Feature Extraction

Feature Extraction

Feature is a distinctive attribute or description of "something" we want to label or differentiate

- ➢ Two principal aspects of image feature extraction
  - ◆ feature detection
    - ✓ Finding the "features" in a region or image
  - ◆ feature description
    - ✓ Assigning quantitative attributes to the detected " features"
- ➢ Example
  - ◆ Suppose that we use object corners as features
  - ◆ feature detection
    - ✓ Finding the corners in a region or image
  - ◆ feature description
    - ✓ Assigning quantitative attributes to the detected corners, such as corner orientation, and location with respect to other corners

Feature Extraction

In general, features should be independent of location, rotation, and scale

- The word "independent" usually has one of two meanings
  - Invariant
    - ✓ A feature descriptor is invariant with respect to a set of transformations if its value remains unchanged after the application (to the entity being described) of any transformation from the family
  - Covariant
    - ✓ A feature descriptor is covariant with respect to a set of transformations if applying to the entity any transformation from the set produces the same result in the descriptor
- Three categories
  - boundary
  - region
  - whole image features

Feature Extraction

Basic relationships between pixels

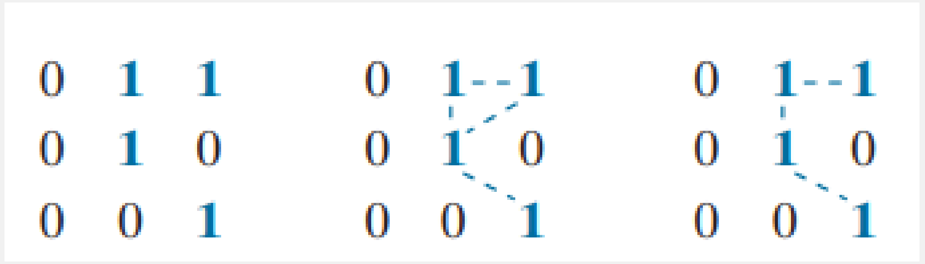
- Neighbors of a pixel
  - 4-neighbors of a pixel (p), is denoted $N_4(p)$
    - ✓ A pixel p at coordinates (x, y) has two horizontal and two vertical neighbors with coordinates (x+1, y), (x-1, y), (x, y+1), (x, y-1)
  - 8-neighbors of a pixel (p), is denoted $N_8(p)$
    - ✓ Four diagonal neighbors of p, denoted $N_D(p)$, have coordinates (x+1, y+1), (x+1, y-1), (x-1, y+1), (x-1, y-1)
    - ✓ These neighbors together with the 4-neighbors forms $N_8(p)$

Feature Extraction

Adjacency

- Let V be the set of intensity values used to define adjacency
  - In binary image, V={1}
  - In grayscale image, V typically contains more elements
  - Three types of adjacency
    1. 4-adjacency: two pixels p and q with values from V are 4-adjacent if q is in the set $N_4(p)$
    2. 8-adjacency: two pixels p and q with values from V are 8-adjacent if q is in the set $N_8(p)$
    3. m-adjacency (mixed-adjacency): two pixels p and q with values from V are m-adjacent if
       (a) q is in $N_4(p)$, or
       (b) q is in $N_D(p)$ and the set $N_4(p) \cap N_4(q)$ has no pixels whose values are from V
  - A digital path (or curve) from pixel p with coordinates (x_0, y_0) to pixel q with coordinates (x_n, y_n) is a sequence of distinct pixels with coordinates

$$(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$$



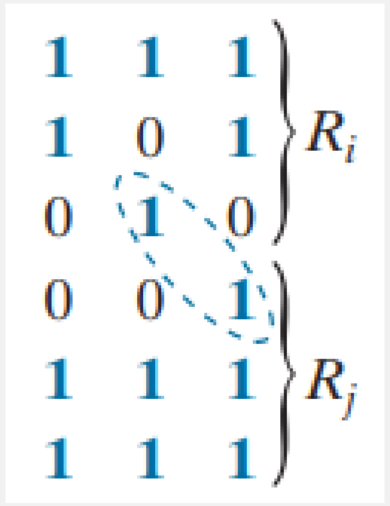
Feature Extraction

Connectivity

- Let S represent a subset of pixels in an image
  - Two pixels p and q are connected if there exists a path between them consisting entirely of pixels in S
  - For any pixel p in S, the set of pixels that are connected to it in S is called a connected component
  - If it only has one component, and that component is connected, then S is called a connected set

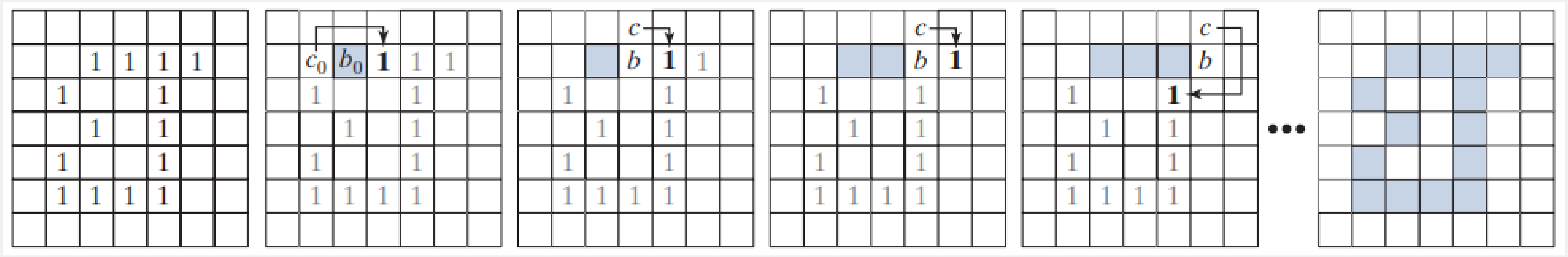
Region

- Let R represent a subset of pixels in an image
  - R is a region of the image if R is a connected set
  - Two regions, R_i and R_j are adjacent if their union forms a connected set; regions that are not adjacent are said to be disjoint



Feature Extraction

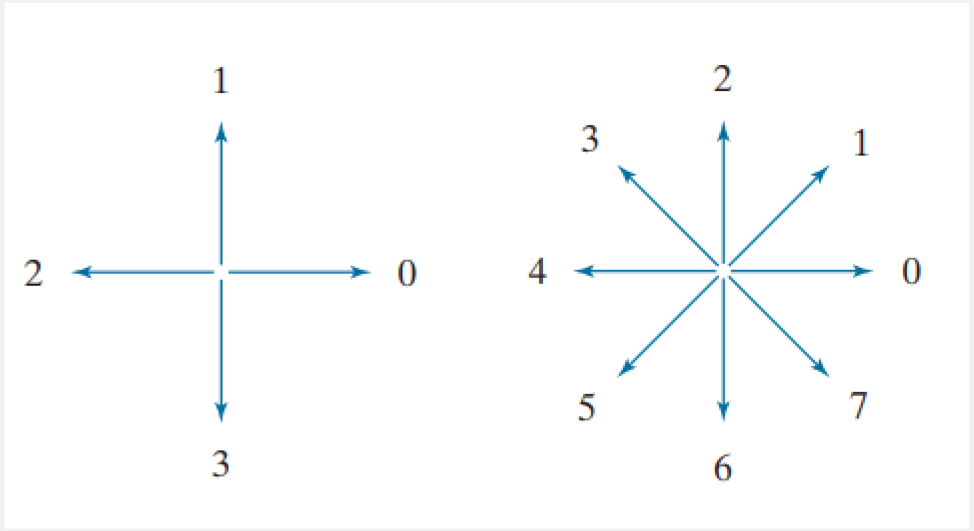
Boundary following (tracing)

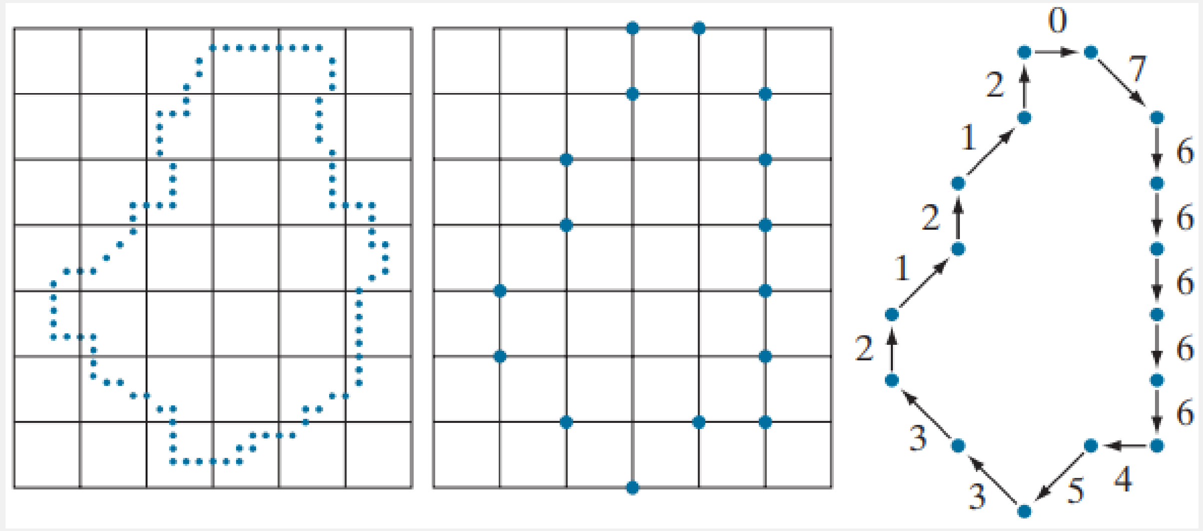


Feature Extraction

Chain codes are used to represent a boundary by a connected sequence of straight-line segments of specified length and direction

- ➢ Freeman chain codes
  - ◆ A boundary code formed as a sequence of such directional numbers is referred to as a Freeman chain code
    - ✓ 0766…12



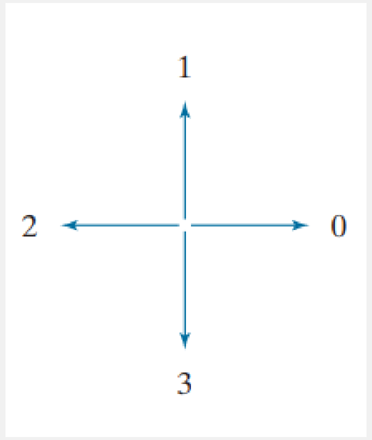


Feature Extraction

The numerical value of a chain code depends on the starting point

However, the code can be normalized with respect to the starting point by a straightforward procedure

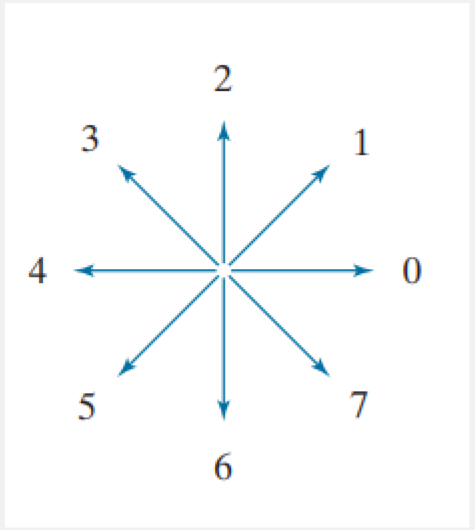
- ➢ The starting point is redefined so that the resulting sequence of numbers forms an integer of minimum magnitude
  - ◆ First difference with the 4-directional chain code
    - ✓ 10103322 -> 3133030
  - ◆ First difference of a circular sequence with the 4-directional chain code
    - ✓ 10103322 -> 33133030

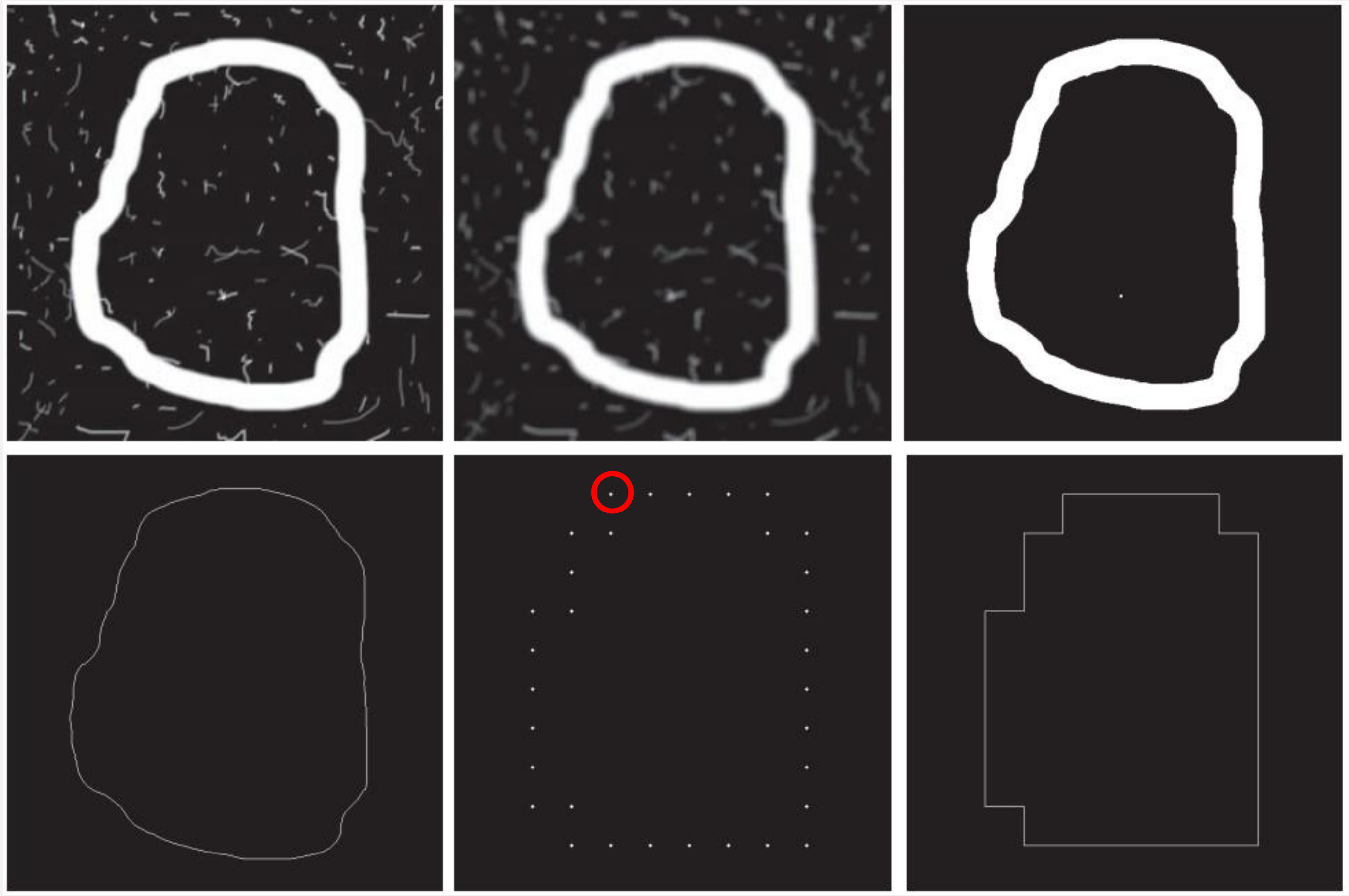


Feature Extraction

Example

- 00006066666666444444242222202202 -> 60006260000000600000626000062062

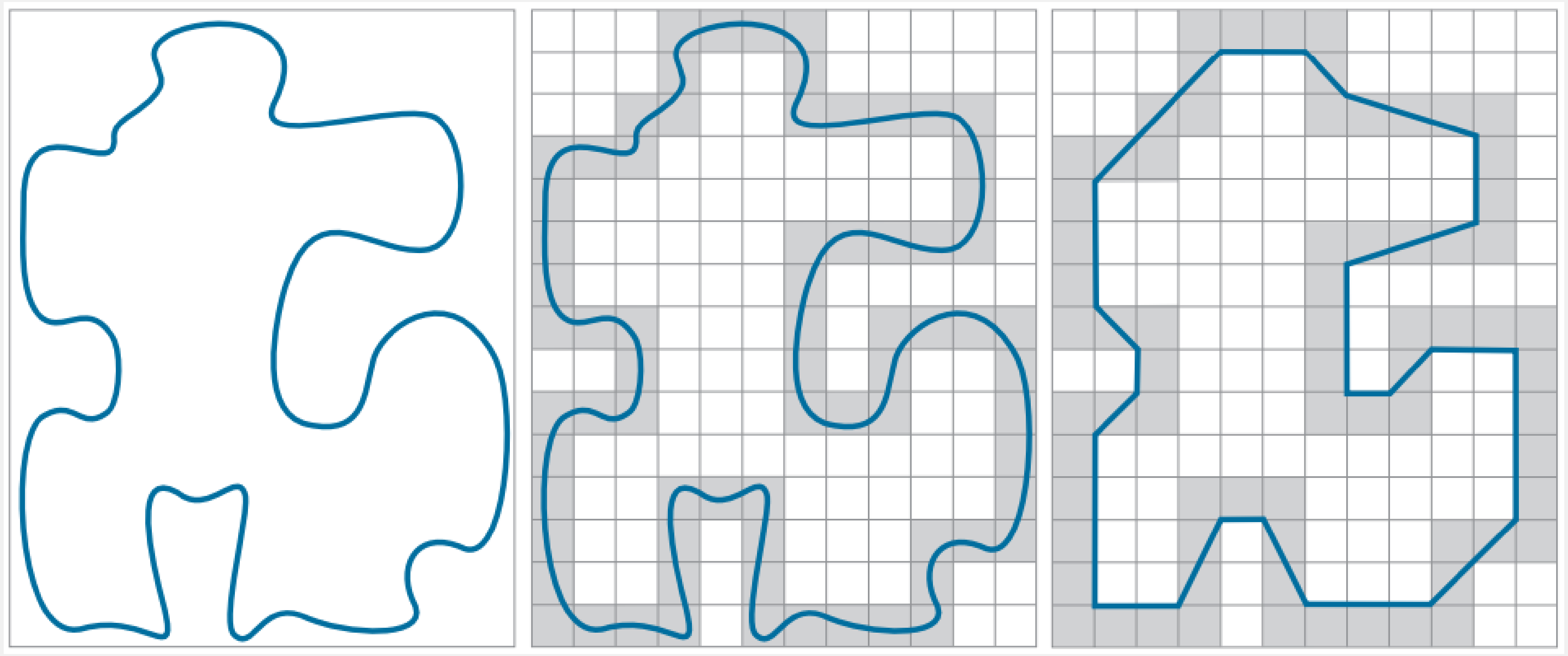




Feature Extraction

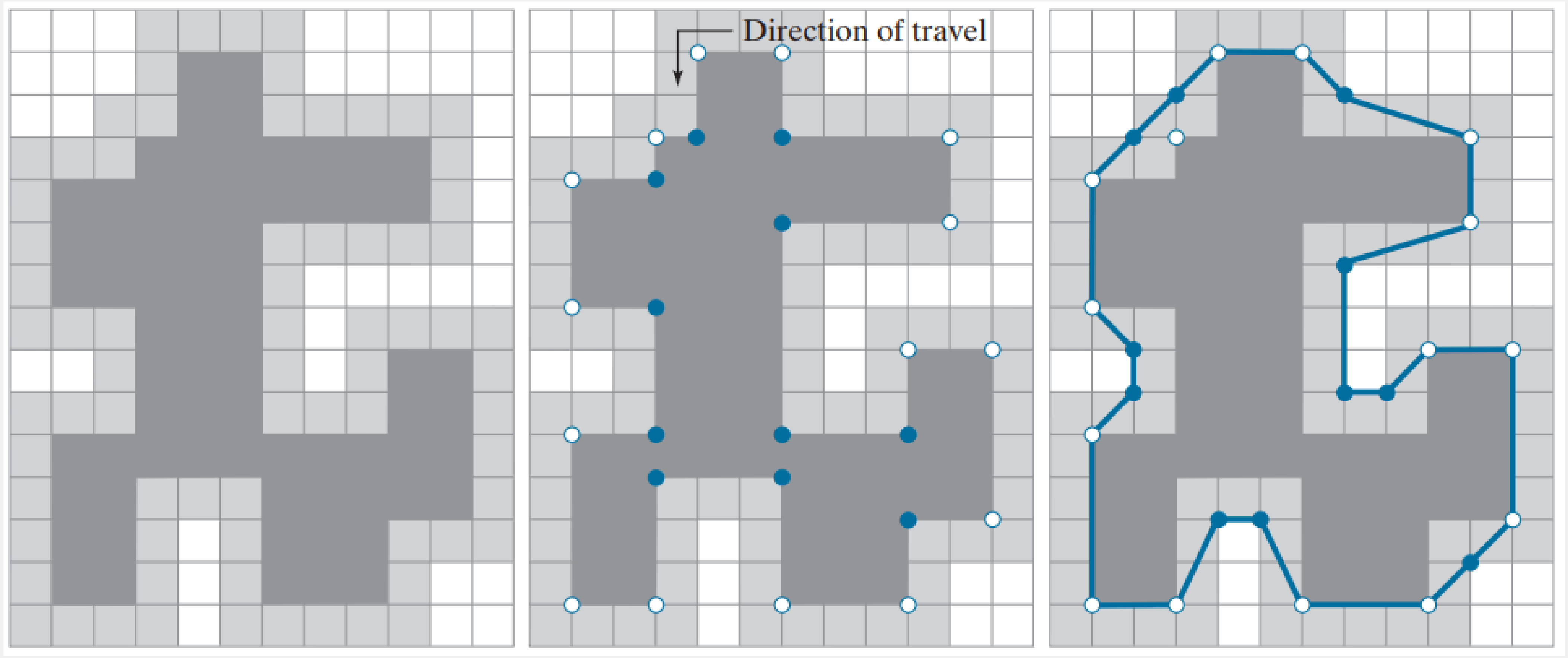
Boundary approximations

- Minimum-perimeter polygons (MPP)
  - A digital boundary can be approximated with arbitrary accuracy be a polygon



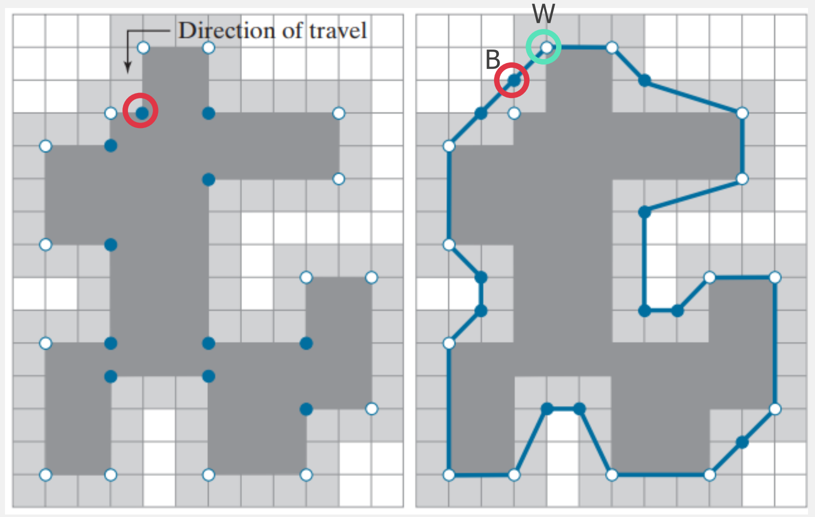
Feature Extraction

- Minimum-perimeter polygons (MPP)
  - A digital boundary can be approximated with arbitrary accuracy be a polygon



