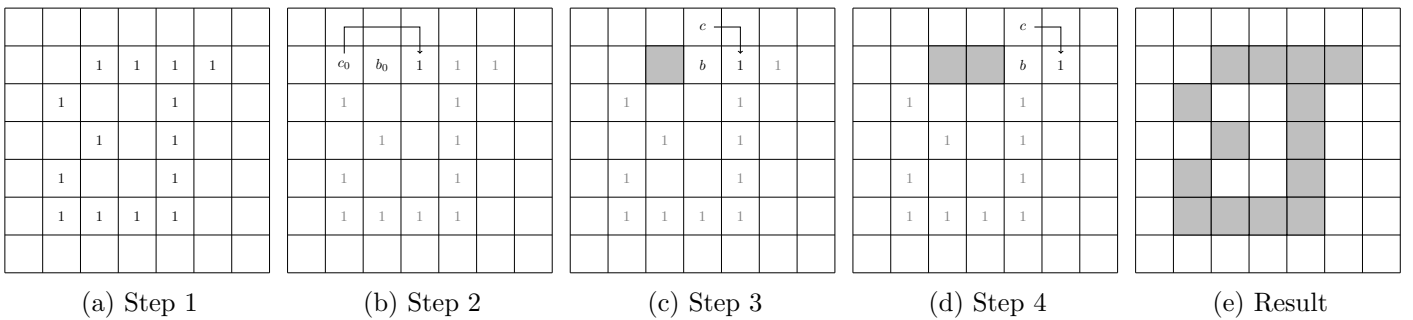
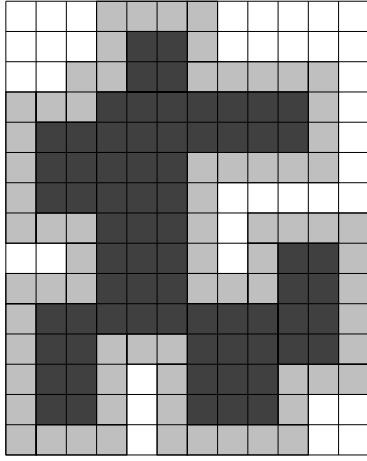
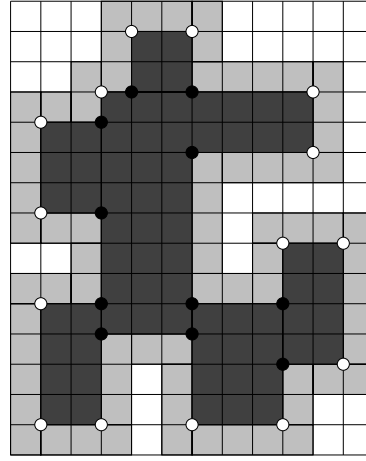


Figure 18: Boundary Following Algorithm

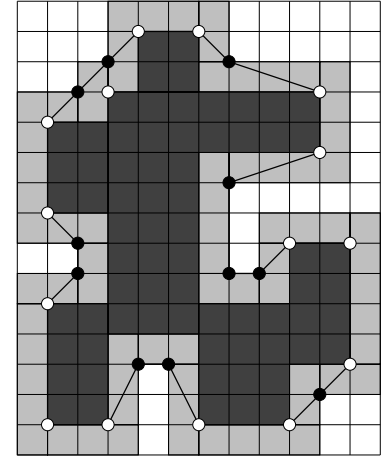
Of course this algorithm also works counter-clock-wise.



(a) Dark gray region resulting from enclosing the original boundary



(b) Convex (white) and concave (black) vertices



(c) Concave vertices displaced to their diagonal mirror locations

Figure 21: MPP obtained by boundary shrinking

3. The orientation (cw or ccw) of a sequence of three points will be necessary. This replaces the convex concave classification

a) $a = (x_1, y_1), b = (x_2, y_2), c = (x_3, y_3)$

b) $A = \begin{bmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{bmatrix}$

c) $\det(A) =$

- > 0 if (a, b, c) is a counterclockwise sequence
- $= 0$ if the points are collinear
- < 0 if (a, b, c) is a clockwise sequence

4. For notational convenience let $\text{sgn}(a, b, c) = \det(A)$

5. Names

- V_L are found vertices of the MPP
- V_K is the next possible candidate
- B_C is a black vertex
- W_C is a white vertex

6. V_K lies to the positive side of the line through pair (V_L, W_C) , that is $\text{sgn}(V_L, W_C, V_K) > 0$. If this condition holds, the next MPP vertex is W_C

7. V_K lies to the negative side of the line through pair (V_L, B_C) , that is $\text{sgn}(V_L, B_C, V_K) < 0$. If this condition holds, the next MPP vertex is B_C .

8. V_K lies on the negative side of the line through pair (V_L, W_C) or is collinear with it, that is $\text{sgn}(V_L, W_C, V_K) \leq 0$. At the same time, V_K lies on the positive side of the line through (V_L, B_C) or is collinear with it, that is $\text{sgn}(V_L, B_C, V_K) \geq 0$. If this condition holds, the next candidate MPP vertex is V_K ; otherwise $B_C = V_K$

9. Continue with the next vertex in the list.

11.1.4 Other Polygonal Approximation Approaches S807

Merging S807 Points on the boundary are merged and a line is fitted to these points. If the fitting error becomes too large, a vertex is set and the procedure starts from the beginning. This approach is very simple, but there is no guarantee for a good approximation. E.g. it does not set vertices at corners, but always a bit later.

