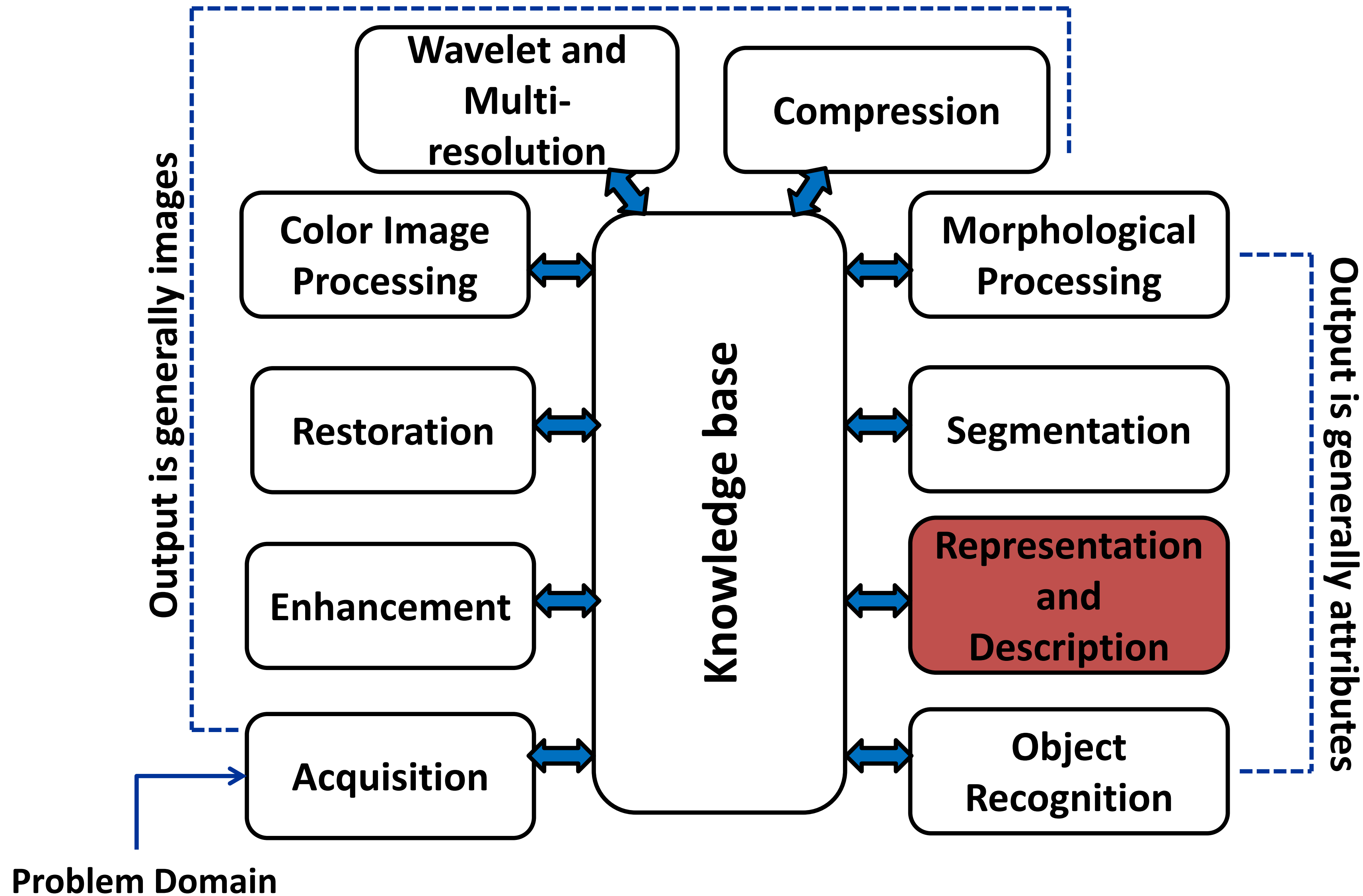
CSC 447 Digital Image Processing

Dr. Maryam Al-Berry
maryam_nabil@cis.asu.edu.eg

Representation and Description

Fundamental Steps of DIP



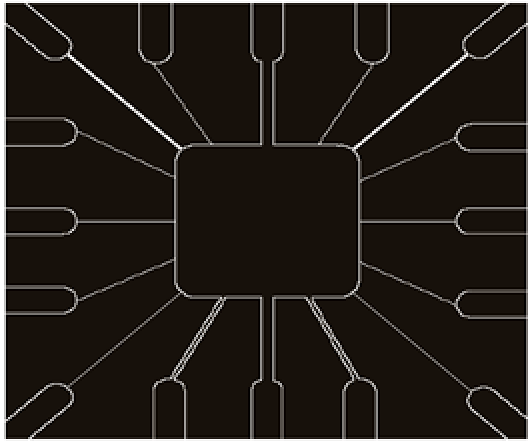
Contents

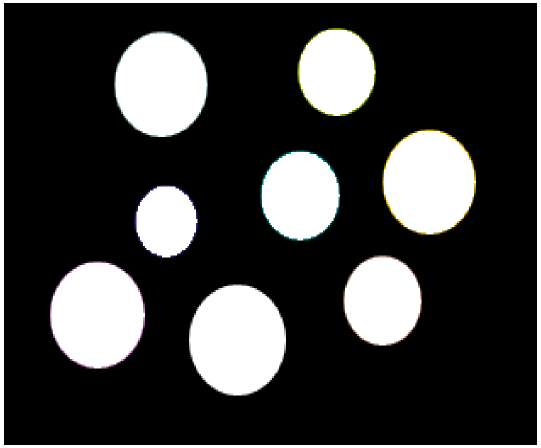
- Representation vs description
- Local feature extraction
- Detectors
- Descriptors

Representation

Why?

- Segmentation results in an aggregate number of pixels that usually