Feature Extraction

- Minimum-perimeter polygons (MPP)
  - Observations (W: convex vertices, B: mirrored concave vertices)
    1. The MPP bounded by a simply connected cellular complex is not self-intersecting
    2. Every convex vertex of the MPP is a W vertex, but not every W vertex of a boundary is a vertex of the MMP
    3. Every mirrored concave vertex of the MPP is a B vertex, but not every B vertex of a boundary is a vertex of the MPP
    4. All B vertices are on or outside the MPP, and all W vertices are on or inside the MPP
    5. The uppermost-leftmost vertex in a sequence of vertices contained in a cellular complex is always a W vertex of the MPP



Feature Extraction

- Minimum-perimeter polygons (MPP)
  - How to calculate the orientation of triplets of points
    - ✓ Using determinant

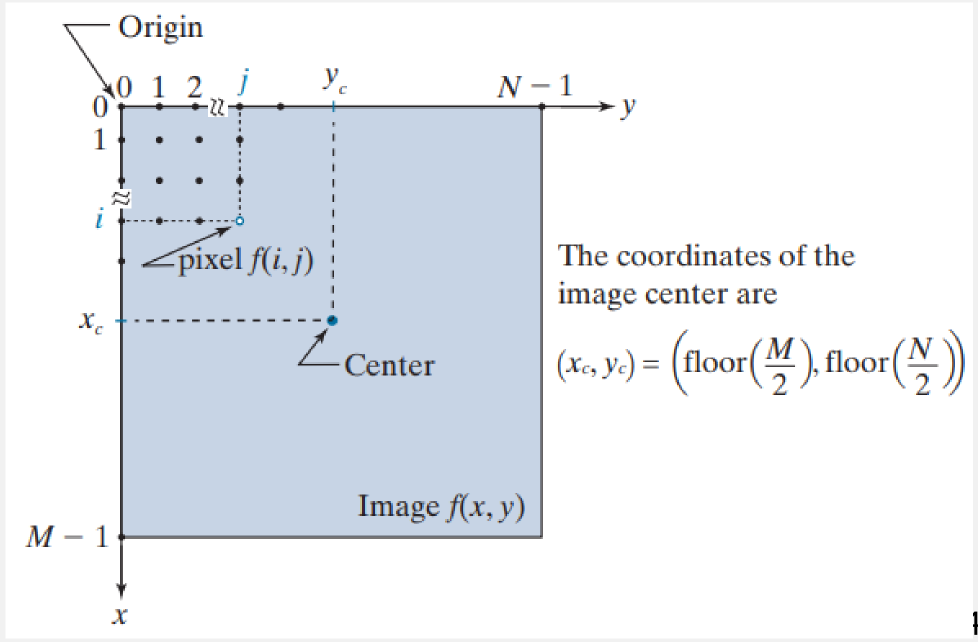
$$a = (a_x, a_y), b = (b_x, b_y), c = (c_x, c_y)$$

$$A = \begin{bmatrix} a_x & a_y & 1 \\ b_x & b_x & 1 \\ c_x & c_x & 1 \end{bmatrix}$$

$$\det(A) = \begin{cases} > 0 & \mathit{if}\ (a, b, c) \mathit{is\ a\ counterclockwise\ sequence} \\ \ 0 & \mathit{if\ the\ points\ are\ collinear} \\ < 0 & \mathit{if}\, (a, b, c)\ \mathit{is\ a\ clockwise\ sequence} \end{cases}$$

$$\mathrm{sgn}(a, b, c) \equiv \det(A)$$

The sequence a=(3,4), b=(2,3), and c=(3,2) is in the counterclockwise direction



Origin
0 1 2 j
y_c
$N-1$
y
0
1
i
pixel $f(i, j)$
x_c
Center
The coordinates of the image center are
$(x_c, y_c) = \left(\mathrm{floor}\left(\frac{M}{2}\right), \mathrm{floor}\left(\frac{N}{2}\right)\right)$
Image $f(x, y)$
$M-1$
x

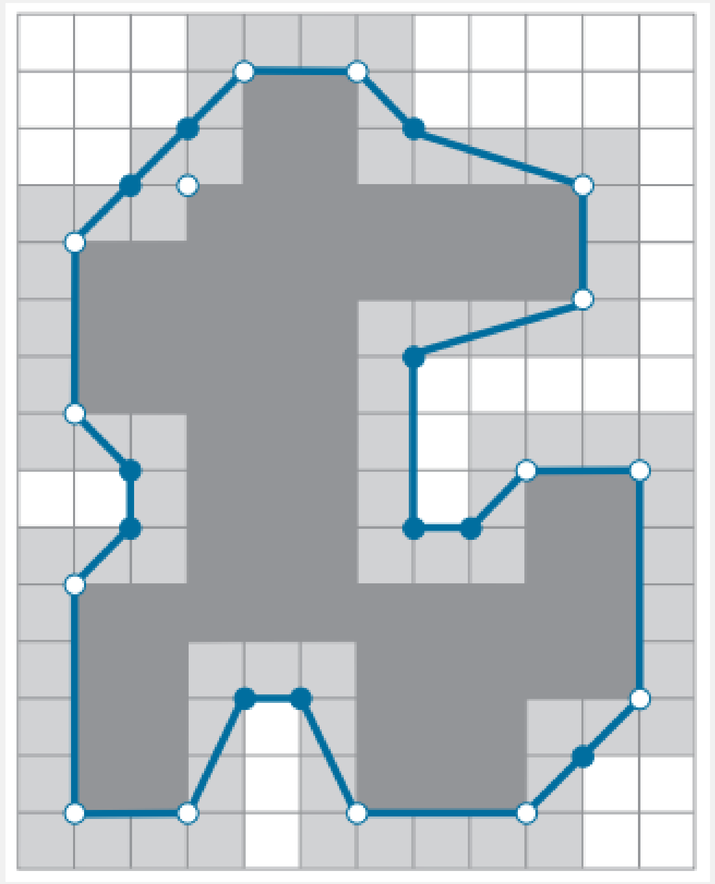
Feature Extraction

- Minimum-perimeter polygons (MPP)
  - Let V_0 is the uppermost-leftmost vertex, V_k is the current vertex being examined, and V_L is the last MPP vertex; W_C and B_C are two crawler points crawls along W and B vertices
  - Algorithm
    a) V_k is on the positive side of the line through the pair of points (V_L, W_C), in which case $\mathrm{sgn}(V_L, W_C, V_K) > 0$
    b) V_k is on the negative side of the line through the pair (V_L, W_C) or is collinear with it, that is $\mathrm{sgn}(V_L, W_C, V_K) \leq 0$; V_k is on the positive side of the line through the pair of points (V_L, B_C) or is collinear with it, that is $\mathrm{sgn}(V_L, B_C, V_K) \geq 0$
    c) V_k is on the negative side of the line through the pair (V_L, B_C), in which case $\mathrm{sgn}(V_L, B_C, V_K) < 0$
  - If condition a) holds, the next MPP vertex is W_C, and let $V_L=W_C$; then reinitialize algorithm by setting $W_C=B_C=V_L$, and start with the next vertex after the newly changed V_L
  - If condition b) holds, V_k becomes a candidate MPP vertex, and set $W_C=V_k$ if V_k is convex; otherwise set $B_C=V_k$. Then continue with the next vertex in the list
  - If condition c) holds, the next MPP vertex is B_C and let $V_L=B_C$; then reinitialize the algorithm by setting $W_C=B_C=V_L$ and start with the next vertex after the newly changed V_L

Feature Extraction

- Minimum-perimeter polygons (MPP)

[V_0 (1,4) W] | [V_1 (2,3) B | [V_2 (3,3) W] | [V_3 (3,2) B] | [V_4 (4,1) W] | [V_5 (7,1) W] | [V_6 (8,2) B] | [V_7 (9,2) B] | …



1. Start by letting V_L and V_0 be equal and initializing the other variables W_C and B_C

$$W_C=B_C=V_L=V_0=(1,4)$$

2. Next vertex is $V_1=(2,3)$: $sgn(V_L,W_C,V_1)=0$ and $sgn(V_L,B_C,V_1)=0$ -> condition b) holds -> Update crawler $B_C = V_1 = (2,3)$

$$V_L=(1,4),\ W_C=(1,4),\ B_C=(2,3)$$

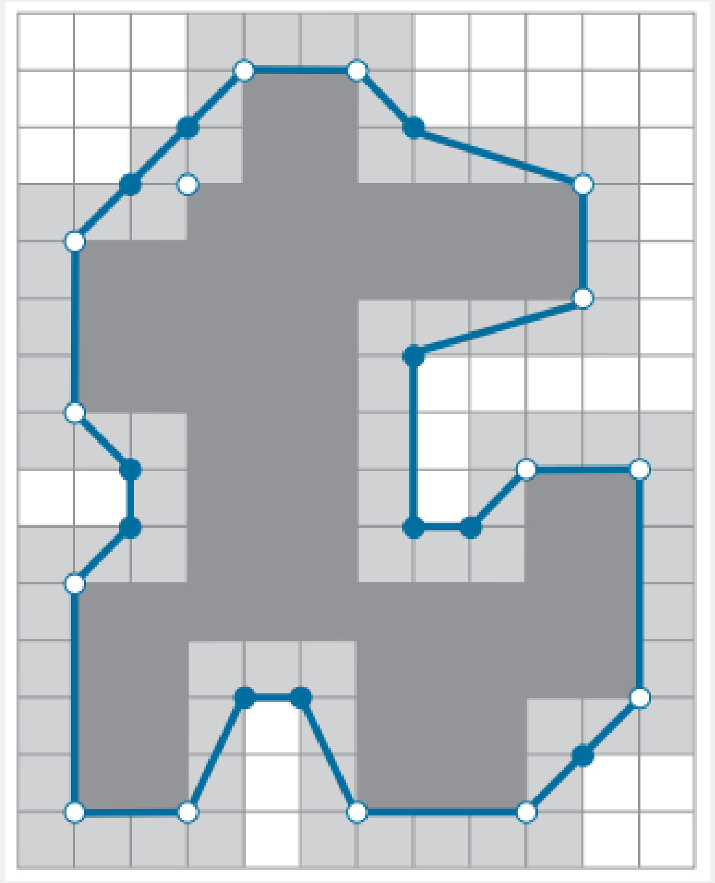
3. Next vertex is $V_2=(3,3)$: $sgn(V_L,W_C,V_2)=0$ and $sgn(V_L,B_C,V_2)=0$ -> condition b) holds -> Update crawler $W_C = V_2 = (3,3)$

$$V_L=(1,4),\ W_C=(3,3),\ B_C=(2,3)$$

Feature Extraction

- Minimum-perimeter polygons (MPP)

[V_0 (1,4) W] | [V_1 (2,3) B | [V_2 (3,3) W] | [V_3 (3,2) B] | [V_4 (4,1) W] | [V_5 (7,1) W] | [V_6 (8,2) B] | [V_7 (9,2) B] | …



4. Next vertex is V_3=(3,2): sgn(V_L,W_C,V_3)=-2 and sgn(V_L,B_C,V_3)=0 -> condition b) holds -> Update crawler B_C = V_3 = (3,2)

V_L=(1,4), W_C=(3,3), B_C=(3,2)

5. Next vertex is V_4=(4,1): sgn(V_L,W_C,V_4)=-3 and sgn(V_L,B_C,V_1)=0 -> condition b) holds -> Update crawler W_C = V_4 = (4,1)

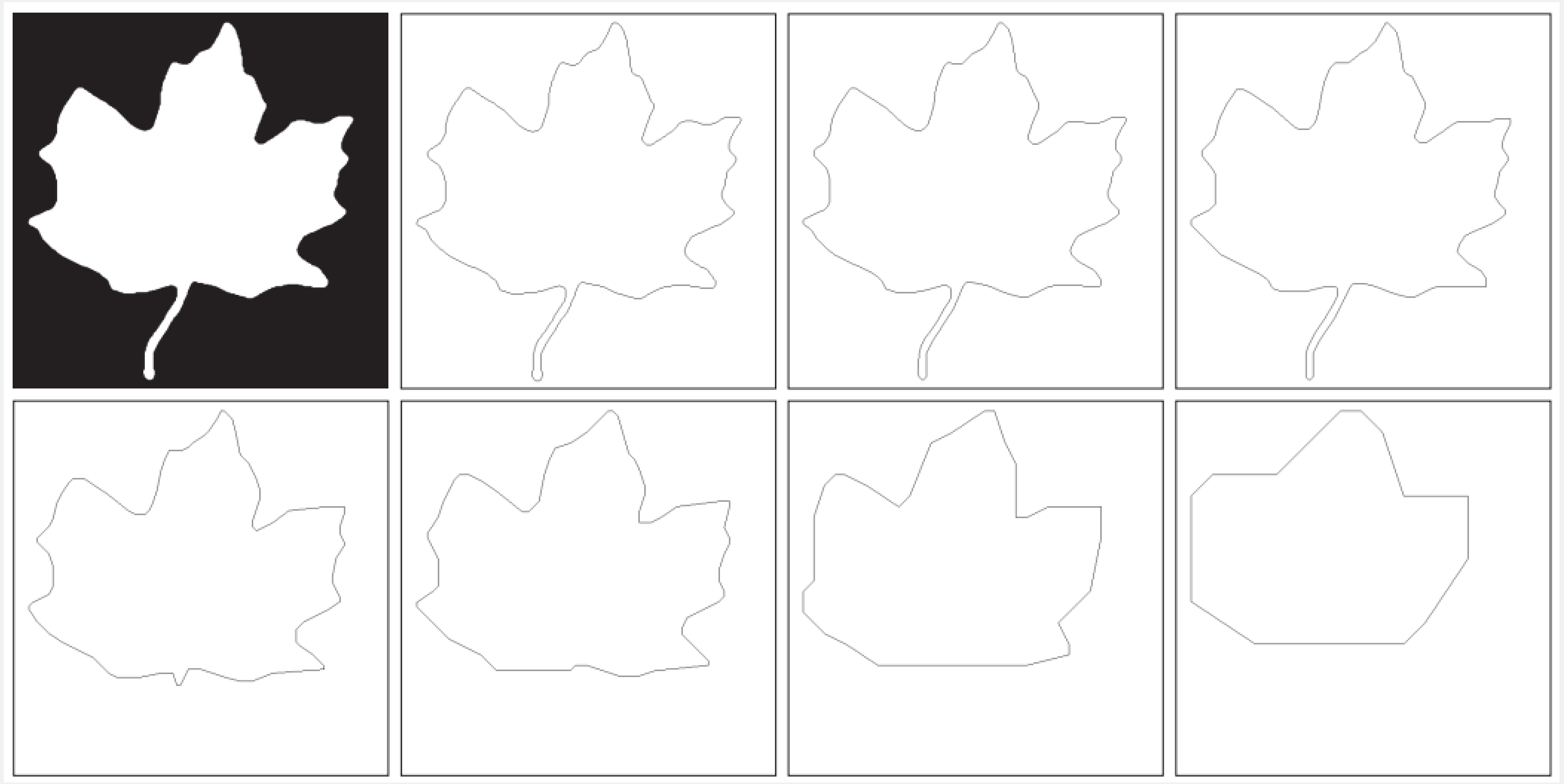
V_L=(1,4), W_C=(4,1), B_C=(3,2)

6. Next vertex is V_5=(7,1): sgn(V_L,W_C,V_5)=9 condition a) holds -> Update VL=WC=(4,1) and reinitialize B_C = W_C = V_L = (4,1) -> Next vertex is also V_5

W_C=B_C=V_L= (4,1)

Feature Extraction

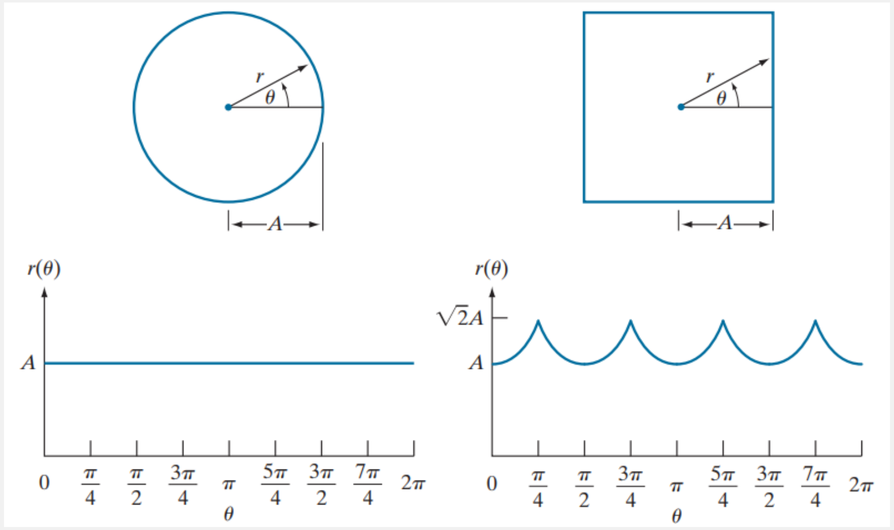
- Minimum-perimeter polygons (MPP)



Feature Extraction

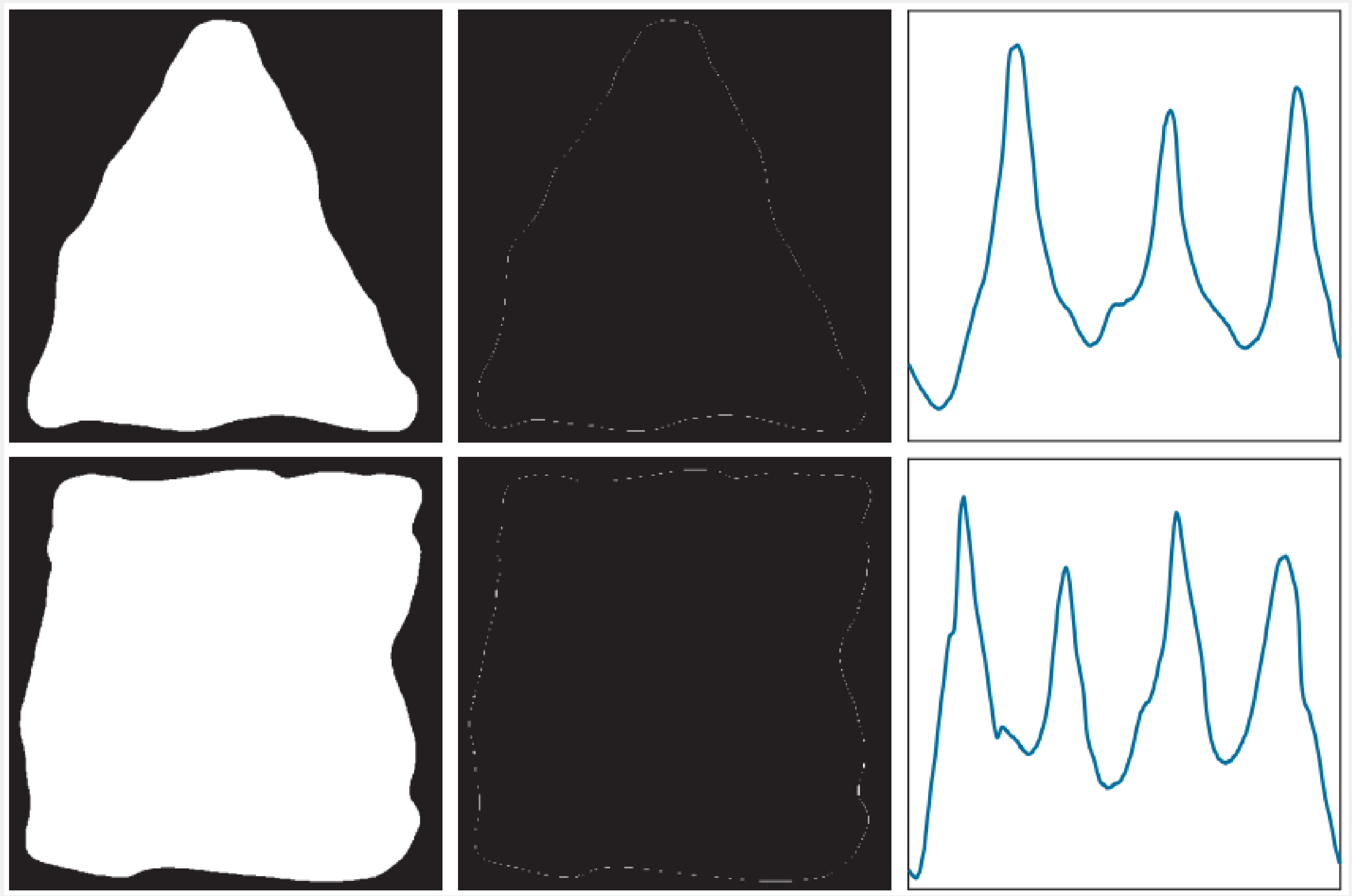
Boundary approximations

- Signatures
  - It is a 1-D functional representation of a 2-D boundary and may be generated in various ways
  - One of the simplest is to plot the distance from the centroid to the boundary as a function of angle
  - The basic idea of using signatures is to reduce the boundary representation to a 1-D function that presumably is easier to describe than the original 2-D boundary



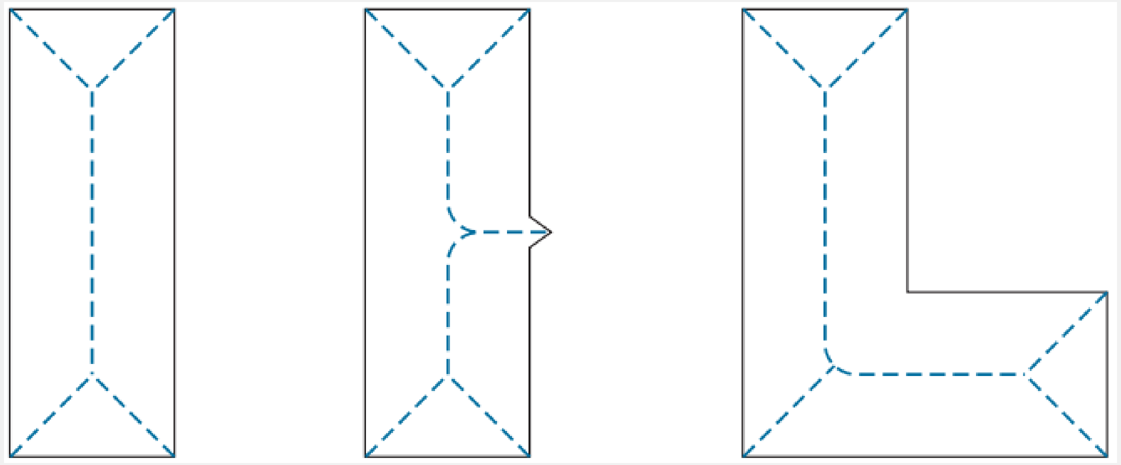
Feature Extraction

- Signatures



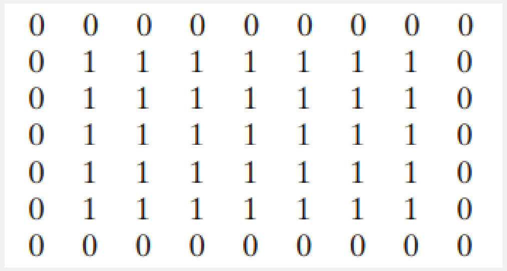
Feature Extraction

- Skeletons, medial axis, and distance transforms
  - Skeletons are related to the shape of a region
  - It can be computed using the coordinates of points in the entire region, including boundary
  - Using one of two principle approaches
    1. Using morphological erosion
    2. Using medial axis transform (MAT)
       - ✓ For each point (p) in region (R) with boundary B (B), finding its closest neighbor in B
       - ✓ If p has more than one such neighbor, it is said to belong to the medial axis of R