Splitting S808 Here the idea is, to divide (split) a boundary segment until a threshold of a criterion is satisfied. E.g. until the distances from the boundary to the approximated lines are below a certain number.

11.1.5 Signatures S808

A signature is a 1D function that represents a boundary. This results in a significant complexity reduction. There are several schemes:

- Find the centroid. The boundary is described as a function of the angle around this point.
 - Rotational invariance: this can be achieved by selecting the point on the eigen axis that is farthest from the centroid.
 - Scaling invariance: this can be achieved by dividing the signature by its standard deviation.

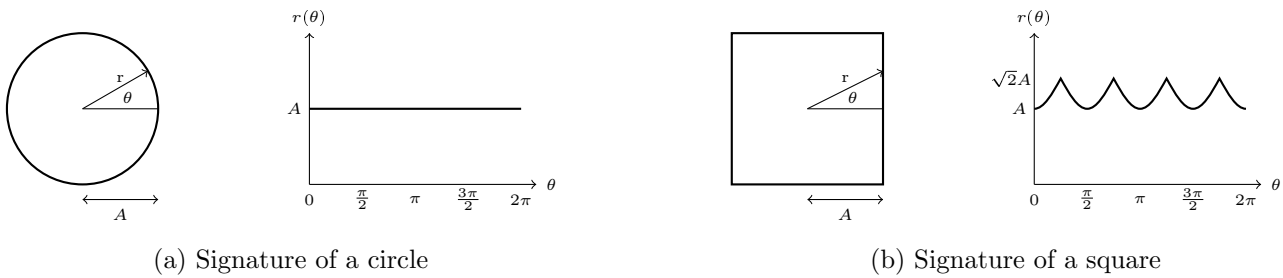


Figure 22: Examples for signatures

11.1.6 Boundary segments S810

The convex hull H of an arbitrary set S is the smallest convex set containing S . The difference $D = H - S$ is the convex deficiency. Follow the contour of S and mark the spots where the contour makes a transition into or out of D . These marking α (0...1) are relative to the total length of the contour D .

This is in theory invariant to:

- Rotation
- Translation
- Scaling

In practice, boundaries have some noise. Often smoothing along the boundaries is used.

11.1.7 Skeletons S812

Medial axis transform (MAT)

- Every point in the region finds its closest boundary point.

- If a point in the region has two closest boundary points, it belongs to the skeleton of the region.
- Direct implementation of this is computationally expensive.

Algorithms for efficient MAT iteratively delete boundary points subject to

1. keep end points
2. keep connectivity (note that morphological thinning does *not* account for this)
3. do not erode that region excessively

Thinning algorithm is 2 steps applied successively to border points

p9	p2	p3
p8	p1	p4
p7	p6	p5

Step 1: south or east boundary points

Flag for deletion if all of these conditions are satisfied:

- a) $2 \leq N(P_1) \leq 6$ with $N(p_1) = p_2 + p_3 + \dots + p_9$
- b) $T(p_1) = 1$ where $T(p_1)$ is the number of 0 – 1 transitions in the ordered sequence $p_2, p_3, \dots, p_8, p_9, p_2$
- c) $p_2 \cdot p_4 \cdot p_6 = 0$
- d) $p_4 \cdot p_6 \cdot p_8 = 0$

The last conditions are satisfied simultaneously by the minimum set of values: ($p_4 = 0$ or $p_6 = 0$) or ($p_2 = 0$ and $p_8 = 0$)

After processing all boundary points, the marked ones are deleted

Step 2: north or west boundary points

Simultaneous to Step 1 except the new conditions c' and d'

- a) $2 \leq N(P_1) \leq 6$ with $N(p_1) = p_2 + p_3 + \dots + p_9$
- b) $T(p_1) = 1$ where $T(p_1)$ is the number of 0 – 1 transitions in the ordered sequence $p_2, p_3, \dots, p_8, p_9, p_2$
- c') $p_2 \cdot p_4 \cdot p_8 = 0$
- d') $p_2 \cdot p_6 \cdot p_8 = 0$

11.2 Boundary Descriptors S815

Having seen different representations of boundaries, the goal is now to develop descriptors for these boundaries.

11.2.1 Simple Descriptors S815

Length

- Number of pixels along a Boundary gives a rough approximation of its length
- 8-directional Chain Code numbers of vertical and horizontal components plus $\sqrt{2}$ times the number of diagonal components, gives the exact length.

Diameter of boundary B

$$\text{Diam}(B) = \max_{i,j} [D(p_i, p_j)]$$

Points on boundary:

$$p_i, p_j$$

Distance measure:

$$D$$

Major axis The line connecting the two extreme points of the boundary is called the major axis of the boundary.

Minor axis The line perpendicular to the major axis and of such length that a box passing through the outer four points of intersection of the boundary with the two axes completely encloses the boundary.

Eccentricity The ratio of the major to the minor axis is called the eccentricity of the boundary. This is a useful descriptor.

Curvature This is the rate of change of the slope in the boundary. i.e. this is a second derivative. Note that the curvature is often very noisy. It is therefore often applied to a polygon approximation.

11.2.2 Shape numbers S816

The shape number of a boundary (based on 4-directional code) is defined as the first difference of smallest magnitude.

The order n of a shape number is defined as the number of digits in its representation.