requires a presentation/description more suitable for computer processing.





Representation

How?

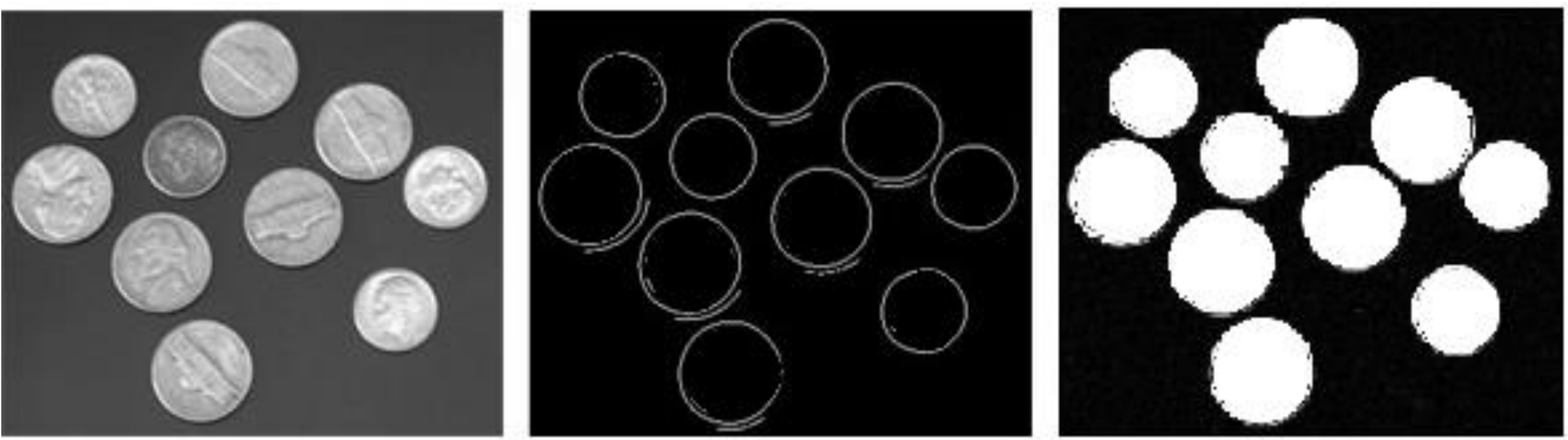
- Choose a representation that facilitate the computation of a descriptor. (detect features)
- Describe the object based on the chosen representation. (select features)

- Sometimes the keypoints are considered features.
- Sometime, the same technique is used for both feature detection and description, e.g. SIFT.