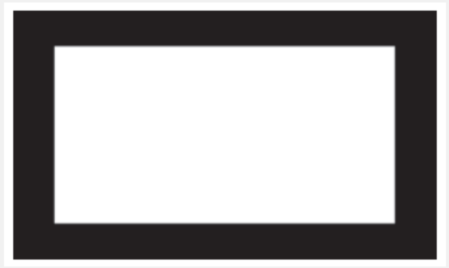


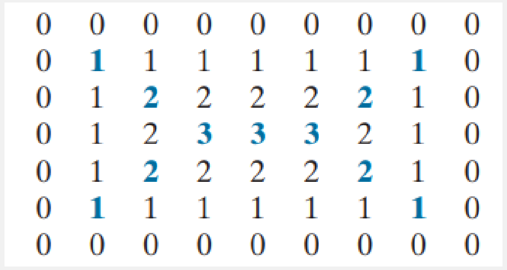
Feature Extraction

- Skeletons, medial axis, and distance transforms
  - Computing MAT of a region requires calculating the distance from every interior points to every point on the border of the region, but it is an impractical endeavor in most applications
  - Distance transform
    - ✓ Finding the distance from the pixels of a region of foreground (white) to their nearest background (zero) pixels, which constitute the region boundary
    - ✓ The MAT is equivalent to the ridge of the distance transform and the ridge is the set of local maxima

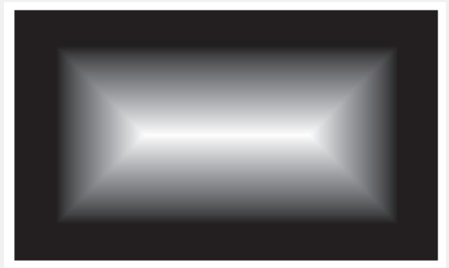


0	0	0	0	0	0	0	0	0
0	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	0
0	0	0	0	0	0	0	0	0





0	0	0	0	0	0	0	0	0
0	**1**	1	1	1	1	1	**1**	0
0	1	**2**	2	2	2	**2**	1	0
0	1	2	**3**	**3**	**3**	2	1	0
0	1	**2**	2	2	2	**2**	1	0
0	**1**	1	1	1	1	1	**1**	0
0	0	0	0	0	0	0	0	0



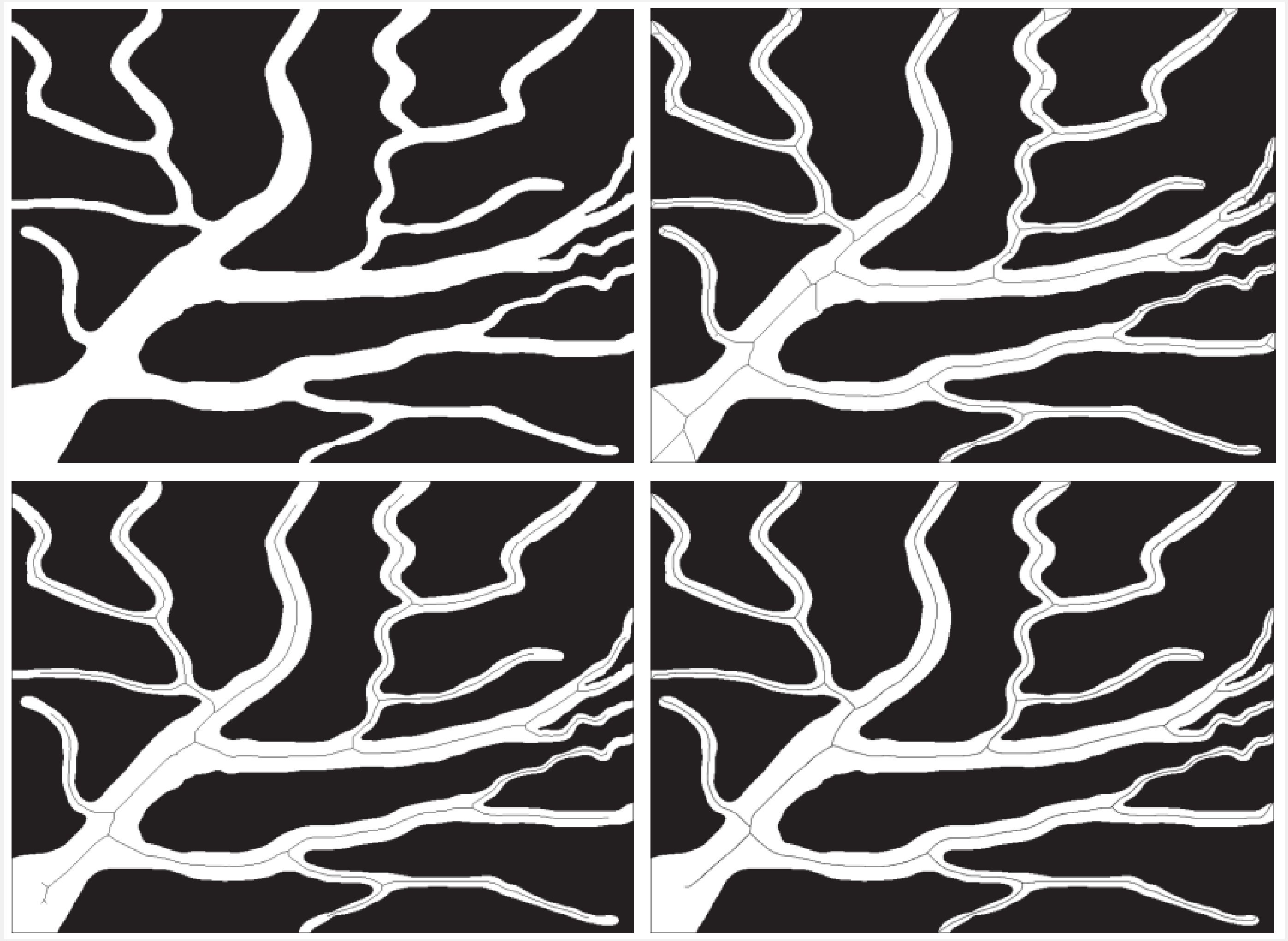
Feature Extraction

- Skeletons, medial axis, and distance transforms
  - Zhang and Suen et al. [paper]
  - Algorithm (satisfy one of two steps -> set the pixel to 0)
    a) step 1: For each pixel p(x, y), checking following conditions
       1. p(x, y) is a foreground pixel (i.e., p1=1)
       2. p(x, y) has between 2 and 6 foreground pixels among its neighbors (i.e. $2 \le B(p1) \le 6$, where B(p1) is the number of foreground pixels among the 8 neighbors))
       3. There is exactly one transition from background to foreground among the neighbors of p(x,y) (i.e., A(p1)=1, where A(p1) is the number of 0->1 transitions)
       4. At least one of the neighbors p2, p4, p6 is a background pixel (i.e., $p2 \times p4 \times p6=0$)
       5. At least one of the neighbors p4, p6, p8 is a background pixel (i.e., $p4 \times p6 \times p8=0$)
    b) step 2: For each pixel p(x, y), checking following conditions
       1. 1-3 same as step1
       2. At least one of the neighbors p2, p4, p8 is a background pixel (i.e., $p2 \times p4 \times p8=0$)
       3. At least one of the neighbors p2, p6, p8 is a background pixel (i.e., $p2 \times p6 \times p8=0$)

p9	p2	p3
p8	p1	p4
p7	p6	p5

Feature Extraction

- Skeletons, medial axis, and distance transforms



Feature Extraction

Basic boundary descriptors

- Length
  - The number of pixels along a boundary is an approximation of its length
  - In chain-coded curve
    - ✓ The number of vertical and horizontal components plus $\sqrt{2}$ multiplied by the number of diagonal components gives its exact length
  - In polygonal curve
    - ✓ The length is equal to the sum of the lengths of the polygonal segments
- Diameter (is called the major axis or longest chord)
  - The diameter of a boundary B is defined as $diameter(B) = \max_{i,j}[D(p_i, p_j)]$
  - The minor axis of a boundary is defined as 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
  - The box just described is called the basic rectangle or bounding box, and the ratio of the major to the minor axis is called the eccentricity of the boundary

$$D_e(p,q) = [(x-u)^2 + (y-v)^2]^{\frac{1}{2}}$$

$$D_4(p,q) = |x-u| + |y-v|$$

$$D_8(p,q) = \max(|x-u|, |y-v|)$$

Feature Extraction

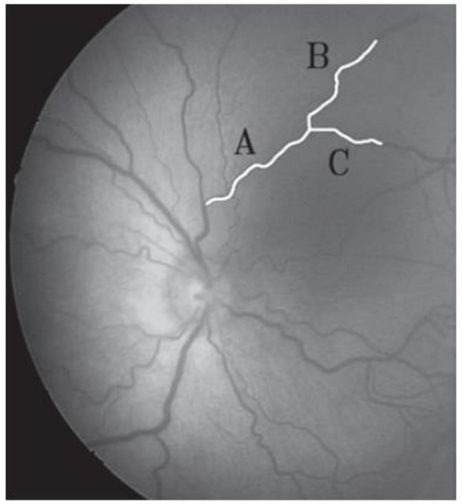
Basic boundary descriptors

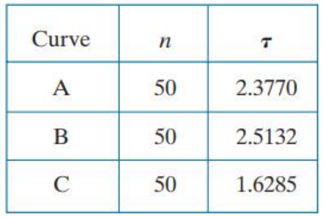
- Curvature is a measure of how sharply a curve bends at a particular point

$$\kappa(t) = \frac{x'(t)y''(t) - y'(t)x''(t)}{(x'(t)^2 + y'(t)^2)^{3/2}}$$

- Tortuosity is a measure of how much a curve deviates from being a straight line

$$\tau = \frac{L_{actual}}{L_{direct}}$$



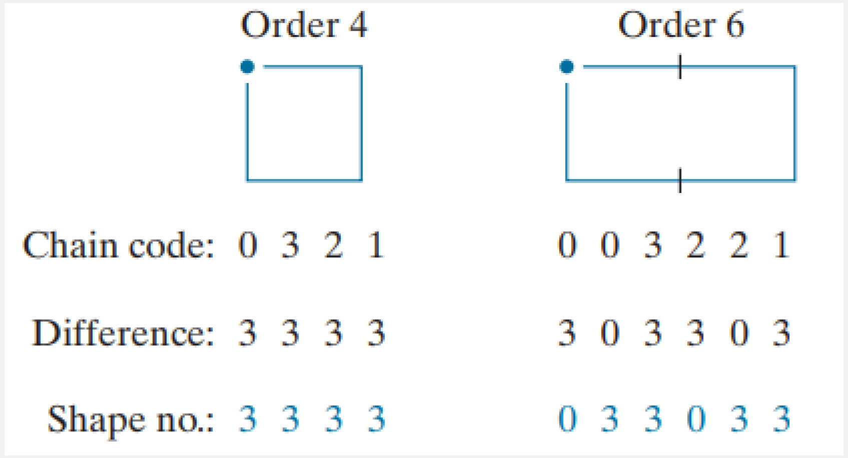


Curve	n	τ
A	50	2.3770
B	50	2.5132
C	50	1.6285

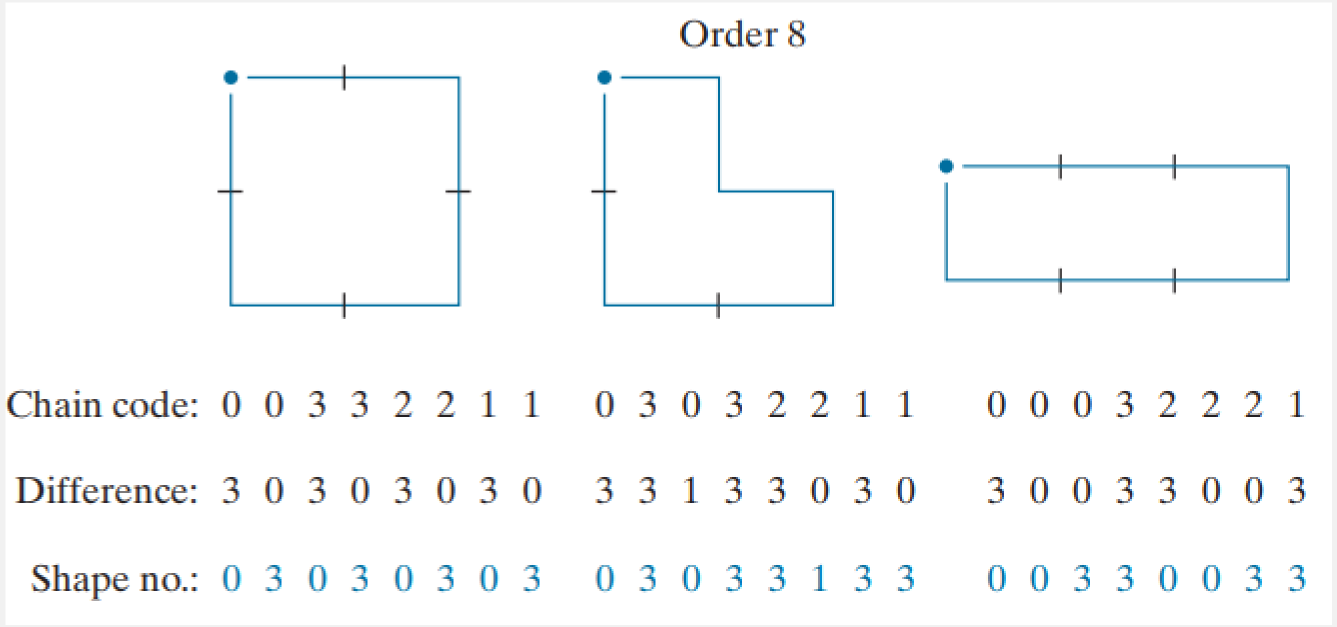
Feature Extraction

Basic boundary descriptors

- ➢ Shape numbers
  - ◆ It is a method to describe and recognize shape based on the chain code
  - ◆ Freeman chain coded
    - ✓ The shape number 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



	Order 4	Order 6
Chain code:	0 3 2 1	0 0 3 2 2 1
Difference:	3 3 3 3	3 0 3 3 0 3
Shape no.:	3 3 3 3	0 3 3 0 3 3



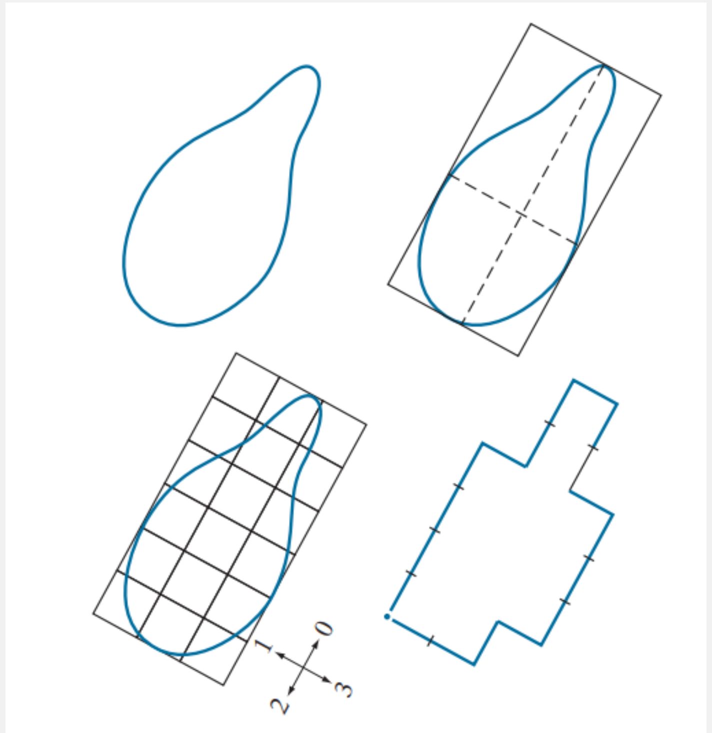
Order 8

Chain code:	0 0 3 3 2 2 1 1	0 3 0 3 2 2 1 1	0 0 0 3 2 2 2 1
Difference:	3 0 3 0 3 0 3 0	3 3 1 3 3 0 3 0	3 0 0 3 3 0 0 3
Shape no.:	0 3 0 3 0 3 0 3	0 3 0 3 3 1 3 3	0 0 3 3 0 0 3 3

Feature Extraction

Basic boundary descriptors

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



Chain code: 0 0 0 0 3 0 0 3 2 2 3 2 2 2 1 2 1 1

Difference: 3 0 0 0 3 1 0 3 3 0 1 3 0 0 3 1 3 0

Shape no.: 0 0 0 3 1 0 3 3 0 1 3 0 0 3 1 3 0 3

Feature Extraction

Basic boundary descriptors

- Statistical moments
  - It is an applicable to 1-D renditions of 2-D boundary, such as signature
  - Histogram
    - ✓ Let r_k, for k=0, 1, 2, …, L-1 denote the intensities of an L-level digital image, f(x, y)
    - ✓ The unnormalized histogram of f is defined as

$$h(r_k) = n_k \quad \text{for k=0, 1, 2, …L-1}$$

where n_k is the number of pixels in f with intensity r_k, and the subdivisions of the intensity scale are called histogram bins

- ✓ Similarly, the normalized histogram of f is defined as

$$p(r_k) = \frac{h(r_k)}{MN} = \frac{n_k}{MN}$$

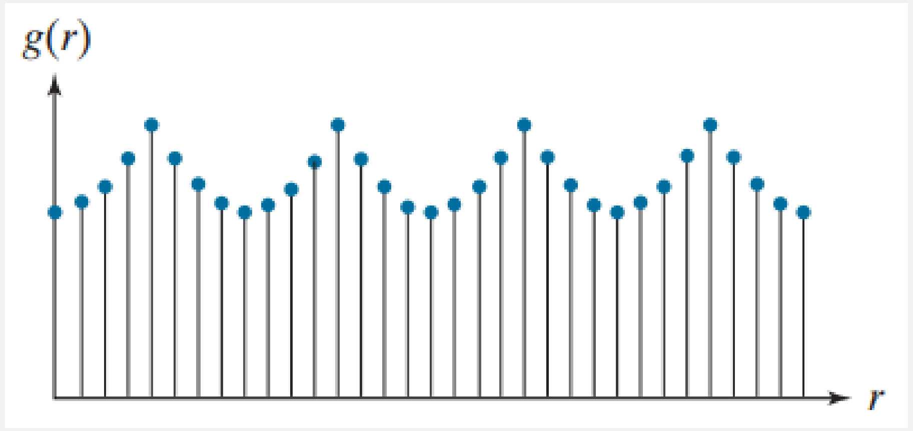
where M and N are the number of image rows and columns

Feature Extraction

Basic boundary descriptors

- Statistical moments
  - Treat the amplitude of g as a discrete random variable z and form an amplitude histogram p(z_i), i=0, 1, 2, …, A-1, so that p(z_i) is an estimate of the probability of intensity value z_i occurring
  - Then, the nth moment of z about its mean is

$$\mu_n(z) = \sum_{i=0}^{A-1} (z_i - m)^n p(z_i) \qquad m = \sum_{i=0}^{A-1} z_i p(z_i)$$



Feature Extraction

Basic region feature descriptors

- Compactness
  - It is a metric used to describe the shape of a region

$$compactness = \frac{p^2}{A}$$ where p is the perimeter of a region and A is area

  - The compactness of a circle is minimal and the value is 4π

- Circularity
  - It is used to describe how close a shape is to a circle

$$circularity = \frac{4\pi A}{p^2}$$

  - The circularity of a circle is 1

- Differences
  - Compactness is a measure of how complex the shape's perimeter is relative to its area
  - Circularity focuses on how similar a shape is to a circle

Feature Extraction

Basic region feature descriptors

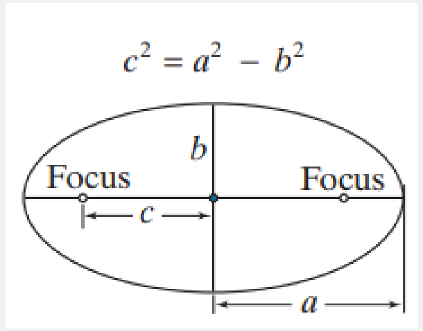
- Effective diameter
  - It is a metric used to describe the effective diameter of a region

$$D_e = 2 \times \sqrt{\frac{Area}{\pi}}$$ where p is the perimeter of a region and A is area

  - It can compare sizes of different shapes
  - Ex: Particle size analysis, biology, and medicine

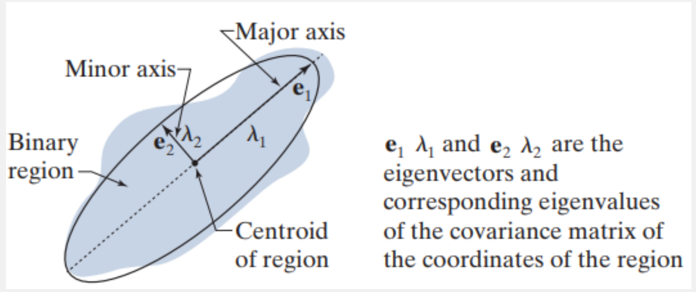
- Eccentricity
  - It is a metric used to describe the ovality or aspect ratio of a shape

$$eccentricity = \frac{c}{a} = \frac{\sqrt{a^2-b^2}}{a} = \sqrt{1-(\frac{b}{a})^2} \quad a \geq b$$



- Eigenvectors of the covariance matrix, C

$$C = \frac{1}{K-1}\sum_{k=1}^{K}(z_k - \bar{z})(z_k - \bar{z})^T \quad \bar{z} = \frac{1}{K}\sum_{k=1}^{K} z_k \quad eccentricity = \sqrt{1-(\frac{\lambda_1}{\lambda_2})^2} \quad \lambda_2 \geq \lambda_1$$

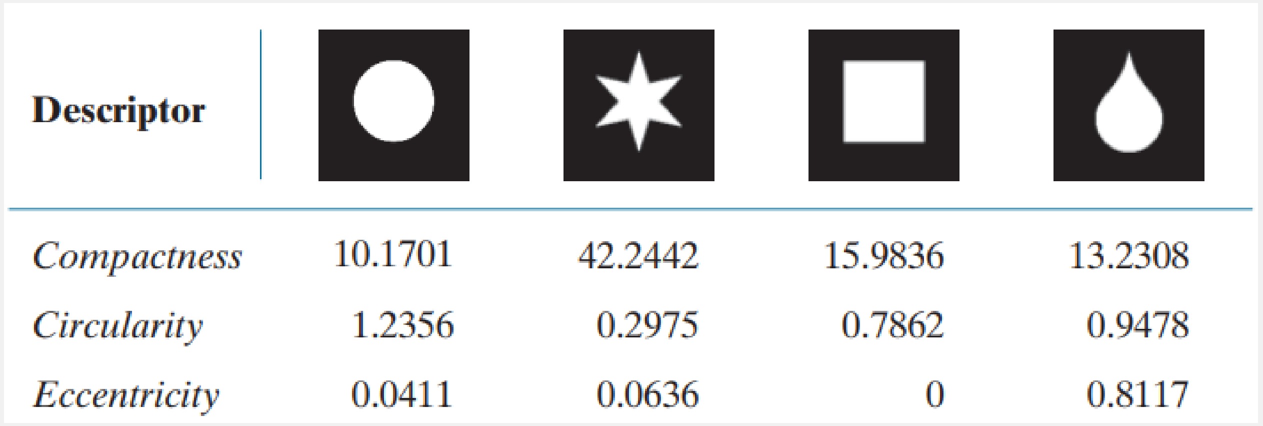


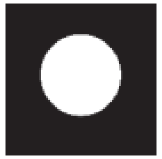
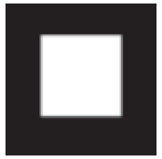
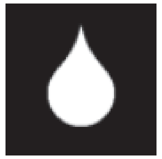
Major axis
Minor axis
e_1
e_2
λ_2
λ_1
Binary region
Centroid of region
e_1 λ_1 and e_2 λ_2 are the eigenvectors and corresponding eigenvalues of the covariance matrix of the coordinates of the region