One way of aligning the chain code grid with the boundary is to use the basic rectangle box based on the major and minor axes

11.2.3 Fourier descriptors S818

The coordinates of the boundary represents a complex sampled signal which can be represented as a complex vector

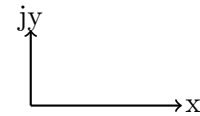
$$s(k) = x(k) + jy(k) \quad k = 0, 1, 2, \dots, K-1$$

which can be analyzed in the frequency domain.

Note that the axes are changed:



(a) Normal



(b) for Fourier descriptors

Figure 23: Axes of image

The DFT of $s(k)$ is

$$\text{Fourier descriptors of the boundary:} \quad a(u) = \sum_{k=0}^{K-1} s(k) e^{-j2\pi uk/K} \quad u = 0, 1, 2, \dots, K-1$$

$$\text{Inverse Fourier transform:} \quad s(k) = \frac{1}{K} \sum_{u=0}^{K-1} a(u) e^{j2\pi uk/K}$$

Since the high frequency components are responsible for the fine detail, using only the first few Fourier descriptors results in a smoothing of the boundary and image compression.

The reconstruction of the image is, when using the first P coefficients

$$\hat{s}(k) = \frac{1}{K} \sum_{u=0}^{P-1} a(u) e^{j2\pi uk/K}$$

Basic properties of Fourier descriptors

Transformation	Boundary	Fourier Descriptor
Identity	$s(k)$	$a(u)$
Rotation	$s_r(k) = s(k)e^{j\theta}$	$a_r(u) = a(u)e^{j\theta}$
Translation	$s_t(k) = s(k) + \Delta_{xy}$	$a_t(u) = a(u) + \Delta_{xy}\delta(u)$
Scaling	$s_s(k) = \alpha s(k)$	$a_s(u) = \alpha a(u)$
Starting point	$s_p(k) = s(k - k_0)$	$a_p(u) = a(u)e^{-j2\pi k_0 u/K}$

11.2.4 Statistical moments S821

The boundary segment is rotated, so that it can be interpreted as a function of a single variable.

The amplitude of $g()$ can be treated as a discrete random variable v and a amplitude histogram $p(v_i)$ can be formed. From the histogram, the statistical moments can be calculated.

$$m = \sum_{i=0}^{A-1} v_i p(v_i)$$

$$\mu_n(v) = \sum_{i=0}^{A-1} (v_i - m)^n p(v_i)$$

The statistical moments are:

- m or μ_1 : Mean
- μ_2 : Variance
- μ_3 : Skewness (this is a measure of asymmetry)
- μ_4 : Kurtosis (peakedness)

If $g()$ is normalized such that it represents a probability density function (pdf), i.e. it sums up to 1, the histogram building can be skipped and $g()$ can directly be analyzed by statistical moments.

11.3 Regional descriptors S822

11.3.1 Simple descriptors S822

- Area: Number of pixels
- Perimeter: Length of boundary
- Compactness: $\text{Perimeter}^2 / \text{area}$
- Circularity ratio $R_C = \frac{4\pi A}{P^2}$ where A is the area of the region in question and P the length of its perimeter
- Mean, median, minimum, maximum of intensity
- Etc.

11.3.2 Topological descriptors S823

Topological descriptors are descriptors that do not change under rubber-sheet distortions. No tearing, no folding allowed.

The number of holes H and connected components C in a figure can be used to define the Euler number $E = C - H$

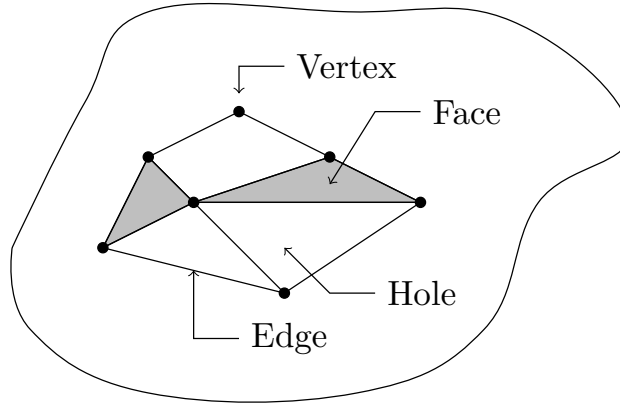


Figure 24: A region containing a polygonal network
In this example:

$$\begin{array}{llll} V = 7 & Q = 11 & F = 2 & \Rightarrow E = -2 \\ & C = 1 & H = 3 & \Rightarrow E = -2 \end{array}$$

- Number of vertices V
- Number of edges Q
- Number of faces F

Euler proved that $E = V - Q + F = C - H$

11.3.3 Texture S827

Smoothness, coarseness, regularity etc.

There are three basic approaches to describe texture:

Statistical approaches S828

- Intensity histogram $p(z_i)$
- Moment $\mu_n(z) = \sum_{i=0}^{L-1} (z_i - m)^n p(z_i)$ and mean $m = \sum_{i=0}^{L-1} z_i p(z_i)$
- The variance $\sigma^2(z) = \mu_2(z)$ is a measure of contrast, which can be used to establish descriptors of smoothness. It should be normalized to $0 \dots 1$ by dividing by $\left(\frac{L-1}{2}\right)^2$. $R(z) = 1 - \frac{1}{1+\sigma^2(z)}$
- A measure for uniformity $U(z) = \sum_{i=0}^{L-1} p^2(z_i)$

- Average entropy $e(z) = - \sum_{i=0}^{L-1} p(z_i) \log_2 p(z_i)$

Note that e is zero for constant regions.

