FEATURES

Outline

- Element of Image Analysis
- Feature Extraction Overview
- Image Main features
- Color Feature
- Texture Feature
- Shape feature
- Features Post processing/Analysis

Element of Image Analysis

Preprocess

Image acquisition, restoration, and enhancement

↓

Intermediate process

Feature extraction & Image segmentation

↓

High level process

Image interpretation and recognition

Features Extraction

- Feature = "point of interest" for image description
  - Features should contain information required to distinguish between classes, be insensitive to irrelevant variability in the input

- The main goal of feature extraction is to obtain the most relevant information from the original data and represent that information in a lower dimensionality space.

Features Extraction

- When the input data to an algorithm is too large to be processed and it is suspected to be redundant (much data, but not much information) then the input data will be transformed into a reduced representation set of features (also named features vector).
- These feature vectors are then used by classifiers to recognize the input unit with target output unit.
- It has been used in many applications such as character recognition, document verification, reading bank deposit slips, extracting information from cheques, applications for credit cards, health insurance, loan, tax forms, data entry, postal address reading, check sorting, tax reading, script recognition etc.

Features Extraction Classification

- Features can be classified as:
  - General features: Application independent features such as color, texture, and shape. According to the abstraction level, they can be further divided into:
    - Pixel-level features: Features calculated at each pixel, e.g. color, location.
    - Local features: Features calculated over the results of subdivision of the image band on image segmentation or edge detection.
    - Global features: Features calculated over the entire image or just regular sub-area of an image.
  - Domain-specific features: Application dependent features such as human faces, fingerprints, and conceptual features. These features are often a synthesis of low-level features for a specific domain.

Features Extraction Classification

- On the other hand, all features can be coarsely classified into:
  - Low-level features: features can be extracted directed from the original images
    - Edges
    - Corners
    - Interest points
  - High-level features: high-level feature extraction must be based on low level features
    - Template Matching
    - Hough Transform

Efficiency of features

- **Identifiability**: shapes which are found perceptually similar by human have the same feature different from the others.
- **Translation, rotation and scale invariance**: the location, rotation and scaling changing of the shape must not affect the extracted features.
- **Affine invariance**: the affine transform performs a linear mapping from 2D coordinates to other 2D coordinates that preserves the "straightness" and "parallelism" of lines. Affine transform can be constructed using sequences of translations, scales, flips, rotations and shears. The extracted features must be as invariant as possible with affine transforms.

Efficiency of features

- **Noise resistance**: features must be as robust as possible against noise, i.e., they must be the same whichever be the strength of the noise in a give range that affects the pattern.
- **Occultation invariance**: when some parts of a shape are occulted by other objects, the feature of the remaining part must not change compared to the original shape.
- **Statistically independent**: two features must be statistically independent. This represents compactness of the representation.
- **Reliable**: as long as one deals with the same pattern, the extracted features must remain the same.

What is best method for feature extraction

- It all depends on your application at hand. Few things you should keep in mind are:
  - Feature extraction is highly subjective in nature
  - There is no generic feature extraction scheme which works in all cases.
    - What kind of problem are you trying to solve? e.g. classification, regression, clustering, etc.
    - Do you have a lot of data?
    - Do your data have very high dimensionality?
    - Is your data labelled?
    - Do you want to use a very computationally intensive method or something rather inexpensive?

Image features

- Image Main features:
  - Color
  - Texture
  - Shape

Color Features

- The color feature is one of the most widely used visual features in image retrieval. Images characterized by color features have many advantages:
  - Robustness. The color histogram is invariant to rotation of the image on the view axis, and changes in small steps when rotated otherwise or scaled
  - Effectiveness. There is high percentage of relevance between the query image and the extracted matching images.
  - Implementation simplicity. The construction of the color histogram is a straightforward process
    - Computational simplicity. The histogram computation has $O(X, Y)$ complexity for images of size $X \times Y$.

Color Features

- Color features are defined subject to a particular color space or model.
- A number of color spaces have been used in literature, such as RGB, HSV, etc.
- Once the color space is specified, color feature can be extracted from images or regions.
- A number of important color features have been proposed in the literatures, including color histogram, color moments(CM), color coherence vector (CCV), color correlogram, *etc.*

Color histogram

- A color histogram H for a given image is defined as a vector H = {h[1], h[2], . . . h[i], . . . , h[N]} where i represents a color in the color histogram, h[i] is the number of pixels in color i in that image, and N is the number of bins in the color histogram, i.e., the number of colors in the adopted color model.
- In order to compare images of different sizes, color histograms should be normalized.
- Can be used to Measures the similarity of
  - images
  - speech
  - music
- Issue:
  - how to capture perceptual similarity of an image

Color histogram

- The standard measure of similarity used for color histograms:
  - A color histogram H(i) is generated for each image h in the database (feature vector),
  - The histogram is *normalized so that its sum equals* unity (removes the size of the image),
  - The histogram is then stored in the database,
  - Now suppose we select a *model image (the new* image to match against all possible targets in the database).

Color histogram

- Histogram distance measures

L_1 distance (Manhattan distance)

$$d_1(H,L) = \sum_{k=1}^{K} | h_k - l_k |$$

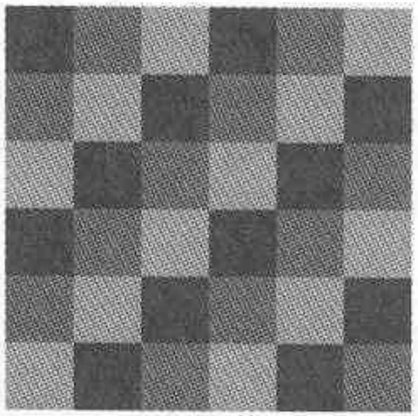
L_2 distance (euklidian distance)

$$d_2(H,L) = \sqrt{\sum_{k=1}^{K} | h_k - l_k |^2}$$

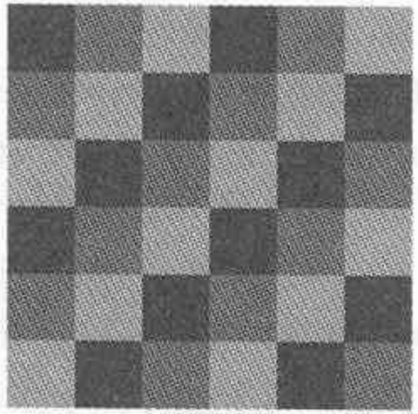
L_∞ distance (maximum distance)

$$d_\infty(H,L) = \max_k (| h_k - l_k |)$$

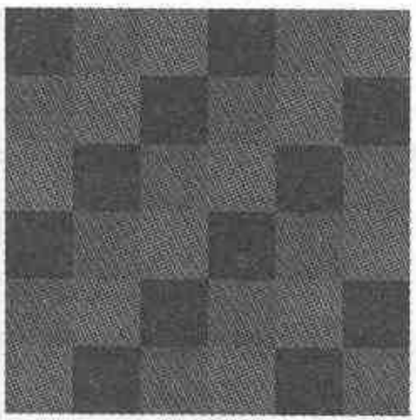
Example for potential problem with histogram distance



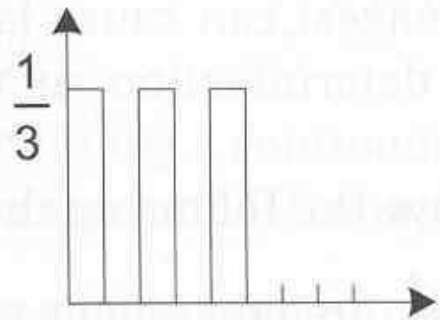
(a)



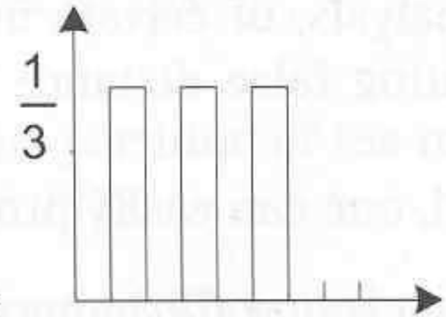
(b)



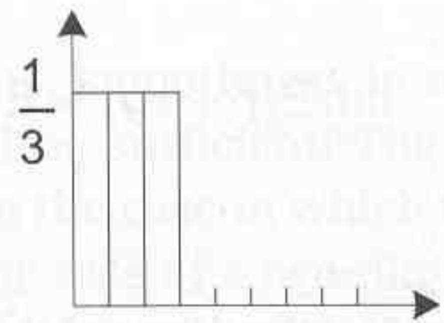
(c)



(a)



(b)



(c)

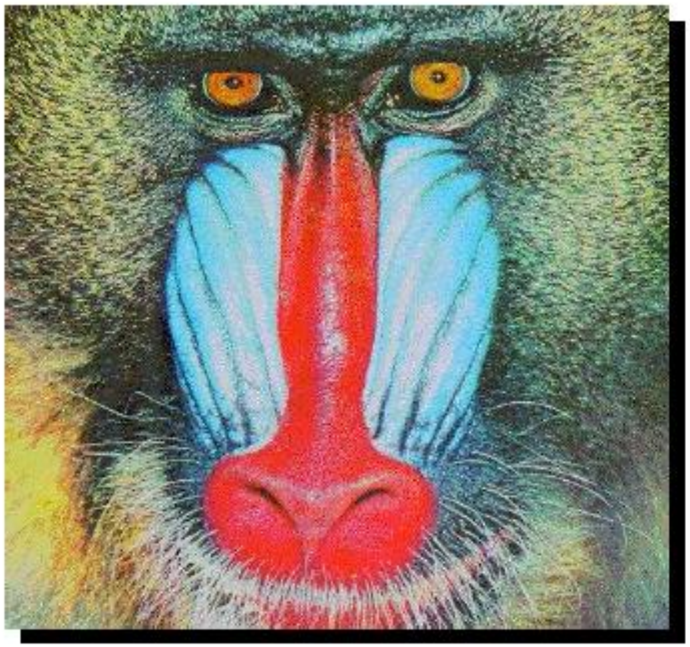
Distances of the three checkerboard images

Distance type	d(a,b)	d(a,c)
L_1	2	~0.67
L_2	~0.82	~0.47
L_∞	~0.33	~0.33

None of the distances captures perceptual similarity

Another Issue: loss of regional information

Partition the image
One histogram per region



Color Moments

- Provide measurement for color similarity between images. These valu es of similarity can then be compared to the values of images indexed in a database for tasks like image retrieval.
- The basis of color moments lays in the assumption that the distribution of color in an image can be interpreted as a probability distribution.
- Probability distributions are characterized by a number of unique moments (e.g. Normal distributions are differentiated by their mean and variance).
- It therefore follows that if the color in an image follows a certain probability distribution, the moments of that distribution can then be used as features to identify that image based on color.
- The first order (mean), the second (variance) and the third order (skewness) color moments have been proved to be efficient and effective in representing color distributions of images.

Color Moments

MOMENT 1 – Mean :

$$E_i = \sum_{N}^{j=1} \frac{1}{N} p_{ij}$$

Mean can be understood as the average color value in the image.

MOMENT 2 - Standard Deviation :

$$\sigma_i = \sqrt{\left(\frac{1}{N} \sum_{N}^{j=1} (p_{ij} - E_i)^2\right)}$$

The standard deviation is the square root of the variance of the distribution.

MOMENT 3 – Skewness :

$$s_i = \sqrt[3]{\left(\frac{1}{N} \sum_{N}^{j=1} (p_{ij} - E_i)^3\right)}$$

Skewness can be understood as a measure of the degree of asymmetry in the distribution.

Color Moments

A function of the similarity between two image distributions is defined as the sum of the weighted differences between the moments of the two distributions. Formally this is:

$$d_{mom}(H, I) = \sum_{i=1}^{r} w_{i1}\left|E_i^1 - E_i^2\right| + w_{i2}\left|\sigma_i^1 - \sigma_i^2\right| + w_{i3}\left|s_i^1 - s_i^2\right|$$

Where:

(H, I)	: are the two image distributions being compared
i	: is the current channel index (e.g. 1 = H, 2 = S, 3 = V)
r	: is the number of channels (e.g. 3)
E_i^1, E_i^2	: are the first moments (mean) of the two image distributions
σ_i^1, σ_i^2	: are the second moments (std) of the two image distributions
s_i^1, s_i^2	: are the third moments (skewness) of the two image distributions
w_i	: are the weights for each moment (described below)

Pairs of images can be ranked based on their d_{mom} values. Those with greater values are ranked lower and considered less similar than those with a higher rank and lower d_{mom} values.

Color Moments example

Index Image

Test Image 1

Test Image 2

$$d_{mom}(Index, Test1) = 0.5878$$

$$d_{mom}(Index, Test2) = 1.5585$$

Color Features Technique Summery

Color method	Pros.	Cons.
Histogram	Simple to compute, intuitive	High dimension, no spatial info, sensitive to noise
CM	Compact, robust	Not enough to describe all colors, no spatial info
CCV	Spatial info	High dimension, high computation cost
Correlogram	Spatial info	Very high computation cost, sensitive to noise, rotation and scale
DCD	Compact, robust, perceptual meaning	Need post-processing for spatial info
CSD	Spatial info	Sensitive to noise, rotation and scale
SCD	Compact on need, scalability	No spatial info, less accurate if compact

CCV: color coherence vector
DCD: dominant color descriptor
CSD: color structure descriptor
SCD: scalable color descriptor respectively

Texture Features: What´s in the image?

- Texture is a tactile or visual characteristic of a surface.
- In general, color is usually a pixel property while texture can only be measured from a group of pixels.
- Texture gives us information about the spatial arrangement of the colors or intensities in an image.
- Aim: to find a unique way of representing the underlying characteristics of textures and represent them in some simpler but unique form, so then they can be used to accurately and robustly classify and segment objects.

Texture Features

- Basically, texture representation methods can be classified into two categories: structural; and statistical.
- Structural approach: Texture is a set of primitive texels in some regular or repeated relationship.
  - Texel: A small geometric pattern that is repeated frequently on some surface resulting in a texture.
- Statistical approach: Texture is a quantitative measure of the arrangement of intensities in a region.
- While the first approach is appealing and can work well for man-made and regular patterns, the second approach is more general and easier to compute and is used more often in practice.

Structural approach

- Structural approaches model texture as a set of texture primitives (also called texels (texture elements)) in a particular spatial relationship (also called lattice or grid layout).
- A structural description of a texture includes a description of the primitives and a specification of their placement patterns.
- Of course, the primitives must be identifiable and their relationships must be efficiently computable.
- Suitable for textures where primitives can be described using a larger variety of properties

Structural approach

A method that involves the detection of interest points, clustering of these points, voting for consistent lattice unit proposals, and iterative fitting of a lattice structure.

Statistical approach

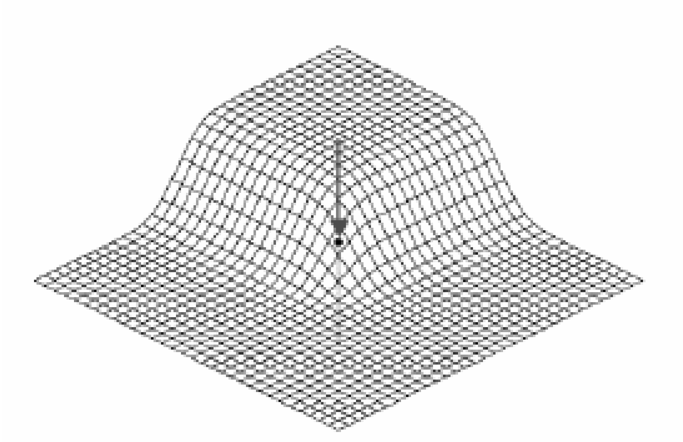
- Usually, segmenting out the texels is difficult or even impossible in real images.
- Instead, numeric quantities or statistics that describe a texture can be computed from the gray tones or colors themselves.
- This approach can be less intuitive, but is computationally efficient and often works well.
- Depending on the number of pixels defining the local feature statistical methods can be further classified into first-order (one pixel), second-order (two pixels) and higher-order (three or more pixels) statistics.

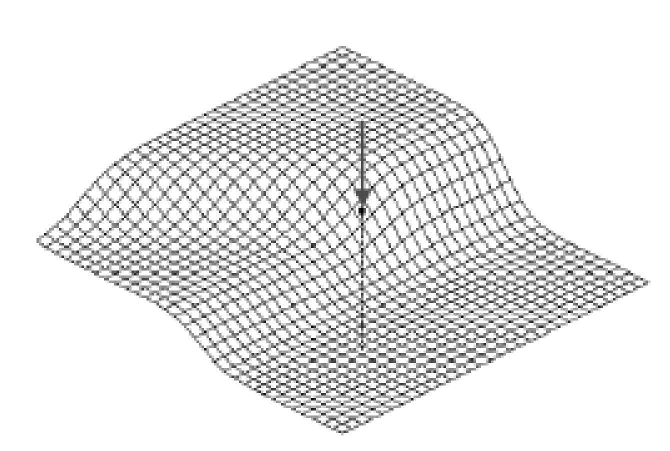
Statistical approach

- Some statistical approaches for texture:
  - Edge density and direction
  - Co- occurrence matrices
  - Local binary patterns
  - Statistical moments
  - Autocorrelation
  - Markov random fields
  - Autoregressive models
  - Mathematical morphology
  - Interest points – SIFT, SURF
  - Fourier power spectrum
  - Gabor filters
- Among these SIFT and SURF (Interest Points) are the most common methods for texture feature extraction

Texture extraction by Interest Points

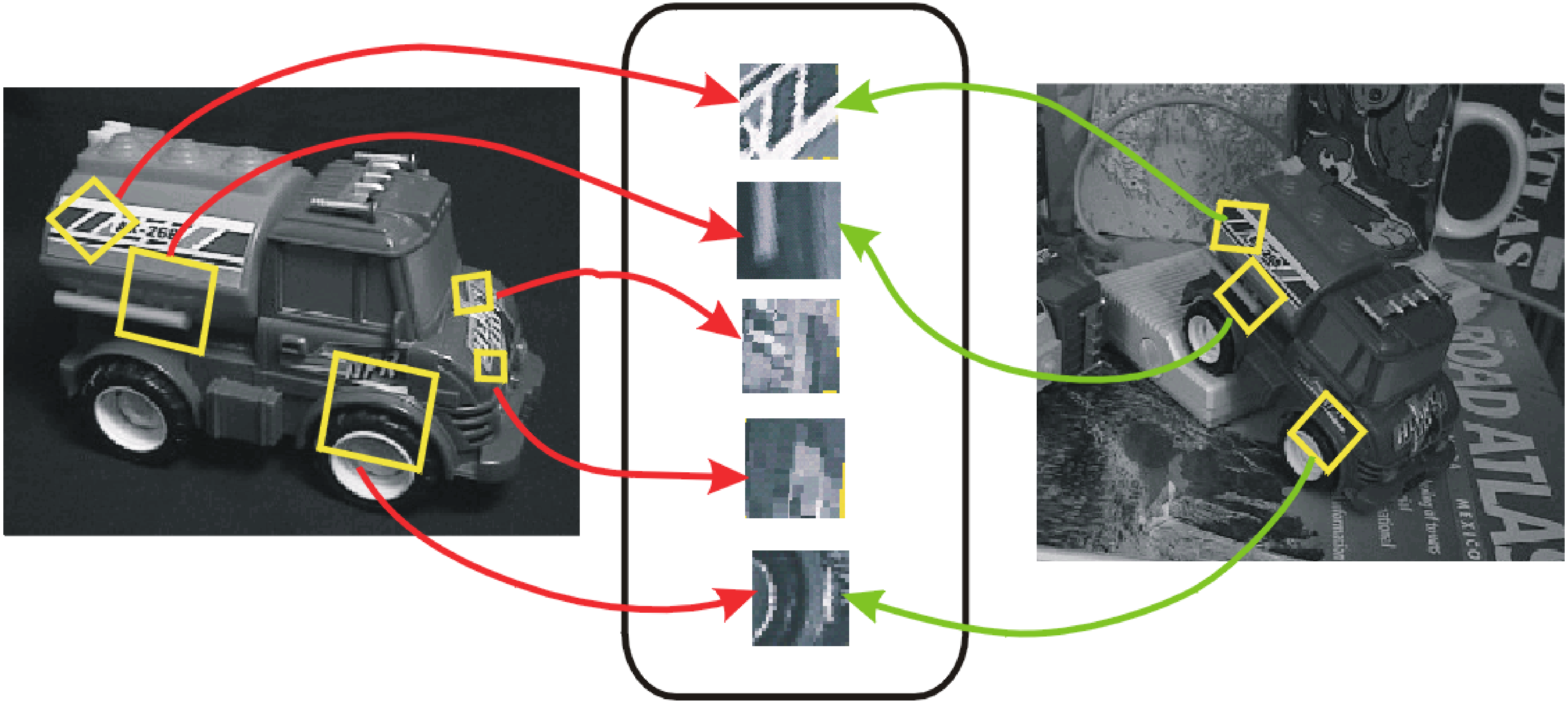
- What is an interest point
  - Expressive texture
    - The point at which the direction of the boundary of object changes abruptly
    - Intersection point between two or more edge segments
  - Goal: Detect points that are *repeatable* and *distinctive*





Interest Point Detection Idea

- Image content is transformed into local feature coordinates that are invariant to translation, rotation, scale, and other imaging parameters

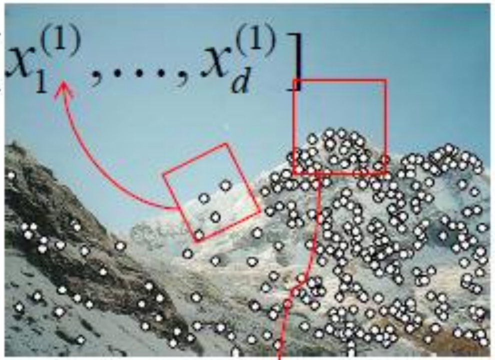


Local features extraction: main components

1) **Detection:** Identify the interest points

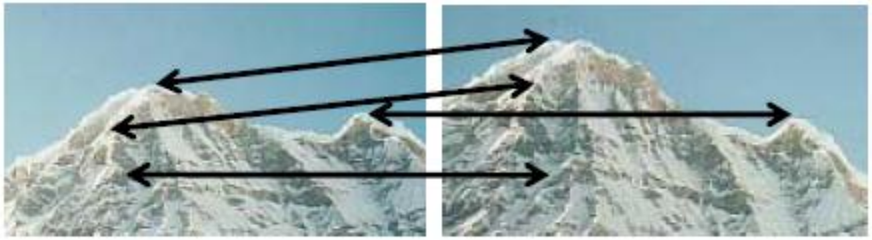
2) **Description** :Extract feature vector descriptor surrounding each interest point.