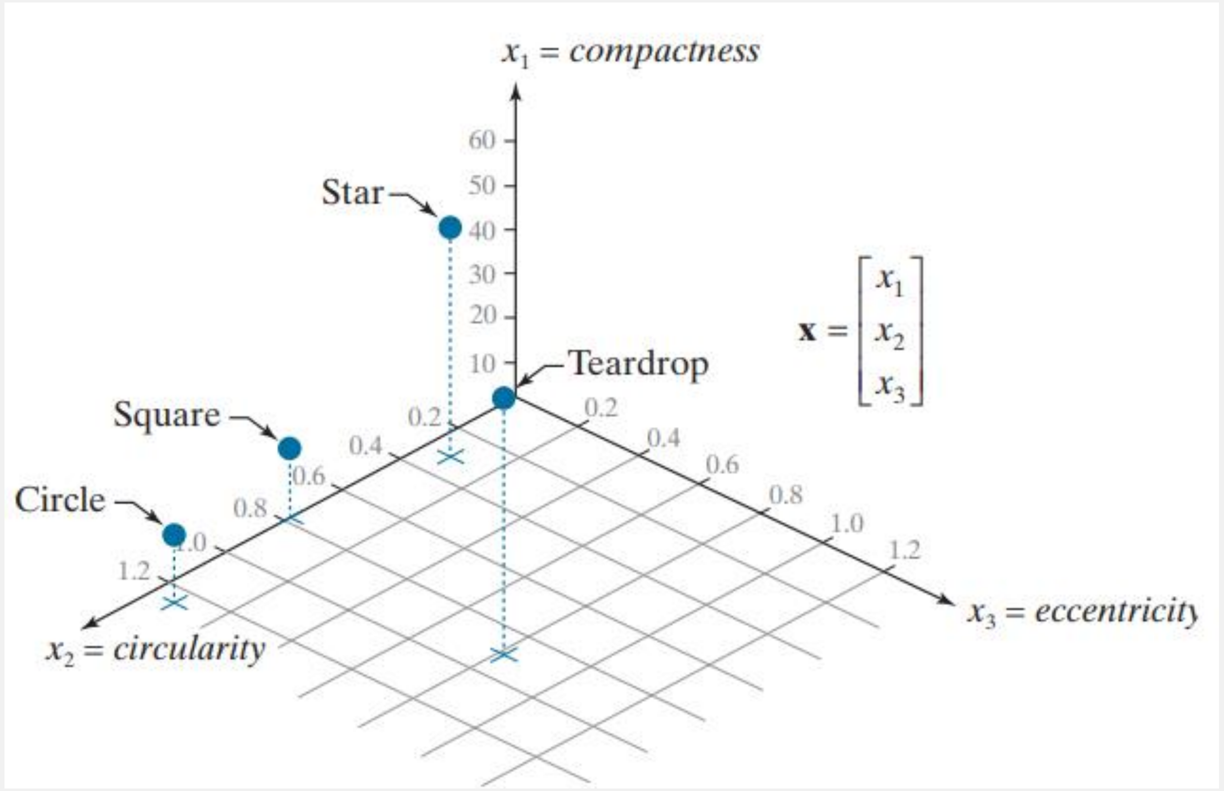
Feature Extraction

Basic region feature descriptors

- Compactness, circularity, and eccentricity



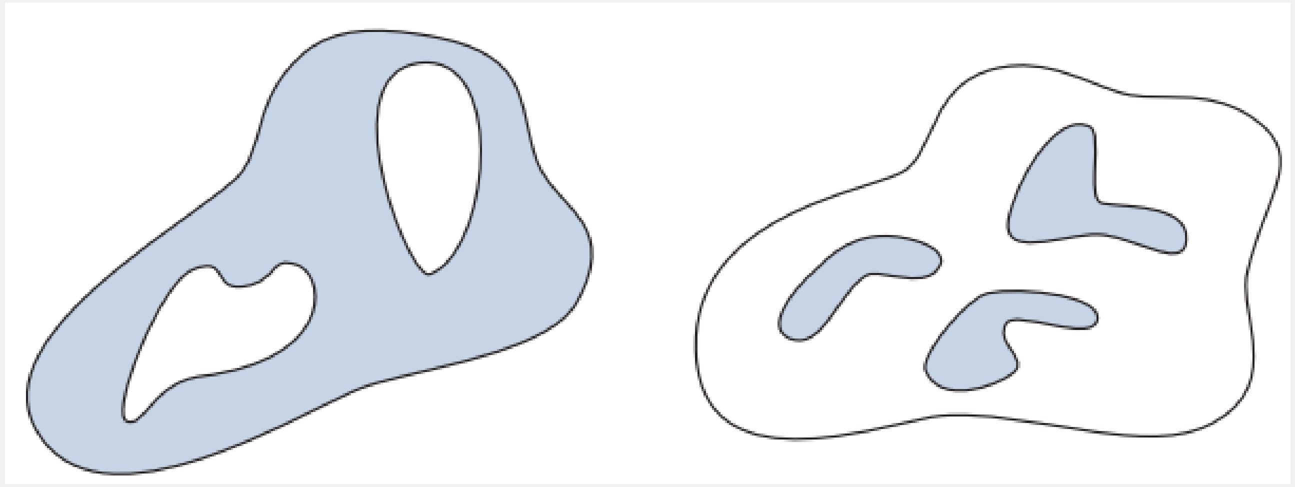
Descriptor				
Compactness	10.1701	42.2442	15.9836	13.2308
Circularity	1.2356	0.2975	0.7862	0.9478
Eccentricity	0.0411	0.0636	0	0.8117

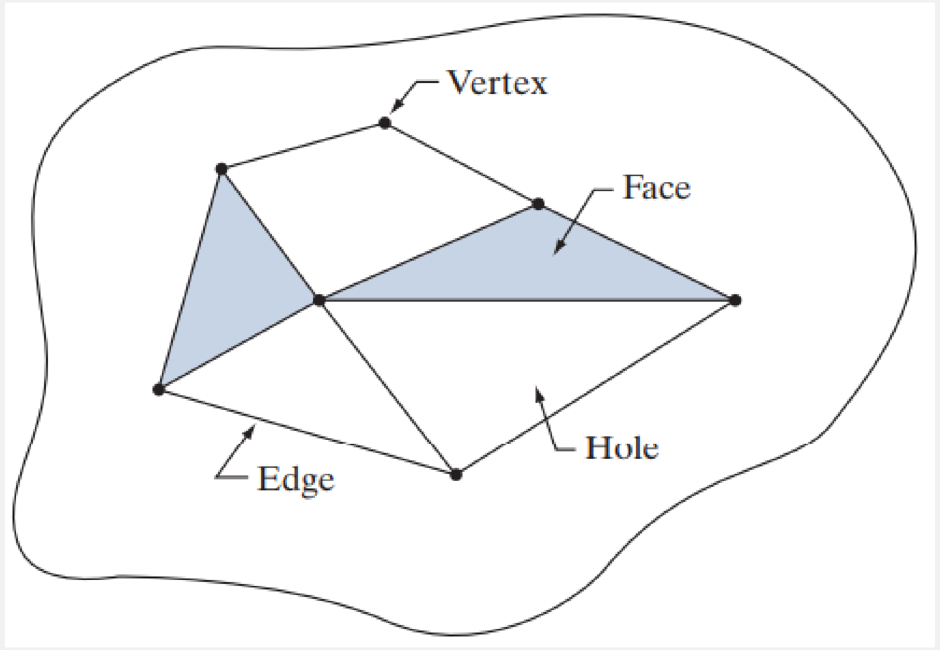


Feature Extraction

Basic region feature descriptors

- Topological
  - It is the study of properties of a figure that are unaffected by any deformation, provided that there is no tearing or joining of the figure

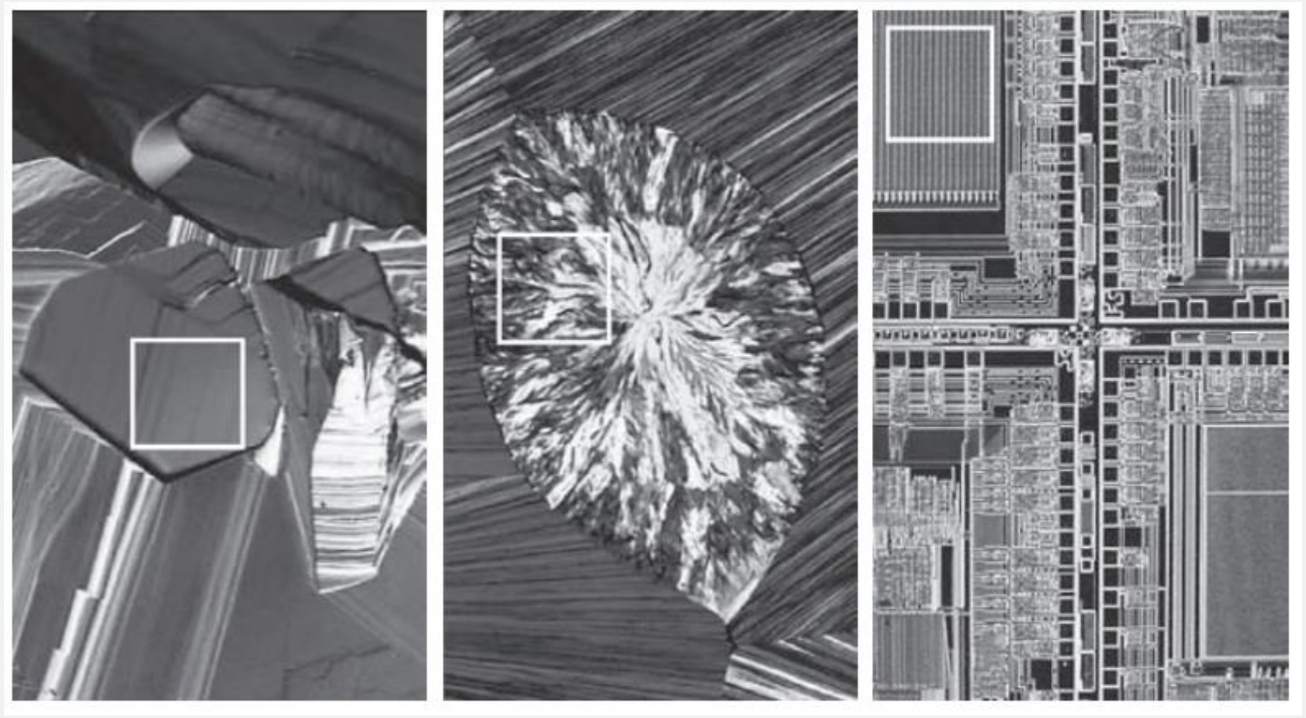




Feature Extraction

Basic region feature descriptors

- Texture
  - It is an important approach to region description for quantifying its texture content
  - Intuitively, smoothness, coarseness, and regularity are common texture descriptors while no formal definition of texture exists

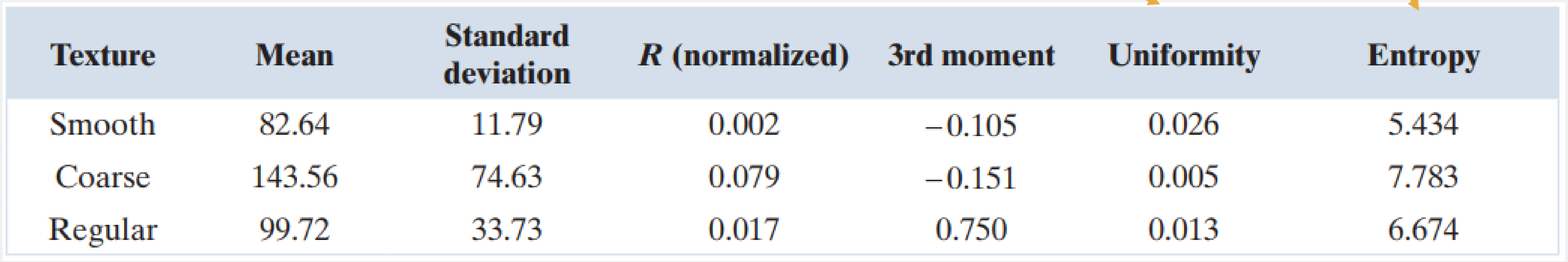


Feature Extraction

Basic region feature descriptors

- ➢ Texture
  - ◆ Statistical Approaches (estimated based on histogram)

$$\mu_n(z) = \sum_{i=0}^{L-1} (z_i - m)^n p(z_i) \qquad \mathrm{U}(z) = \sum_{i=0}^{L-1} p^2\,(zi) \qquad \mathrm{e}(z) = -\sum_{i=0}^{L-1} p\,(zi) log_2 p(zi)$$



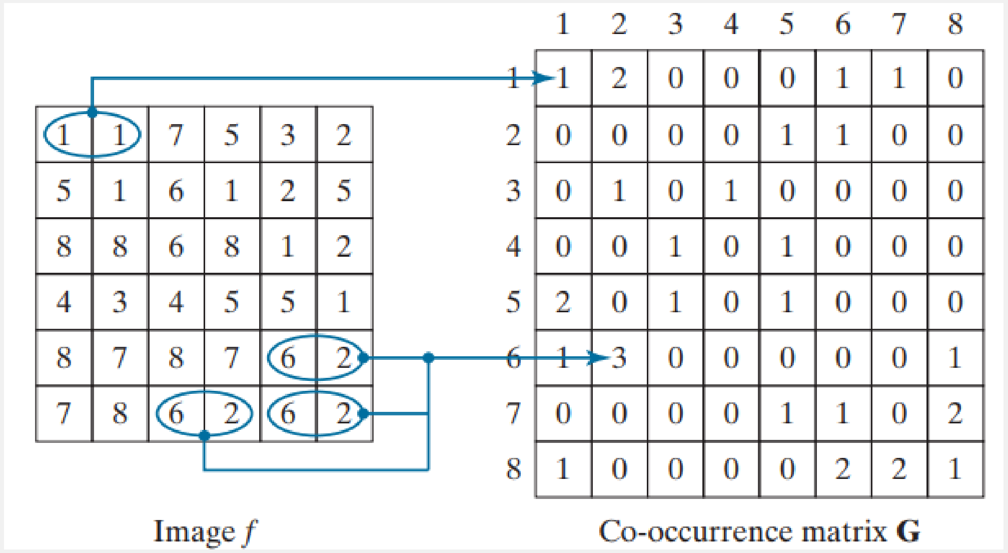
Texture	Mean	Standard deviation	*R* (normalized)	3rd moment	Uniformity	Entropy
Smooth	82.64	11.79	0.002	−0.105	0.026	5.434
Coarse	143.56	74.63	0.079	−0.151	0.005	7.783
Regular	99.72	33.73	0.017	0.750	0.013	6.674

Feature Extraction

However, measures of texture computed using only histograms carry no information regarding spatial relationships between pixels, which is important when describing texture

- ➢ Relative position
  - ◆ A co-occurrence matrix (G) with operator Q that defines the position of two pixels relative to each other
    - ✓ For an 8-bit image, G will be of size 256×256
  - ◆ An approach for reducing computation
    - ✓ Quantize the intensities into a few bands, such as letting the first 32 intensity levels equal to 1, the next 32 equal to 2, and so on

Q: one pixel immediately to its right



1	1	7	5	3	2
5	1	6	1	2	5
8	8	6	8	1	2
4	3	4	5	5	1
8	7	8	7	6	2
7	8	6	2	6	2

Image f

	1	2	3	4	5	6	7	8
1	1	2	0	0	0	1	1	0
2	0	0	0	0	1	1	0	0
3	0	1	0	1	0	0	0	0
4	0	0	1	0	1	0	0	0
5	2	0	1	0	1	0	0	0
6	1	3	0	0	0	0	0	1
7	0	0	0	0	1	1	0	2
8	1	0	0	0	0	2	2	1

Co-occurrence matrix **G**

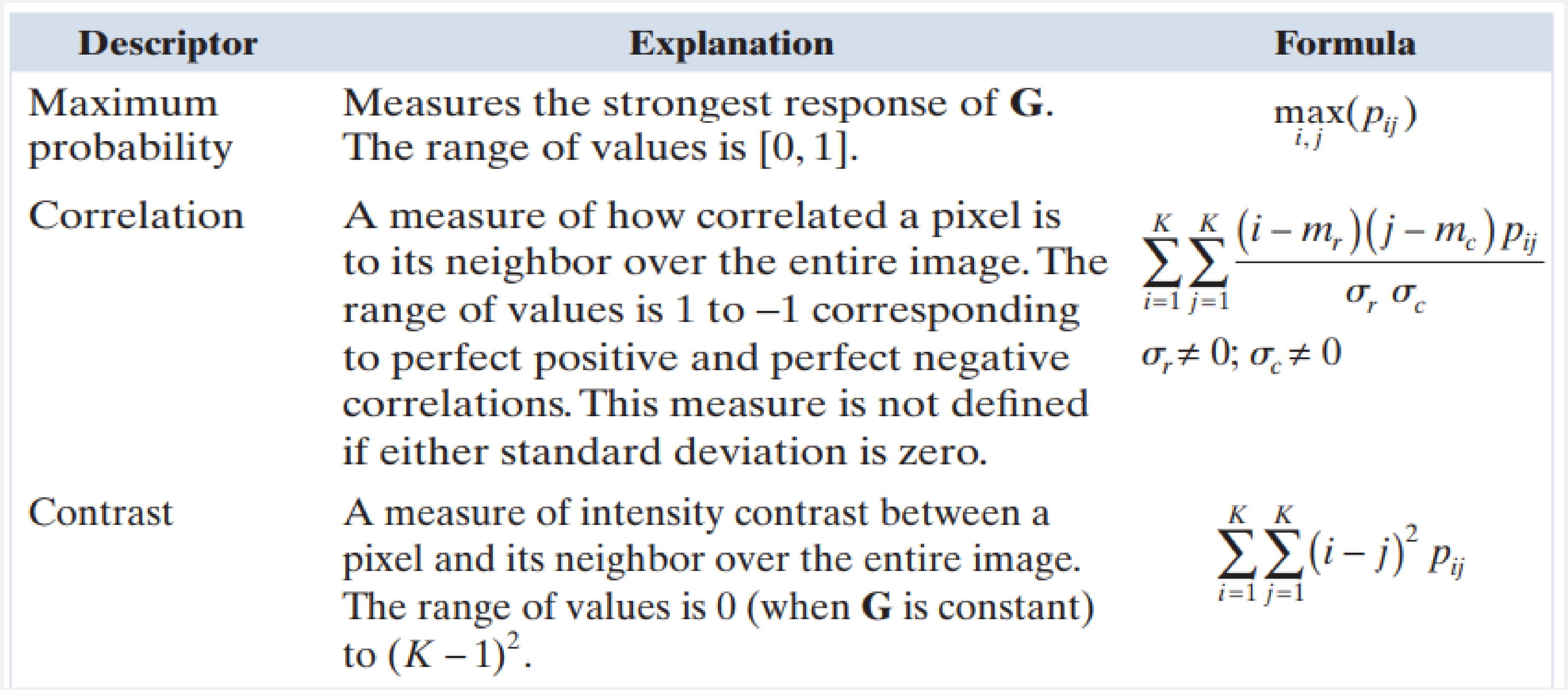
Feature Extraction

- Co-occurrence matrix
  - n: the total number of pixel pairs
  - Quantify

$$p_{ij} = \frac{g_{ij}}{n} \qquad \sum_{i=1}^{K}\sum_{j=1}^{K} p_{ij} = 1$$

$$m_r = \sum_{i=1}^{K} i \sum_{j=1}^{K} p_{ij} \qquad \sigma_r^2 = \sum_{i=1}^{K}(i - m_r)^2 \sum_{j=1}^{K} p_{ij}$$

$$m_c = \sum_{j=1}^{K} j \sum_{i=1}^{K} p_{ij} \qquad \sigma_c^2 = \sum_{j=1}^{K}(j - m_c)^2 \sum_{i=1}^{K} p_{ij}$$



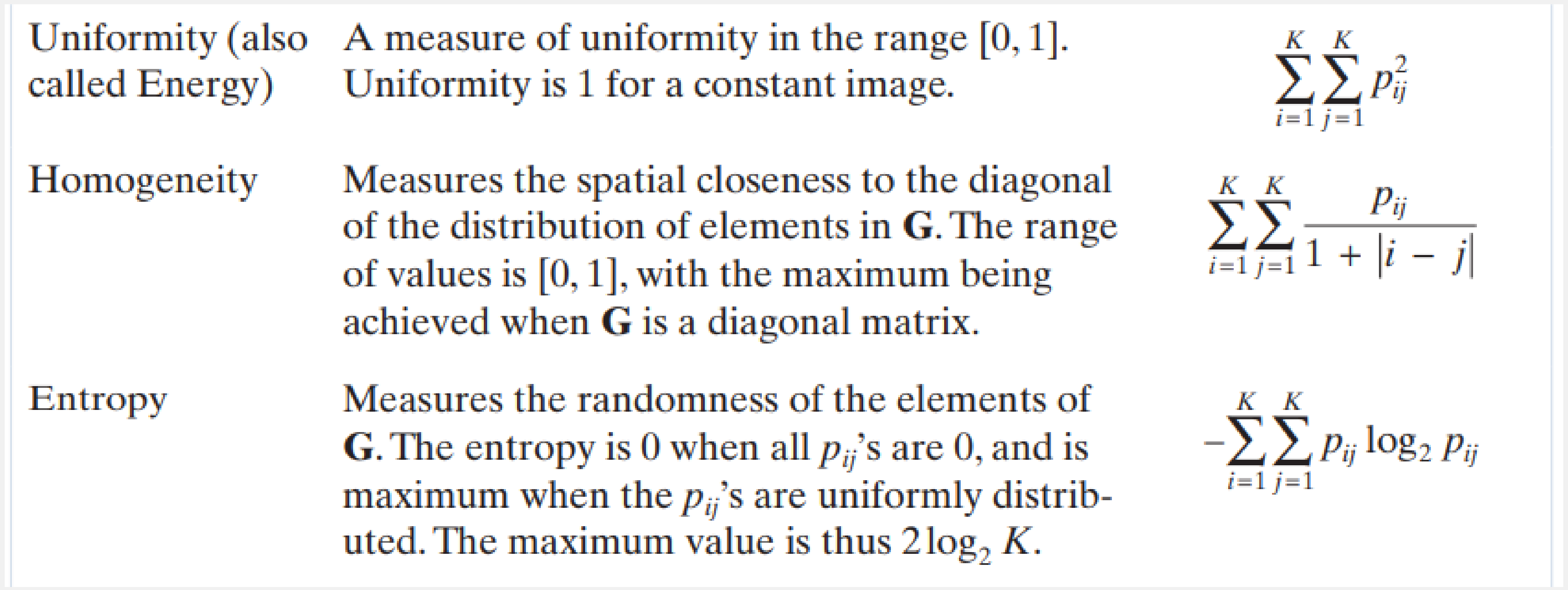
Descriptor	Explanation	Formula
Maximum probability	Measures the strongest response of **G**. The range of values is $[0, 1]$.	$\max_{i,j}(p_{ij})$
Correlation	A measure of how correlated a pixel is to its neighbor over the entire image. The range of values is 1 to -1 corresponding to perfect positive and perfect negative correlations. This measure is not defined if either standard deviation is zero.	$\sum_{i=1}^{K}\sum_{j=1}^{K} \frac{(i-m_r)(j-m_c)p_{ij}}{\sigma_r \sigma_c}$ $\sigma_r \neq 0;\ \sigma_c \neq 0$
Contrast	A measure of intensity contrast between a pixel and its neighbor over the entire image. The range of values is 0 (when **G** is constant) to $(K-1)^2$.	$\sum_{i=1}^{K}\sum_{j=1}^{K}(i-j)^2 p_{ij}$

Feature Extraction

- Co-occurrence matrix
  - n: the total number of pixel pairs
  - Quantify

$$m_r = \sum_{i=1}^{K} i \sum_{j=1}^{K} p_{ij} \qquad \sigma_r^2 = \sum_{i=1}^{K} (i - m_r)^2 \sum_{j=1}^{K} p_{ij}$$