Relative positions Measures with only histograms have no information about the relative positions of pixels with respect to each other. Thus, a Q operator, that defines the positions of two pixels relative to each other, is introduced. A *co-occurrence matrix* $\mathbf{G}$ is generated, whose elements $g_{i,j}$ are how many times a certain pixel pair (z_i, z_j) appears.

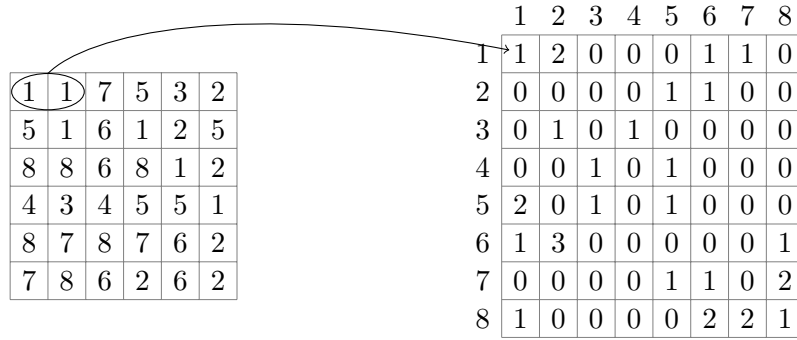


Figure 25: Co-occurrence matrix $\mathbf{G}$

n	sum of elements in $\mathbf{G}$
K	size of co-occurrence matrix $\mathbf{G}$ (size: $K \times K$)
g_{ij}	element of $\mathbf{G}$
$p_{ij} = g_{i,j}/n$	estimate for probability of a pair (z_i, z_j)
$m_r = \sum_{i=1}^K i \sum_{j=1}^K p_{ij}$	mean, computed along rows of $\mathbf{G}$
$m_c = \sum_{j=1}^K j \sum_{i=1}^K p_{ij}$	mean, computed along columns of $\mathbf{G}$
$\sigma_r^2 = \sum_{i=1}^K (i - m_r)^2 \sum_{j=1}^K p_{ij}$	variance, computed along rows of $\mathbf{G}$
$\sigma_c^2 = \sum_{j=1}^K (j - m_c)^2 \sum_{i=1}^K p_{ij}$	variance, computed along columns of $\mathbf{G}$

First, all probabilities p_{ij} and the means m_r, m_c and variances σ_r^2, σ_c^2 are calculated. Then, the following descriptors can be calculated, if needed. Note that the correlation and the entropy are the most common descriptors.

Structural approaches S836 Some texture can be described as a repetition of a simple primitive. For such textures, a description based on rewriting rules is useful.

Spectral approaches S837 Good for periodic patterns

The energy spectrum $S(r, \theta)$ is analyzed in polar coordinates. It is analyzed for a fixed angle, $S_\theta(r)$ and/or a fixed radius $S_r(\theta)$. The advantage of this is, that these are 1D functions and therefore easier to deal with.

$$S(r) = \sum_{\theta=0}^{\pi} S_\theta(r)$$

$$S(\theta) = \sum_{r=1}^{R_0} S_r(\theta)$$

Descriptor	Explanation	Formula
Maximum probability	Measures the strongest response of $\mathbf{G}$. The range of values is $[0, 1]$.	$\max_{i,j}(p_{ij})$
Correlation	A measure of how correlated a pixel is to its neighbor over the entire image. Range of values is 1 to -1 , corresponding to perfect positive or negative correlations.	$\sum_{i=1}^K \sum_{j=1}^K \frac{(i - m_r)(j - m_c)p_{ij}}{\sigma_r \sigma_c}$
Contrast	A measure of intensity contrast between a pixel and its neighbor over the entire image. The range of values is 0 (when $\mathbf{G}$ is constant) to $(K - 1)^2$.	$\sum_{i=1}^K \sum_{j=1}^K (i - j)^2 p_{ij}$
Uniformity (Energy)	A measure of uniformity in the range $[0, 1]$. Uniformity is 1 for a constant image.	$\sum_{i=1}^K \sum_{j=1}^K p_{ij}^2$
Homogeneity	Measures the spatial closeness of the distribution of elements in $\mathbf{G}$ to the diagonal. The range of values is $[0, 1]$, with the maximum being achieved when $\mathbf{G}$ is a diagonal matrix.	$\sum_{i=1}^K \sum_{j=1}^K \frac{p_{ij}}{1 + i - j }$
Entropy	Measures the randomness of the elements of $\mathbf{G}$. The entropy is 0 when all p_{ij} 's are 0 and is maximum when all p_{ij} 's are equal. The maximum value is $2 \log_2 K$.	$-\sum_{i=1}^K \sum_{j=1}^K p_{ij} \log_2 p_{ij}$

11.3.4 Moment Invariants S839

A set of seven moments for a given image region can be generated as regional descriptors. They are intrinsically invariant to translation, scale change, rotation and even mirroring except for a sign change.

$$\text{2D moments of order } (p+q) \quad m_{pq} = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} x^p y^q f(x, y)$$

$$\text{The central moments} \quad \mu_{pq} = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} (x - \bar{x})^p (y - \bar{y})^q f(x, y) \quad \text{The seven invariant moments:}$$

$$\bar{x} = \frac{m_{10}}{m_{00}} \text{ and } \bar{y} = \frac{m_{01}}{m_{00}}$$

$$\text{Normalized central moments} \quad \eta_{pq} = \frac{\mu_{pq}}{\mu_{00}} \text{ with } \gamma = \frac{p+q}{2} + 1$$

$$\phi_1 = \eta_{20} + \eta_{02}$$

$$\phi_2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2$$

$$\phi_3 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2$$

$$\phi_4 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2$$

$$\phi_5 = (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] + (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]$$

$$\phi_6 = (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03})$$

$$\phi_7 = (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] + (3\eta_{12} - \eta_{30})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]$$

11.4 Use of Principal Components for Description S842

Principle component analysis (PCA) is a very general tool. It is applicable to boundaries and regions but also to color images and multispectral images. A given pixel in a multispectral image with n spectral components can be written as a vector containing these components. It is assumed that the components are random variables and

hence these pixels are random vectors.