$$\mathbf{x}_1 = [x_1^{(1)}, \ldots, x_d^{(1)}]$$



$$\mathbf{x}_2 = [x_1^{(2)}, \ldots, x_d^{(2)}]$$

3) **Matching:** Determine correspondence between descriptors in two views

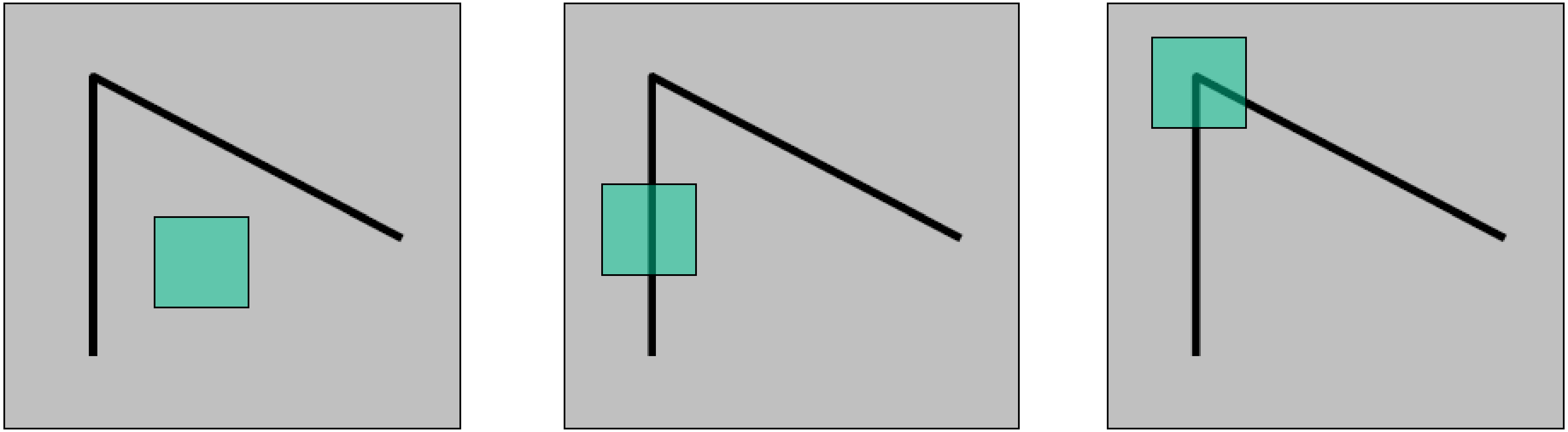


Choosing interest points

- What about edges?
  - Edges are usually defined as sets of points in the image which have a strong gradient magnitude
  - Edges can be invariant to brightness changes but typically not invariant to other transformations

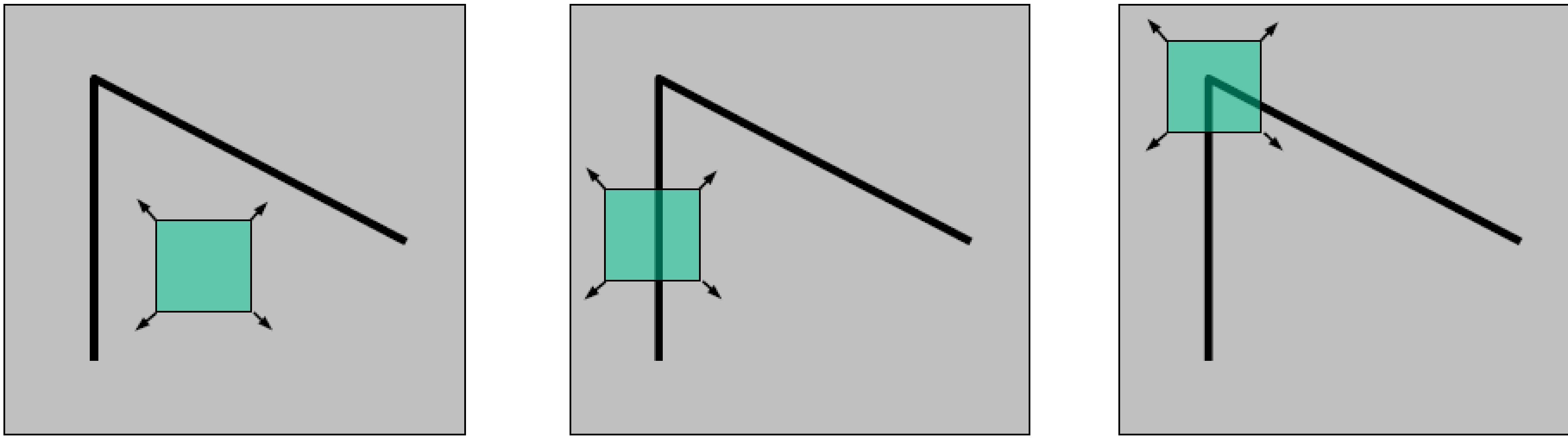
Finding interest points in an image

- Suppose we only consider a small window of pixels
  - What defines whether a feature is a good or bad candidate?



Finding interest points in an image

How does the window change when you shift it?



"flat" region:
no change in all directions

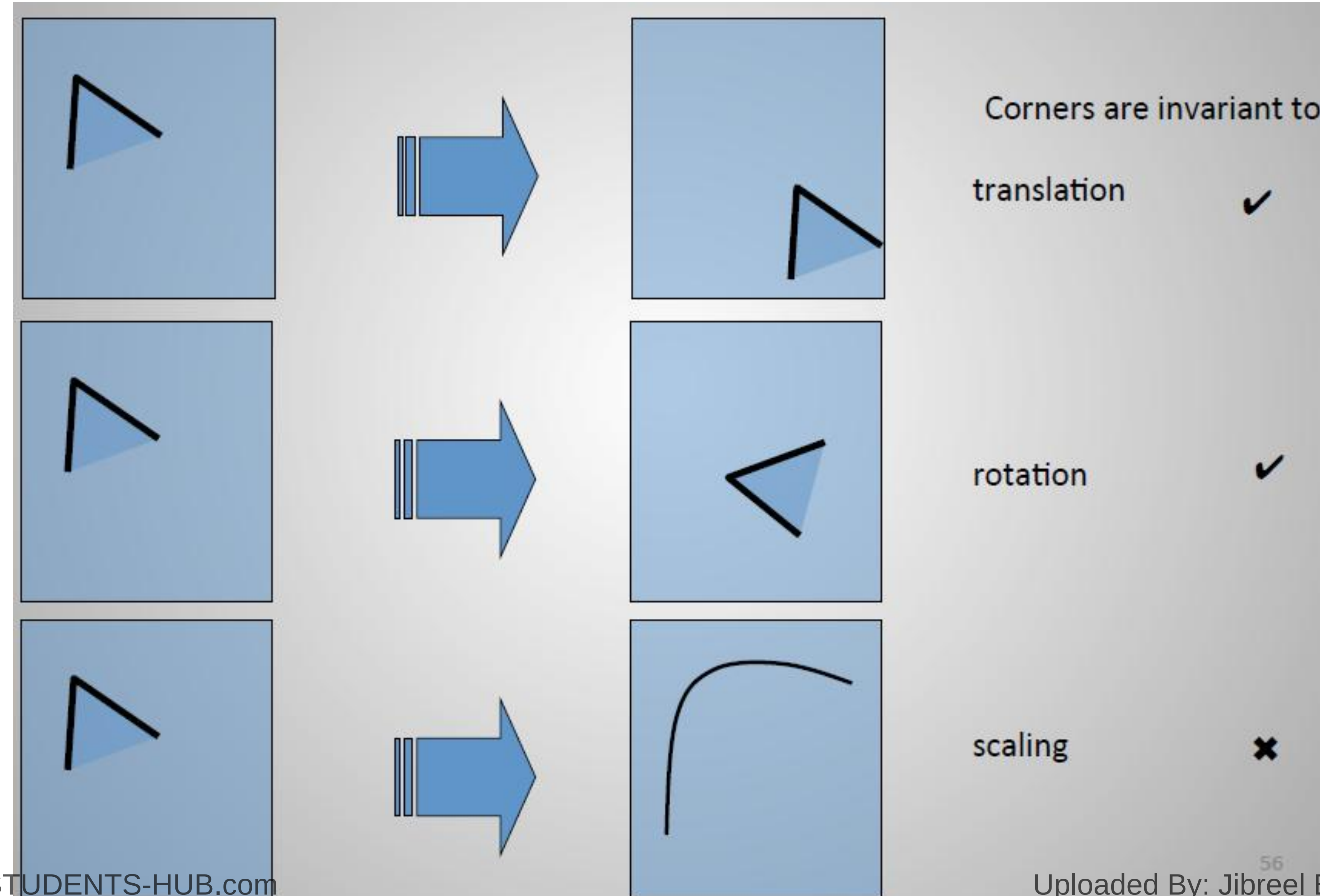
"edge":
no change along the edge direction

"corner":
significant change in all directions, i.e., even the minimum change is large

Find locations such that the minimum change caused by shifting the window in any direction is large

Corners detection By Harris Detector technique

Corner detection with respect to invariants



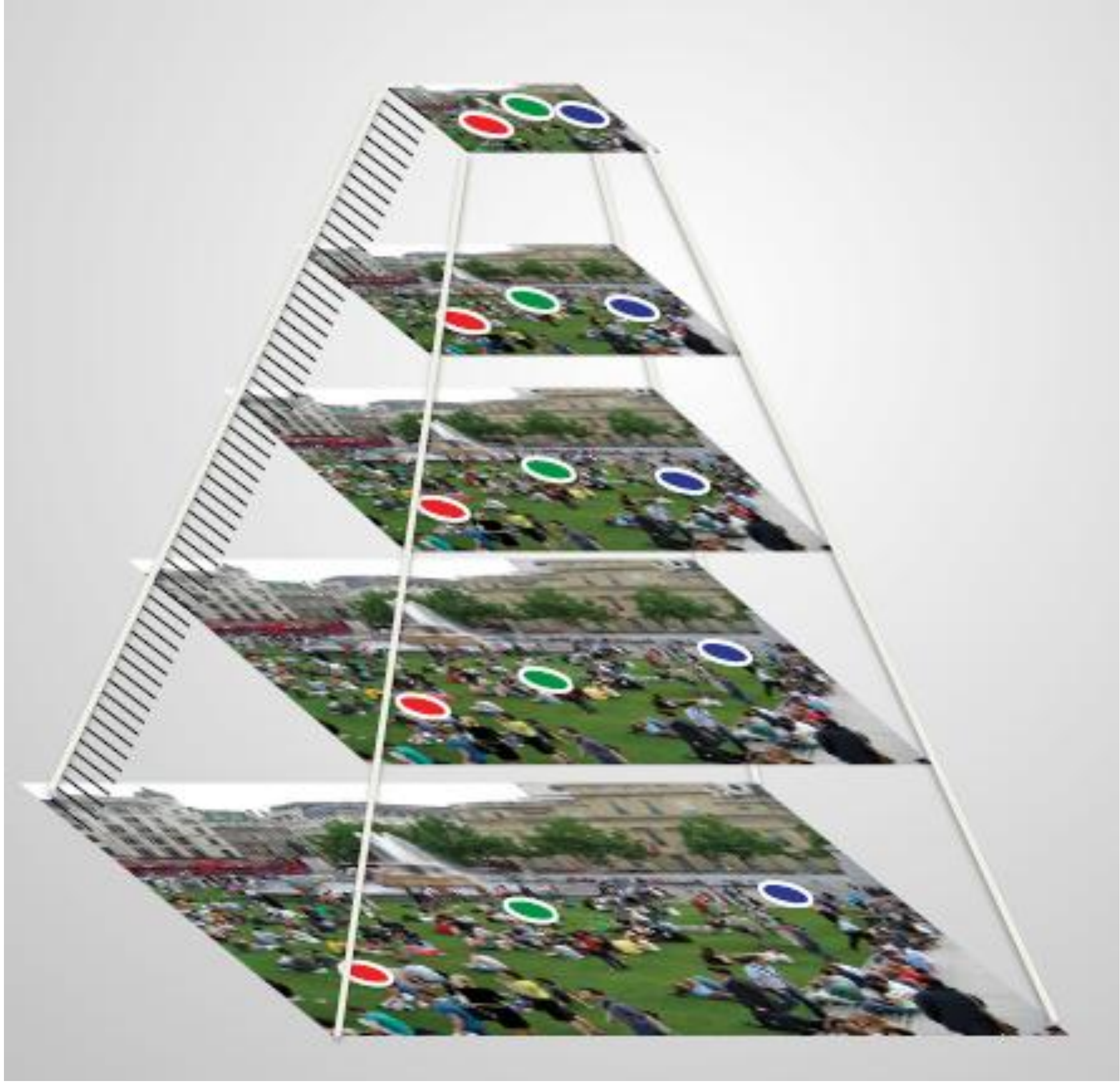
Scale Space

- The concept of scale is essential when computing features and descriptors from image data.
- Real-world objects may contain different types of structures at different scales and may therefore appear in different ways depending on the scale of observation.
- When observing objects by a camera or an eye, there is an additional scale problem due to perspective effects, implying that distant objects will appear smaller than nearby objects.
- A vision system intended to operate autonomously on image data acquired from a complex environment must therefore be able to handle and be robust to such scale variations.

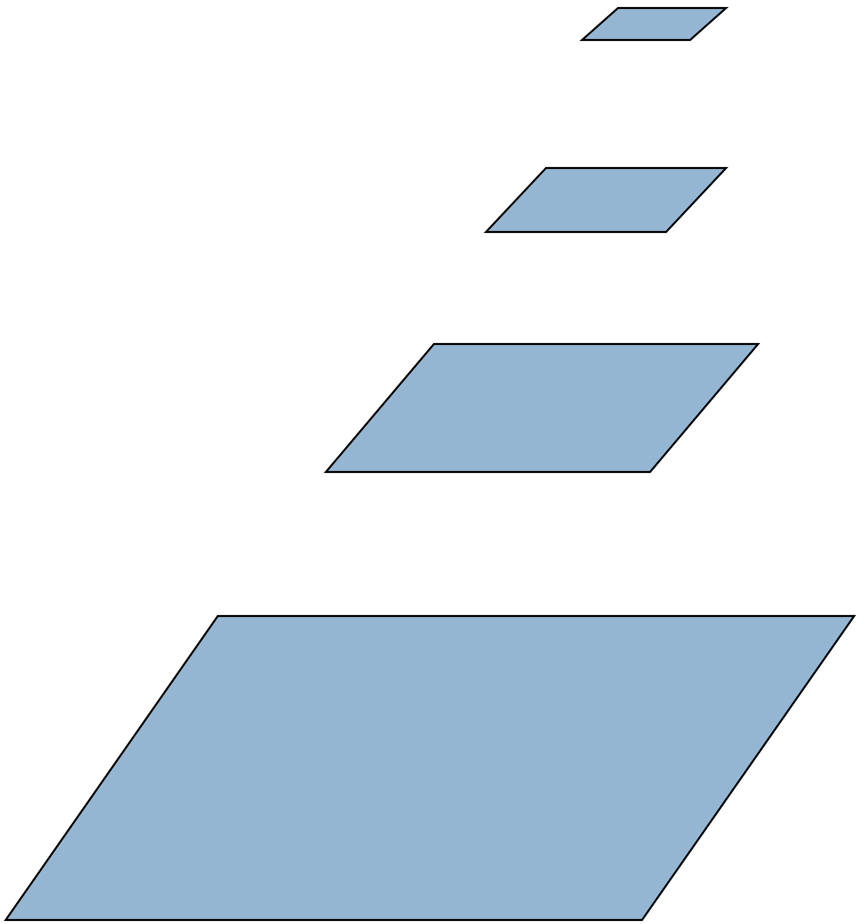
Scale Space

- The notion of scale selection refers to methods for estimating characteristic scales in image data and for automatically determining locally appropriate scales in a scale-space representation, so as to adapt subsequent processing to the local image structure and compute scale invariant image features and image descriptors.
- An essential aspect of the approach is that it allows for a bottom-up determination of inherent scales of features and objects without first recognizing them or delimiting alternatively segmenting them from their surrounding.
- Scale selection methods have also been developed from other viewpoints of performing noise suppression and exploring top-down information
- **Pyramid** is one way to represent images in Multi-Scale

Pyramid can capture global and local features



Aside: Image Pyramids



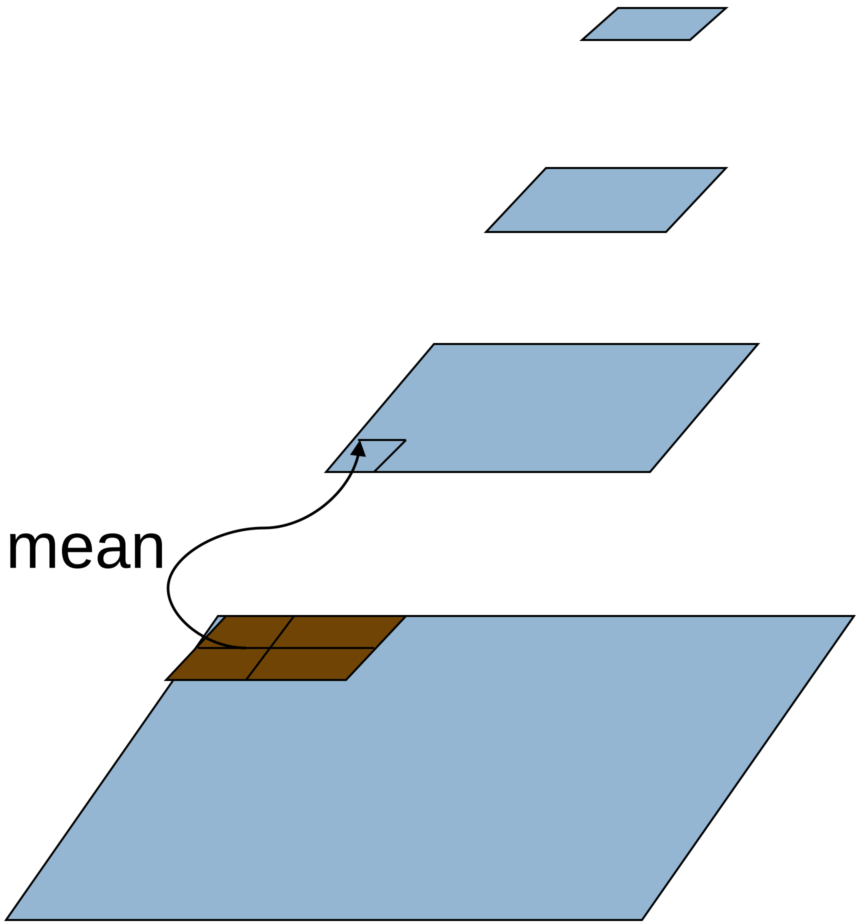
And so on.

3^{rd} level is derived from the 2^{nd} level according to the same funtion

2^{nd} level is derived from the original image according to some function

Bottom level is the original image.

Aside: Mean Pyramid



And so on.

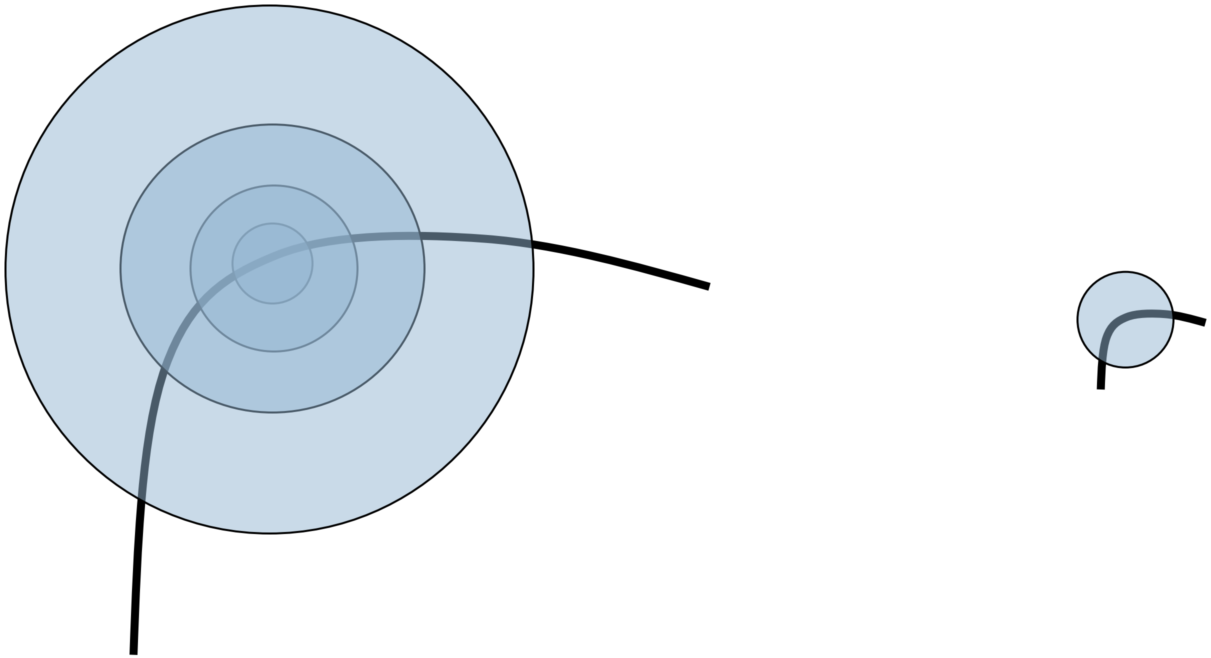
At 3rd level, each pixel is the mean of 4 pixels in the 2nd level.

At 2nd level, each pixel is the mean of 4 pixels in the original image.

Bottom level is the original image.