Representation – (cont.)

Based on

- External characteristics (boundary → shape).
- Internal characteristics (region pixels → color and texture).



Representation – (cont.)

Example 1

- Choose a representation.

    e.g. *boundary*

- Describe the object.

    boundary described by its

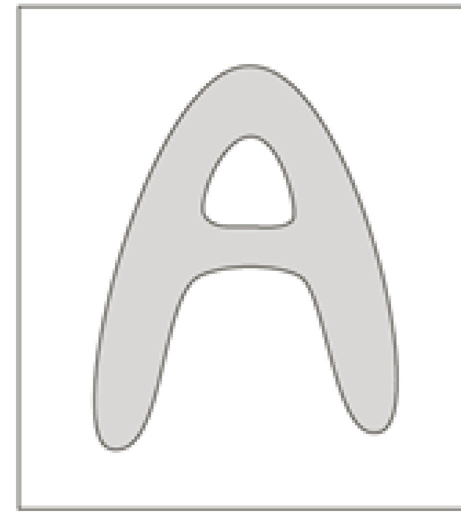
    signature

corners, lines, length,

orientation, concavities, etc.

Representation – (cont.)

Example 2



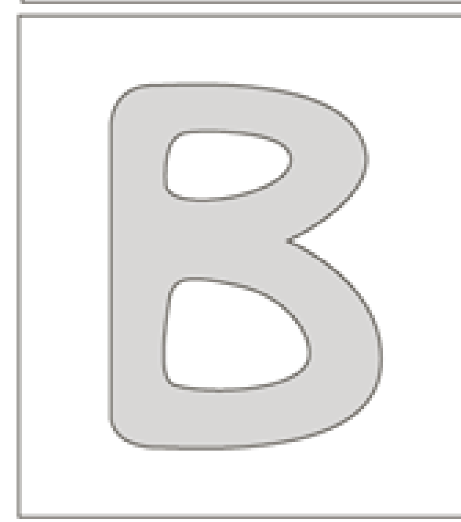
- Choose a representation.

  e.g. *region*

- Describe the object.

  region described by its

  area, color, Euler number, texture, etc.



Representation – (cont.)

Feature Extraction

- The process by which certain features of interest are detected/represented for future processing.

- A critical step in IP and CV, as it marks the transition for pictorial to non-pictorial data representation.

- Result is used as input to pattern recognition and classification techniques, which will label, classify, and recognize the contents of the image and its objects.

Representation – (cont.)

Feature Vector

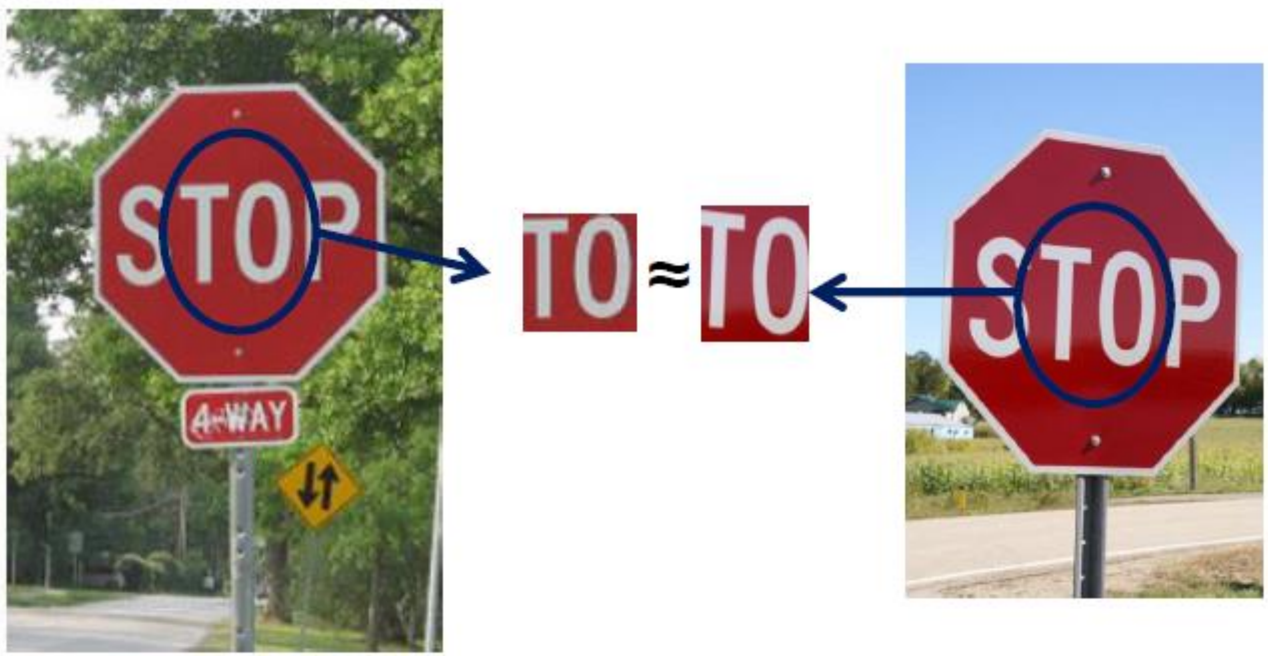
- A vector that encodes (represents) the features describing an image or its objects.