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

$$\mathbf{m}_x = E\{\mathbf{x}\} = \frac{1}{K} \sum_{k=1}^K \mathbf{x}_k$$

$$\mathbf{C}_x = E\{(\mathbf{x} - \mathbf{m}_x)(\mathbf{x} - \mathbf{m}_x)^T\} = \frac{1}{K} \sum_{k=1}^K \mathbf{x}_k \mathbf{x}_k^T - \mathbf{m}_x \mathbf{m}_x^T$$

Since the covariance matrix is real and symmetric, there exists a set of n orthonormal eigenvectors $\mathbf{e}_i$ and corresponding eigenvalues λ_i

$$\mathbf{C}\mathbf{e}_i = \lambda_i \mathbf{e}_i$$

The eigenvectors can be arranged into a matrix $\mathbf{A}$:

$$\mathbf{A} = [\mathbf{e}_1^T, \mathbf{e}_2^T, \dots, \mathbf{e}_n^T]^T$$

Now $\mathbf{A}$ is used as a transformation matrix to map the vectors $\mathbf{x}$ into the vectors $\mathbf{y}$. This is called the *Hotelling transform*.

$$\mathbf{y} = \mathbf{A}(\mathbf{x} - \mathbf{m}_x)$$

Clearly, the mean vector of these new $\mathbf{y}$ vectors is zero $\mathbf{m}_y = E\{\mathbf{y}\} = 0$

$$\mathbf{C}_y = \mathbf{A}\mathbf{C}_x\mathbf{A}^T = \begin{bmatrix} \lambda_1 & & & 0 \\ & \lambda_2 & & \\ & & \dots & \\ 0 & & & \lambda_n \end{bmatrix}$$

The elements of the $\mathbf{y}$ vectors are uncorrelated! $\mathbf{C}_x$ and $\mathbf{C}_y$ have the same eigenvalues.

The Hotelling transform can easily be inverted using the fact that because $\mathbf{A}$ is orthonormal, $\mathbf{A}^{-1} = \mathbf{A}^T$:

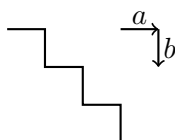
$$\mathbf{x} = \mathbf{A}^T \mathbf{y} + \mathbf{m}_x$$

If however a matrix $\mathbf{A}_k$ from the k eigenvectors with the largest k eigenvalues is formed, the reconstruction is no longer exact, but has a mean square error of $e_{ms} = \sum_{j=k+1}^n \lambda_j$, which decreases monotonically. Thus an image compression is possible by using only k eigenvectors. The reconstructed image is then

$$\hat{\mathbf{x}} = \mathbf{A}_k^T \mathbf{y}_k + \mathbf{m}_x$$

11.5 Relational Descriptors S852

Rewriting Rules Example: Two primitive elements a and b are defined. They can be repetitively used to describe staircases. Rewriting rules are defined as follows: S is the starting symbol, A is a variable. Now a recursive definition can be made for describing the staircase.



- | | | |
|----|-----------|--------------------|
| 1) | start | $S \rightarrow aA$ |
| 2) | repeat 1) | $A \rightarrow bS$ |
| 3) | end | $A \rightarrow b$ |

Example: staircase and the used elements a and b

Now any staircase made up of a s and b can be described by repeating these rules. E.g. $(1, 3)$ is a staircase a, b . $(1, 2, 1, 3)$ is $a, b, a, b, \dots$

Directed line segments The idea is, that section of an image are described by directed line segments

- There needs to be a set of operations acting on the primitives, such that a region or a boundary can be described
- The abstracted primitives need some specific primitives, which contain all directions
- A shape can now be described as a string of operations.



Figure 26: Primitives

Tree descriptors Information describing the meaning of a node and relating it to its neighbors can be stored in trees. A texture may some times not be connected, then it can be described by trees.

Each tree has a unique node called the root $\$$ and is made up of subtrees. All leaves of a tree are called the frontier of the tree.

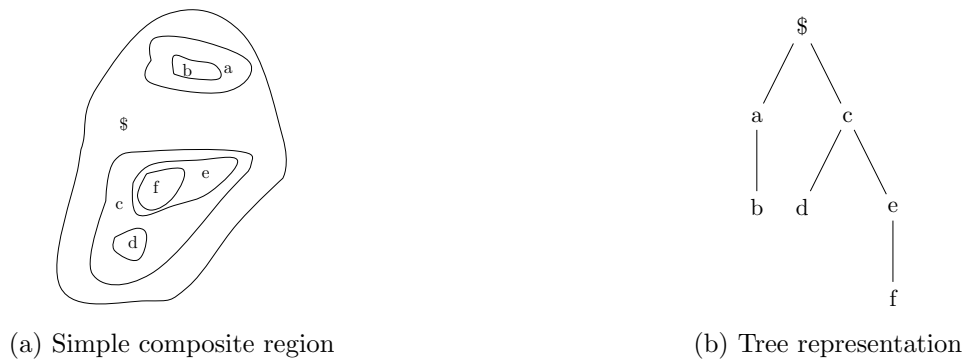


Figure 27: Tree descriptors

12 Object Recognition Ch. 12 S861-907

- Recognition of *individual* image regions called *object* or *pattern*
- Concept of “learning” from sample patterns