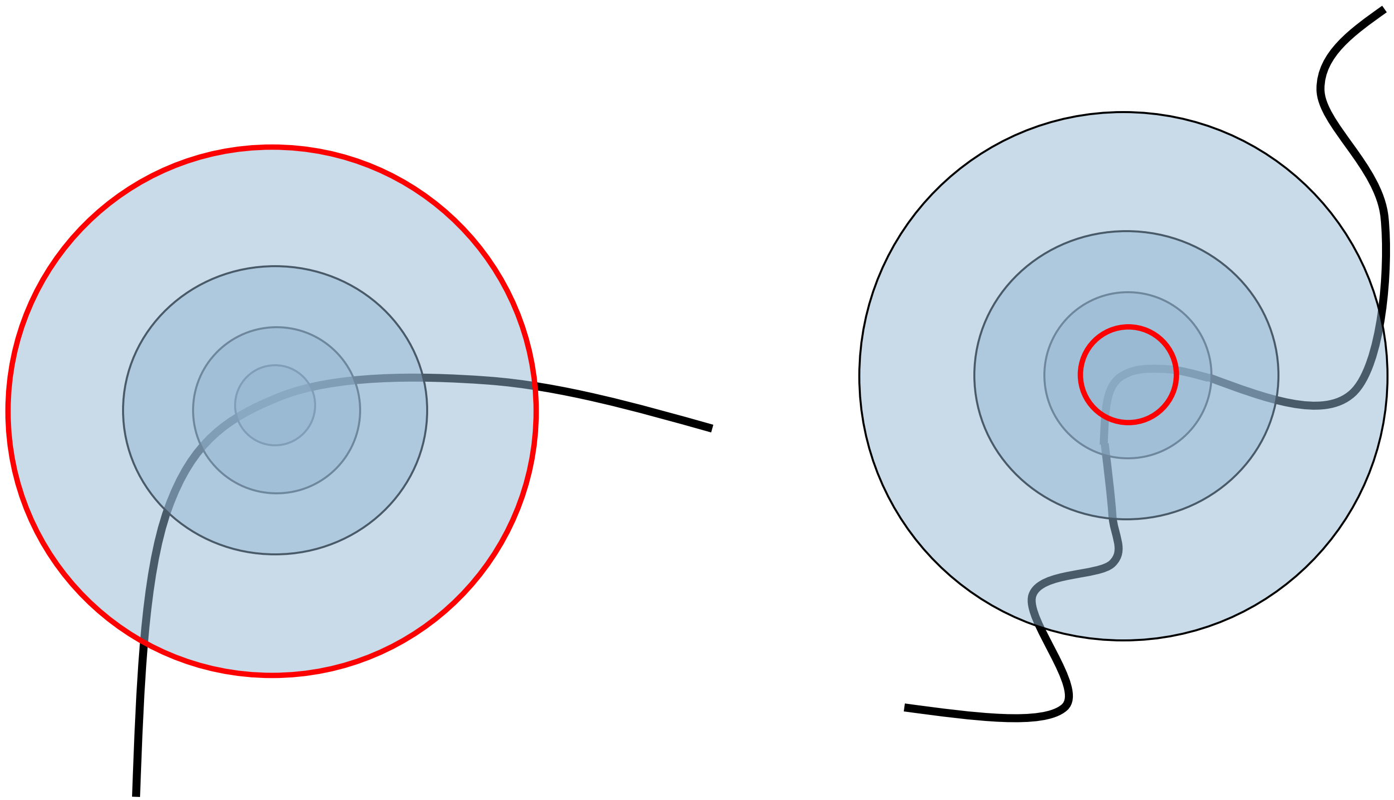
Scale Invariant Detection

- Consider regions (e.g. circles) of different sizes around a point
- Regions of corresponding sizes will look the same in both images



Scale Invariant Detection

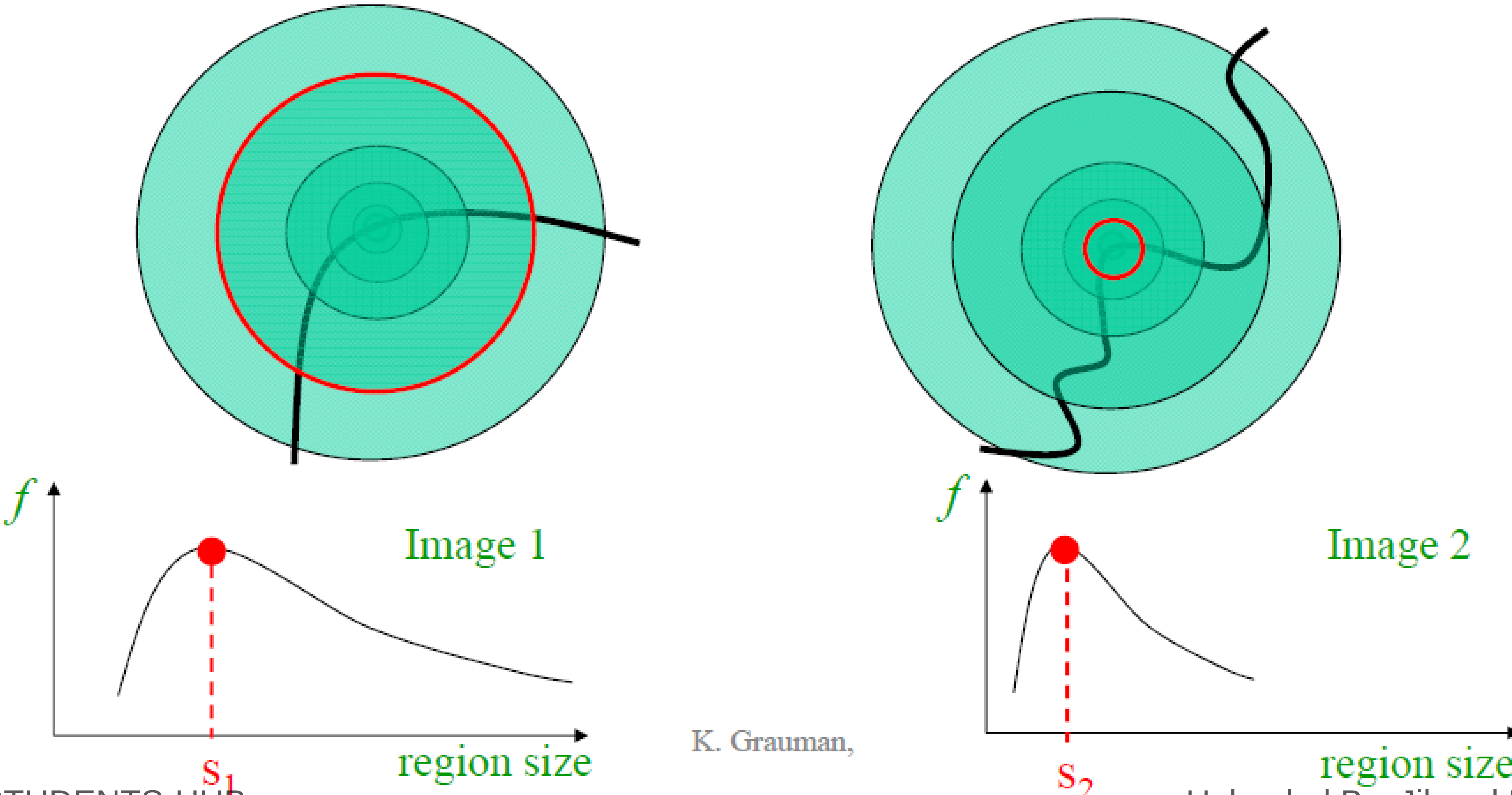
- The problem: how do we choose corresponding circles ***independently*** in each image?



Scale Invariant Detection

Intuition:

- Find scale that gives local maxima of some function *f* in both position and scale.



K. Grauman,

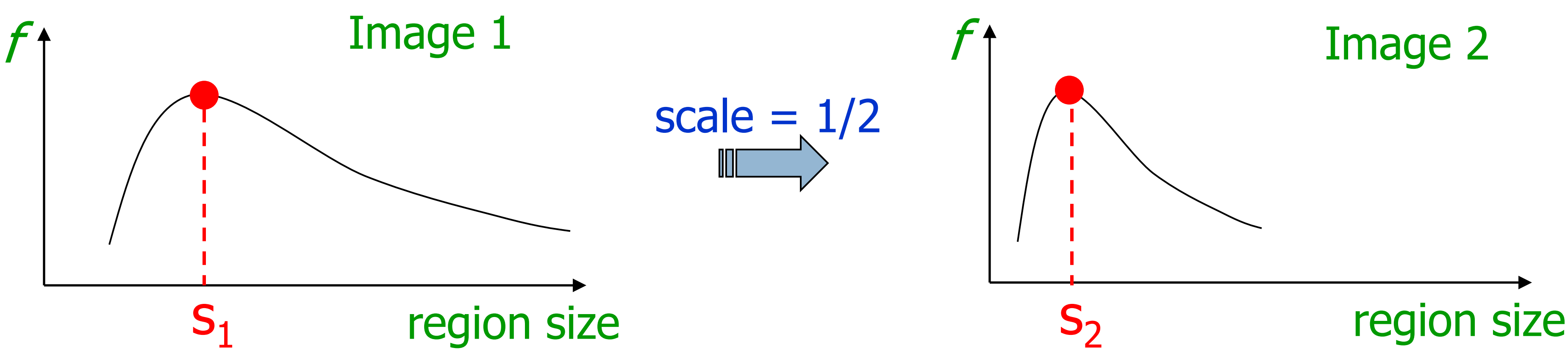
Scale Invariant Detection

- Common approach:

  Take a local maximum of this function

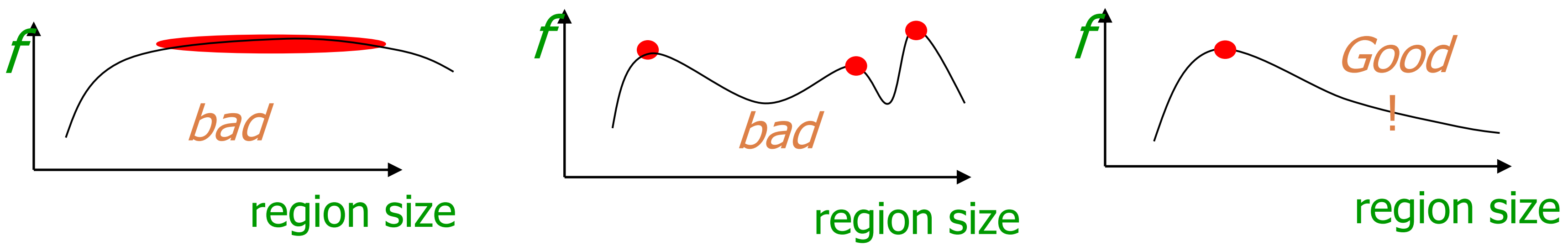
  Observation: region size, for which the maximum is achieved, should be *invariant* to image scale.

> Important: this scale invariant region size is found in each image independently!



Scale Invariant Detection

- A “good” function for scale detection:
  has one stable sharp peak



- For usual images: a good function would be a one which responds to contrast (sharp local intensity change)

Scale invariance

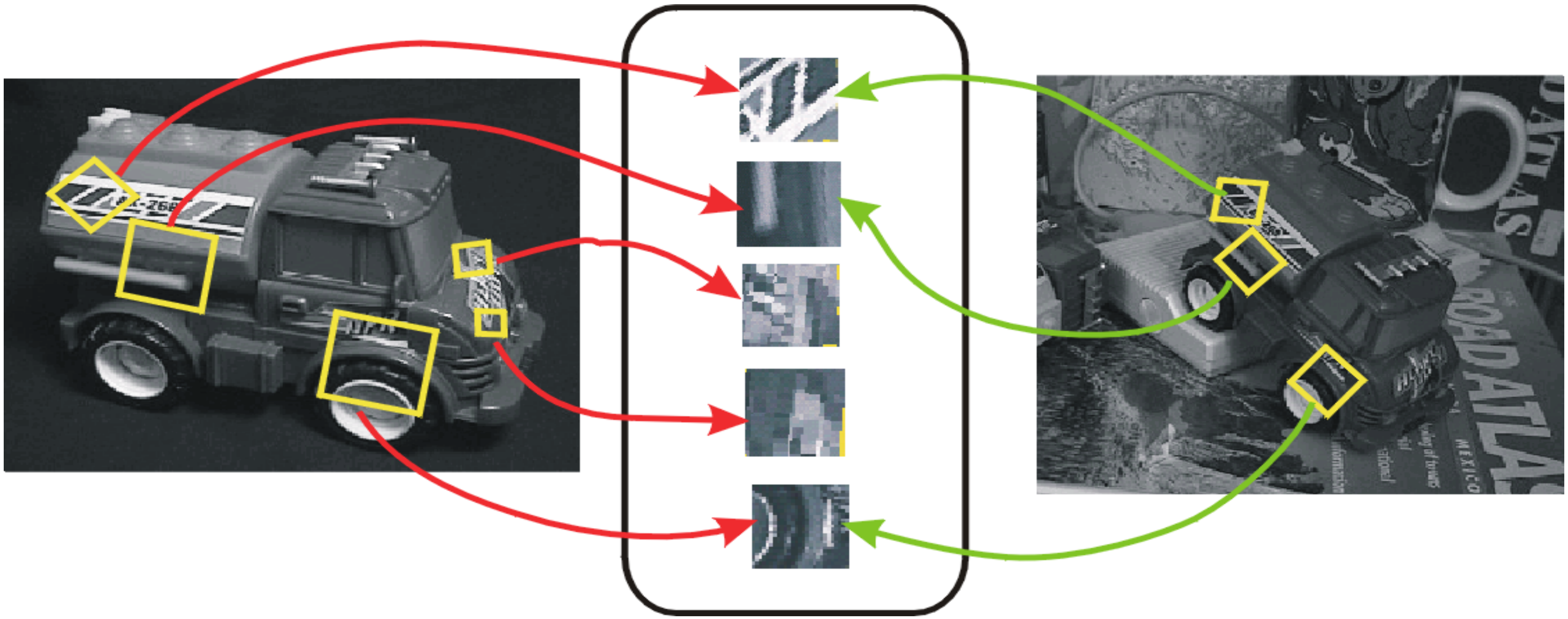
- **Requires a method to repeatably select points in location and scale:**
- **The** only reasonable scale-space kernel is a Gaussian
- An efficient choice is to detect peaks in the difference of Gaussian pyramid
- Difference-of-Gaussian with constant ratio of scales is a close approximation to scale-normalized Laplacian

SIFT - Scale Invariant Feature Transforms

- SIFT image features provide a set of features of an object that are not affected by many of the complications experienced in other methods, such as object scaling and rotation.
- While allowing for an object to be recognized in a larger image SIFT image features also allow for objects in multiple images of the same location, taken from different positions within the environment, to be recognized.
- SIFT features are also very resilient to the effects of "noise" in the image.
- The SIFT approach, for image feature generation, takes an image and transforms it into a "large collection of local feature vectors"

SIFT - Idea

- Image content is transformed into local feature coordinates that are invariant to translation, rotation, scale, and other imaging parameters



Claimed Advantages of SIFT

- **Locality:** features are local, so robust to occlusion and clutter (no prior segmentation)
- **Distinctiveness:** individual features can be matched to a large database of objects
- **Quantity:** many features can be generated for even small objects
- **Efficiency:** close to real-time performance
- **Extensibility:** can easily be extended to wide range of differing feature types, with each adding robustness

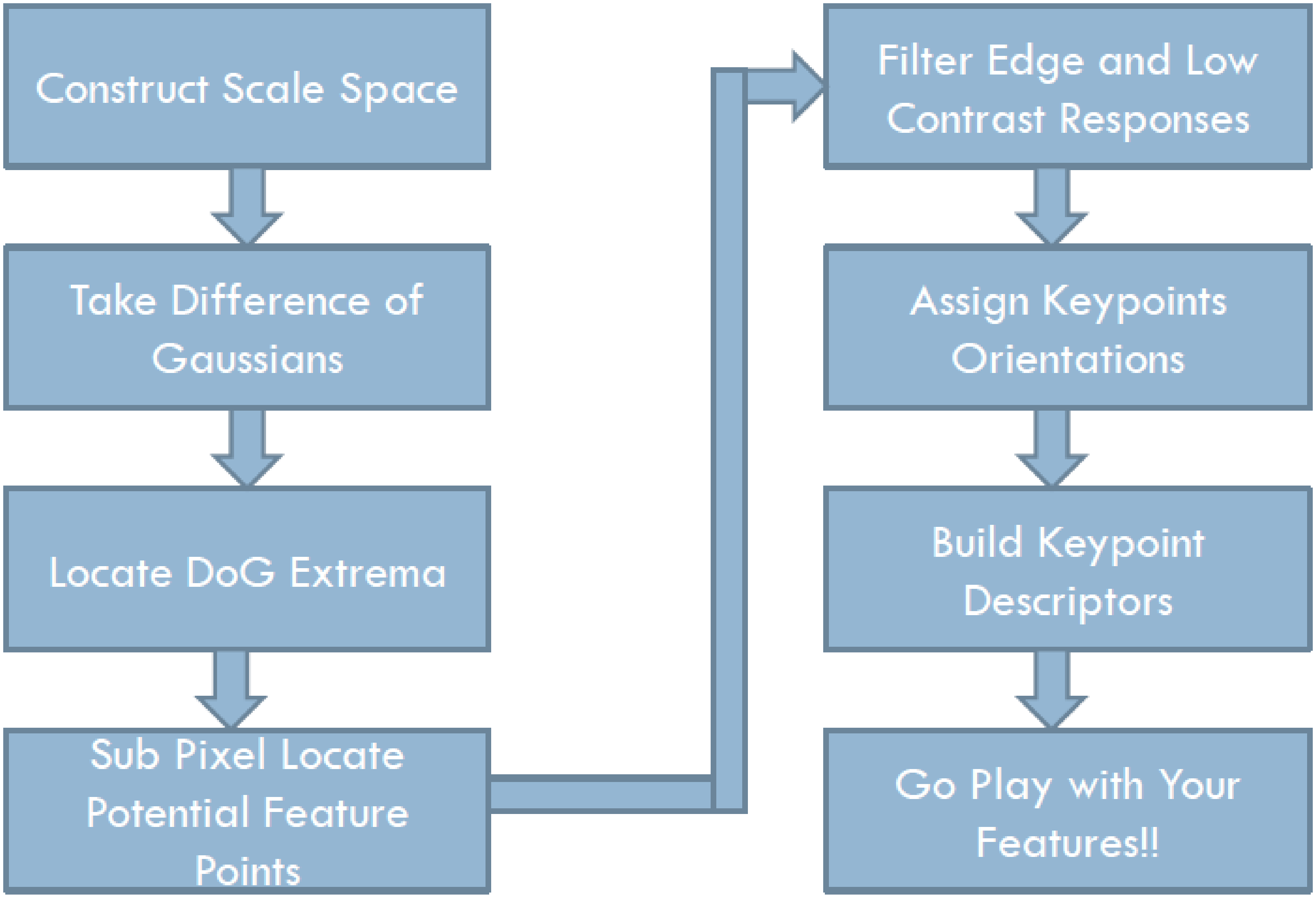
Uses for SIFT

- Feature points are used also for:
  - Image alignment
  - 3D reconstruction
  - Motion tracking
  - Object recognition
  - Indexing and database retrieval
  - Robot navigation
  - … many others

Overall Procedure at a High Level

1. Scale-space extrema detection
   - Search over multiple scales and image locations
2. Keypoint localization
   - Fit a model to detrmine location and scale.
   - Select keypoints based on a measure of stability.
3. Orientation assignment
   - Compute best orientation(s) for each keypoint region
4. Keypoint description
   - Use local image gradients at selected scale and rotation to describe each keypoint region.

Overview Of Algorithm



Scale-Space Extrema Detection

- This stage of the filtering attempts to identify those locations and scales that are identifiable from different views of the same object. This can be efficiently achieved using a "scale space" function.
- Further it has been shown under reasonable assumptions it must be based on the Gaussian function. The scale space is defined by the function:

  $L(x, y, \sigma) = G(x, y, \sigma) * I(x, y)$Where * is the convolution operator, $G(x, y, \sigma)$ is a variable-scale Gaussian and $I(x, y)$ is the input image.

- Various techniques can then be used to detect stable keypoint locations in the scale-space.
- Difference of Gaussians is one such technique, locating scale-space extrema, $D(x, y, \sigma)$ by computing the difference between two images, one with scale *k* times the other. $D(x, y, \sigma)$ is then given by:

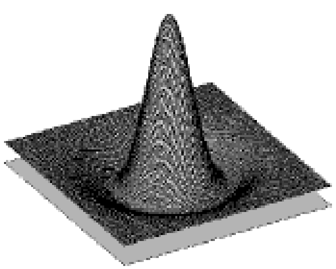
  $D(x, y, \sigma) = L(x, y, k\sigma) - L(x, y, \sigma)$

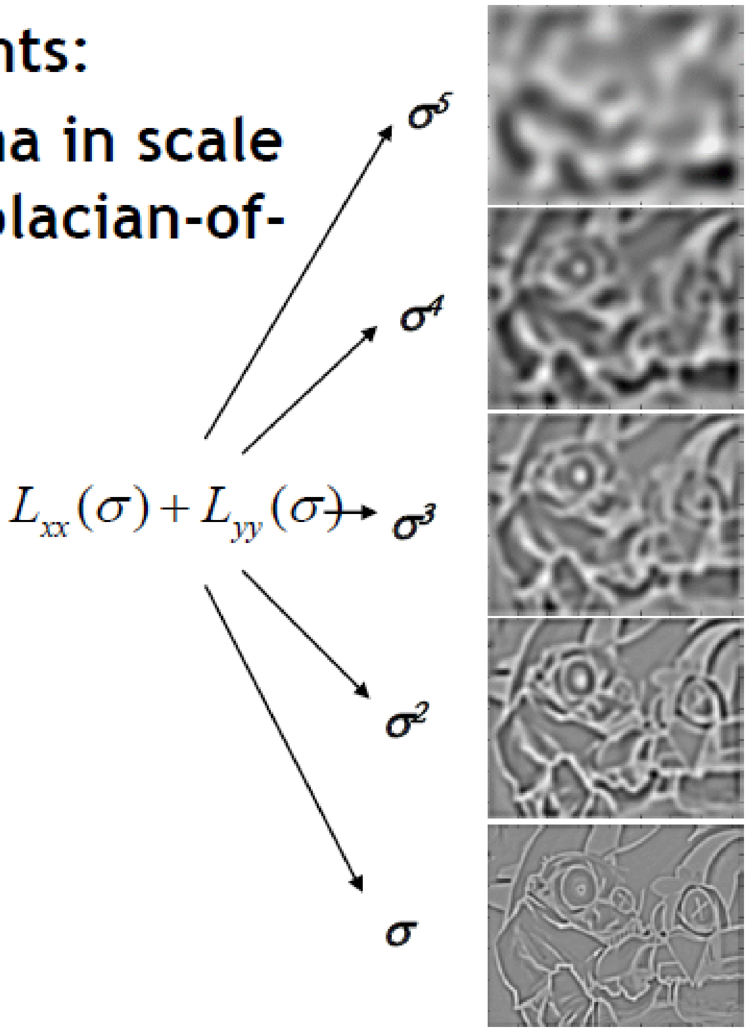
- To detect the local maxima and minima of D(x, y, σ) each point is compared with its 8 neighbours at the same scale, and its 9 neighbours up and down one scale. If this value is the minimum or maximum of all these points then this point is an extrema.