- It is a compact representation of the image/object and it can be numeric or symbolic.

Why are Features Important?

- Correspondence: matching points, patches, edges, or regions across images.

Example: classification.



James Hayes

Why are Features Important? – (cont.)

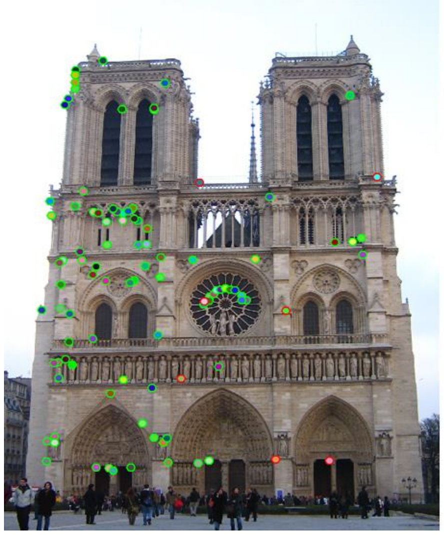
- Correspondence: matching points, patches, edges, or regions across images.

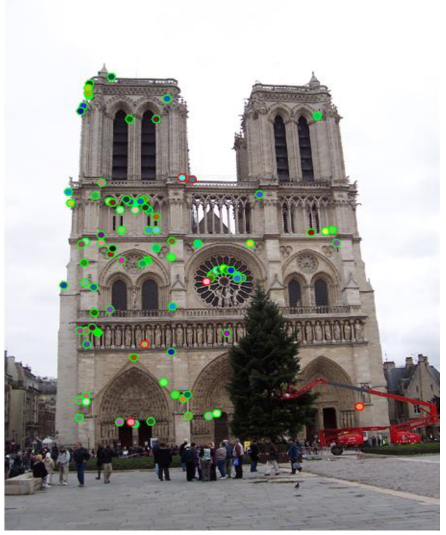
Example: panoramic stitching.

Why are Features Important? – (cont.)

Applications

- Image alignment
- 3D reconstruction
- Motion tracking
- Robot navigation
- Indexing and retrieval
- Object recognition





Features

Types

- Global
  - Histogram
  - Texture
  - Statistical
- Local
  - Detectors: keypoints (boundary, region...)
  - Descriptors: binary, spectra, basis space, polygon, multi-modal...

Local Feature Extraction

1. Detection

Find a set of keypoints and define a region around each.

2. Description

Compute a local descriptor from the normalized region.

3. Matching

Determine correspondence between local descriptors in two views.

Local Feature Extraction – (cont.)