12.1 Patterns and Pattern Classes Ch. 12.1

- A *pattern* is an arrangement of descriptors (as in 11) in pattern classification called *features*.
- A *pattern class* is a set of patterns, denoted $\omega_1, \omega_2, \dots, \omega_W$ W is the number of classes
- The goal is to assign a given patter into a class

12.1.1 Vectors

- Used for quantitative descriptors
- every dimension contains the numerical value of a descriptor

$$\bullet \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = (x_1, x_2, \dots, x_n)^T$$

Example: Classifying flowers based on petal length and width into one of three classes.

12.1.2 Strings

Structural information can be captured in strings. Example Boundary Descriptors S815

- Connectivity patterns (order is important)
- Compact description, better than sampling into a feature vector

12.1.3 Tree

If there is hierarchy then a descriptor which is based on a tree structure should be used.

12.2 Recognition Based on Decision Theory Ch. 12.2

Discriminant functions The core of decision theory $d_i(\mathbf{x})$ defined for each class ω_i . These are not known and finding them is the main goal.

A class ω_i is selected if

$$d_i(\mathbf{x}) > d_j(\mathbf{x}) \quad j = 1, 2, \dots, W; j \neq i$$

The interesting parts are decision boundaries between classes

$$d_i(\mathbf{x}) - d_j(\mathbf{x}) = 0$$

For two classes, a single discriminant function can be introduced

$$d_{ij}(\mathbf{x}) = d_i(\mathbf{x}) - d_j(\mathbf{x}) = 0$$

12.2.1 Matching S866

Each class is represented by prototypes. A vector x belongs to the class with the shortest distance.

Minimum distance classifier The mean vector of a class is a powerful prototype vector.

$$\mathbf{m}_j = \frac{1}{N_j} \sum_{\mathbf{x} \in \omega_j} \mathbf{x}_j \quad j = 1, 2, \dots, W$$

A simple matching function is the Euclidean distance ($\|a\| = (a^T a)^{1/2}$).

$$D_j(\mathbf{x}) = \|\mathbf{x} - \mathbf{m}_j\| = ((\mathbf{x} - \mathbf{m}_j)^T (\mathbf{x} - \mathbf{m}_j))^{1/2}$$

The resulting discriminant function which is the largest, when the distance is the smallest.

$$d_j(\mathbf{x}) = \mathbf{x}^T \mathbf{m}_j - \frac{1}{2} \mathbf{m}_j^T \mathbf{m}_j$$

The decision boundary between class ω_i and ω_j will be

$$\begin{aligned} d_{ij}(\mathbf{x}) &= d_i(\mathbf{x}) - d_j(\mathbf{x}) \\ &= \mathbf{x}^T (\mathbf{m}_i - \mathbf{m}_j) - \frac{1}{2} (\mathbf{m}_i - \mathbf{m}_j)^T (\mathbf{m}_i + \mathbf{m}_j) = 0 \end{aligned}$$

Which is a line through the midpoint of m_i and m_j for $n = 2$. For $n = 3$ its a plane and for $n > 3$ it's a hyperplane.

Matching by correlation S869 Calculate cross-correlation between a mask $w(x, y)$ of size $m \times n$ and an image $f(x, y)$ to find the mask in the image.

$$c(x, y) = \sum_s \sum_t w(s, t) f(x + s, y + t)$$

This is sensitive to contrast and average intensity in f and w . Hence the normalized cross correlation is used. Note $\bar{f}_{xy}$ is the average value of f in the region w .

$$\gamma(x, y) = \frac{\sum_s \sum_t [w(s, t) - \bar{w}] [f(x + s, y + t) - \bar{f}_{xy}]}{\left\{ \sum_s \sum_t [w(s, t) - \bar{w}]^2 \sum_s \sum_t [f(x + s, y + t) - \bar{f}_{xy}]^2 \right\}^{\frac{1}{2}}}$$

$\gamma(x, y)$ is always in range $[-1, 1]$, where +1 is perfect match and -1 a perfect anti-match. This is sensitive to scale and rotation. Often the square root is ignored and absolute values are used instead of squares to save computational effort.

12.2.2 Optimum Statistical Classifiers S872

A classifier which results in the lowest average risk of incorrect classification.

The probability that a pattern $\mathbf{x}$ comes from class ω_i is the conditional probability $p(\omega_i|\mathbf{x})$. The loss that occurs if a pattern $\mathbf{x}$ is put to class ω_j instead of ω_i is denoted as L_{ij} .