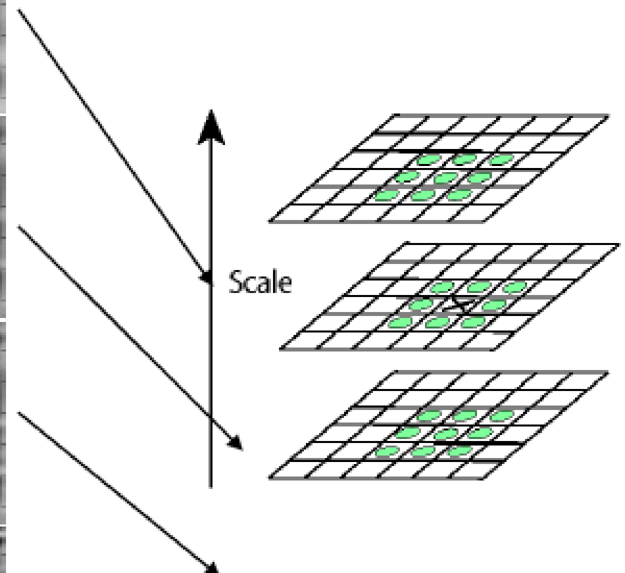
Laplacian-of-Gaussian (LoG)

Interest points:

Local maxima in scale space of Laplacian-of-Gaussian



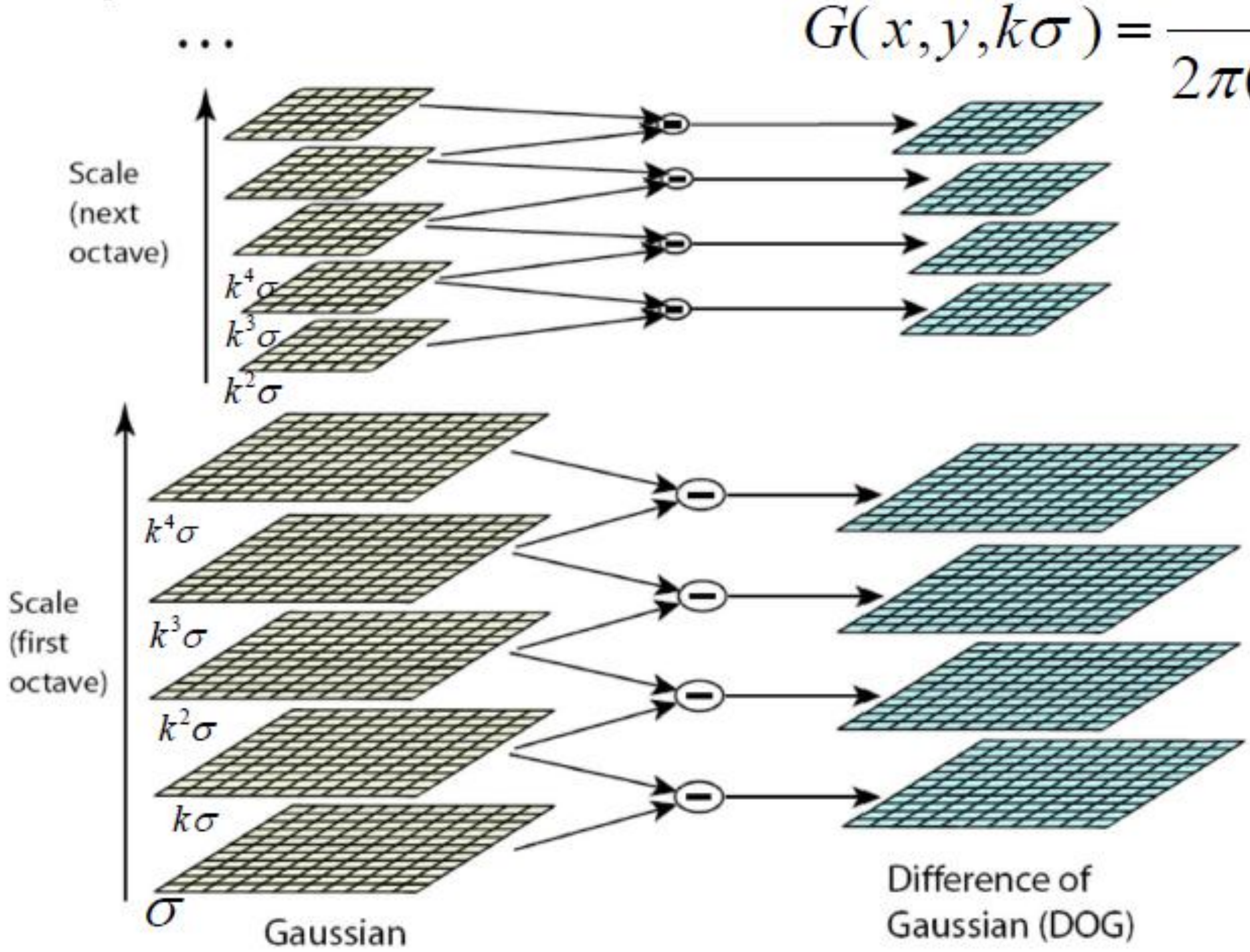




$\Rightarrow$ List of (x, y, σ)

Building a Scale Space

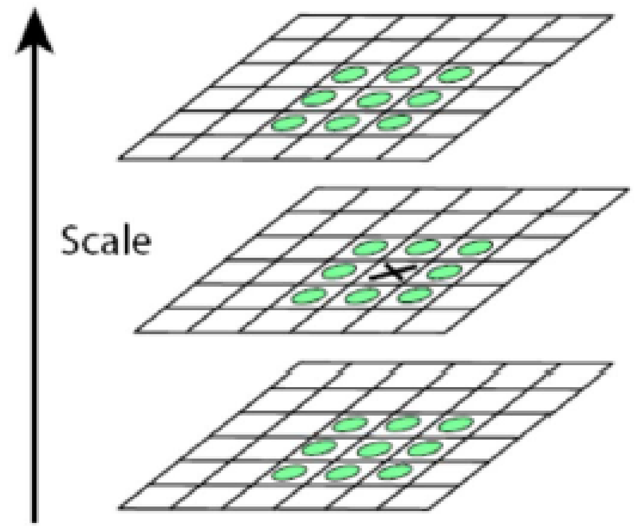
$$G(x,y,k\sigma) = \frac{1}{2\pi(k\sigma)^2} e^{-(x^2+y^2)/2k^2\sigma^2}$$



$$k = \sqrt{2}$$

Scale Space Peak Detection

- Compare a pixel (**X**) with 26 pixels in current and adjacent scales (Green Circles)
- Select a pixel (**X**) if larger/smaller than all 26 pixels
- Large number of extrema, computationally expensive
  - Detect the most stable subset with a coarse sampling of scales



Keypoint Localistaion

- This stage attempts to eliminate more points from the list of keypoints by finding those that have low contrast or are poorly localised on an edge. This is achieved by calculating the Laplacian value for each keypoint found in stage 1. The location of extremum, **z**, is given by:

$$-\frac{\partial^2 D}{\partial \mathbf{x}^2}^{-1}\frac{\partial D}{\partial \mathbf{x}}$$

- If the function value at **z** is below a threshold value then this point is excluded. This removes extrema with low contrast.
- To eliminate extrema based on poor localisation it is noted that in these cases there is a large principle curvature across the edge but a small curvature in the perpendicular direction in the Difference of Gaussian function.
- If this difference is below the ratio of largest to smallest eigenvector, from the 2x2 Hessian matrix at the location and scale of the keypoint, the keypoint is rejected.

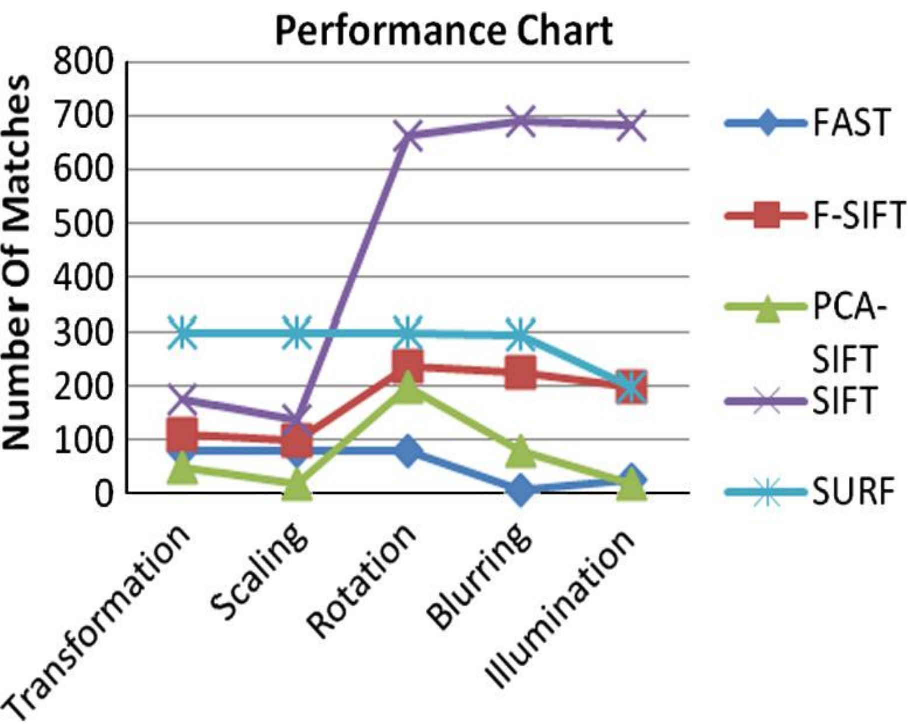
Orientation Assignment

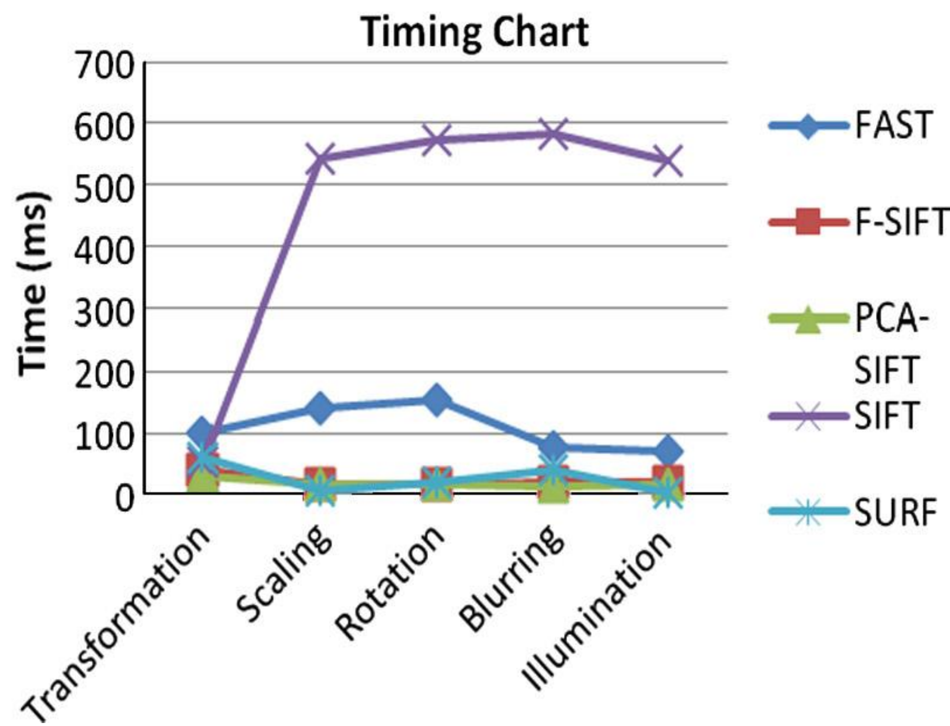
- This step aims to assign a consistent orientation to the keypoints based on local image properties. The keypoint descriptor, can then be represented relative to this orientation, achieving invariance to rotation. The approach taken to find an orientation is:
  - Use the keypoints scale to select the Gaussian smoothed image L, from above
  - Compute gradient magnitude, *m*
  - Compute orientation, θ
  - Form an orientation histogram from gradient orientations of sample points
  - Locate the highest peak in the histogram. Use this peak and any other local peak within 80% of the height of this peak to create a keypoint with that orientation
  - Some points will be assigned multiple orientations
  - Fit a parabola to the 3 histogram values closest to each peak to interpolate the peaks position

Keypoint Descriptor

- The local gradient data, used above, is also used to create keypoint descriptors. The gradient information is rotated to line up with the orientation of the keypoint and then weighted by a Gaussian with variance of 1.5 * keypoint scale. This data is then used to create a set of histograms over a window centred on the keypoint.
- Keypoint descriptors typically uses a set of 16 histograms, aligned in a 4x4 grid, each with 8 orientation bins, one for each of the main compass directions and one for each of the mid-points of these directions. This results in a feature vector containing 128 elements.

SIFT compared to other algorithms



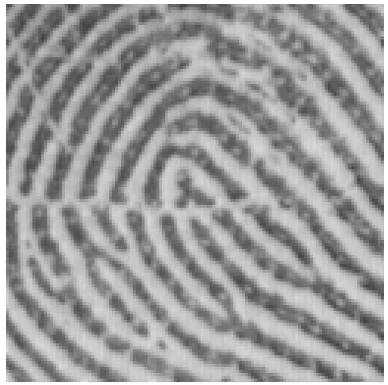


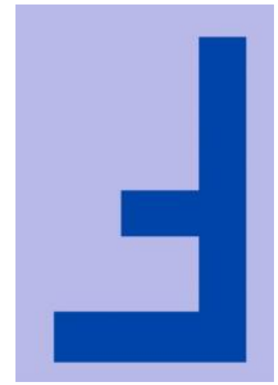
Shape features

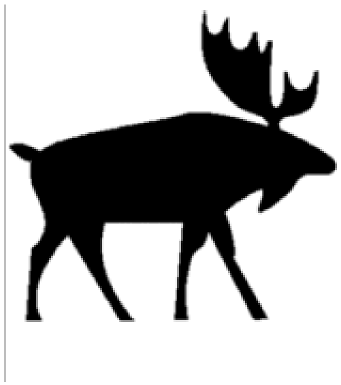
- Shape is probably the most important property that is perceived about objects. It allows to predict more facts about an object than other features, e.g. color (Palmer 1999)
- Thus, recognizing shape is crucial for object recognition. In some applications it may be the only feature present, e.g. logo recognition
- Shape content description is difficult to define because measuring the similarity between shapes is difficult.
- In order to extract object features, we need an image that has undergone image segmentation and any necessary morphological filtering.
- This will provide us with a clearly defined object which can be labeled and processed independently.

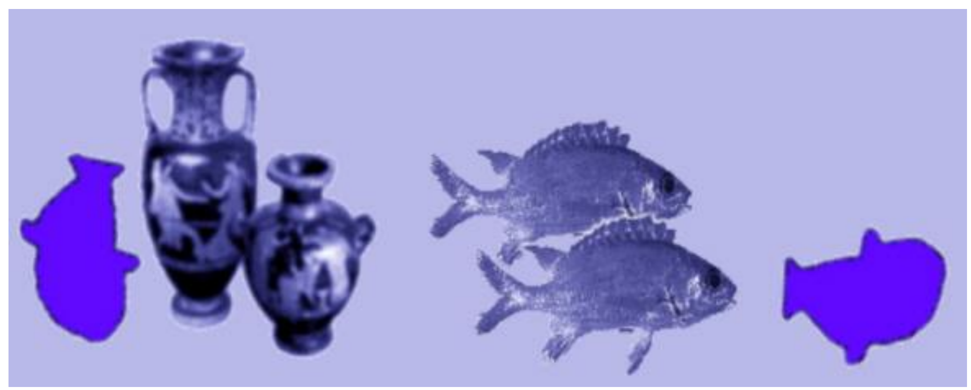
Why Shape ?

These objects are recognized by…



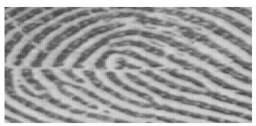
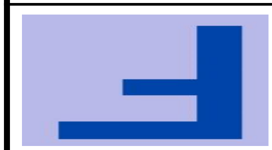






Why Shape ?

These objects are recognized by…

	Texture	Color	Context	Shape
	X	X		
	X			X
				X
				X
				X
			X	X

Object Recognition by Shape

Source:
2D image of a 3D object

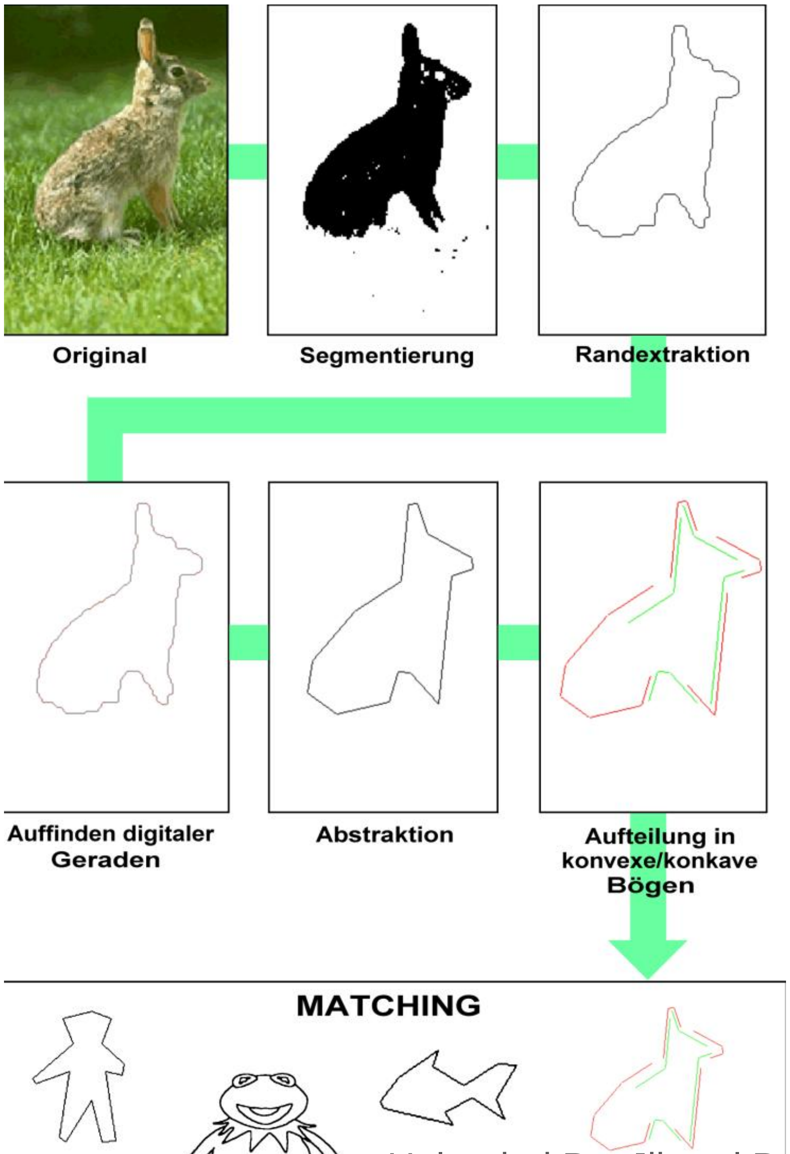
Object Segmentation

Contour Extraction

Contour Cleaning, e.g.,
Evolution

Contour Segmentation

Matching: Correspondence
of Visual Parts



Shape feature categories of applications:

- **Shape retrieval**: searching for all shapes in a typically large database of shapes that are similar to a query shape. Usually all shapes within a given distance from the query are determined or the first few shapes that have the smallest distance.
- **Shape recognition and classification**: determining whether a given shape matches a model sufficiently, or which of representative class is the most similar.
- **Shape alignment and registration**: transforming or translating one shape so that it best matches another shape, in whole or in part.
- **Shape approximation and simplification**: constructing a shape with fewer elements (points, segments, triangles, etc.), so that it is still similar to the original.

Typical problems:

- How to describe shape ?
- What is the matching transformation?
  - No one-to-one correspondence
- Occlusion
- Noise

Shape descriptor

- Shape descriptor is a set of numbers that are produced to represent a given shape feature.
- A descriptor attempts to quantify the shape in ways that agree with human intuition.
- Good retrieval accuracy requires a shape descriptor to be able to effectively find perceptually similar shapes from a database.
- Usually, the descriptors are in the form of a vector.

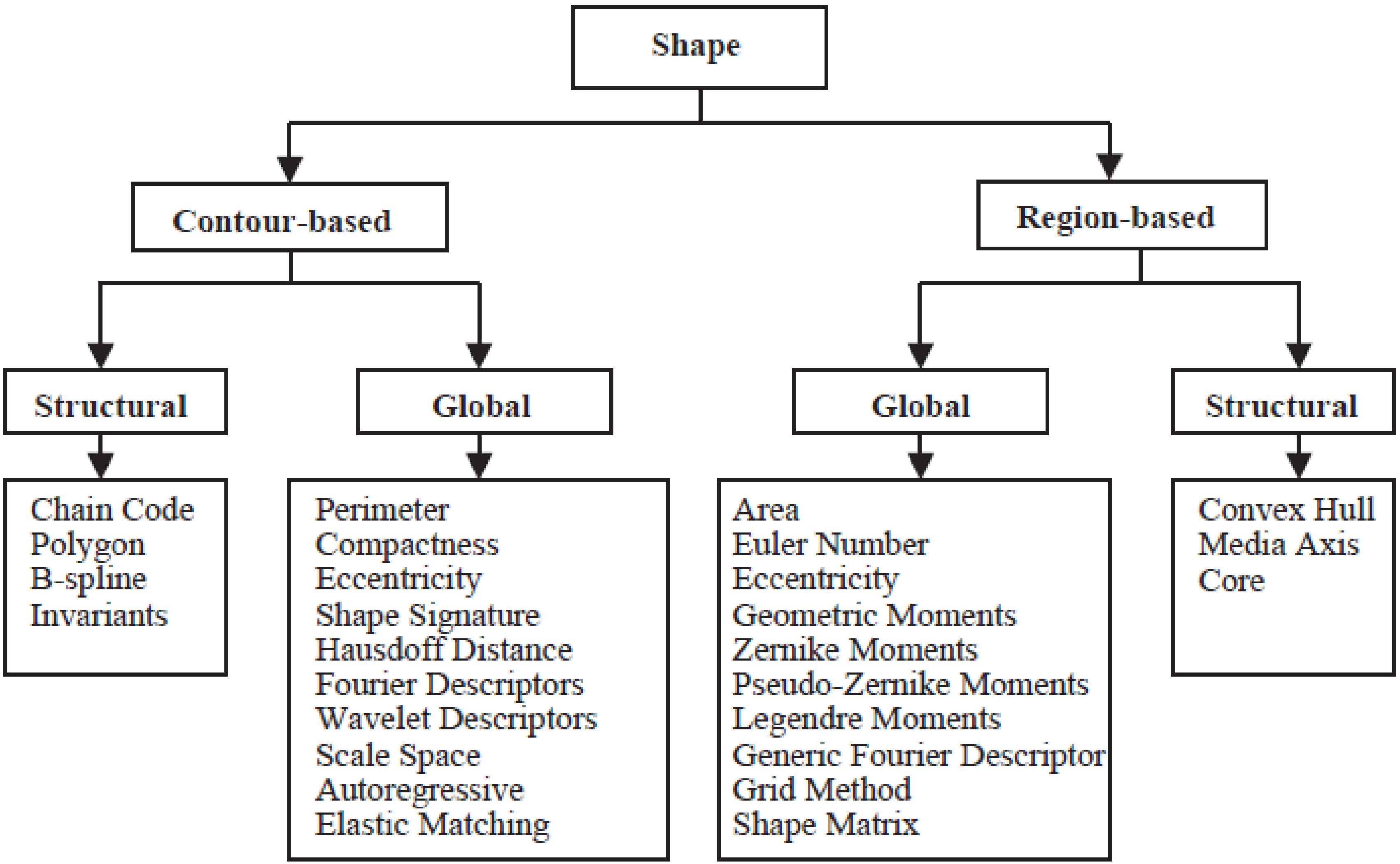
Shape descriptor

- Shape descriptors should meet the following requirements:
  - The descriptors should be as complete as possible to represent the content of the information items.
  - The descriptors should be represented and stored compactly. The size of a descriptor vector must not be too large.
  - The computation of the similarity or the distance between descriptors should be simple; otherwise the execution time would be too long.

Shape features categories

- Shape descriptors can be divided into two main categories: region based and contour-based methods.
  - **Region-based methods** use the whole area of an object for shape description
  - **Contour-based methods** use only the information present in the contour of an object.
- Under each class, the different methods are further divided into structural approaches and global approaches. This sub-class is based on whether the shape is represented as a whole or represented by segments/sections (primitives).
- These approaches can be further distinguished into space domain and transform domain, based on whether the shape features are derived from the spatial domain or from the transformed domain.

Shape features categories



Boundary Descriptors

1. **Simple boundary descriptors:** we can use
   - Length of the boundary
   - The size of smallest circle or box that can totally enclosing the object

2. **Chain Code**

3. **Polygon Approximation**

4. **Shape number**

5. **Fourier descriptor**

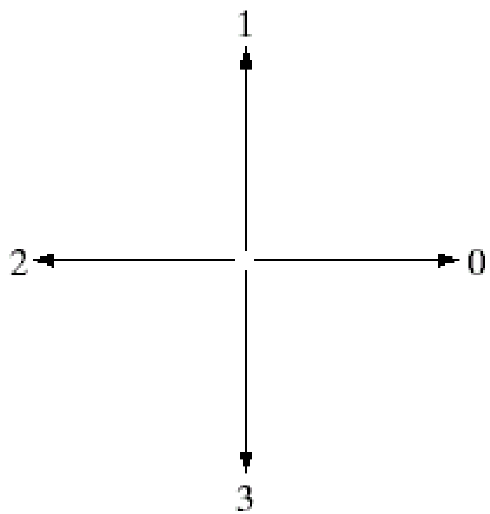
6. **Statistical moments**

Shape Representation by Using Chain Codes

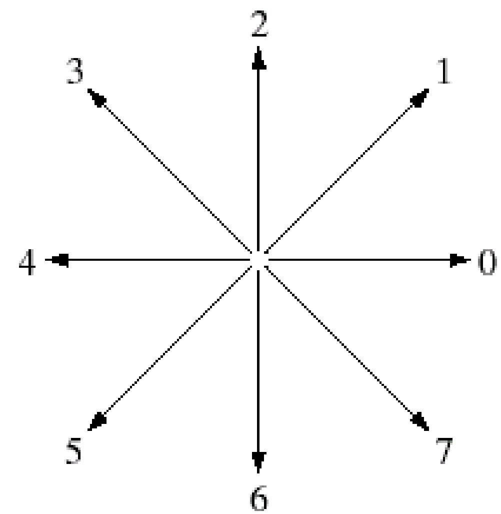
Why we focus on a boundary?