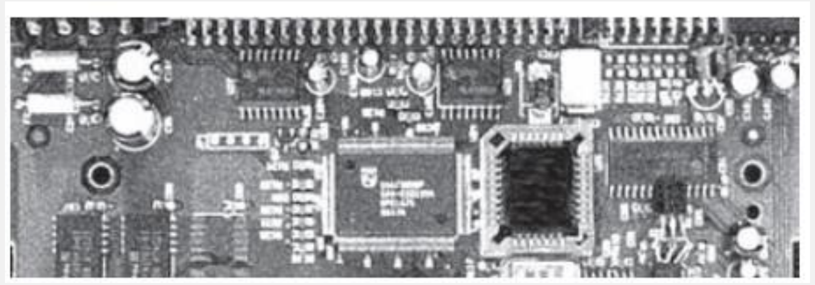
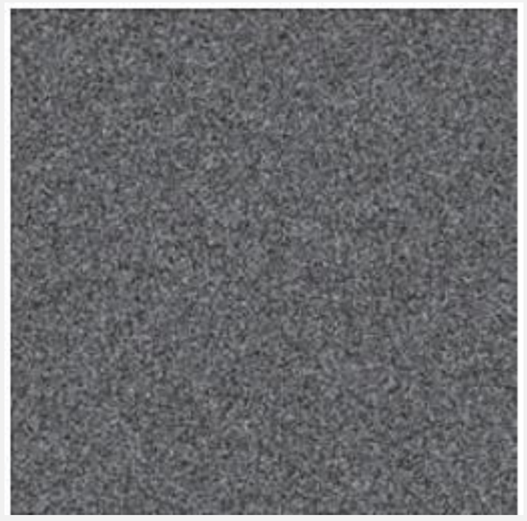
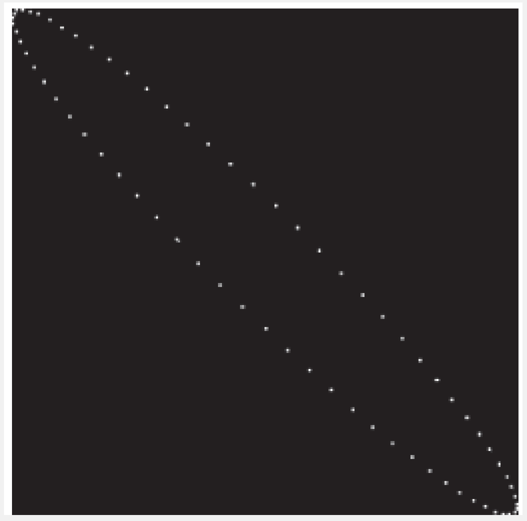
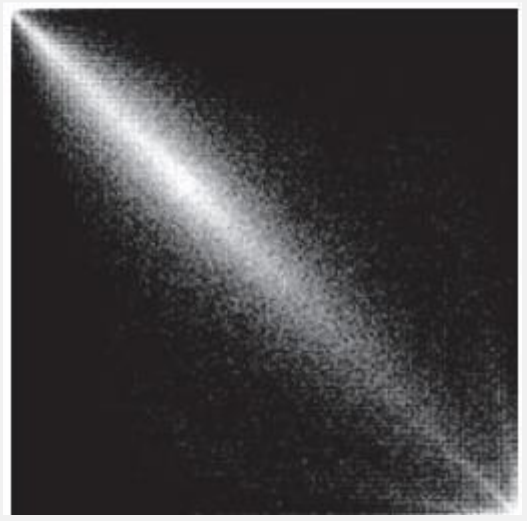
$$p_{ij} = \frac{g_{ij}}{n} \qquad \sum_{i=1}^{K} \sum_{j=1}^{K} p_{ij} = 1 \qquad m_c = \sum_{j=1}^{K} j \sum_{i=1}^{K} p_{ij} \qquad \sigma_c^2 = \sum_{j=1}^{K} (j - m_c)^2 \sum_{i=1}^{K} p_{ij}$$



Uniformity (also called 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 to the diagonal of the distribution of elements in **G**. The range of values is [0, 1], with the maximum being achieved when **G** is a diagonal matrix.	$\sum_{i=1}^{K} \sum_{j=1}^{K} \frac{p_{ij}}{1 + \lvert i - j \rvert}$
Entropy	Measures the randomness of the elements of **G**. The entropy is 0 when all p_{ij}'s are 0, and is maximum when the p_{ij}'s are uniformly distributed. The maximum value is thus $2\log_2 K$.	$-\sum_{i=1}^{K} \sum_{j=1}^{K} p_{ij} \log_2 p_{ij}$

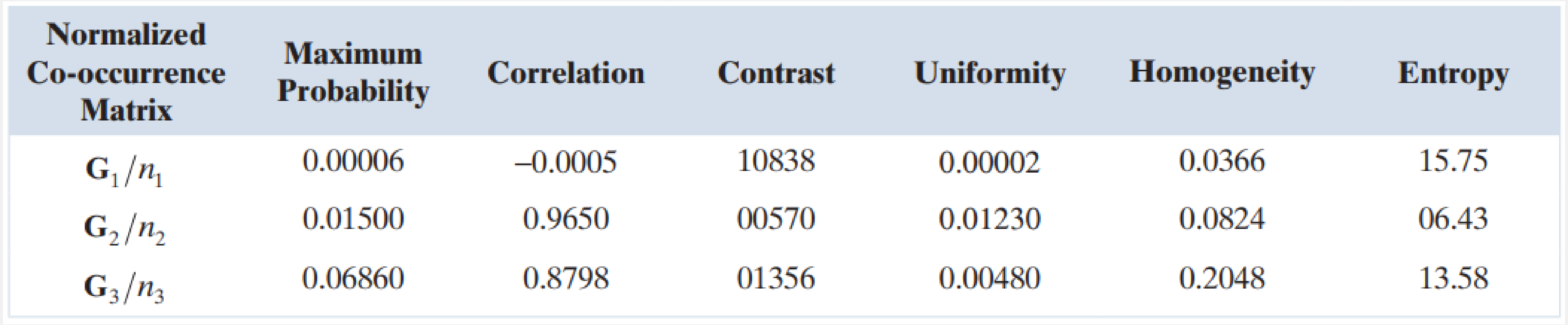
Feature Extraction

- Co-occurrence matrix

Feature Extraction

➢ Co-occurrence matrix

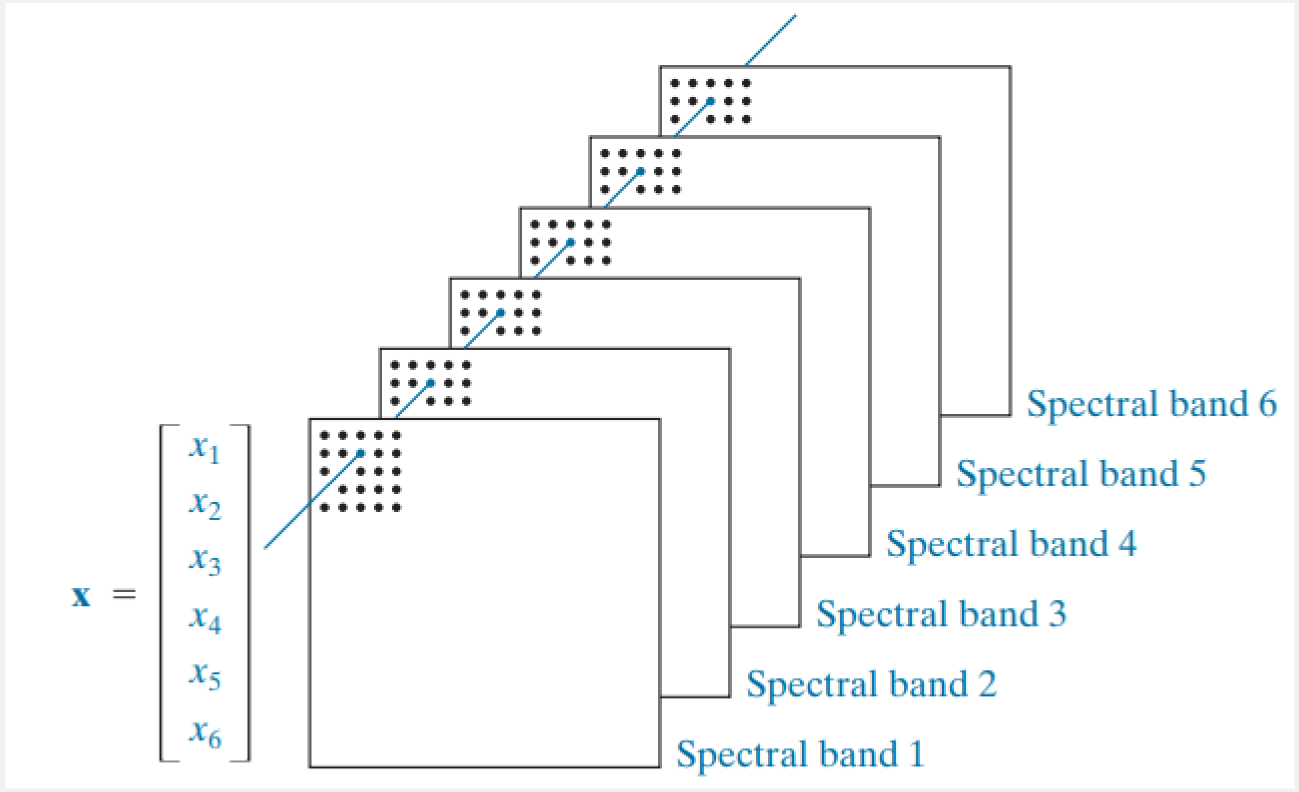


Normalized Co-occurrence Matrix	Maximum Probability	Correlation	Contrast	Uniformity	Homogeneity	Entropy
$\mathbf{G}_1/n_1$	0.00006	–0.0005	10838	0.00002	0.0366	15.75
$\mathbf{G}_2/n_2$	0.01500	0.9650	00570	0.01230	0.0824	06.43
$\mathbf{G}_3/n_3$	0.06860	0.8798	01356	0.00480	0.2048	13.58

Feature Extraction

Basic boundary and region feature descriptors

- Principal components analysis (PCA, Hotelling transform, eigenvector transform, or Karhunen–Loève transform (KL transform))



$$x = (x_1, x_2, \dots, x_n)^T$$

Mean vector: $m_x = E\{x\}$ where E{x} is the expected value of x

Covariance vector: $C_x = E\{(x - m_x)(x - m_x)^T\}$

Because x is n dimensional, C_x is an n×n matrix, and element c_{ii} of is the variance of x_i and element c_{ij} of is the covariance between x_i and x_j

If elements x_i and x_j are uncorrelated, their covariance is zero -> $c_{ij} = 0$

Feature Extraction

Basic boundary and region feature descriptors

- PCA
  - The purpose is to reduce the dimensionality of data by removing redundancy in the data, while maintaining the main information of the data

Because C_x is real and symmetric, finding a set of n orthonormal eigenvectors is always possible

$y = A(x - m_x)$ where A is a matrix whose rows are formed from the eigenvectors of C_x, arranged in descending values of their eigenvalues

$m_y = E\{y\} = 0$ $\quad C_y = AC_xA^T$ $\quad C_y = \begin{bmatrix} \lambda_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \lambda_n \end{bmatrix}$ where the main diagonal are the eigenvalues of C_x

The off-diagonal elements of this covariance matrix are 0, so the elements of the y vectors are uncorrelated, and C_x and C_y have the same eigenvalues

Feature Extraction

Basic boundary and region feature descriptors

- PCA
  - By hotelling transform, x can be reconstructed from y

$$y = A(x - m_x)$$

$$A^{-1}y = A^{-1}A(x - m_x)$$

$\because$ the rows of A are orthonormal vectors $\therefore$ $A^{-1}=A^T$

$$A^T y = A^T A(x - m_x) = x - m_x$$

$$x = A^T y + m_x$$

  - The vector reconstructed by using A_k is

$$\hat{x} = A_k^T y + m_x$$

  - The mean squared error between x and $\hat{x}$ is

$$e_{ms} = \sum_{j=1}^{n} \lambda_j - \sum_{j=1}^{k} \lambda_j = \sum_{j=k+1}^{n} \lambda_j$$

Selecting the k eigenvectors associated with the largest eigenvalues

Feature Extraction

Basic boundary and region feature descriptors

- How to calculate PCA
  1. Data standardization
     - Center the data by subtracting the mean, making the mean of the data zero, which eliminates the bias among different features
  2. Calculate the covariance matrix
     - Compute the covariance matrix of the data
  3. Compute eigenvalues and eigenvectors
  4. Select principal components
     - Sort the eigenvalues in descending order and select the most significant principal components
  5. Project the data

Feature Extraction

Basic boundary and region feature descriptors

- How to calculate PCA
  - Example
    1. Data standardization

Data

x	y
2	4
3	5
4	6
5	8

Centered data

x	y
-1.5	-1.75
-0.5	-0.75
0.5	0.25
1.5	2.25

$m_x=(2+3+4+5)/4 = 3.5$

$m_y=(4+5+6+8)/4 = 5.75$

Feature Extraction

Basic boundary and region feature descriptors

- How to calculate PCA
  - Example
    2. Covariance matrix

$$Cov(x,y) = \frac{1}{n-1}\sum_{i=1}^{n}(x_i' - m_{x'})(y_i' - m_{y'})$$

Centered data

x'	y'
-1.5	-1.75
-0.5	-0.75
0.5	0.25
1.5	2.25

$m_{x'}$=(-1.5-0.5+0.5+1.5)/4 = 0

$m_{y'}$=(-1.75-0.75+0.25+2.25)/4 = 0

Covariance matrix

$$Cov = \begin{bmatrix} Var(x') & Cov(x',y') \\ Cov(y',x') & Var(y') \end{bmatrix} = \begin{bmatrix} 1.6667 & 1.9167 \\ 1.9167 & 2.9167 \end{bmatrix}$$

Var(x')=((-1.5)^2+(-0.5)^2+(0.5)^2+(1.5)^2)/3=5/3=1.6667

Cov(x', y')=((-1.5)×(-1.75)+(-0.5)×(-0.75)+(0.5)×(0.25) +(1.5)×(2.25))/3=1.9167

Feature Extraction

Basic boundary and region feature descriptors

- How to calculate PCA
  - Example

3. Eigenvalues and eigenvectors of covariance matrix

$$Cov = \begin{bmatrix} 1.6667 & 1.9167 \\ 1.9167 & 2.9167 \end{bmatrix}$$

Eigenvalues (λ)

det(Cov-λI)=0

$$\begin{vmatrix} 1.6667-\lambda & 1.9167 \\ 1.9167 & 2.9167-\lambda \end{vmatrix} = 0$$

$(1.6667-\lambda)\times(2.9167-\lambda)-(1.9167)\times(1.9167)=0$

$(\lambda-4.3077)\times(\lambda-0.2757)=0$

$\lambda_1=4.3077$, $\lambda_2=0.2757$

Eigenvectors (ν)

(Cov-λI)ν=0

For $\lambda_1=4.3077$, $\nu^1=[\begin{smallmatrix} v_x^1 \\ v_y^1 \end{smallmatrix}]$, ($Cov$-4.3077I)$\nu^1$ =0

$(1.6667-4.3077)v_x^1+1.9167v_y^1=0$
$1.9167v_x^1+(2.9167-4.3007)v_y^1=0$

$$v_y^1 = \frac{2.641}{1.9167}v_x^1 = 1.3779v_x^1$$

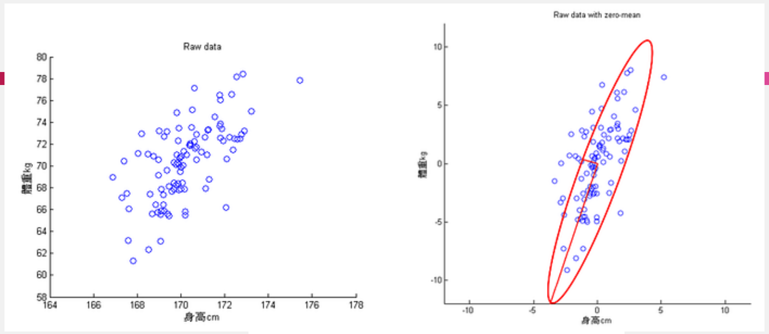
$$\nu^1=\begin{bmatrix} v_x^1 \\ v_y^1 \end{bmatrix} = \begin{bmatrix} 0.5874 \\ 0.8093 \end{bmatrix}$$

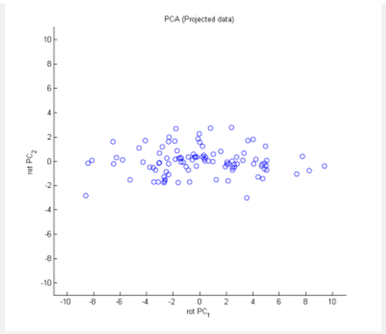
Feature Extraction

Basic boundary and region feature descriptors

- How to calculate PCA
  - Example
    4. Select the most significant principal components: λ_1=4.3077 and ν^1
    5. Project the data

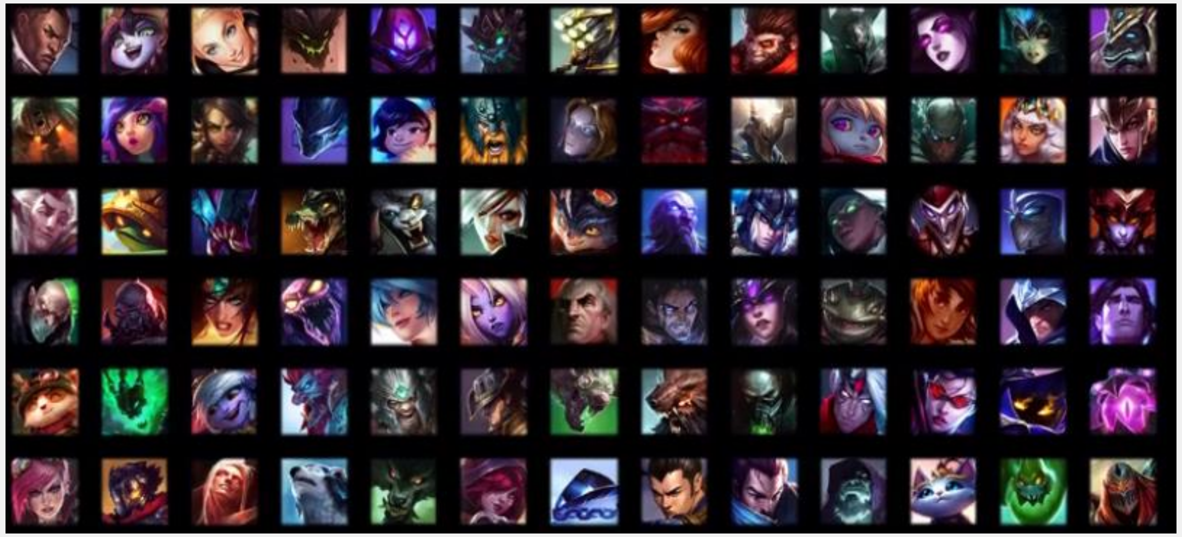
$$\text{Project} = \text{Data} \times \nu^1$$



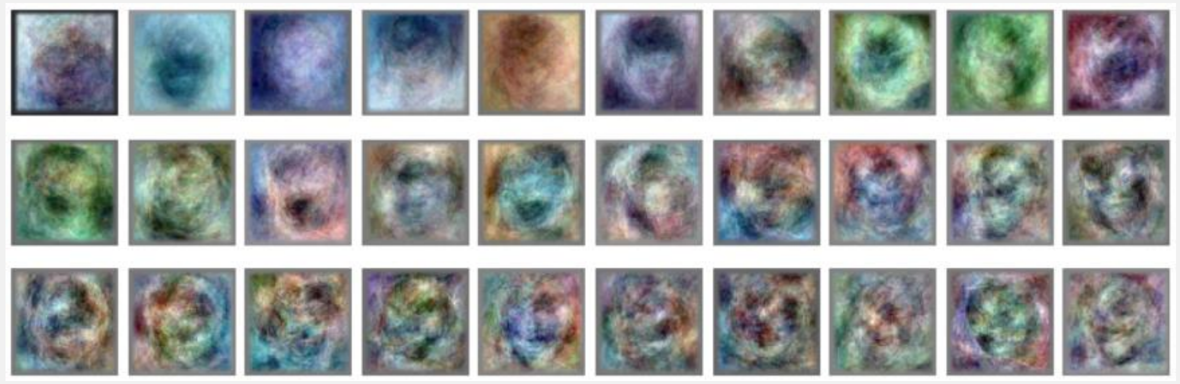


Link

Data



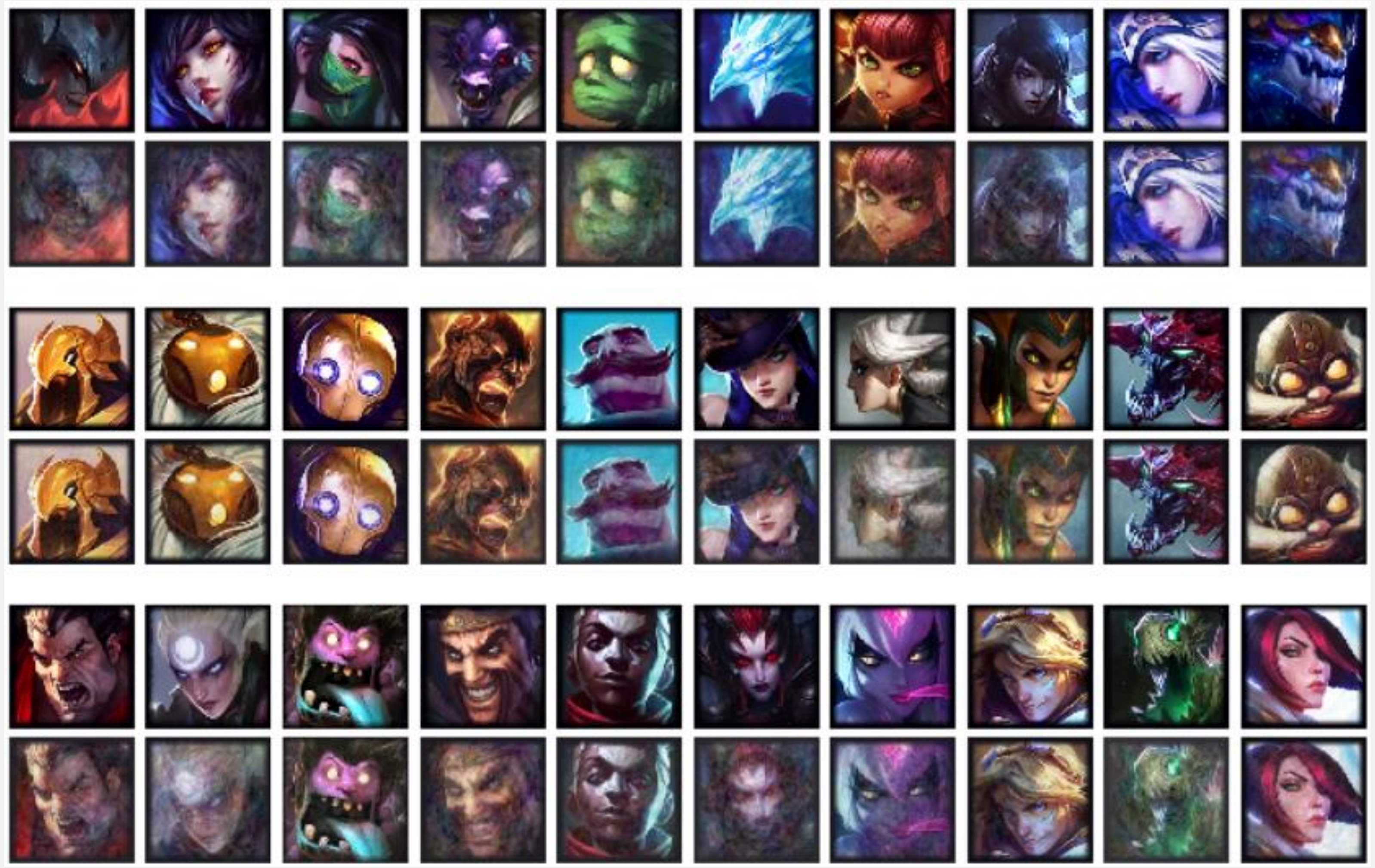
Project data with 30 principal components