The conditional average risk or loss is the average loss if $\mathbf{x}$ is assigned to class j :

$$r_j(\mathbf{x}) = \sum_{k=1}^W L_{kj} p(\omega_k|\mathbf{x}) = \frac{1}{p(\mathbf{x})} \sum_{k=1}^W L_{kj} p(\mathbf{x}|\omega_k) P(\omega_k)$$

As $p(\mathbf{x})$ is equal to all r_j and assuming that the loss for every misclassification is equal (e.g. 1) this is simplified to

$$r_j(\mathbf{x}) = p(\mathbf{x}) - p(\mathbf{x}|\omega_j) P(\omega_j)$$

Or written in the same form as before

$$d_j(\mathbf{x}) = p(\mathbf{x}|\omega_j) P(\omega_j) \quad j = 1, 2, \dots, W$$

In order to estimate $p(\mathbf{x}|\omega_j)$ and $P(\omega_j)$, one uses a Gaussian assumption. If N_j is the number of pattern vectors from class ω_j , the mean vector $\mathbf{m}_j$ and the covariance matrix $\mathbf{C}_j$ are calculated as

$$\mathbf{m}_j = \frac{1}{N_j} \sum_{\mathbf{x} \in \omega_j} \mathbf{x} \quad \mathbf{C}_j = \frac{1}{N_j} \sum_{\mathbf{x} \in \omega_j} \mathbf{x} \mathbf{x}^T - \mathbf{m}_j \mathbf{m}_j^T$$

The decision function can then be written as

$$d_j(\mathbf{x}) = \ln P(\omega_j) - \frac{1}{2} \ln |\mathbf{C}_j| - \frac{1}{2} [(\mathbf{x} - \mathbf{m}_j)^T \mathbf{C}_j^{-1} (\mathbf{x} - \mathbf{m}_j)]$$

12.2.3 Neural Networks S882

Class of algorithms which find decision functions directly by training. Neural Networks (NN) make no Gaussian assumption.

Perceptron for two pattern classes The perceptron learns a linear decision function that separates two linearly separable training sets.

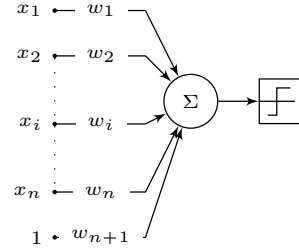


Figure 28: Perceptron model for two pattern classes

$$d(\mathbf{x}) = \sum_{i=1}^n w_i x_i + w_{n+1}$$

If $d(\mathbf{x}) > 0$, then the pattern belongs to ω_1 , if $d(\mathbf{x}) < 0$ it belongs to ω_2 .

The function which maps the weighted sum of the input to the output is called *activation function*. In 28 a sign function is used. For multilayer NN a sigmoid (30) is common.

The constant input of 1 allows an offset. A convenient notation is the augmented pattern vector, which allows to handle the offset as if it was another input

$$\begin{aligned} y_i &= x_i & i &= 1, 2, \dots, n \\ y_{n+1} &= 1 \\ d(y) &= \sum_{i=1}^{n+1} w_i y_i = w^T y \end{aligned}$$

Training algorithms S886 Train the weights of the perceptron so that it can distinguish between classes, using training sets. For linear separable classes, a simple, iterative algorithm is:

- Pattern vectors of the training set are presented repetitively, the weight vector is changed if a misclassification happens.
- The training stops, when there are no more misclassifications.

for $d(y) > 0 \rightarrow \omega_1$ and $d(y) < 0 \rightarrow \omega_2$

$$\begin{aligned} \text{if } y(k) \in \omega_1 \text{ and } w^T(k)y(k) \leq 0 & \rightarrow w(k+1) = w(k) + cy(k) \\ \text{if } y(k) \in \omega_2 \text{ and } w^T(k)y(k) \geq 0 & \rightarrow w(k+1) = w(k) - cy(k) \end{aligned}$$

c is a positive correction increment. This algorithm converges in a finite number of steps for linearly separable training data.

This is not possible for nonseparable classes. Hence minimizing the squared error is the target.

$$J(w) = \frac{1}{2}(r - w^T y)^2 \quad r \text{ is the desired response and } y \text{ the pattern vector}$$

To minimize this we use the delta correction algorithm

$$\begin{aligned} \Delta w &= \alpha e(k) y(k) \\ \Delta w &= w(k+1) - w(k) && \text{weighting change} \\ e(k) &= r(k) - w^T(k) y(k) && \text{error} \end{aligned}$$

The change of the error becomes

$$\begin{aligned} \Delta e &= -\alpha e(k) y^T(k) y(k) \\ &= -\alpha e(k) \|y(k)\|^2 \end{aligned}$$

So α controls the speed of error reduction. For stability α must be smaller than 2. In practice α is picked below 1.

Multilayer feedforward neural networks S889 These allow for multiclass pattern recognition, also for non linear separable classes. They are stacked banks of perceptrons with the sigmoid as activation function (30).