The boundary is a good representation of an object shape and also requires a few memory.

Chain codes: represent an object boundary by a connected sequence of straight line segments of specified length and direction.



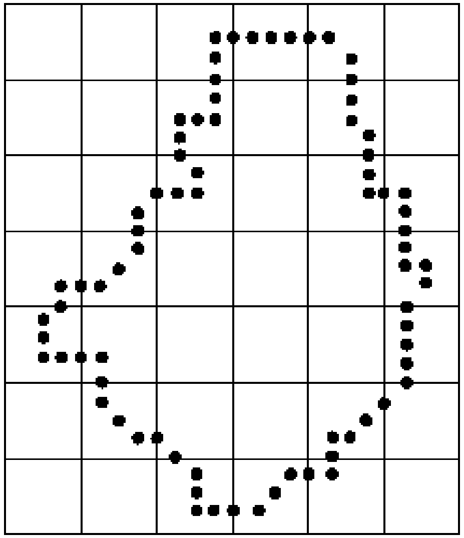
4-directional chain code

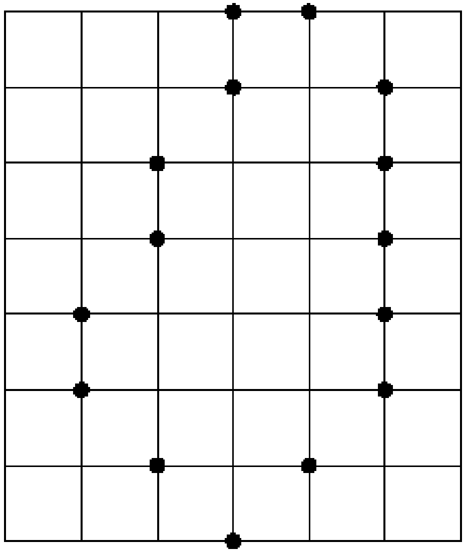


8-directional chain code

Examples of Chain Codes

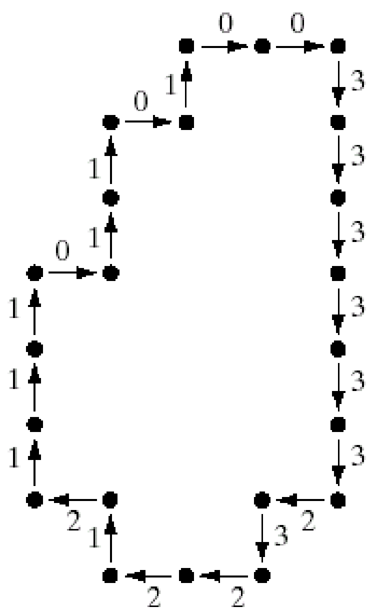
Object boundary (resampling)

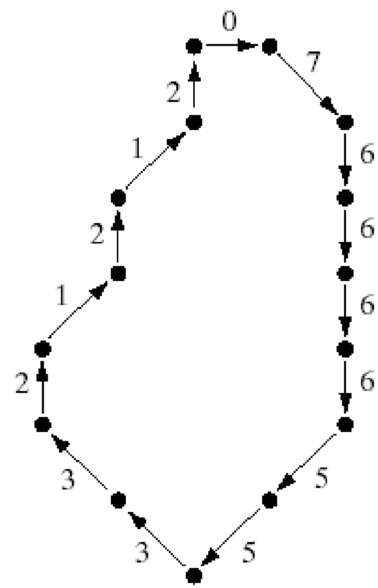




Boundary vertices

4-directional chain code





8-directional chain code

The First Difference of a Chain Codes

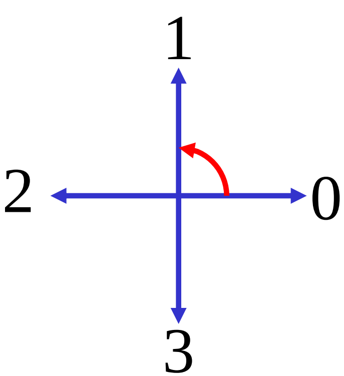
Problem of a chain code:

a chain code sequence depends on a starting point.

Solution: treat a chain code as a circular sequence and redefine the starting point so that the resulting sequence of numbers forms an integer of minimum magnitude.

The first difference of a chain code: counting the number of direction change (in counterclockwise) between 2 adjacent elements of the code.

Example:



Chain code :	The first difference
0 → 1	1
0 → 2	2
0 → 3	3
2 → 3	1
2 → 0	2
2 → 1	3

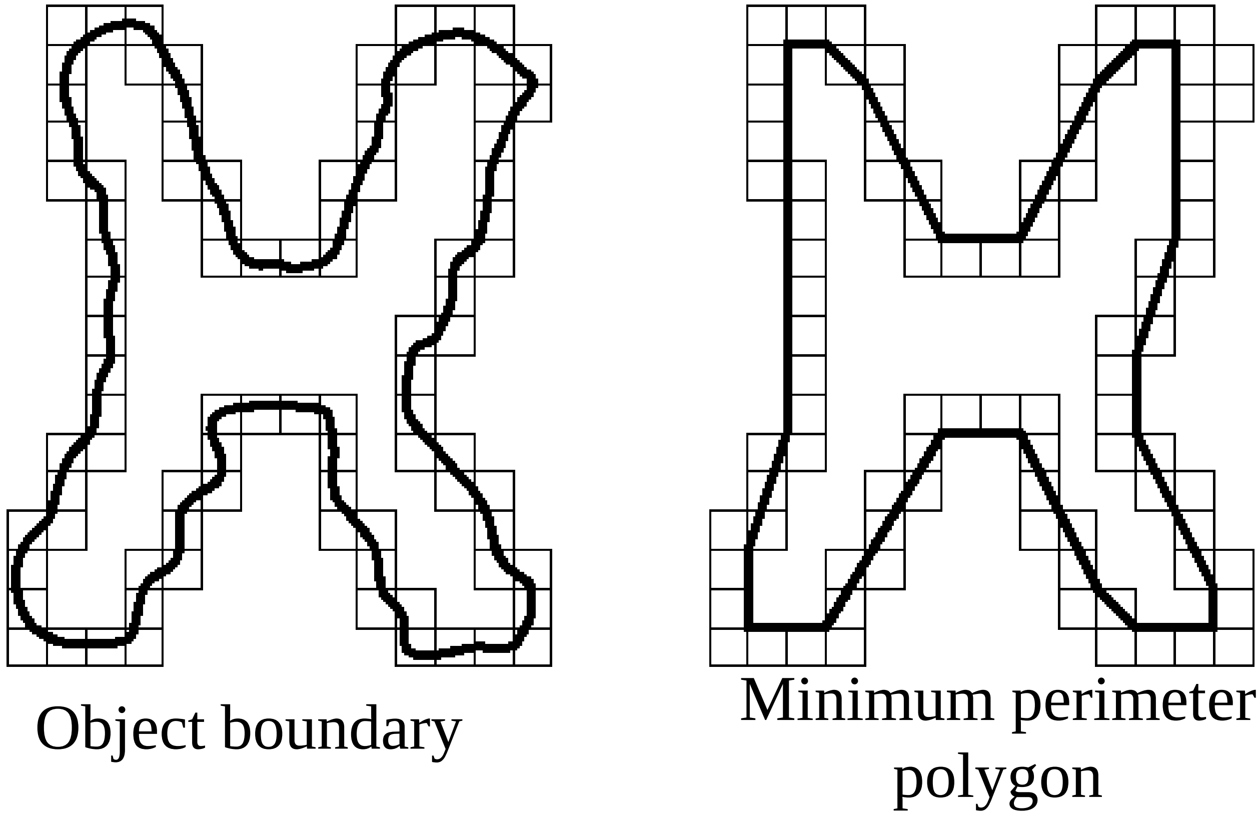
Example:

- a chain code: 10103322
- The first difference = 3133030
- Treating a chain code as a circular sequence, we get the first difference = 33133030

The first difference is rotational invariant.

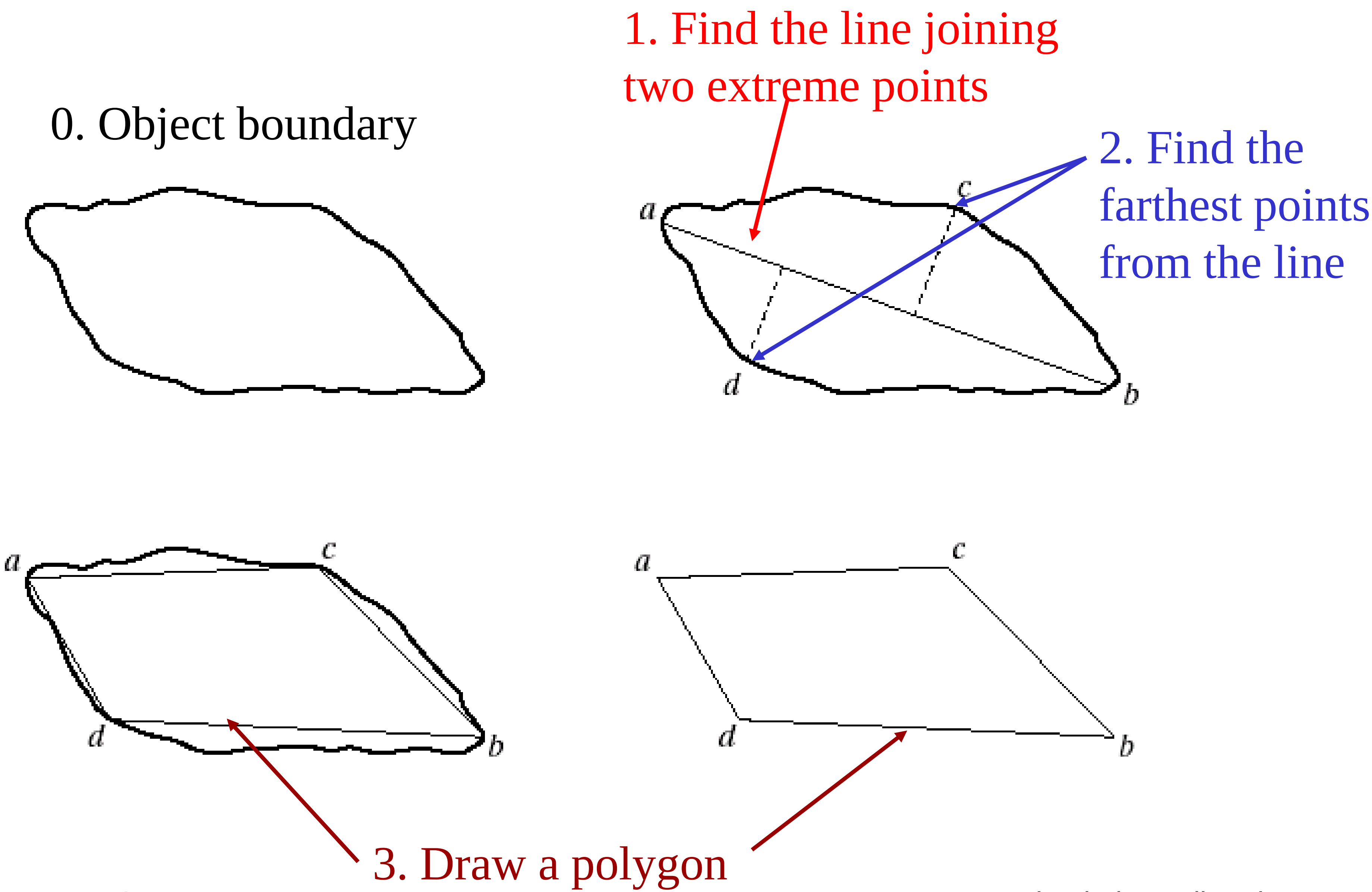
Polygon Approximation

Represent an object boundary by a polygon



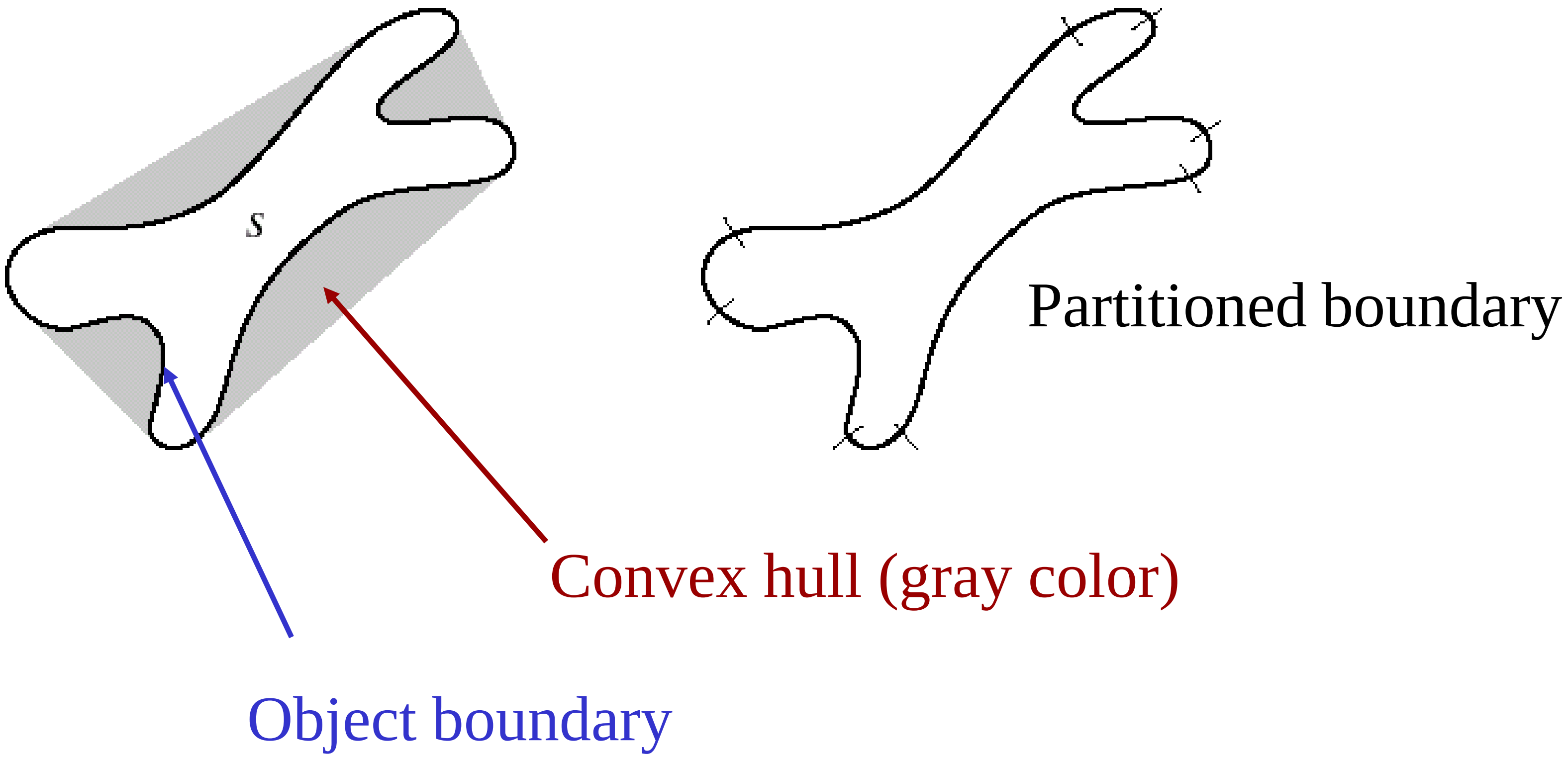
Minimum perimeter polygon consists of line segments that minimize distances between boundary pixels.

Polygon Approximation:Splitting Techniques



Boundary Segments

Concept: Partitioning an object boundary by using vertices of a convex hull.



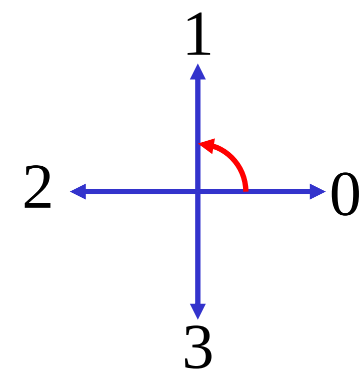
Shape Number

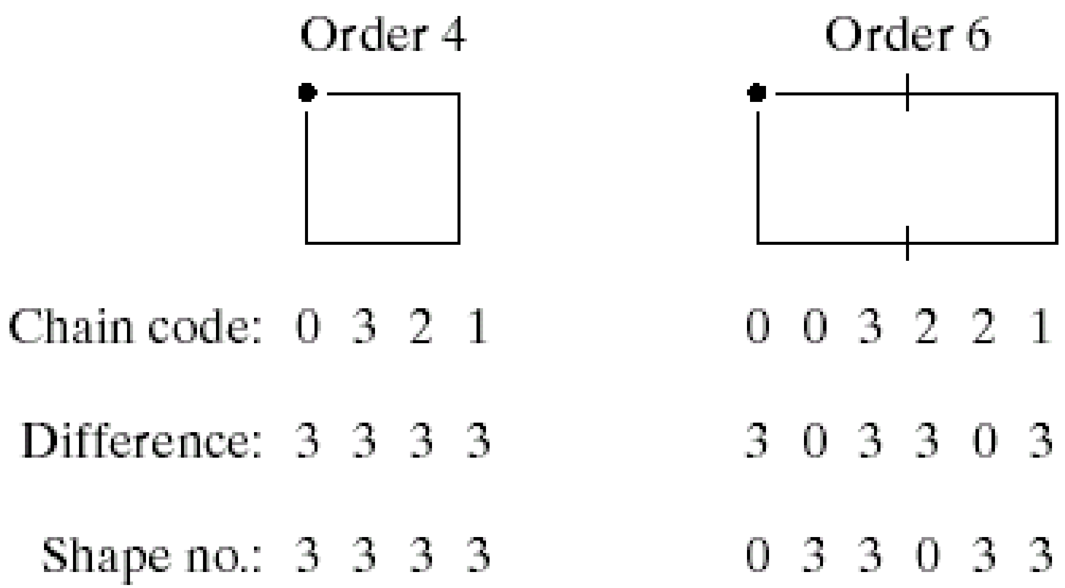
Shape number of the boundary definition:

the first difference of smallest magnitude

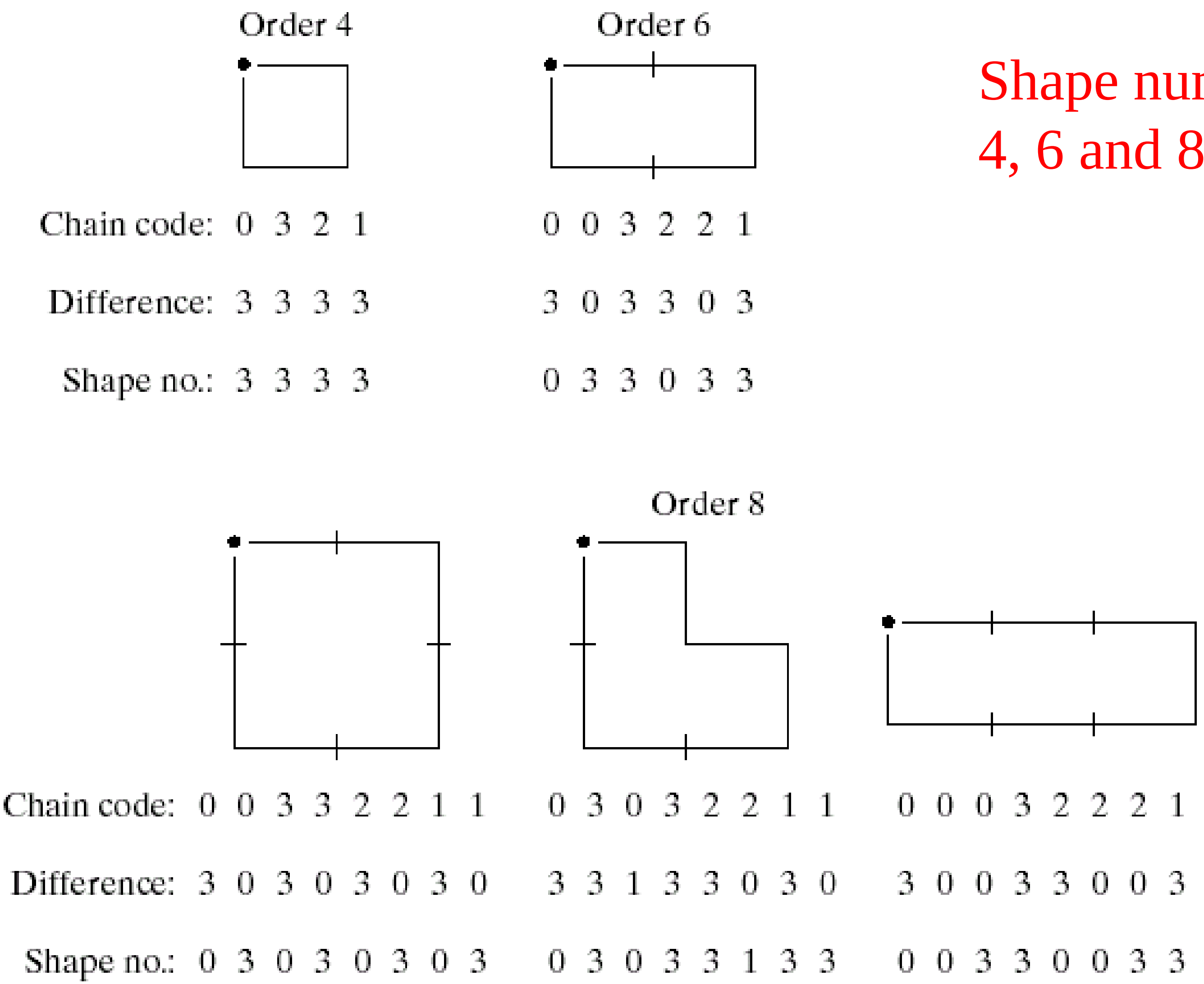
The order n of the shape number:

the number of digits in the sequence



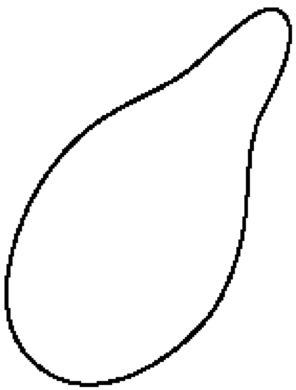


Shape Number (cont.)