Link

Feature Extraction

Reconstruct with 108 principal components

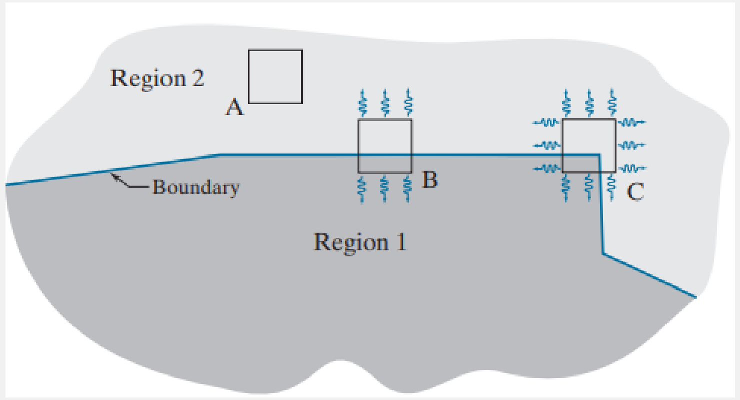


Link

Feature Extraction

Basic whole image feature descriptors

- Harris-Stephens corner detector
  - Basic approach: : Corners are detected by running a small window over an image and the window is designed to compute intensity changes
  - Three scenarios:
    1. Areas of zero (or small) intensity changes in all directions, which happens when the window is located in a constant (or nearly constant) region, as in location A
    2. Areas of changes in one direction but no (or small) changes in the orthogonal direction, which this happens when the window spans a boundary between two regions, as in location B
    3. Areas of significant changes in all directions, a condition that happens when the window contains a corner (or isolated points), as in location C



Feature Extraction

- Harris-Stephens corner detector

f is an image and f(s, t) is a patch of the image defined by the values of (s, t)

A patch of the same size, but shifted by (x, y), is given by f(s+x, t+y)

The weighted sum of squared difference between the two patches is given by

$$C(x,y) = \sum_s \sum_t \omega(s,t)[f(s+t, t+y) - f(s,t)]^2 \ \ where\ w(s,t)\ is\ a\ weighting\ function$$

$$f(s+x, t+y) \approx f(s,t) + x f_x(s,t) + y f_y(s,t)\ by\ Taylon\ expansion$$

$$f(x_0+\Delta x, y_0+\Delta y) = f(x_0, y_0) + \frac{\partial f}{\partial x}\Delta x + \frac{\partial f}{\partial y}\Delta y$$

$$C(x,y) = \sum_s \sum_t \omega(s,t)[x f_x(s,t) + y f_y(s,t)]^2$$

$$C(x,y) = [x\ y] M \begin{bmatrix} x \\ y \end{bmatrix}\ where\ M = \sum_s \sum_t \omega(s,t) A\ \ and\ \ A = \begin{bmatrix} f_x^2 & f_x f_y \\ f_x f_y & f_y^2 \end{bmatrix}$$

Harris matrix

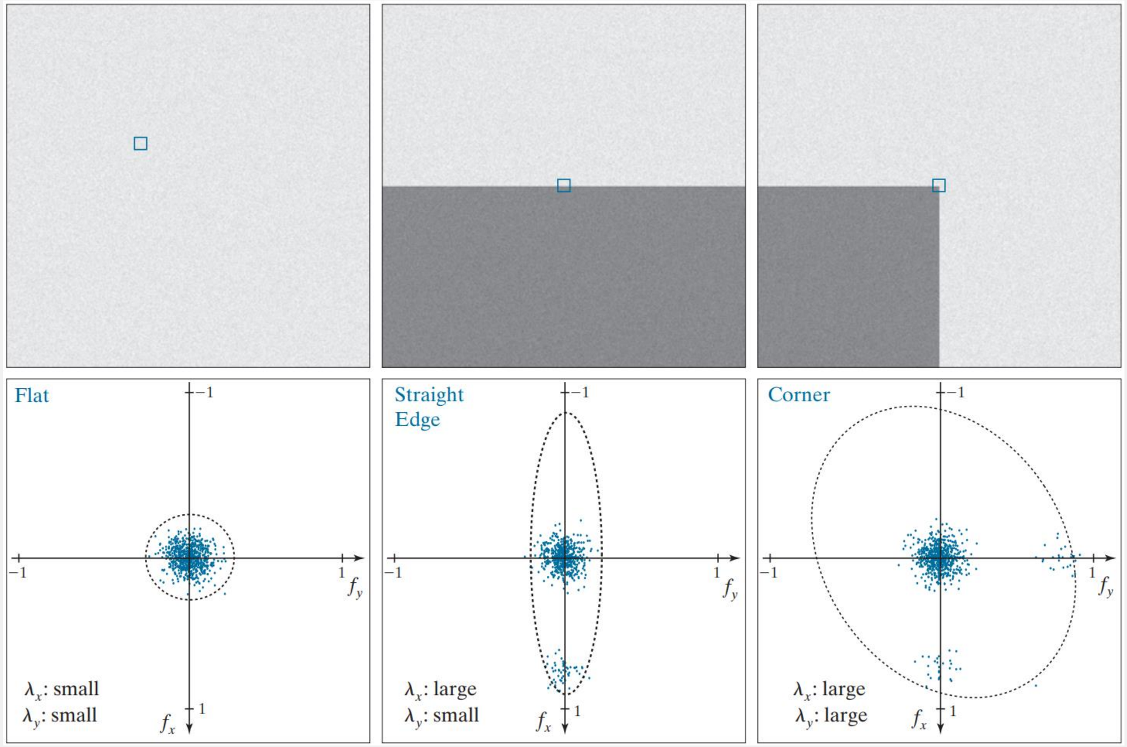
In general, ω has one of two forms:

1) 1 inside in the patch and 0 elsewhere: when computational speed is paramount and the noise level is low

2) $\omega(s,t) = e^{-(s^2+t^2)/2\sigma^2}$: when data smoothing is important

Feature Extraction

- Harris-Stephens corner detector
  - As discussed in PCA, the eigenvector of a real, symmetric matrix point in the direction of maximum data spread, and the corresponding eigenvalues are proportional to the amount of data spread in the direction of the eigenvectors
  - The eigenvectors are the major axes of an ellipse fitting the data, and the magnitude of the eigenvalues are the distances from the center of the ellipse to the points where it intersects the major axes



Feature Extraction

- Harris-Stephens corner detector
  - However, instead of using the eigenvalues (which are expensive to compute), the HS detector utilizes a measure of corner response based on the fact that the trace of a square matrix is equal to the sum of its eigenvalues, and its determinant is equal to the product of its eigenvalues

$$R=\lambda_x\lambda_y-k(\lambda_x+\lambda_y)^2=\det(M)-k\times \mathrm{trace}^2(M) \quad \text{where } k \text{ is a constant}$$

a) R is large positive values when both eigenvalues are large -> corner

b) R is large negative values when one eigenvalue is large and the other small -> edge

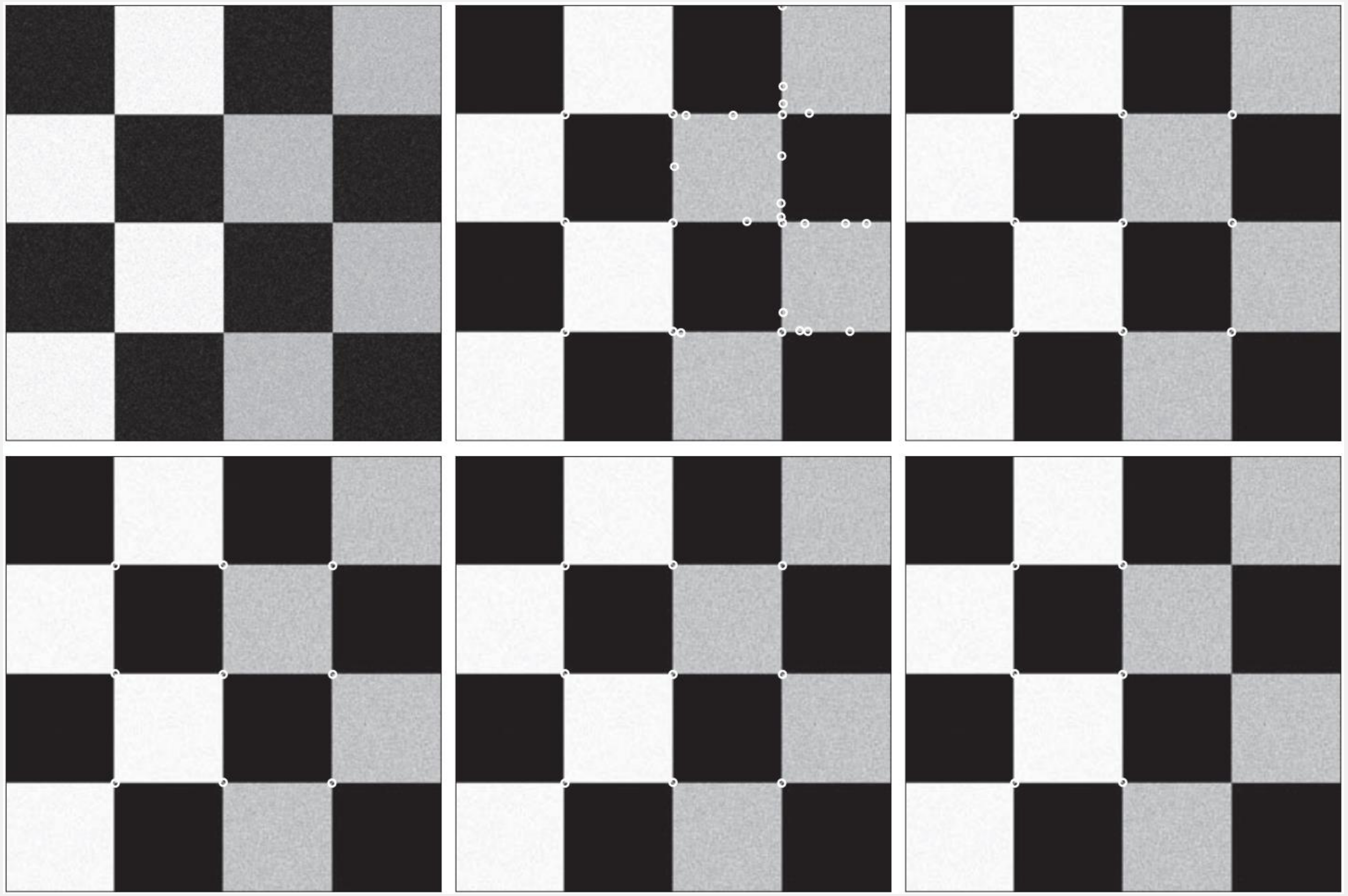
c) The absolute of R is small when both eigenvalues are small -> flat

- Constant k is determined empirically, the smaller it is, the more likely the detector is to find corners
- A corner at an image location has been detected only if R > T, where T is threshold

Feature Extraction

- Harris-Stephens corner detector

k=0.04, T=0.01 k=0.1, T=0.01

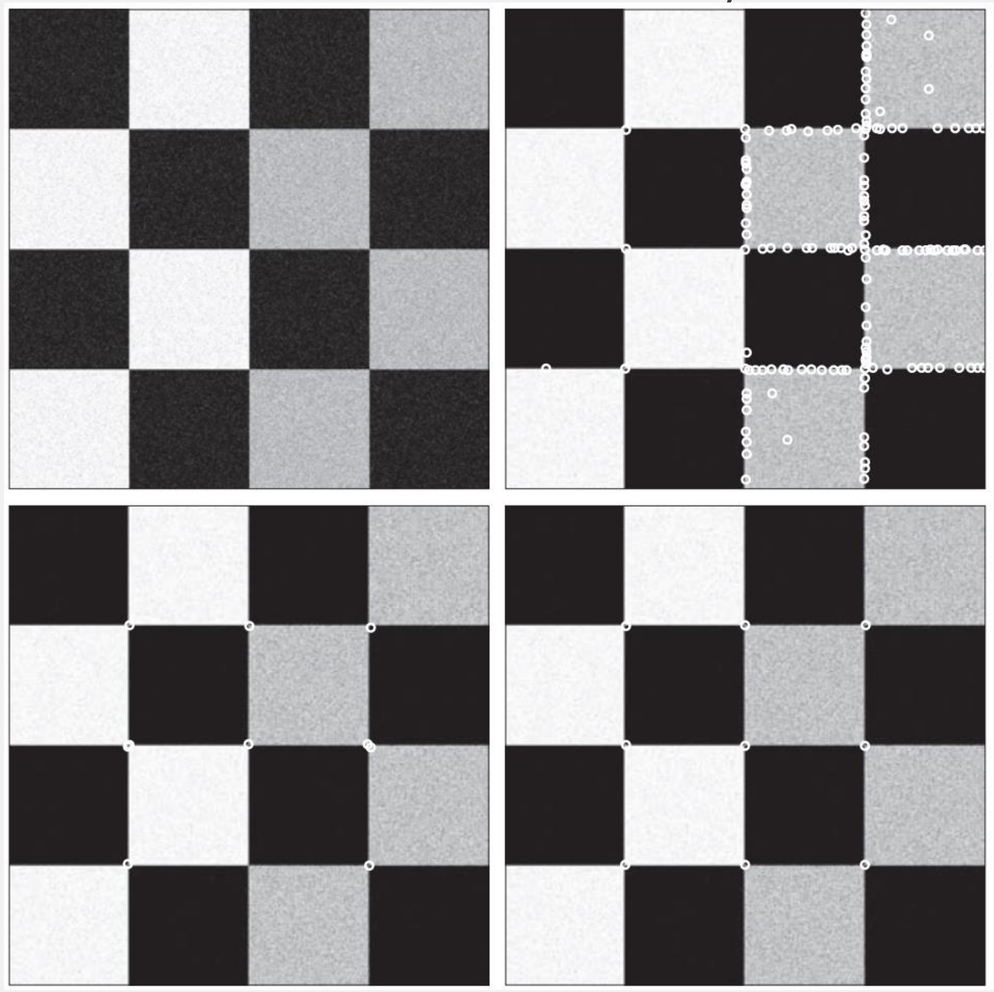


k=0.1, T=0.1 k=0.04, T=0.1 k=0.04, T=0.3

Feature Extraction

- Harris-Stephens corner detector

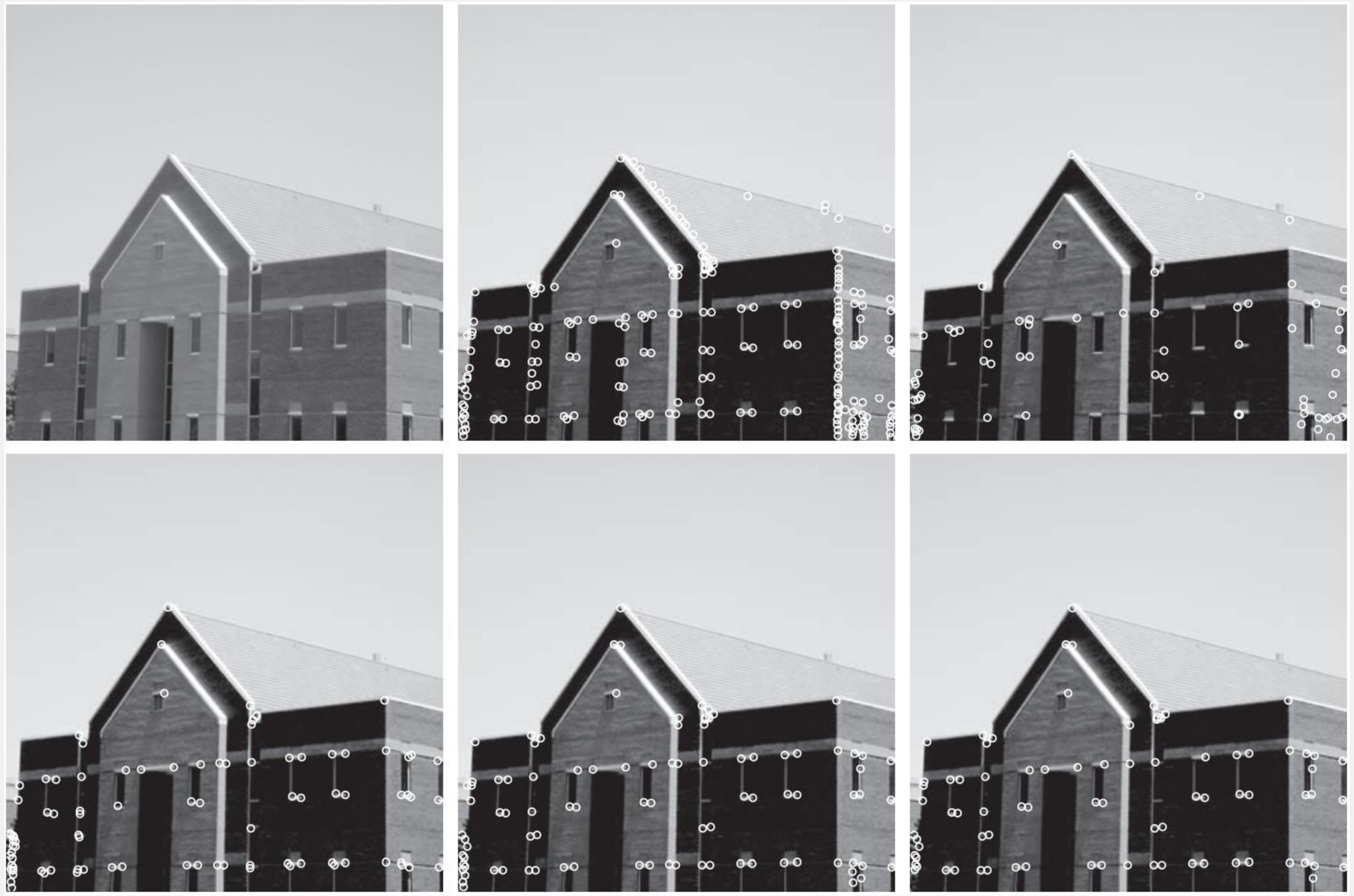
k=0.04, T=0.01



k=0.249, T=0.01 k=0.04, T=0.15

Feature Extraction

- Harris-Stephens corner detector



Feature Extraction

Basic whole image feature descriptors

- Scale-invariant feature transform (SIFT)
    - It transforms image data into scale-invariant coordinates for extracting invariant features from an image
    - SIFT features (called keypoints) are invariant to image scale and rotation, and are robust across a range of affine distortions, changes in 3-D viewpoint, noise, and changes of illumination
    - The input is an image and the out is an n-dimensional feature vector whose elements are the invariant feature descriptors

Feature Extraction

- SIFT algorithm
  1. Construct the scale space
  2. Obtain the initial keypoints by local extrema
  3. Improve the accuracy of the location of the keypoints
  4. Compute keypoint orientations
  5. Compute keypoint descriptors

Feature Extraction

- SIFT algorithm
  1. Construct the scale space
     - Find image locations that are invariant to scale change by searching for stable features across all possible scales using a scale function (i.e. scale space)
     - Scale space is a multi-scale representation suitable for handling image structures at difference scales in a consistent manner
     - Scale space represents an image as a one-parameter (scale parameter) family of smoothed images, with the objective of simulating the loss of detail that would occur as the scale of an image decreases

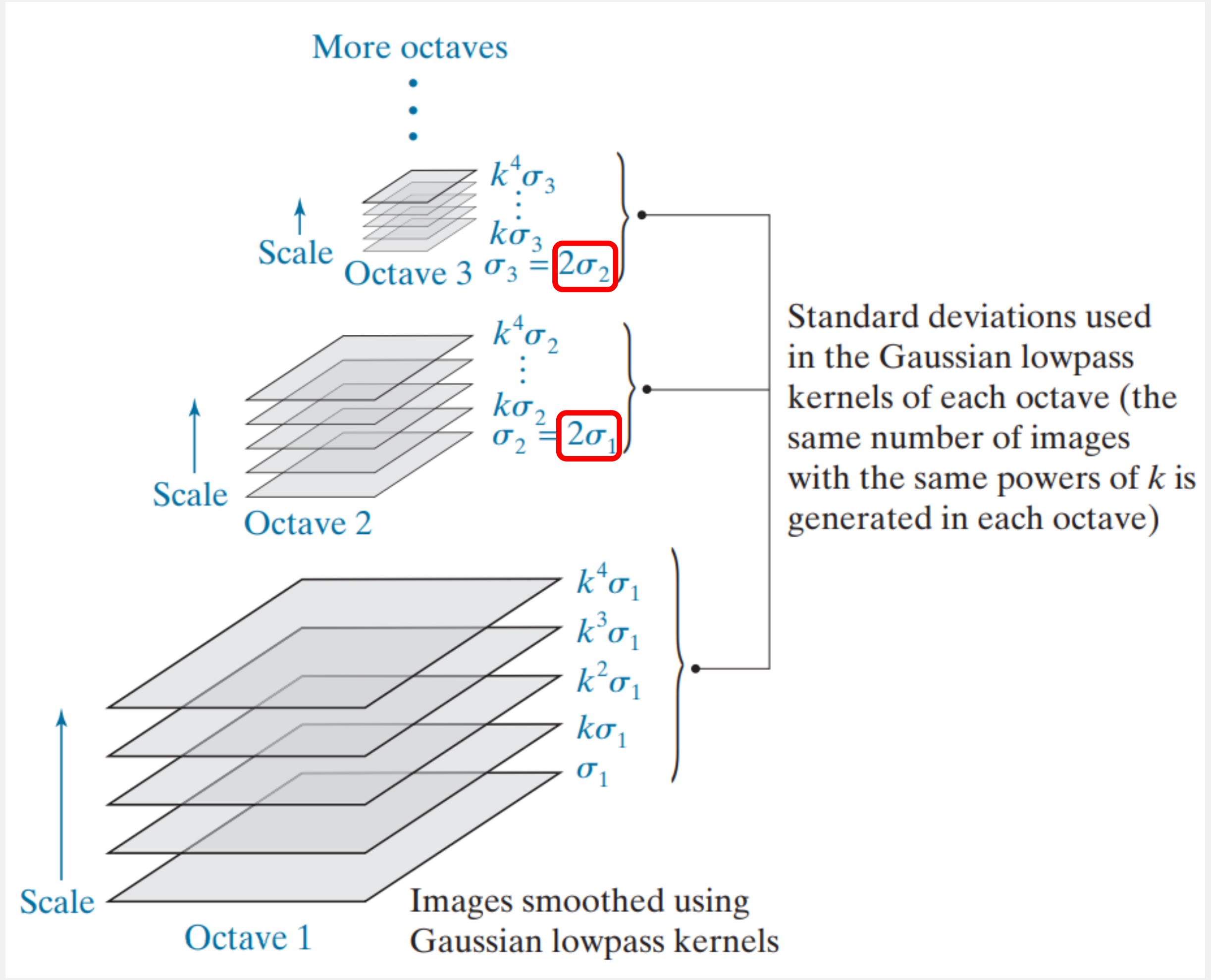
$$L(x,y,\sigma) = G(x,y,\sigma) \bigstar f(x,y)$$

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

     - f(x, y) is successively convolved with Gaussian kernels have standard deviations σ, $k\sigma$, $k^2\sigma$, $k^3\sigma$, … to generate a "stack" of Gaussian-filtered (smoothed) images

Feature Extraction

- f(x, y) is successively convolved with Gaussian kernels have standard deviations σ, $k\sigma$, $k^2\sigma$, $k^3\sigma$, … to generate a "stack" of Gaussian-filtered (smoothed) images