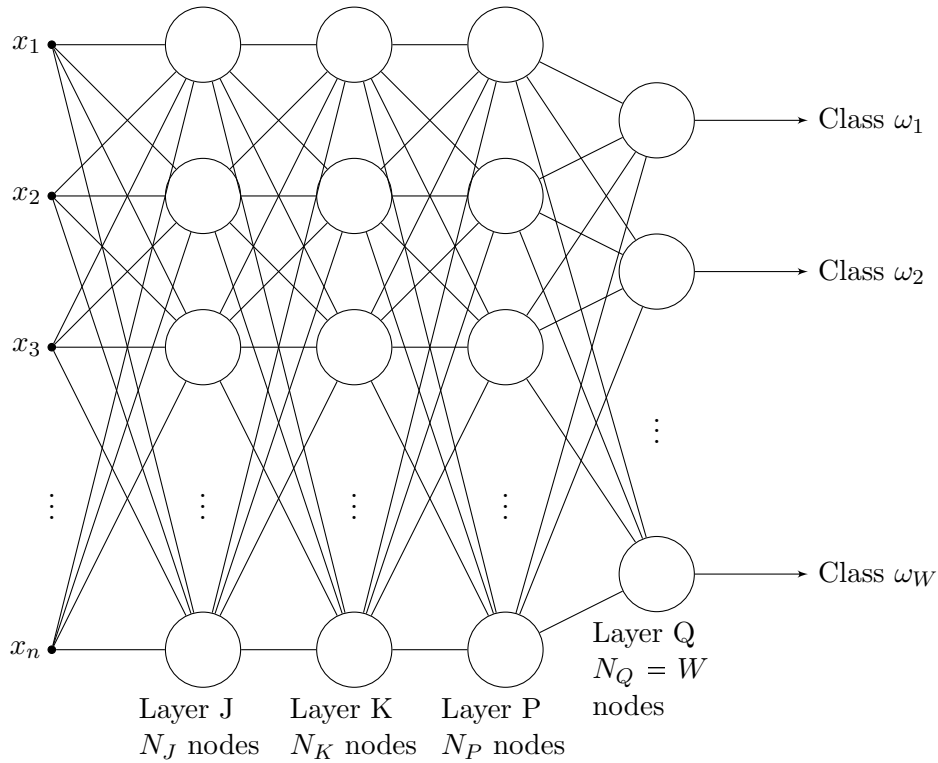


Figure 29: Multilayer feed forward network

If K is the layer preceding layer J then the input to the activation function of node j in layer J is the weighted sum of the outputs of Layer K .

$$I_j = \sum_{k=1}^{N_k} w_{jk} O_k$$

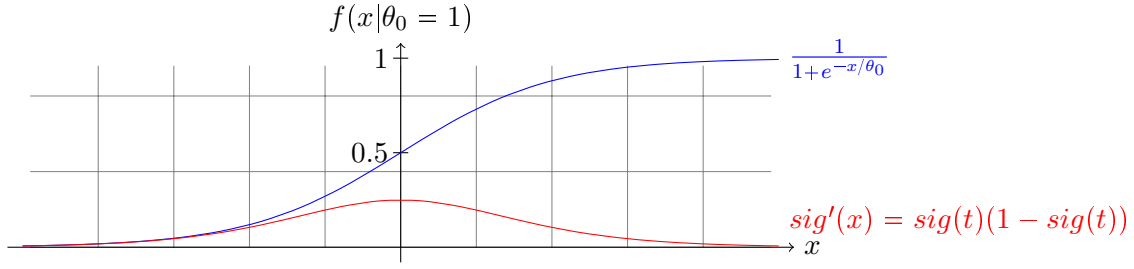


Figure 30: Sigmoid function

Training by back propagation Target is to minimize the total squared error between the desired responses r_q and the actual response O_q in the output layer Q

$$E_Q = \frac{1}{2} \sum_{q=1}^{N_Q} (r_q - O_q)^2$$

Back-Propagation is deriving the Error with respect to the weights

$$\Delta w_{qp} = -\alpha \frac{\partial E_Q}{\partial w_{qp}}$$

Δw is the change in weight vector, α the correction increment, E_Q the error and w_{qp} the current weight vector

- A training vector is presented at the input and the desired response at the output
- For all nodes, the input and output values are calculated
- Starting at the output layer, the deltas are calculated and back propagated
- Now the weights are updated

$$\Delta w_{jk} = \alpha \delta_j O_k$$

$$\delta_j = (r_j - O_j) h'_j(I_j)$$

if layer J is the output Layer

$$\delta_j = h'_j(I_j) \sum_{p=1}^{N_p} \delta_p w_{jp}$$

if layer J is an internal layer and layer P is the next layer

Since $sig'(t) = sig(t)(1 - sig(t))$ for $\theta_0 = 1$ the derivative of the activation function becomes

$$h'_j(I_j) = O_j(1 - O_j)$$

Some facts about NN

- Training with noisy data helps improving performance
- Training with more patterns also
- Single layer perceptron results in hyperplanes
- Two layers result in open or closed convex regions
- Three layers result in arbitrary (limited by the number of nodes) regions
- More than three layers are never needed

12.3 Structural Methods Ch. 12.3

12.3.1 Shape numbers

Remember that the order n of a shape number is the number of digits. This is clearly even for closed boundaries.

To match shape numbers, the degree of similarity k between two shapes a and b is introduced. It is defined as the largest order j , for which the shape numbers s_j still coincide.

$$\begin{aligned} s_j(a) &= s_j(b) & \text{for } j &= 4, 6, 8, \dots, k \\ s_j(a) &\neq s_j(b) & \text{for } j &= k+2, k+4, \dots \end{aligned}$$

The distance between shape a and b is now

$$D(a, b) = \frac{1}{k}$$

From this definition follows that

$$D(a, b) \geq 0 \qquad D(a, b) = 0 \quad \text{if } a = b \qquad D(a, c) \leq \max[D(a, b), D(b, c)]$$

12.3.2 String Matching

Assume that two shapes a, b are coded into strings $a_1 a_2 \dots a_n$ and $b_1 b_2 \dots b_n$. Let α be the number of matches where $a_k = b_k$. The number of symbols that do not match is $\beta = \max[|a|, |b|] - \alpha$

A simple measure of similarity between a and b is now

$$R = \frac{\alpha}{\beta} = \frac{\alpha}{\max[|a|, |b|] - \alpha}$$

R is 0 for no match at all, and is ∞ for the perfect match. Of course it is important that the strings are aligned. Otherwise, one string must be shifted circularly and R is calculated at each position. The largest R now represents the best match.