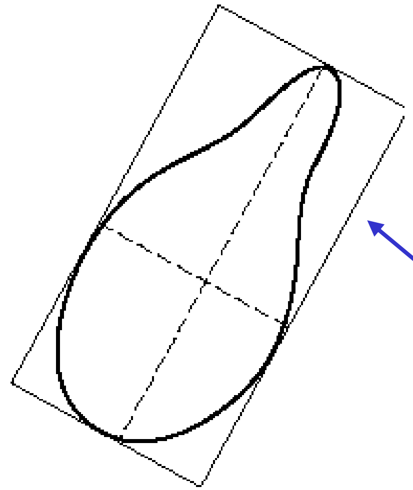


Shape numbers of order 4, 6 and 8

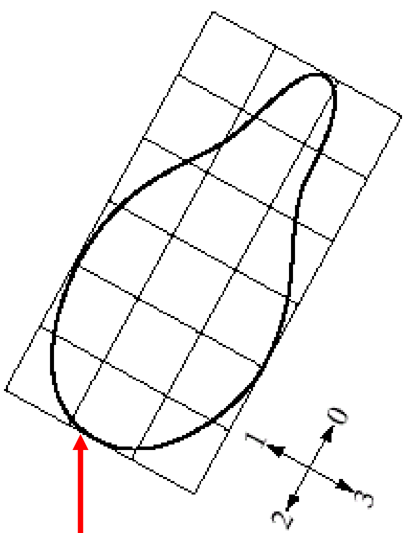
Example: Shape Number



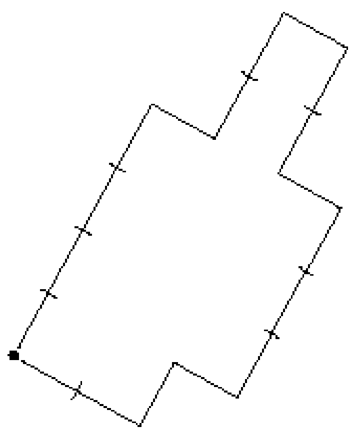
1. Original boundary



2. Find the smallest rectangle that fits the shape



3. Create grid



4. Find the nearest Grid.

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

First 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

Fourier Descriptor

Fourier descriptor: view a coordinate (x,y) as a complex number (x = real part and y = imaginary part) then apply the Fourier transform to a sequence of boundary points.

Let s(k) be a coordinate of a boundary point k :

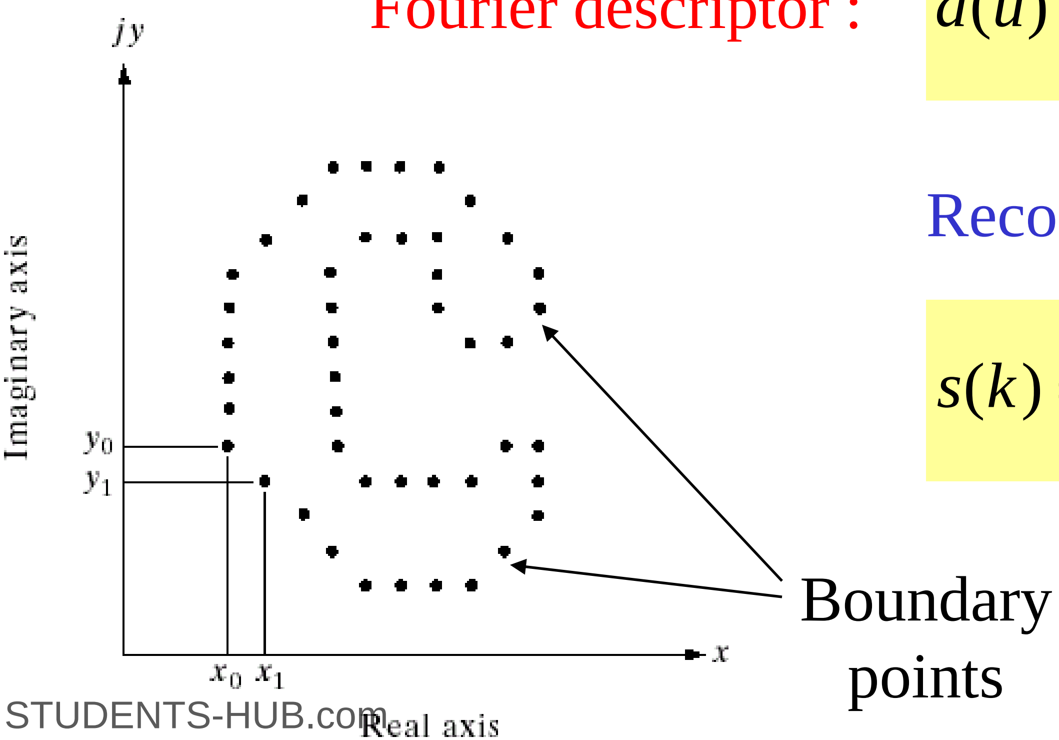
$$s(k) = x(k) + jy(k)$$

Fourier descriptor :

$$a(u) = \frac{1}{K}\sum_{k=0}^{K-1} s(k) e^{-2\pi uk/K}$$

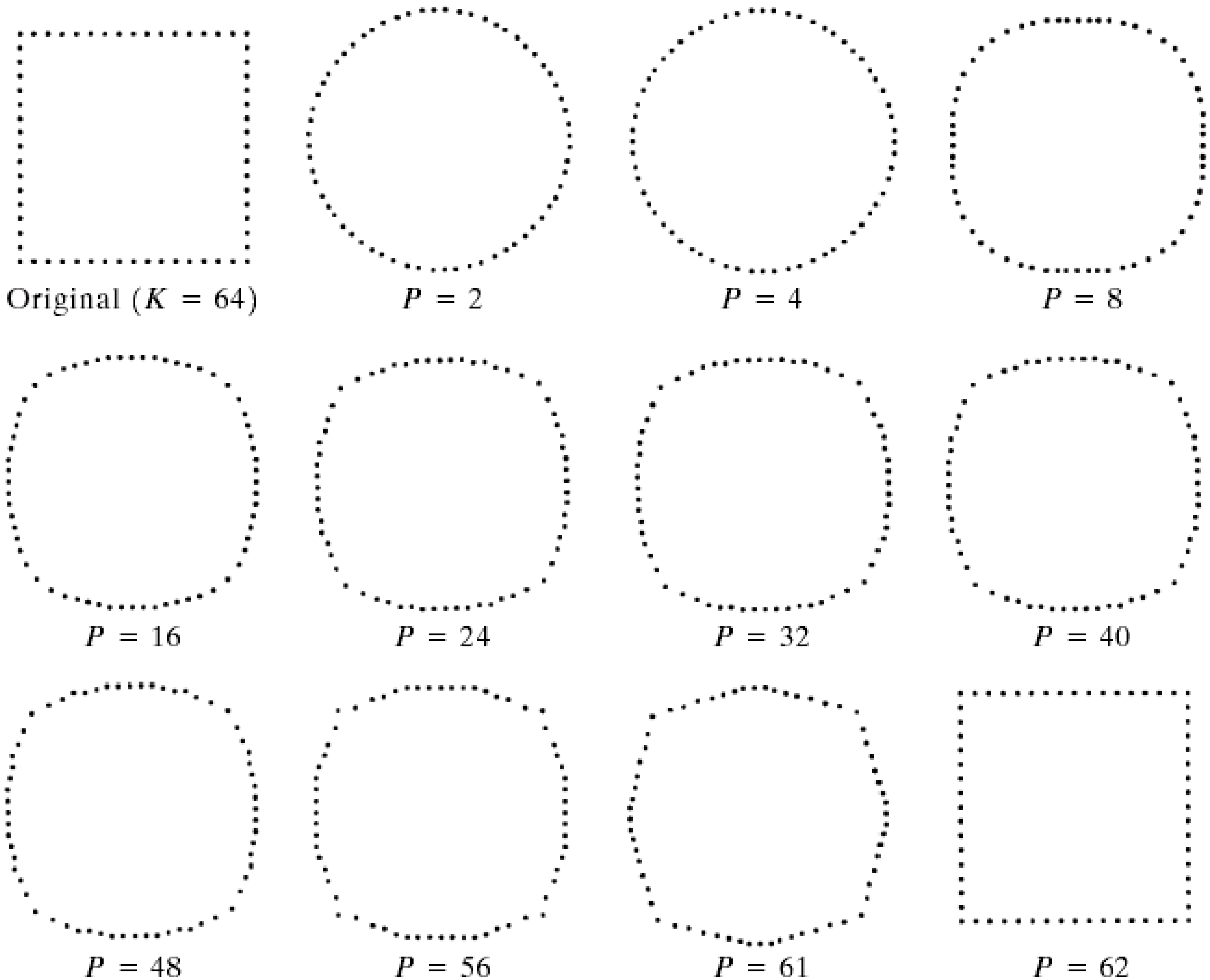
Reconstruction formula

$$s(k) = \frac{1}{K}\sum_{k=0}^{K-1} a(u) e^{2\pi uk/K}$$



Example: Fourier Descriptor

Examples of reconstruction from Fourier descriptors



$$\hat{s}(k) = \frac{1}{K}\sum_{k=0}^{P-1} a(u)e^{2\pi uk/K}$$

P is the number of Fourier coefficients used to reconstruct the boundary

Fourier Descriptor Properties

Some 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	$s_p(k) = s(k - k_0)$	$a_p(u) = a(u)e^{-j2\pi k_0 u/K}$

Statistical Moments

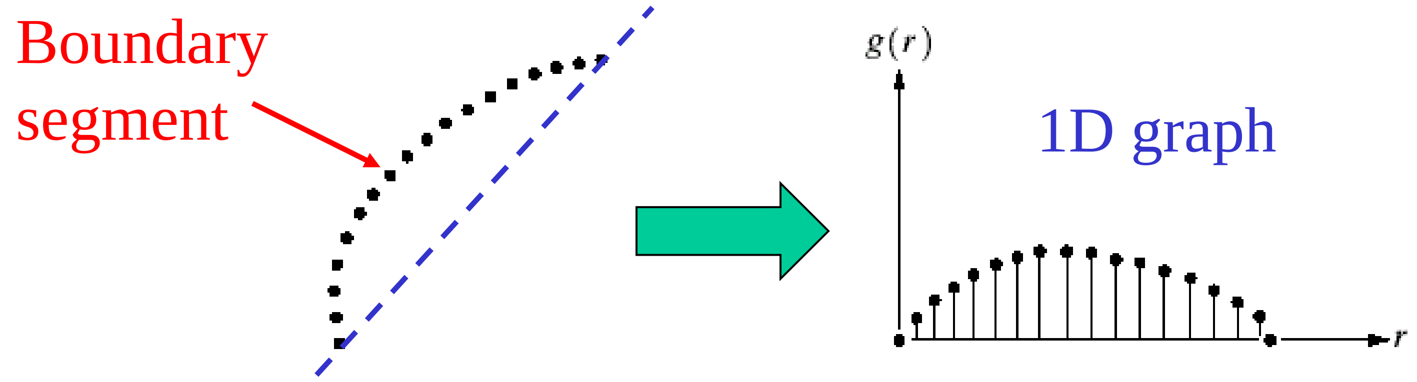
Definition: the n^{th} moment

$$\mu_n(r) = \sum_{i=0}^{K-1} (r_i - m)^n g(r_i)$$

where

$$m = \sum_{i=0}^{K-1} r_i g(r_i)$$

Example of moment:
The first moment = mean
The second moment = variance



1. Convert a boundary segment into 1D graph
2. View a 1D graph as a PDF function
3. Compute the n^{th} order moment of the graph

Regional Descriptors

Purpose: to describe regions or “areas”

1. Some simple regional descriptors

- area of the region
- length of the boundary (perimeter) of the region
- Compactness

$$C = \frac{A(R)}{P^2(R)}$$

where $A(R)$ and $P(R)$ = area and perimeter of region R

Example: a circle is the most compact shape with C = 1/4π

2. Topological Descriptors

3. Texture

4. Moments of 2D Functions

Example: Regional Descriptors

Infrared image of America at night

White pixels represent "light of the cities"

Region no.	% of white pixels compared to the total white pixels
1	20.4%
2	64.0%
3	4.9%
4	10.7%

Topological Descriptors

Use to describe holes and connected components of the region

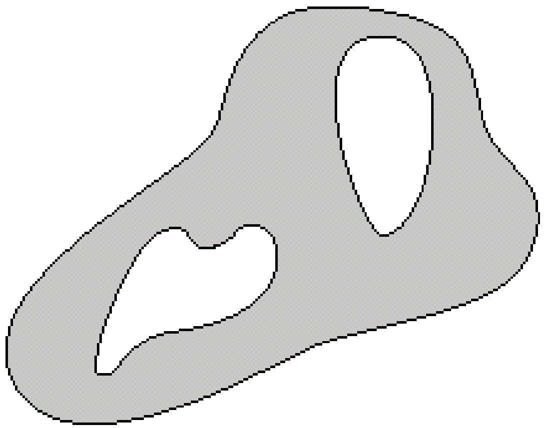


FIGURE 11.17 A region with two holes.

Euler number (E):

$$E = C - H$$

C = the number of connected components

H = the number of holes

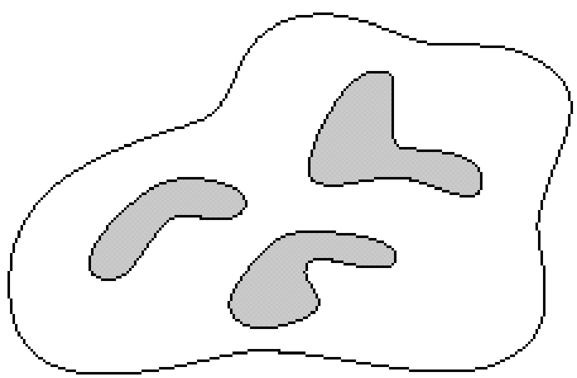
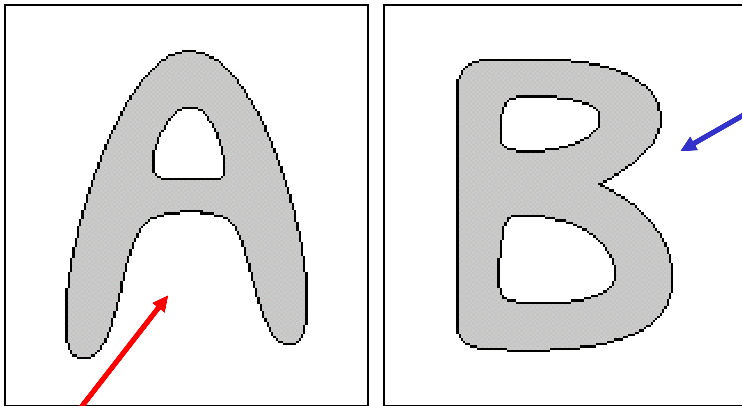


FIGURE 11.18 A region with three connected components.

Topological Descriptors (cont.)

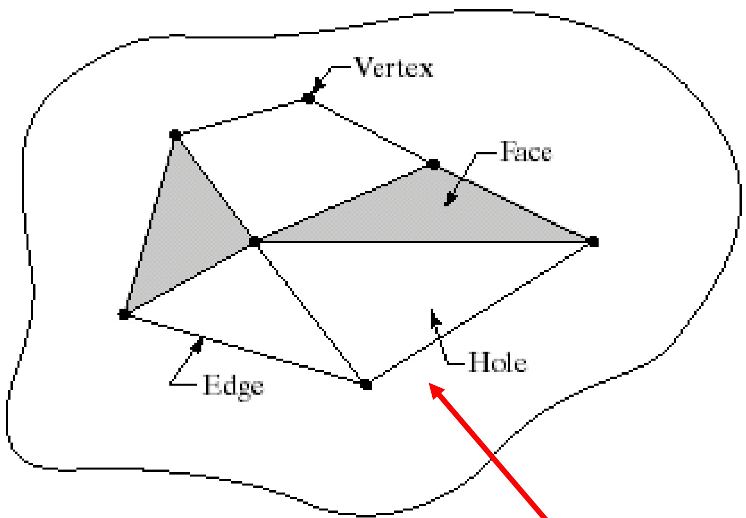


E = -1

E = 0

Euler Formula

$$V - Q + F = C - H = E$$



E = -2

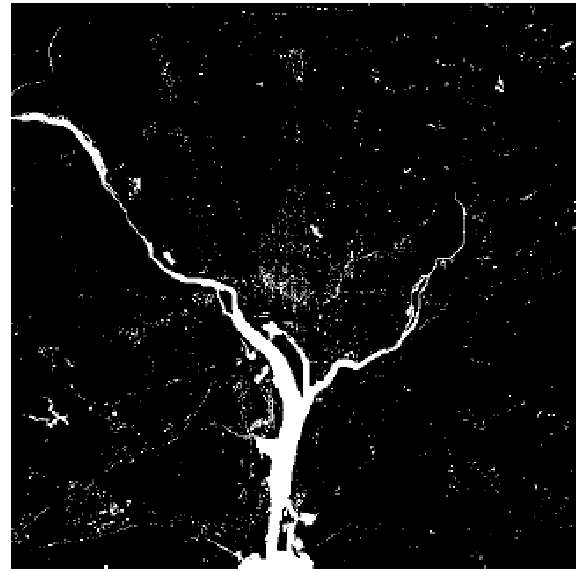
V = the number of vertices

Q = the number of edges

F = the number of faces

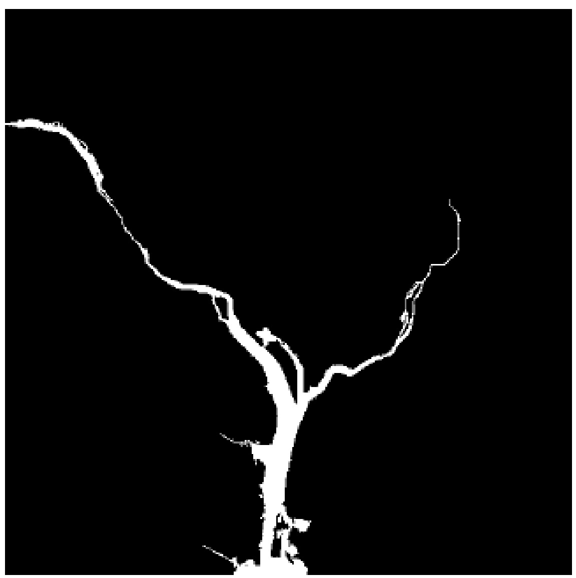
Example: Topological Descriptors

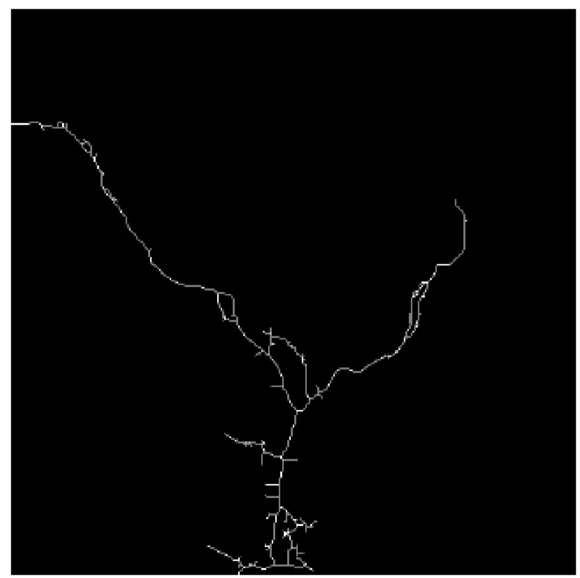
Original image:
Infrared image
Of Washington
D.C. area



After intensity
Thresholding
(1591 connected
components
with 39 holes)
Euler no. = 1552

The largest
connected
area
(8479 Pixels)
(Hudson river)



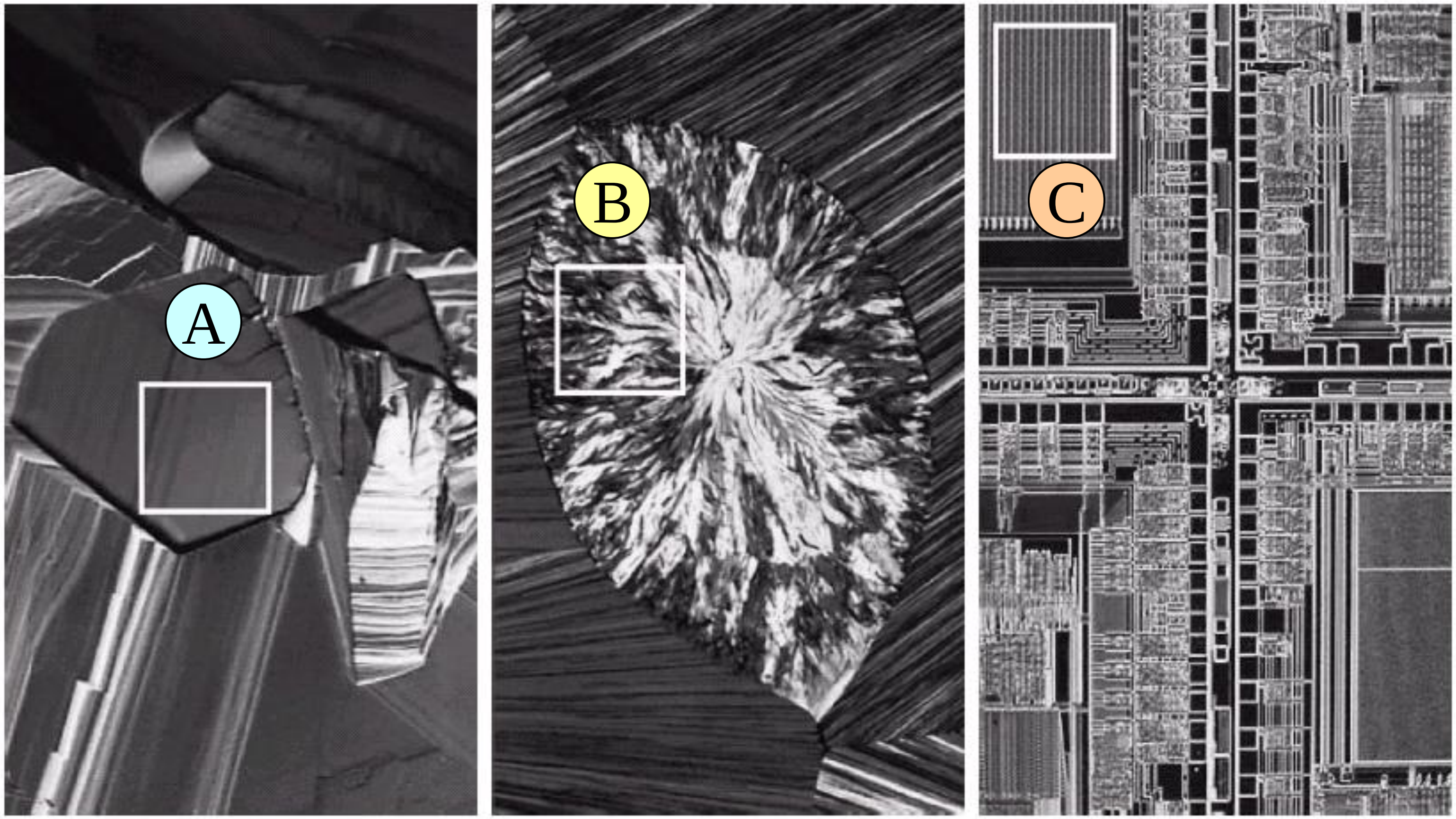


After thinning

Texture Descriptors

Purpose: to describe "texture" of the region.