More octaves
$k^4\sigma_3$
$k\sigma_3$
$\sigma_3 = \boxed{2\sigma_2}$
Scale
Octave 3
$k^4\sigma_2$
$k\sigma_2$
$\sigma_2 = \boxed{2\sigma_1}$
Scale
Octave 2
Standard deviations used in the Gaussian lowpass kernels of each octave (the same number of images with the same powers of k is generated in each octave)
$k^4\sigma_1$
$k^3\sigma_1$
$k^2\sigma_1$
$k\sigma_1$
σ_1
Scale
Octave 1
Images smoothed using Gaussian lowpass kernels

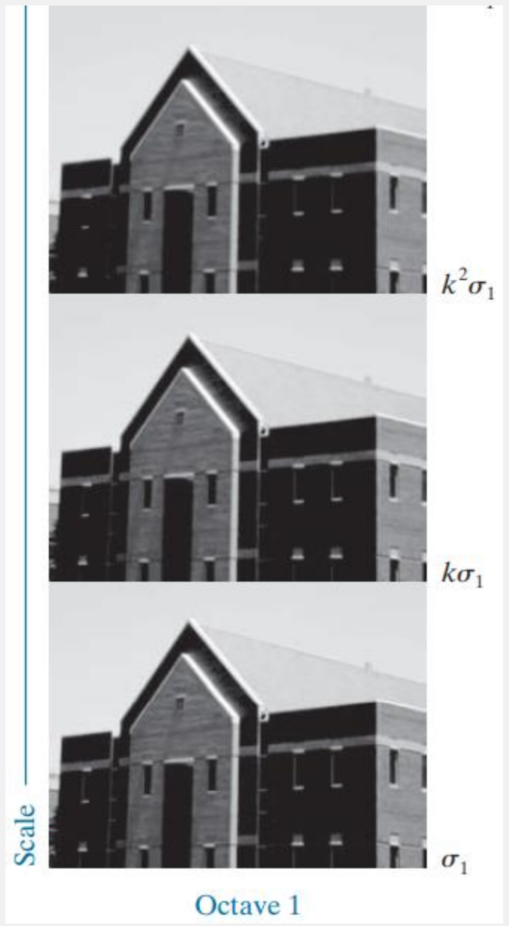
Scale space is subdivide into octaves, with each octave corresponding to a doubling of σ

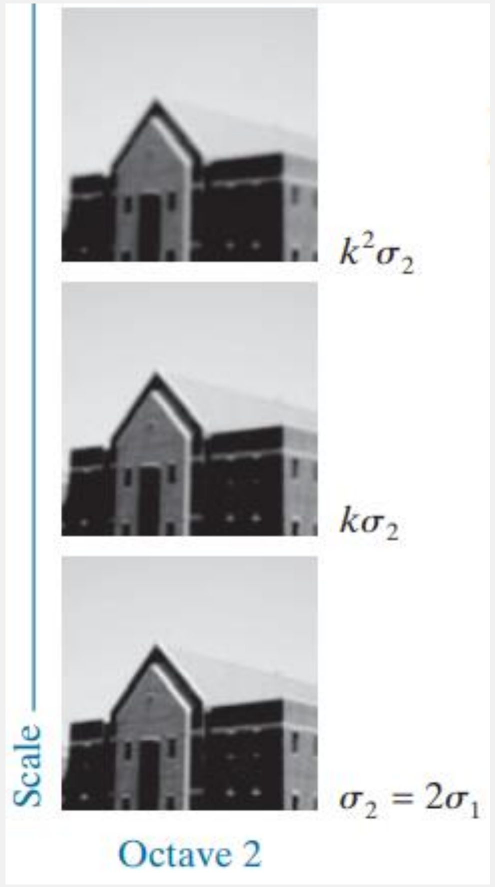
Each octave is subdivide into an integer number, s, of intervals, so that an interval of 1 consists of two images, an interval of 2 consists of three images because these smoothed images will be used to compute differences of Gaussians

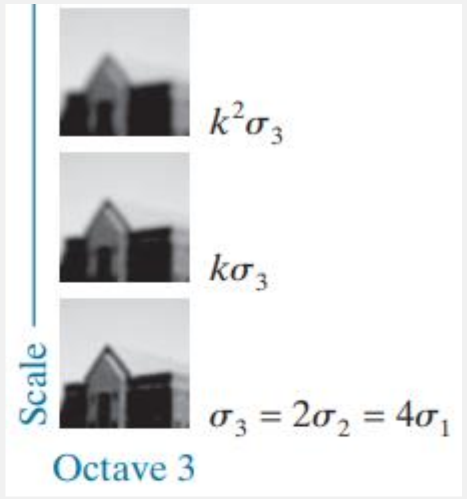
The size of image in next octave is formed by downsampling the previous image, and then smoothing it using a kernel with twice the standard deviation used in the previous octave

Feature Extraction

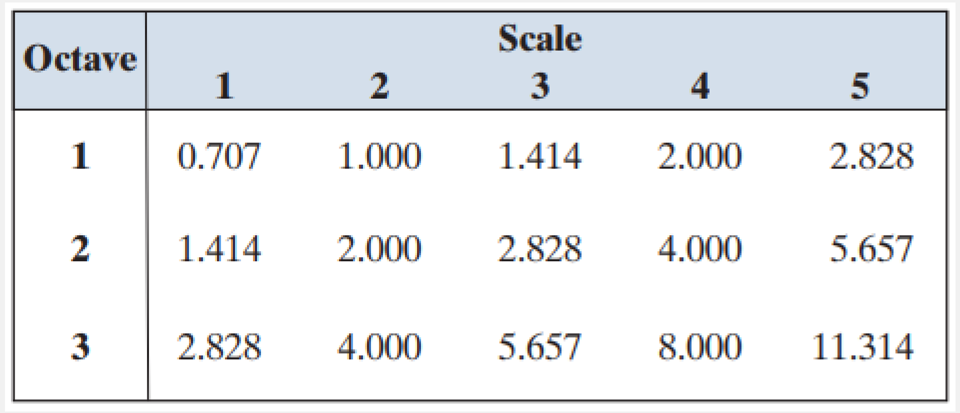
- f(x, y) is successively convolved with Gaussian kernels have standard deviations σ, kσ, k^2σ, k^3σ, … to generate a "stack" of Gaussian-filtered (smoothed) images







$$\sigma_1 = \sqrt{2}/2 = 0.707 \qquad k = \sqrt{2} = 1.414$$



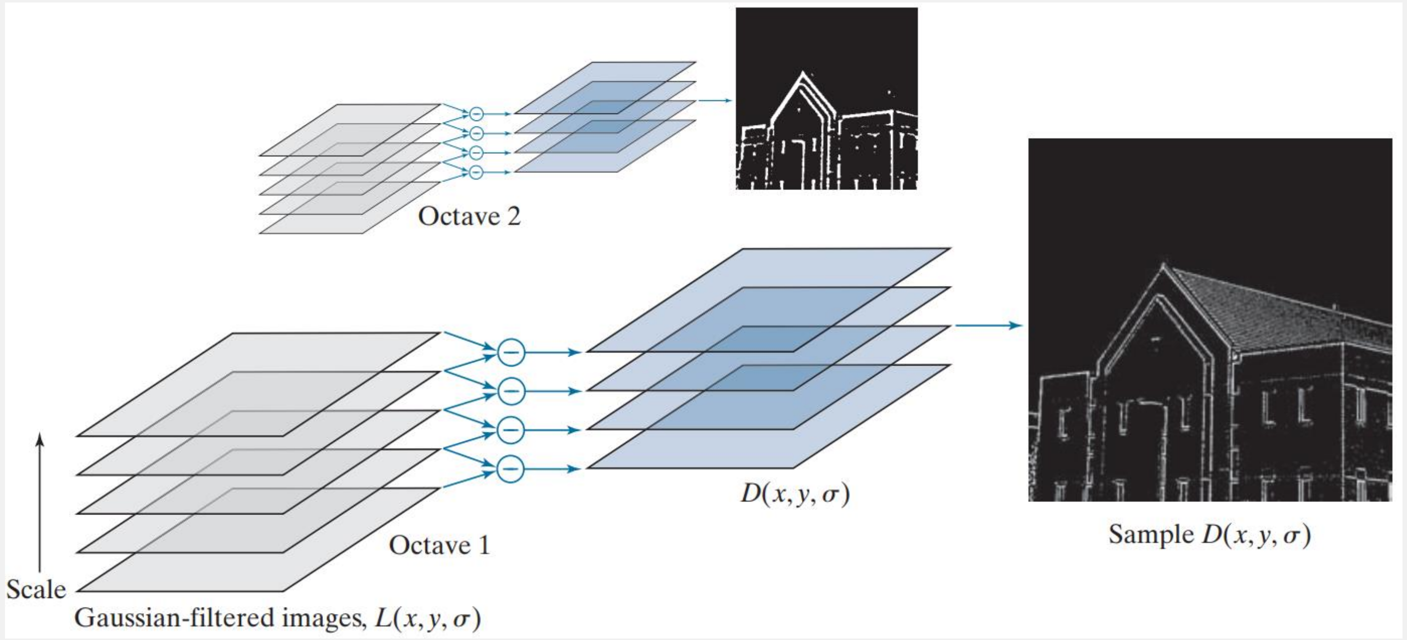
Octave	Scale 1	2	3	4	5
1	0.707	1.000	1.414	2.000	2.828
2	1.414	2.000	2.828	4.000	5.657
3	2.828	4.000	5.657	8.000	11.314

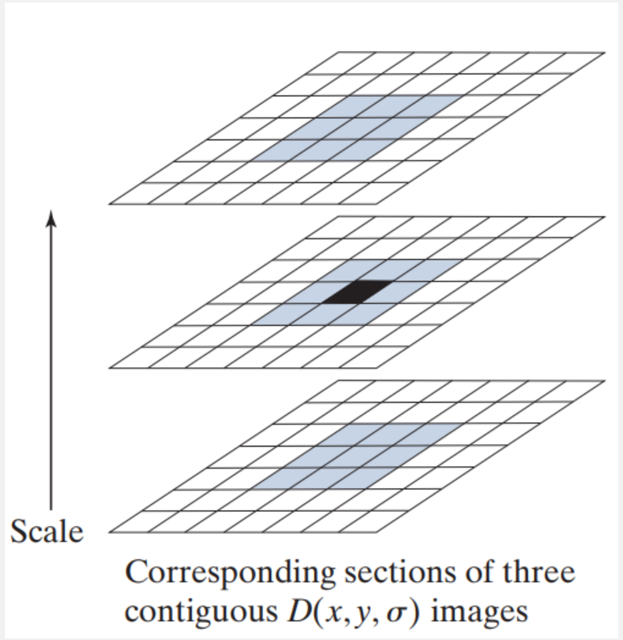
Feature Extraction

- SIFT algorithm
  2. Obtain the initial keypoints by local extrema
     - Using the difference of Gaussians (DoGs) of two adjacent scale-space images in an octave, convolved with the input image that corresponds to that octave

$$D(x, y, \sigma) = [G(x, y, k\sigma) - G(x, y, \sigma)] \bigstar f(x, y)$$
$$G(x, y, k\sigma) - G(x, y, \sigma) \approx (k - 1)\sigma^2 \nabla^2 G$$

A point is select as keypoint if the value is largest or smallest than its eight neighbors in the current image and its nine neighbors in the images above and below





Scale
Corresponding sections of three contiguous $D(x, y, \sigma)$ images

Feature Extraction

- SIFT algorithm
  3. Improve the accuracy of the location of the keypoints
     a. if $D(\hat{x}) = D + \frac{1}{2}(\nabla D)^T \hat{x}$, where ∇D is the differential of D, is low, then it is rejected (eliminated)
     b. if $\frac{[Tr(H)]^2}{\mathrm{Det}(H)} < \frac{(r+1)^2}{r}$, where r is a threshold and H is the Hessian matrix, is true, then it is preserved

In function a, the ∇D is defined as

$$\nabla D = \frac{\partial D}{\partial x} = \begin{bmatrix} \partial D/\partial x \\ \partial D/\partial y \\ \partial D/\partial \sigma \end{bmatrix}$$

In function b, the Tr(H) and Det(H) are defined as

$$H = \begin{bmatrix} \partial^2 D/\partial x^2 & \partial^2 D/\partial x \partial y \\ \partial^2 D/\partial y \partial x & \partial^2 D/\partial y^2 \end{bmatrix} = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{yx} & D_{yy} \end{bmatrix}$$

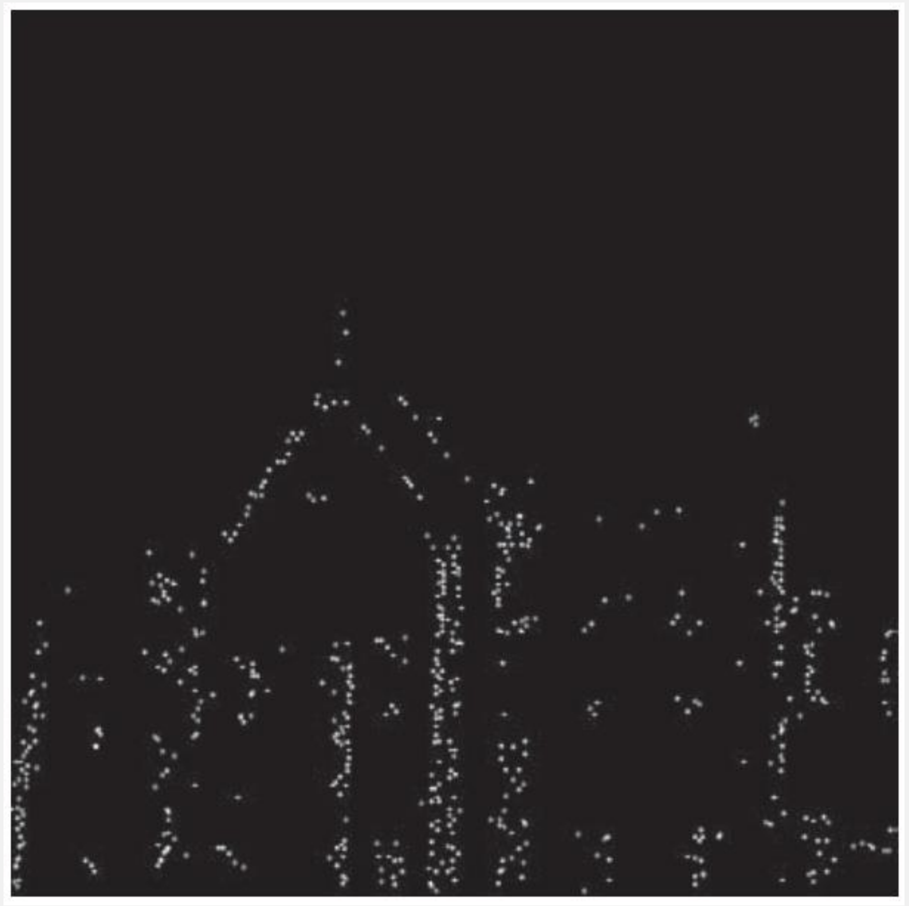
$$Tr(H) = D_{xx} + D_{yy} = \alpha + \beta$$

$$Det(H) = D_{xx}D_{yy} - (D_{xu})^2 = \alpha\beta$$

where α and β are the eigenvalues of H with the largest and smallest magnitude

Feature Extraction

- SIFT algorithm
  3. Improve the accuracy of the location of the keypoints
     - If $D(\hat{x}) < 0.03\ and\ r = 10$



Feature Extraction

- SIFT algorithm
  4. Compute keypoint orientations
     - Based on L(x, y), the gradient magnitude M(x, y), and orientation angle θ are computed using

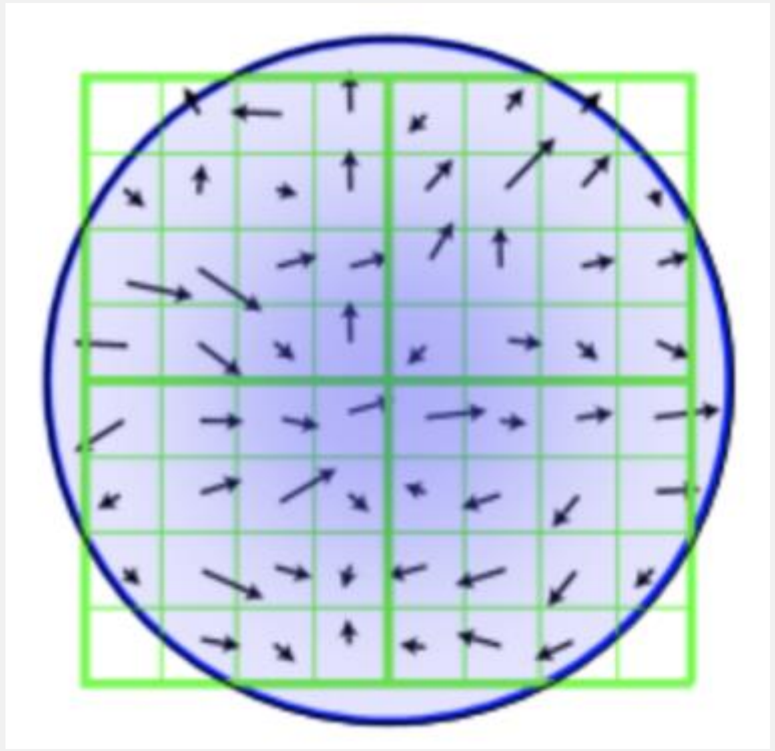
$$M(x, y) = [(L(x+1, y) - L(x-1, y))^2+(L(x, y+1) - L(x, y-1))^2]^{1/2}$$

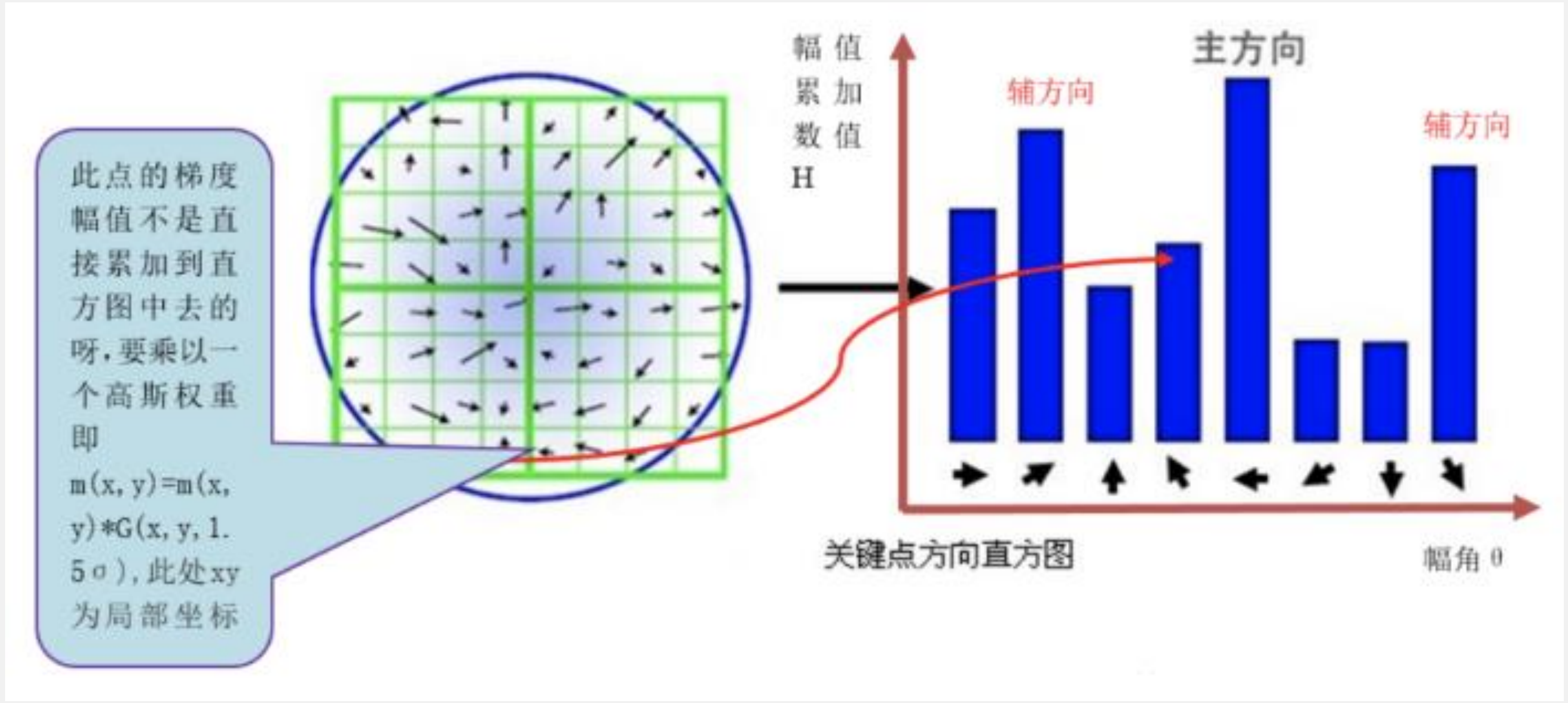
$$\theta(x, y)= \tan^{-1}[(L(x, y+1) - L(x, y-1))/L(x+1, y) - L(x-1, y))]$$

     - Then, a histogram of orientations is formed from the gradient orientations of simple points in a neighborhood of each keypoint
     - Each sample added to the histogram is weighed by its gradient magnitude, and by a circular Gaussian function with a standard deviation 1.5 times the scale of the keypoint