Examples: optical microscope images:



Superconductor (smooth texture) | Cholesterol (coarse texture) | Microprocessor (regular texture)

Statistical Approaches for Texture Descriptors

We can use statistical moments computed from an image histogram:

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

z = intensity
$p(z)$ = PDF or histogram of z

where

$$m = \sum_{i=0}^{K-1} z_i p(z_i)$$

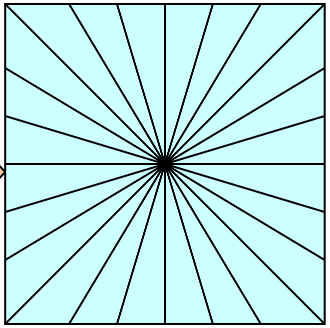
Example: The 2nd moment = variance → measure "smoothness"
The 3rd moment → measure "skewness"
The 4th moment → measure "uniformity" (flatness)

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

Fourier Approach for Texture Descriptor

Concept: convert 2D spectrum into 1D graphs

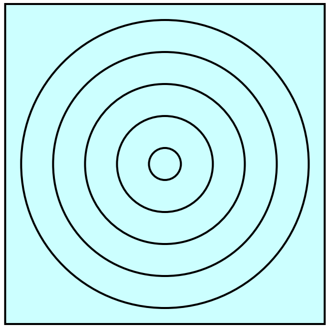
Original image → FFT2D +FFTSHIFT → Fourier coefficient image → Divide into areas by angles



Sum all pixels in each area

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

Fourier coefficient image → Divide into areas by radius

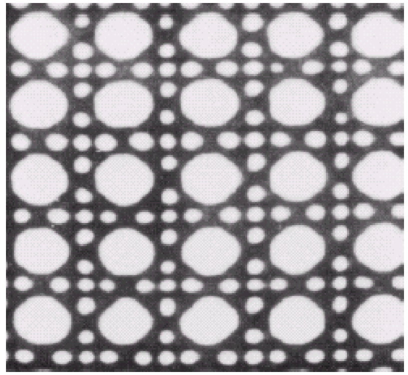


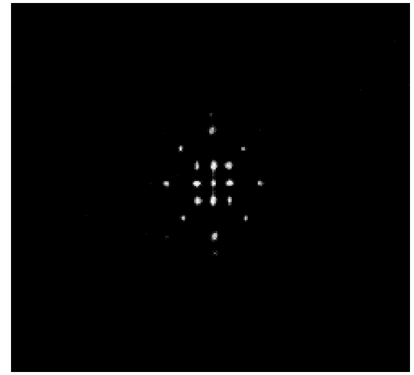
Sum all pixels in each area

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

Fourier Approach for Texture Descriptor

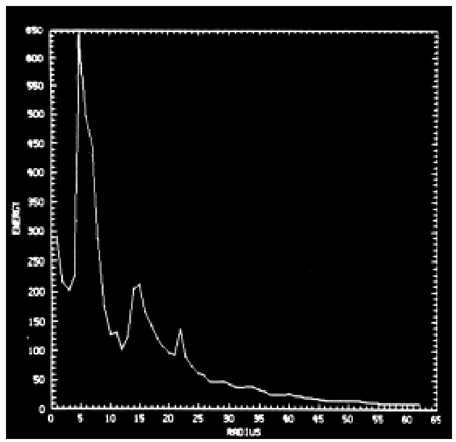
Original image

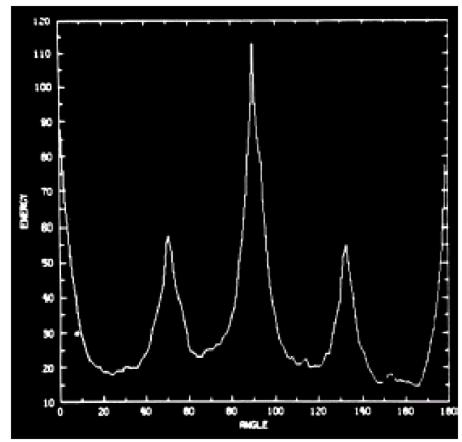




2D Spectrum (Fourier Tr.)

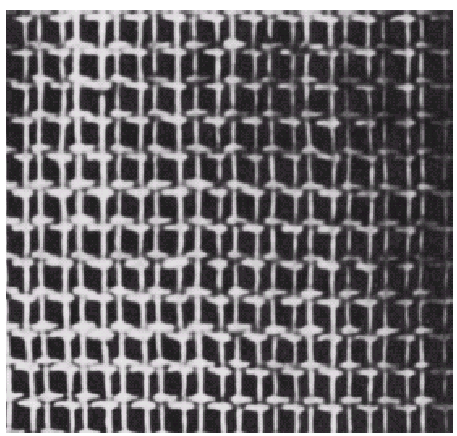
$S(r)$

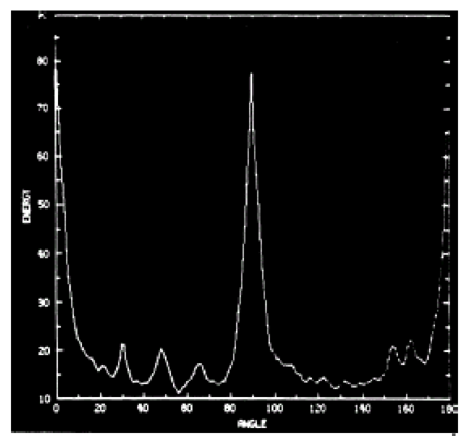




$S(\theta)$

Another image





Another $S(\theta)$

(Images from Rafael C. Gonzalez and Richard E. Wood, Digital Image Processing, 2nd Edition.

Moments of Two-D Functions

The moment of order *p* + *q*

$$m_{pq} = \sum_x \sum_y x^p y^q f(x,y)$$

$$\bar{x} = \frac{m_{10}}{m_{00}}$$

$$y = \frac{m_{01}}{m_{00}}$$

The central moments of order *p* + *q*

$$\mu_{pq} = \sum_x \sum_y (x-\bar{x})^p (y-\bar{y})^q f(x,y)$$

$$\mu_{00} = m_{00}$$

$$\mu_{01} = \mu_{10} = 0$$

$$\mu_{11} = m_{11} - \bar{x}m_{01} = m_{11} - \bar{y}m_{10}$$

$$\mu_{20} = m_{20} - \bar{x}m_{10}$$

$$\mu_{02} = m_{02} - \bar{y}m_{01}$$

$$\mu_{21} = m_{21} - 2\bar{x}m_{11} - \bar{y}m_{20} + 2\bar{x}^2 m_{01}$$

$$\mu_{30} = m_{30} - 3\bar{x}m_{20} + 2\bar{x}^2 m_{10}$$

$$\mu_{12} = m_{12} - 2\bar{y}m_{11} - \bar{x}m_{02} + 2\bar{y}^2 m_{10}$$

$$\mu_{03} = m_{03} - 3\bar{y}m_{02} + 2\bar{y}^2 m_{01}$$

Invariant Moments of Two-D Functions

The normalized central moments of order $p+q$

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{\gamma}} \qquad \text{where} \qquad \gamma = \frac{p+q}{2} + 1$$

Invariant moments: independent of rotation, translation, scaling, and reflection

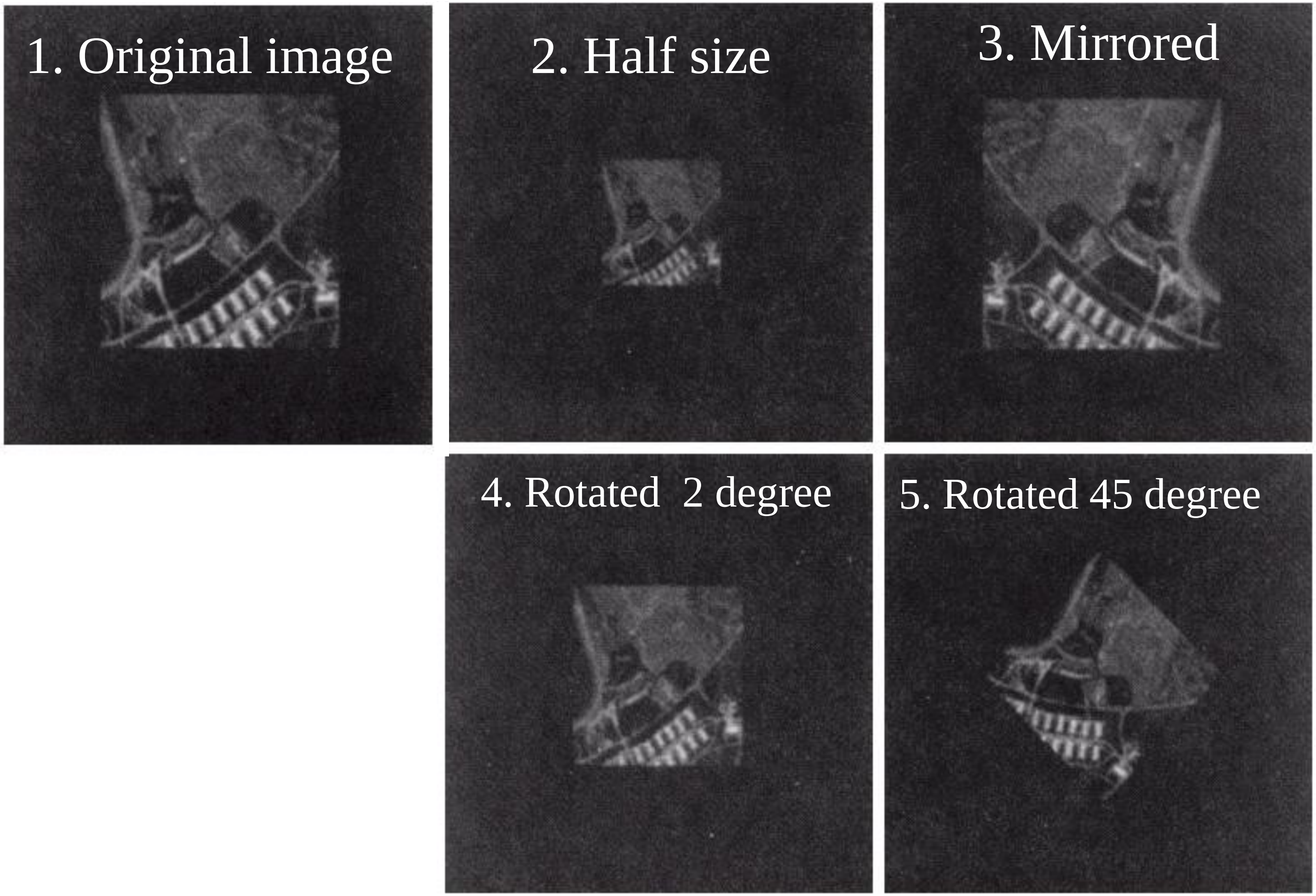
$$\phi_1 = \eta_{20} + \eta_{02} \qquad \phi_2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2$$

$$\phi_3 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2 \qquad \phi_4 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2$$

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

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

Example: Invariant Moments of Two-D Functions



Example: Invariant Moments of Two-D Functions

Invariant moments of images in the previous slide

Invariant (Log)	Original	Half Size	Mirrored	Rotated 2°	Rotated 45°
ϕ_1	6.249	6.226	6.919	6.253	6.318
ϕ_2	17.180	16.954	19.955	17.270	16.803
ϕ_3	22.655	23.531	26.689	22.836	19.724
ϕ_4	22.919	24.236	26.901	23.130	20.437
ϕ_5	45.749	48.349	53.724	46.136	40.525
ϕ_6	31.830	32.916	37.134	32.068	29.315
ϕ_7	45.589	48.343	53.590	46.017	40.470

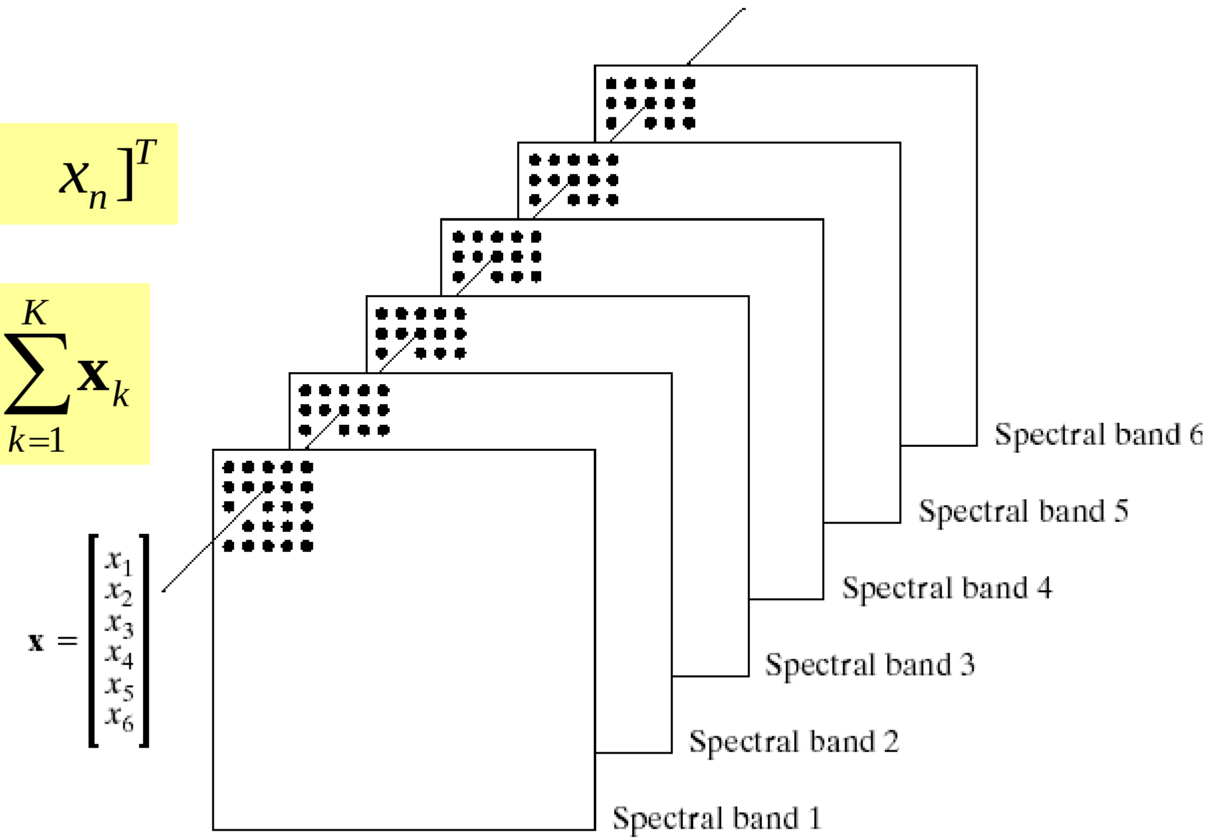
Invariant moments are independent of rotation, translation, scaling, and reflection

Principal Components for Description

Purpose: to reduce dimensionality of a vector image while maintaining information as much as possible.

Let $\mathbf{x} = [x_1 \quad x_2 \quad \ldots \quad x_n]^T$

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



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

Hotelling transformation

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

Where A is created from eigenvectors of C_x as follows
Row 1 contain the 1st eigenvector with the largest eigenvalue.
Row 2 contain the 2nd eigenvector with the 2nd largest eigenvalue.
....

Then we get

$$\mathbf{m}_y = E\{\mathbf{y}\} = 0$$

$$\mathbf{C}_y = \mathbf{A}\mathbf{C}_x\mathbf{A}^T$$

and

$$\mathbf{C}_y = \begin{bmatrix} \lambda_1 & 0 & \ldots & 0 \\ 0 & \lambda_1 & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots \\ 0 & \ldots & \ldots & \lambda_1 \end{bmatrix}$$

Then elements of $\mathbf{y} = \mathbf{A}(\mathbf{x} - \mathbf{m}_x)$ are uncorrelated. The component of y with the largest λ is called the principal component.

Eigenvector and Eigenvalue

Eigenvector and eigenvalue of Matrix **C** are defined as

Let **C** be a matrix of size NxN and **e** be a vector of size Nx1. If

$$\mathbf{Ce} = \lambda \mathbf{e}$$

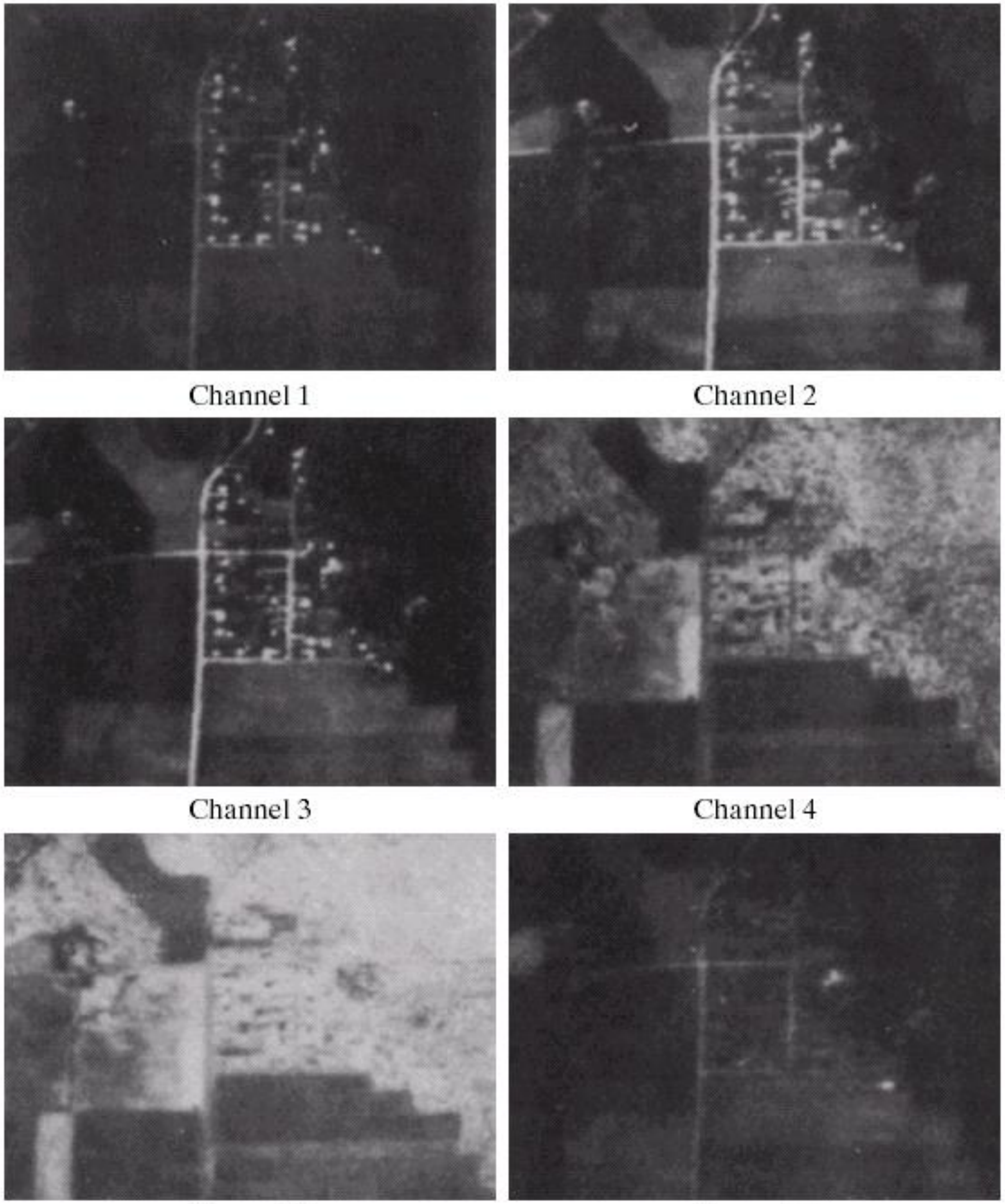
Where λ is a constant

We call **e** as an eigenvector and λ as eigenvalue of **C**

Example: Principal Components

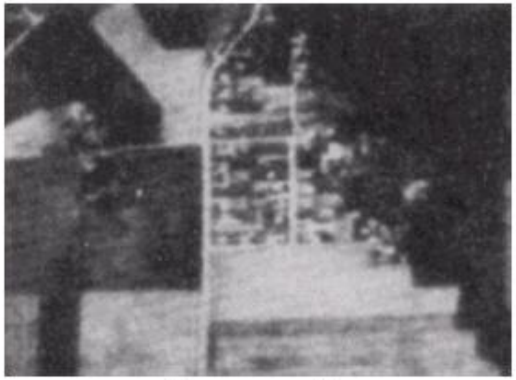
6 spectral images from an airborne Scanner.

Channel	Wavelength band (microns)
1	0.40–0.44
2	0.62–0.66
3	0.66–0.72
4	0.80–1.00
5	1.00–1.40
6	2.00–2.60



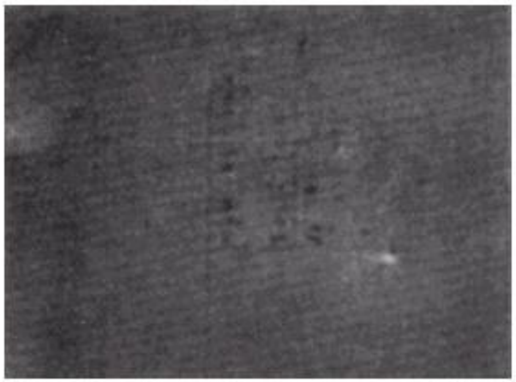
Channel 1
Channel 2
Channel 3
Channel 4
Channel 5
Channel 6

Example: Principal Components (cont.)



Component 1

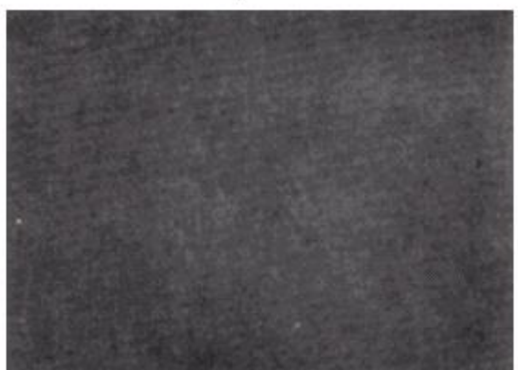
Component 2



Component 3

Component 4

Component 5



Component 6

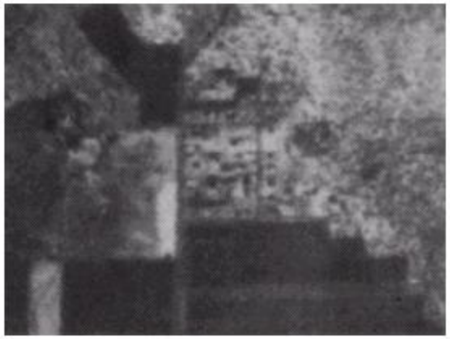
Component	λ
1	3210
2	931.4
3	118.5
4	83.88
5	64.00
6	13.40

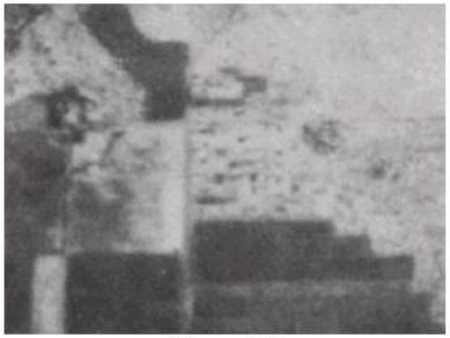
Example: Principal Components (cont.)