Feature Extraction

- SIFT algorithm
  4. Compute keypoint orientations

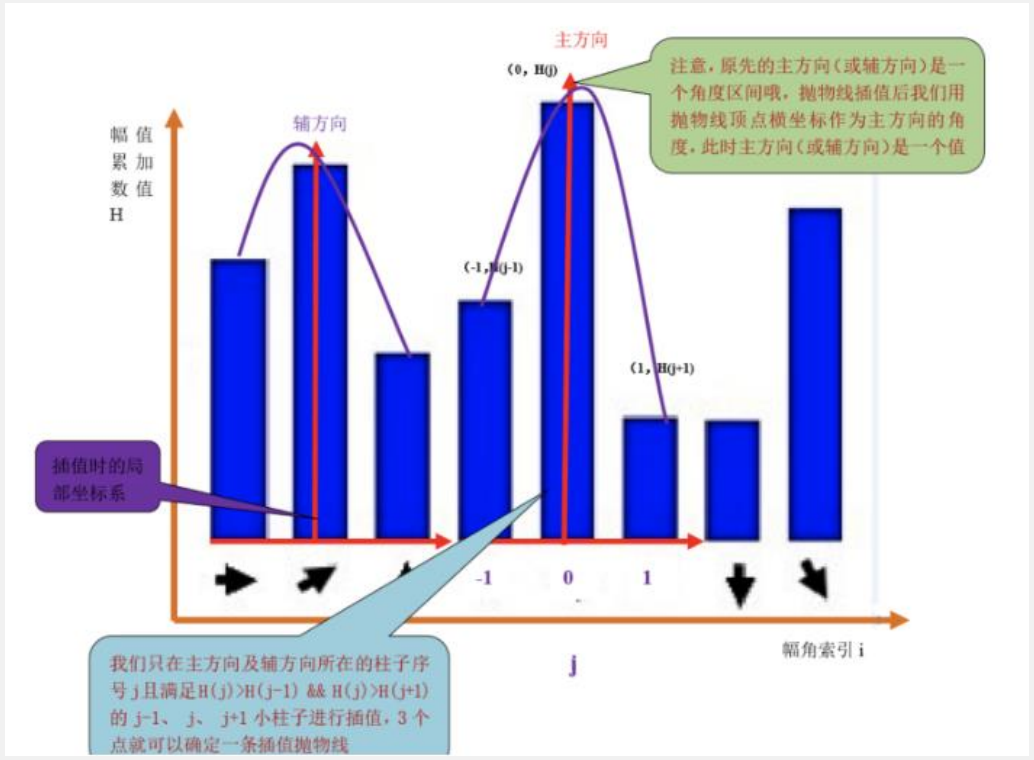


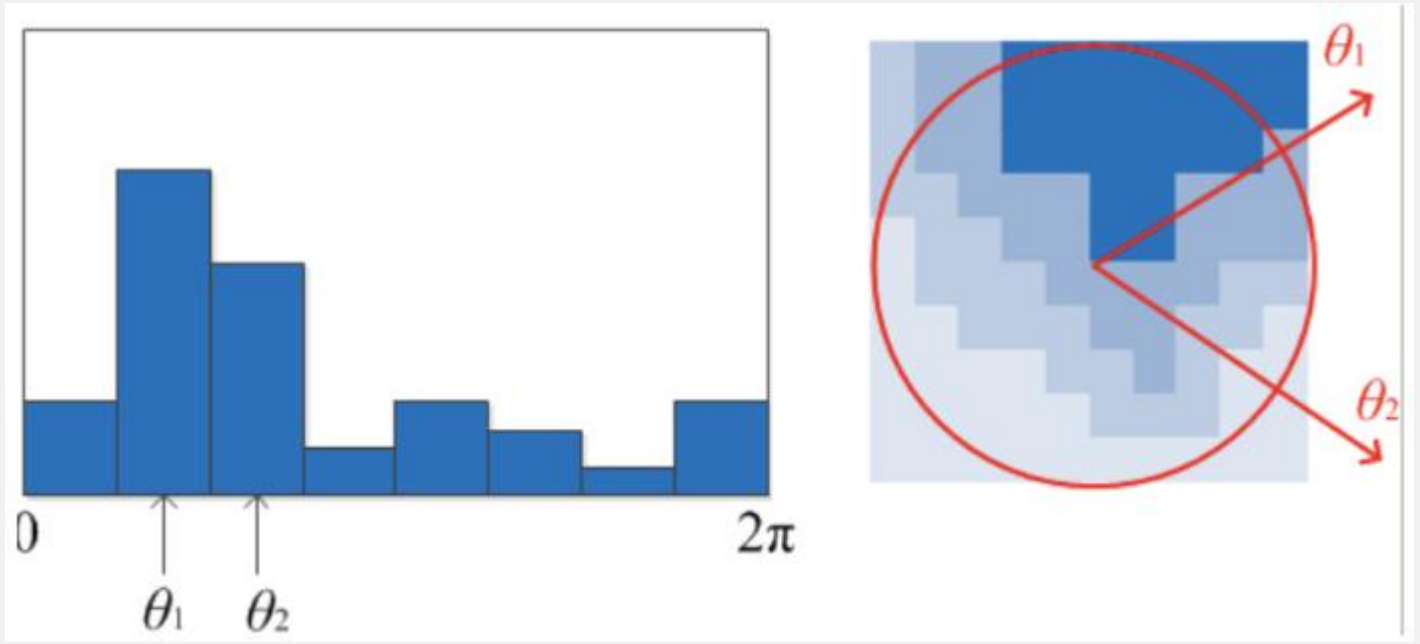

此点的梯度幅值不是直接累加到直方图中去的呀，要乘以一个高斯权重即m(x, y)=m(x, y)*G(x, y, 1.5σ)，此处xy为局部坐标
幅值累加数值H
主方向
辅方向
辅方向
关键点方向直方图
幅角 θ

Link

Feature Extraction

- SIFT algorithm
  4. Compute keypoint orientations

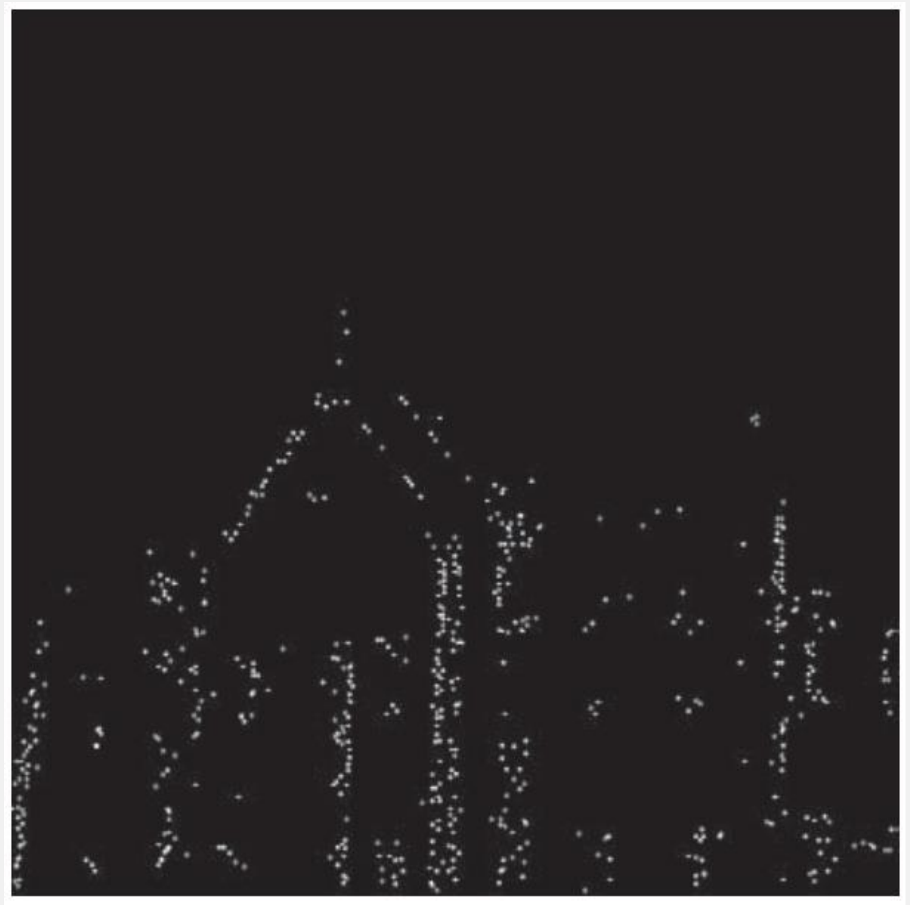

主方向
(0, H(j)
注意，原先的主方向（或辅方向）是一个角度区间哦，抛物线插值后我们用抛物线顶点横坐标作为主方向的角度，此时主方向（或辅方向）是一个值
辅方向
幅值累加数值 H
(-1, H(j-1)
(1, H(j+1)
插值时的局部坐标系
-1
0
1
幅角索引 i
j
我们只在主方向及辅方向所在的柱子序号 j 且满足H(j)>H(j-1) && H(j)>H(j+1) 的 j-1、 j、 j+1 小柱子进行插值，3 个点就可以确定一条插值抛物线

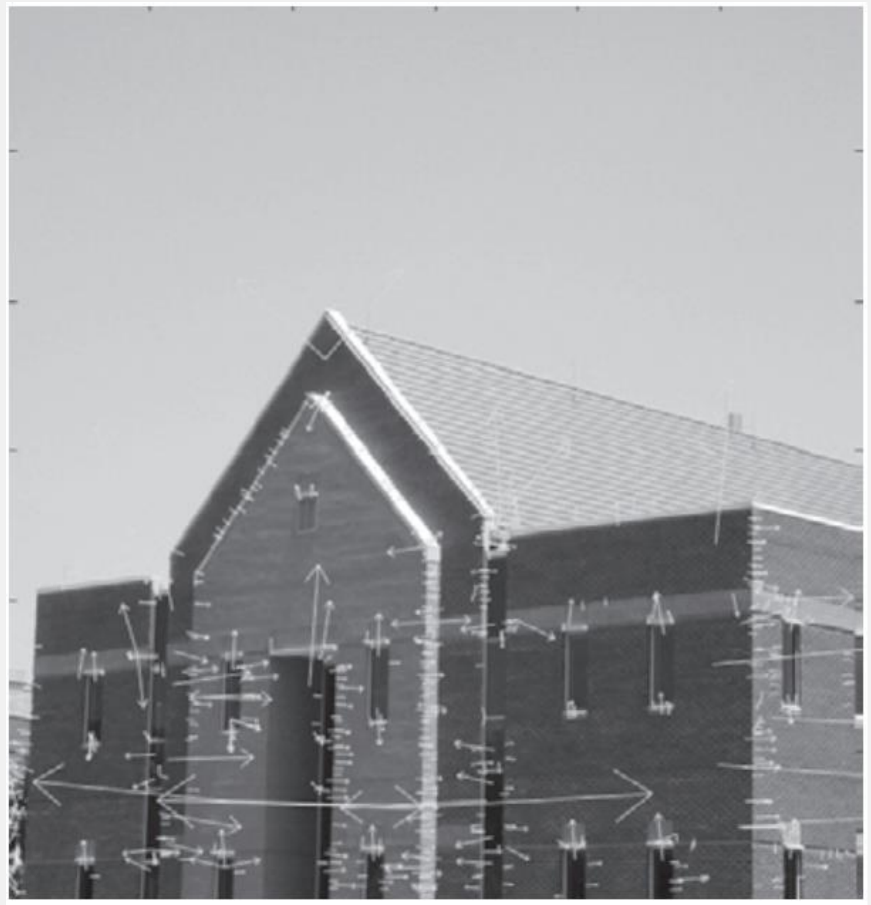


Link

Feature Extraction

- SIFT algorithm
  4. Compute keypoint orientations



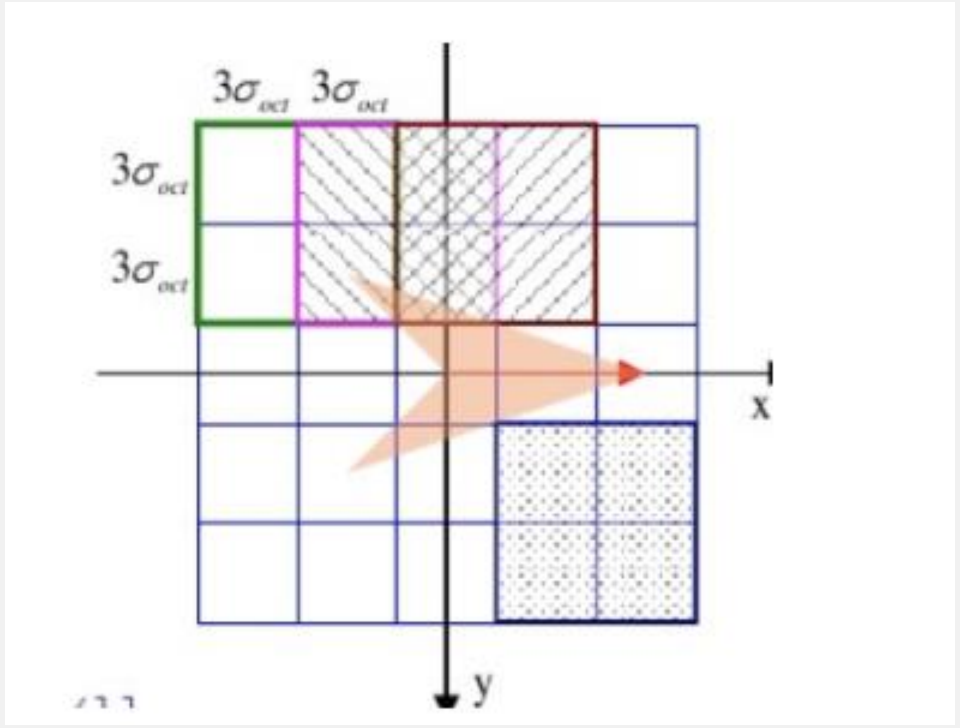


Feature Extraction

- SIFT algorithm
  5. Compute keypoint descriptors
     - Build a descriptor for each keypoint that is both distinctive and invariant to certain variables, such as lightling and viewpoint. Additionally, the descriptor not only includes the keypoint itself but also the surrounding pixels that contribute to it
     - The main idea is to divide the image area around the keypoint into blocks, calculate the gradient histogram within each block, generate feature vectors, and abstract the image information
     - Steps:
       1. Taking the feature point as the center, divide its nearby neighborhood into d*d sub-regions (usually d=4), each sub-region is a square with a side length of 3σ
       2. In order to ensure the rotation invariance of the feature point, the coordinate axis is rotated to the main direction of the key point with the feature point as the center
       3. Calculate the gradient of the pixels in the sub-region, perform Gaussian weighting according to σ=0.5d, and then interpolate to calculate the gradient of each seed point in eight directions

Feature Extraction

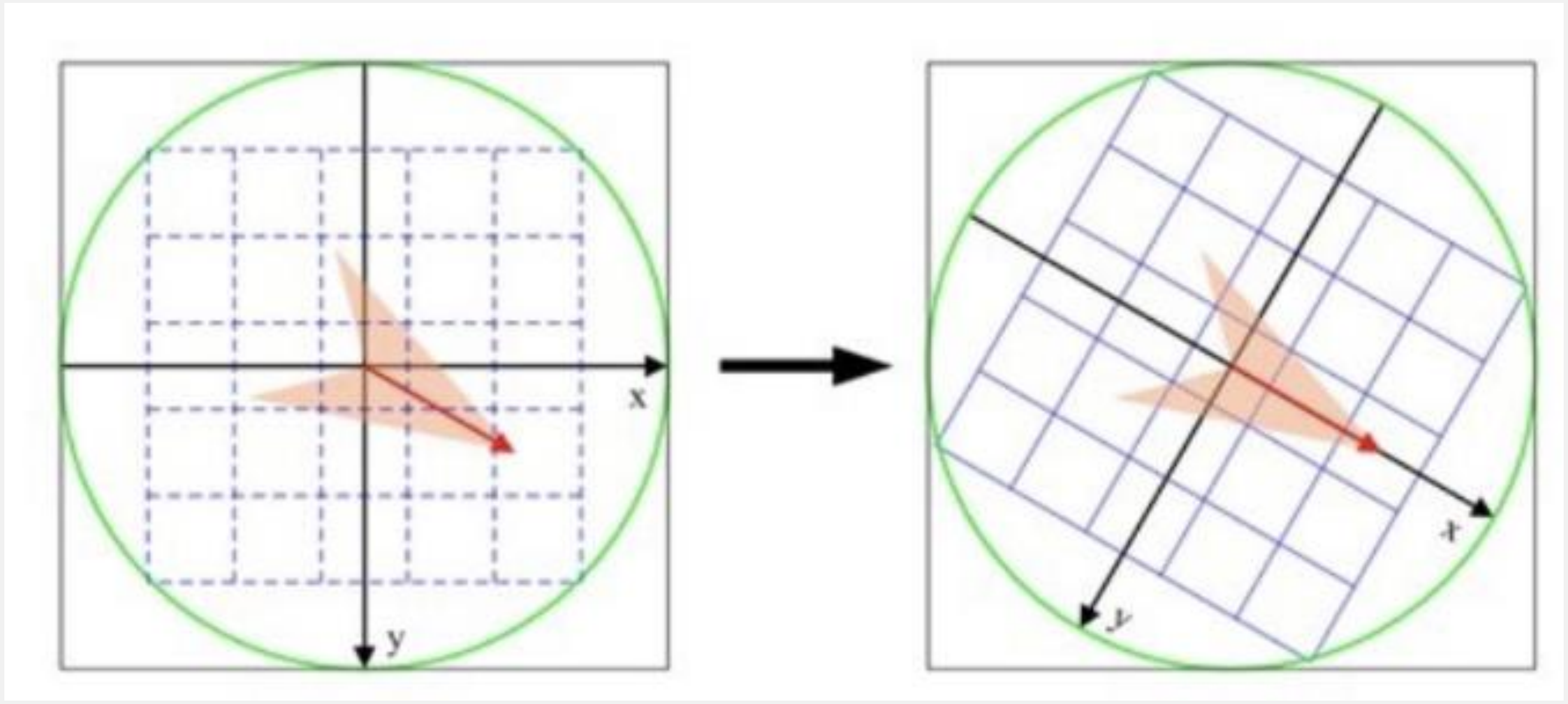
- SIFT algorithm
  5. Compute keypoint descriptors
     - Steps:
       1. Taking the feature point as the center, divide its nearby neighborhood into d*d sub-regions (usually d=4), each sub-region is a square with a side length of 3σ



Link

Feature Extraction

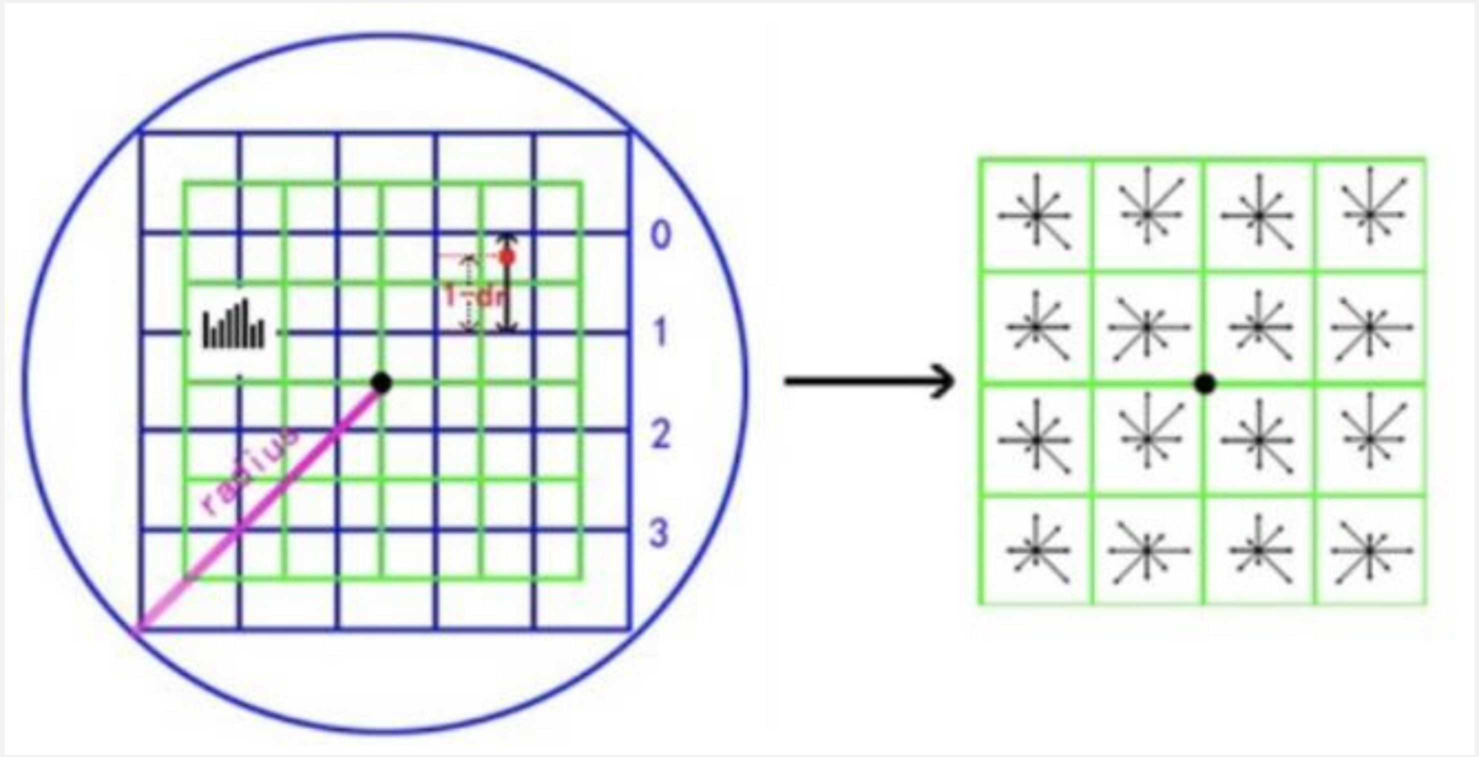
- SIFT algorithm
  5. Compute keypoint descriptors
     - Steps:
       2. In order to ensure the rotation invariance of the feature point, the coordinate axis is rotated to the main direction of the key point with the feature point as the center



Link

Feature Extraction

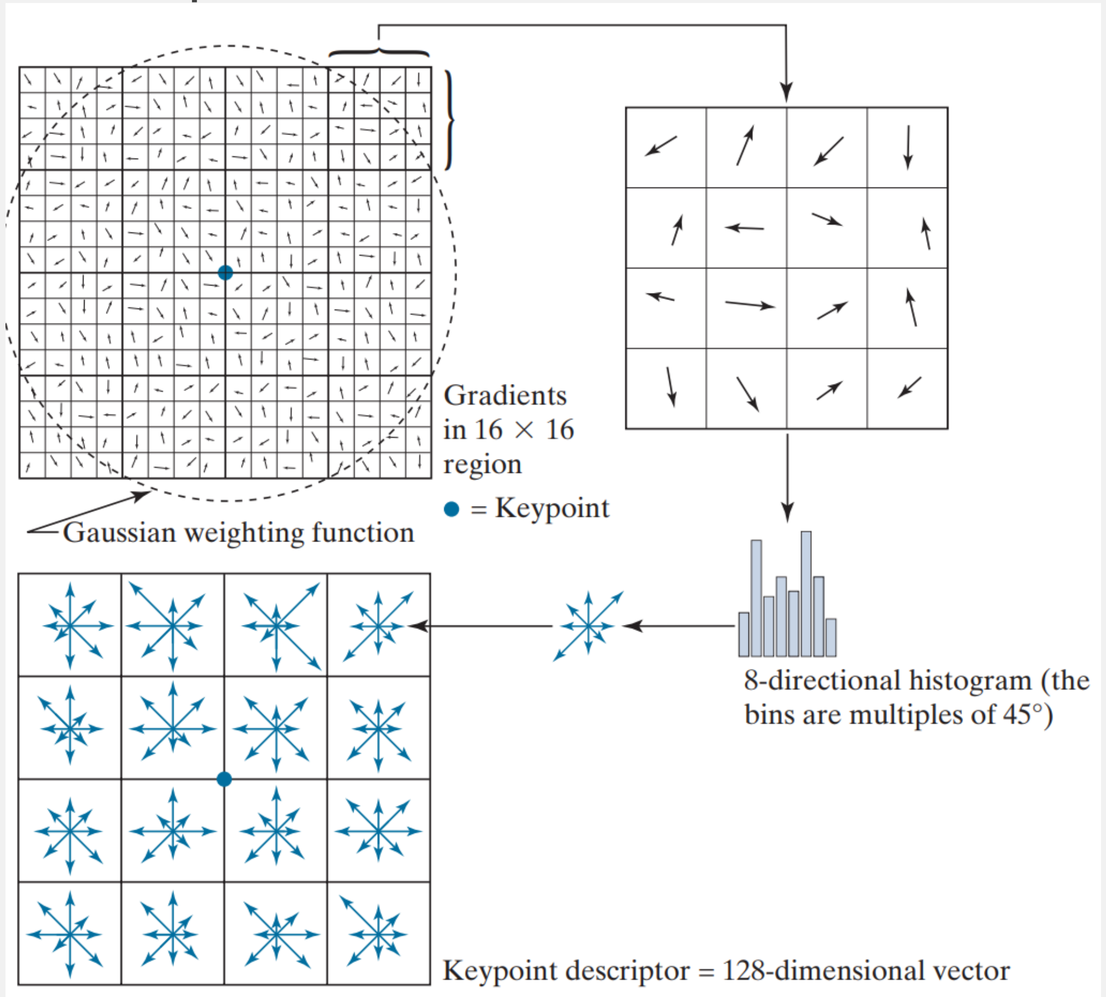
- SIFT algorithm
  5. Compute keypoint descriptors
     - Steps:
       3. Calculate the gradient of the pixels in the sub-region, perform Gaussian weighting according to σ=0.5d, and then interpolate to calculate the gradient of each seed point in eight directions



Link

Feature Extraction

- SIFT algorithm
  5. Compute keypoint descriptors


Gradients in 16 × 16 region
● = Keypoint
Gaussian weighting function
8-directional histogram (the bins are multiples of 45°)
Keypoint descriptor = 128-dimensional vector