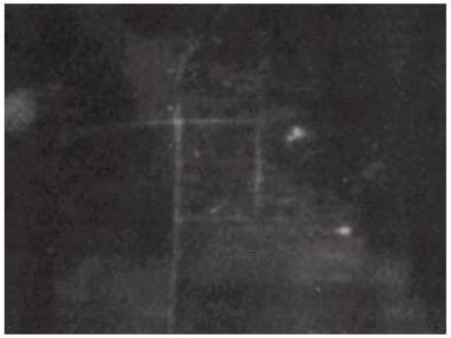

Channel 1


Channel 2

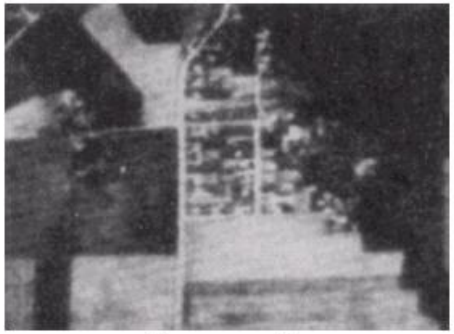

Channel 3


Channel 4

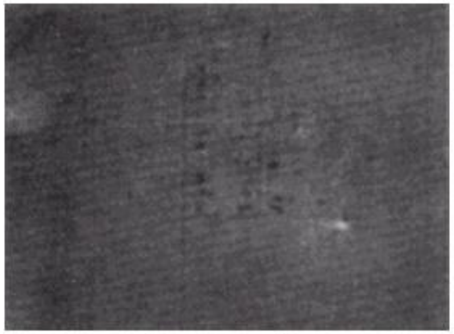

Channel 5


Channel 6

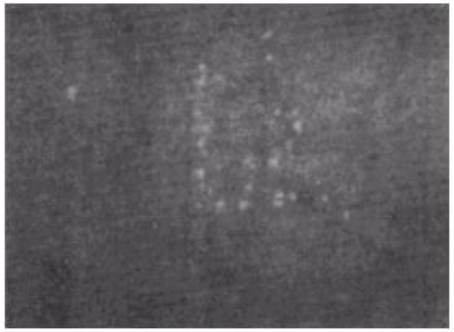
Original image

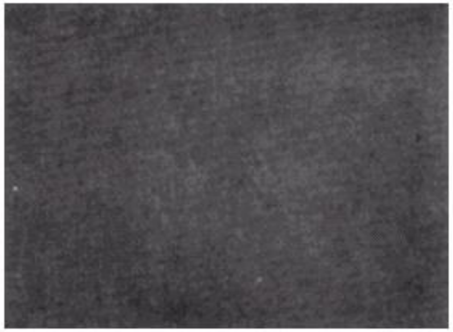

Component 1


Component 2


Component 3


Component 4


Component 5


Component 6

After Hotelling transform

Principal Components for Description

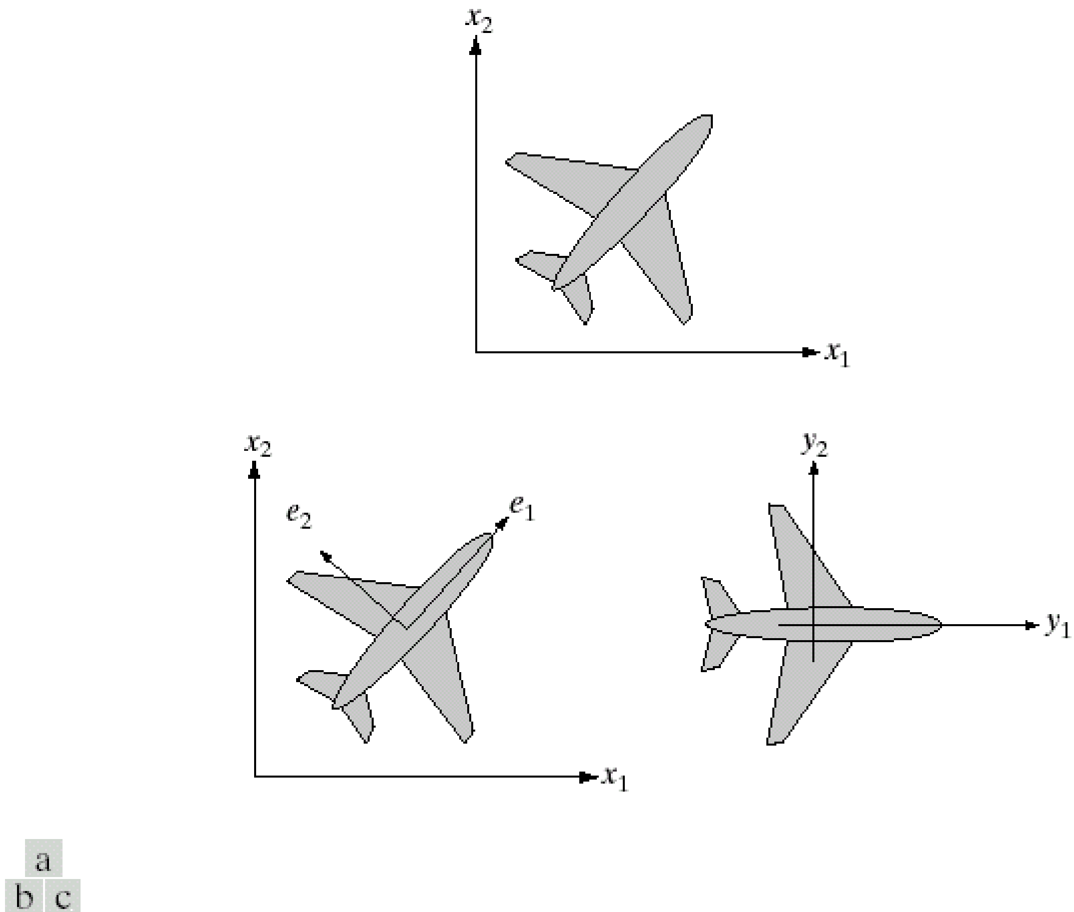
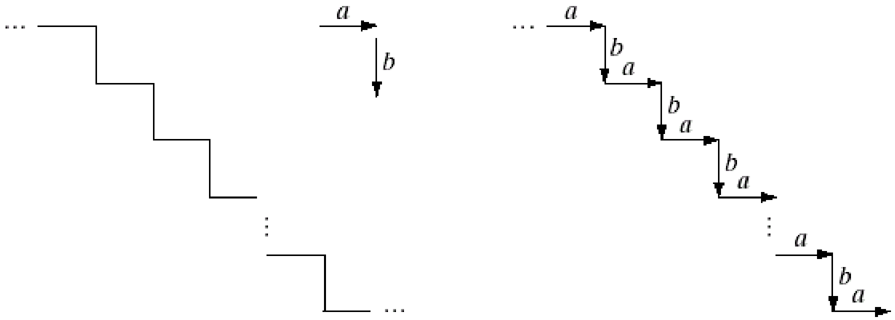


FIGURE 11.29 (a) An object. (b) Eigenvectors. (c) Object rotated by using Eq. (11.4-6). The net effect is to align the object along its eigen axes.

Relational Descriptors

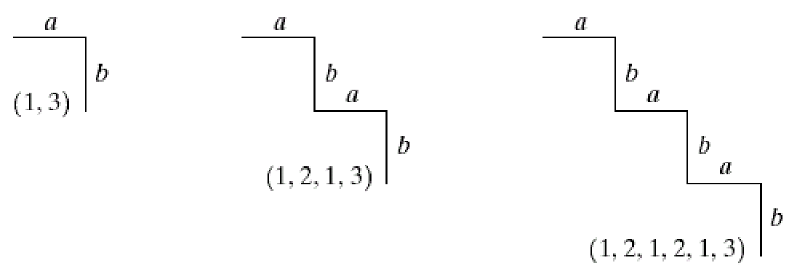


a b

FIGURE 11.30
(a) A simple staircase structure.
(b) Coded structure.

Relational Descriptors

FIGURE 11.31 Sample derivations for the rules $S \rightarrow aA$, $A \rightarrow bS$, and $A \rightarrow b$.



Relational Descriptors

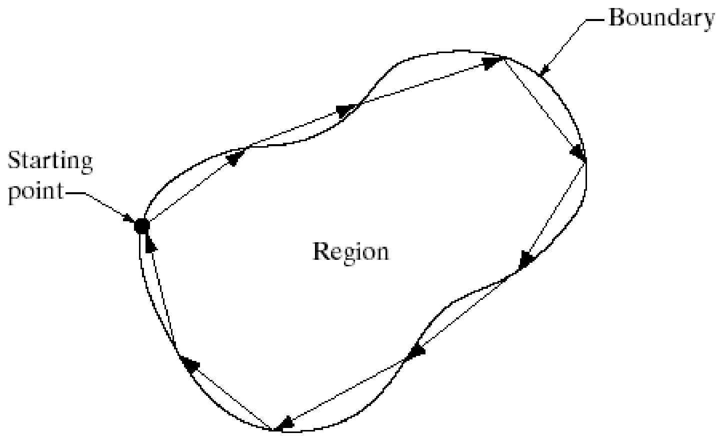


FIGURE 11.32 Coding a region boundary with directed line segments.

Relational Descriptors

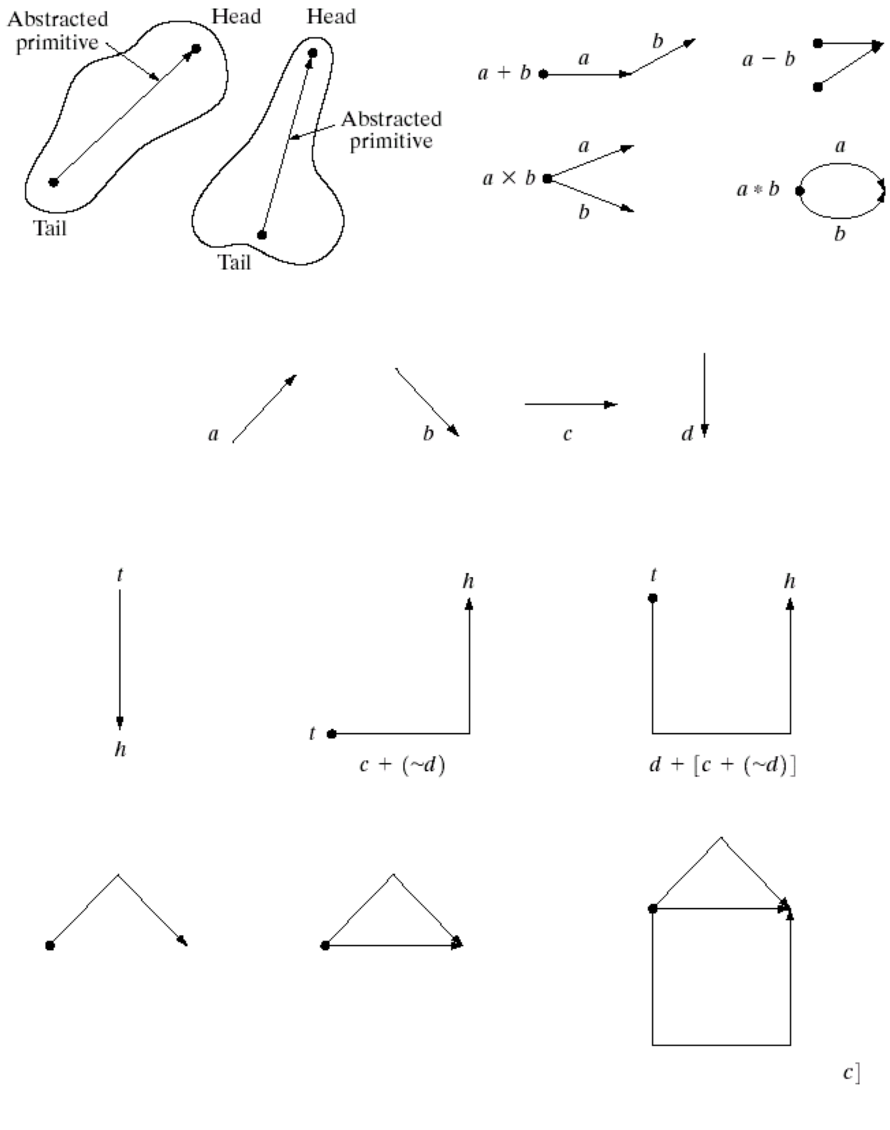


FIGURE 11.33 (a) Abstracted primitives. (b) Operations among primitives. (c) A set of specific primitives. (d) Steps in building a structure.

Relational Descriptors

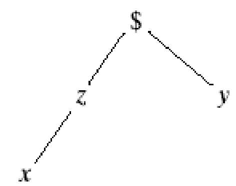
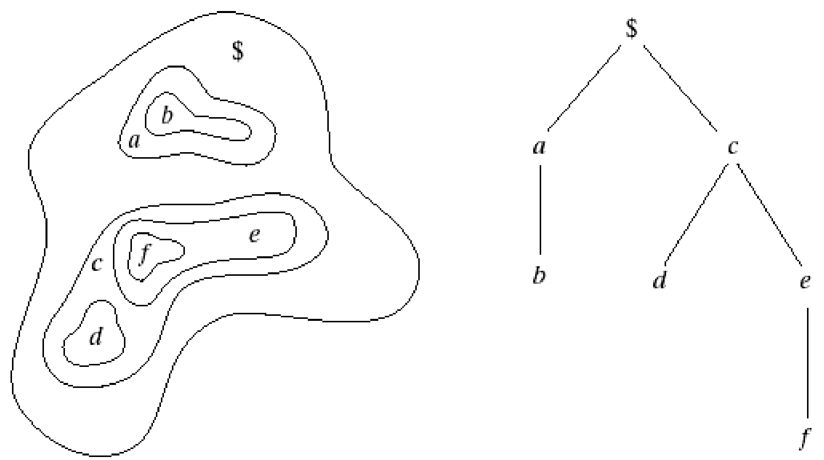


FIGURE 11.34 A simple tree with root $ and frontier *xy*.



a b

FIGURE 11.35 (a) A simple composite region. (b) Tree representation obtained by using the relationship "inside of."

(Images from Rafael C. Gonzalez and Richard E. Wood, Digital Image Processing, 2nd Edition.)

Structural Approach for Texture Descriptor

a
b
c

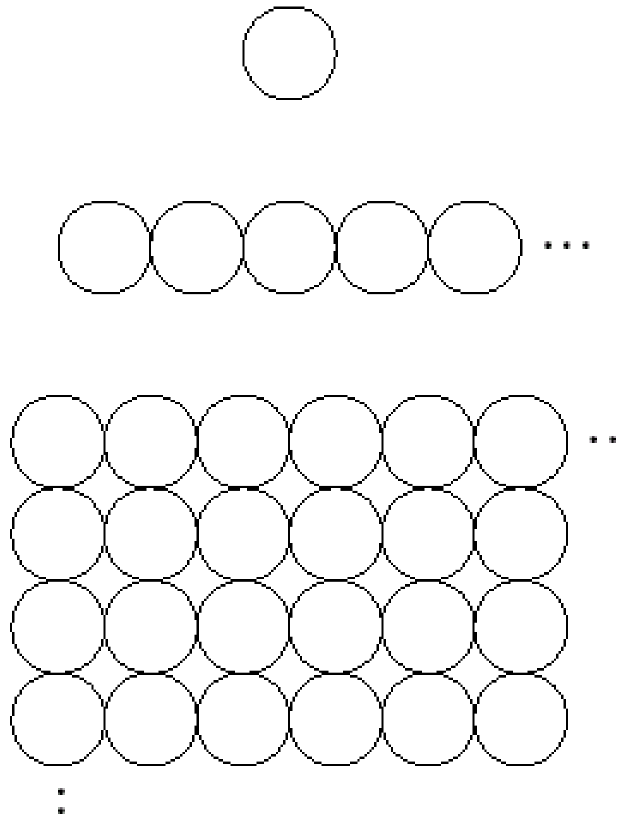


FIGURE 11.23
(a) Texture primitive.
(b) Pattern generated by the rule $S \rightarrow aS$.
(c) 2-D texture pattern generated by this and other rules.

(Images from Rafael C. Gonzalez and Richard E. Wood, Digital Image Processing, 2nd Edition.)

Shape features categories Comparison

- Contour-based approaches are more popular than region-based approaches in literature.
  - Human beings are thought to discriminate shapes mainly by their contour features.
- However, there are several limitations with contour-based methods.
  - contour shape descriptors are generally sensitive to noise and variations because they only use a small part of shape information
  - In many cases, the shape contour is not available.
  - In some applications, shape content is more important than the contour features.

Shape features categories Comparison

- Region-base methods are more robust as they use all the shape information available; they can be applied to general applications; and
  - they generally provide more accurate retrieval. In addition,
  - region-based methods can cope well with shape defection, which is a common problem for contour-based shape representation techniques.
- Although region-based methods make use of all the shape information, it is not necessarily more complex than contour-based methods

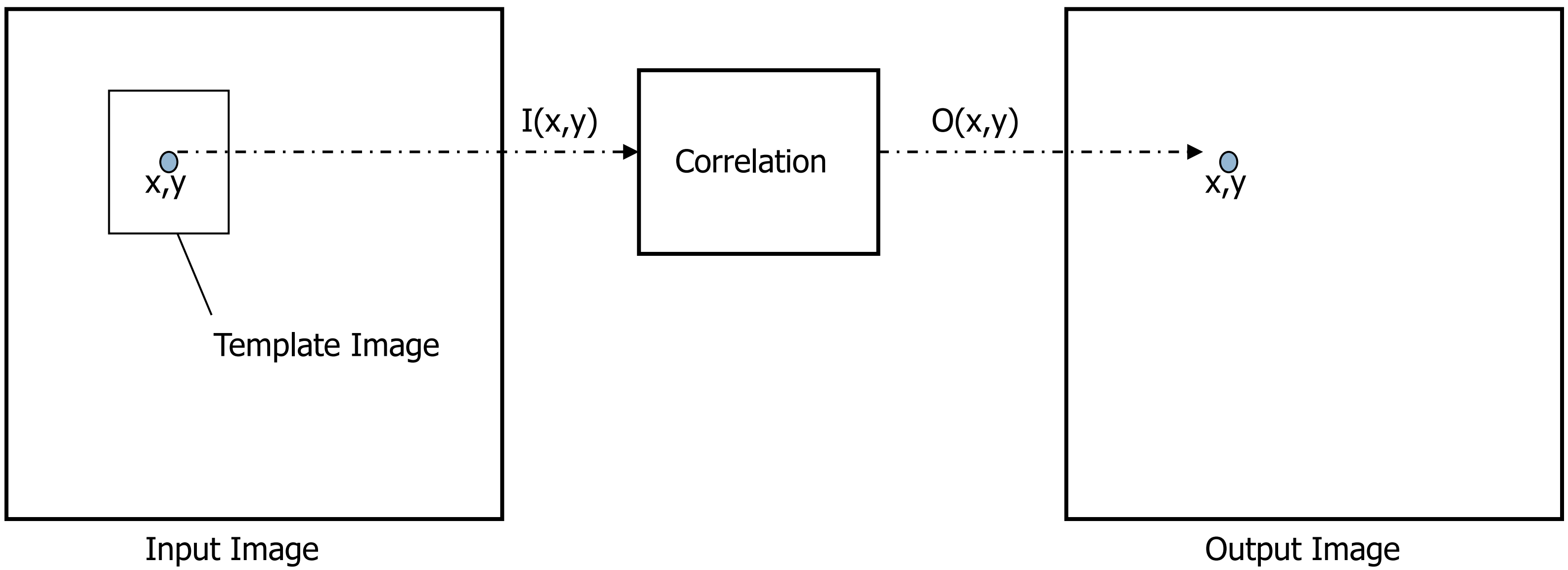
Shape Matching

- High-level *feature extraction concerns finding shapes in computer images.*
- *To be able to recognize* faces automatically, for example, one approach is to extract the component features.
- This requires extraction of, say, the eyes, the ears and the nose, which are the major facial features.
- *Shape matching is an important ingredient in shape retrieval, recognition and classification, alignment and registration, and approximation and simplification.*
- *Approaches*
  - Template Matching
  - Hough Transform

Shape Matching

- Template Matching techniques compare portions of images against one another.
- Sample image may be used to recognize similar objects in source image.
- If standard deviation of the template image compared to the source image is small enough, template matching may be used.
- Templates are most often used to identify printed characters, numbers, and other small, simple objects.
- Template Matching Approaches
  - intensity based (correlation measures)
  - feature based (distance transforms)

Template Matching Method



The matching process moves the template image to all possible positions in a larger source image and computes a numerical index that indicates how well the template matches the image in that position.

Match is done on a pixel-by-pixel basis.

Machine Vision Example

- Load printed circuit board into a machine
- Teach template image (select and store)
- Load printed circuit board
- Capture a source image and find template

Features Post Processing

- Number and types of features have a direct effect on the performance of classification and clustering techniques besides the model efficiency in time and memory
- Two Approaches for analyzing features the
  - Dimensionality Reduction
  - Feature Selection

Dimensionality

- The dimension of the data is the number of variables that are measured on each observation.
- If the data x lies in high dimensional space, then an enormous amount of data is required to learn distributions or decision rules.
- Example: 50 dimensions. Each dimension has 20 levels. This gives a total of 20^{50} cells. But the no. of data samples will be far less. There will not be enough data samples to learn.
  - For example in cancer classification tasks the number of variables (i.e. the number of genes for which expression is measured) may range from 6000 to 60000.

Dimensionality Reduction

- In machine learning and statistics, **dimensionality reduction** or **dimension reduction** is the process of reducing the number of random variables under consideration
  - transforms the data in the high-dimensional space to a space of fewer dimensions.
- Advantages of dimensionality reduction
  - It reduces the time and storage space required.
  - Removal of multi-collinearity improves the performance of the machine learning model.
  - It becomes easier to visualize the data when reduced to very low dimensions such as 2D or 3D.

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ \\ x_N \end{bmatrix} \xrightarrow{\text{feature extraction}} \begin{bmatrix} y_1 \\ y_2 \\ \\ y_M \end{bmatrix} = f\left(\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ \\ x_N \end{bmatrix}\right)$$

Dimensionality Reduction

- Techniques for dimension reduction:
  - Principal Component Analysis (PCA)
    - SIFT has 4x4x8 = 128 dimensional feature vector
    - The PCA-SIFT is demonstrated to achieve better results when it reduces its descriptor to a 36-dimensional feature vector
  - Fisher's Linear Discriminant
  - Multi-dimensional Scaling.
  - Independent Component Analysis.

Feature Selection

- Given a set of **n** features, the role of **feature selection** is to select a subset of **d** features (**d** < **n**) in order to minimize the classification error.
  - Selecting some subset of features upon which to focus its attention, while ignoring the rest.

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ \\ x_N \end{bmatrix} \xrightarrow{\text{feature selection}} \begin{bmatrix} x_{i_1} \\ x_{i_2} \\ \\ x_{i_M} \end{bmatrix}$$

Feature Selection Methods

- **Filter methods**: they are preprocessing methods. They attempt to assess the merits of features from the data, ignoring the effects of the selected feature subset on the performance of the learning algorithm. Examples: Using information gain
  - Pros: easily scale to very high-dimensional datasets, computationally simple and fast, and independent of the classification algorithm. Feature selection needs to be performed only once, and then different classifiers can be evaluated.
  - Cons: they ignore the interaction with the classifier. They are often univariate or low-variate. This means that each feature is considered separately, thereby ignoring feature dependencies, which may lead to worse classification performance when compared to other types of feature selection techniques.

Feature Selection Methods

- **Wrapper methods**: these methods assess subsets of variables/features according to their usefulness to a given predictor. The method conducts a search for a good subset using the learning algorithm itself as part of the evaluation function. The problem boils down to a problem of stochastic state space search. E.g. the stepwise methods in linear regression.
  - Pros: interaction between feature subset search and model selection, and the ability to take into account feature dependencies.
  - Cons: higher risk of overfitting than filter techniques and are very computationally intensive, especially if building the classifier has a high computational cost.

Feature Selection Methods

- **Embedded methods**: they perform variable selection as part of the learning procedure and are usually specific to given learning machines. Examples are classification trees, random forests, and methods based on regularization techniques (e.g. lasso)
  - Pros: less computationally intensive than wrapper methods.
  - Cons: specific to a learning machine.