1. Detection

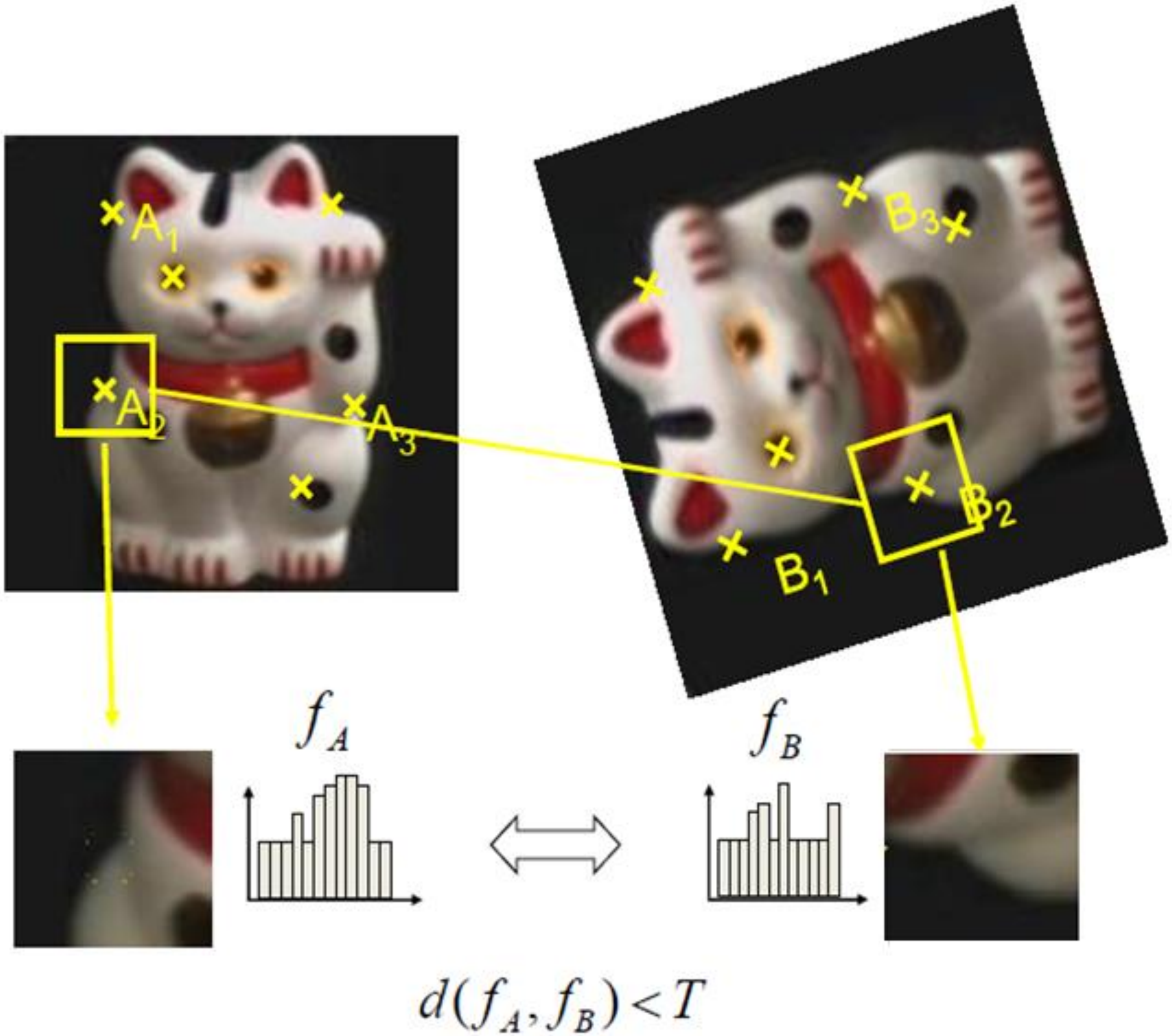
Identify interest points.

2. Description

Compute a local descriptor.

3. Matching

Match local descriptors.



Outline

- Detection
  - Edges
  - Lines
  - Contour
  - Corner
  - Region
- Boundary Descriptors
  - 1. Chain Code
  - 2. Shape Number
  - 3. Signature
  - 4. Simple Descriptors
  - 5. Fourier Descriptor
  - 6. Statistical Moments
- Regional Descriptors
  - 1. Simple Descriptors
  - 2. Topological
  - 3. Texture
  - 4. Moment Invariants
- Matching
  - Euclidian Distance
  - Nearest Neighbor
  - Mean Square Error

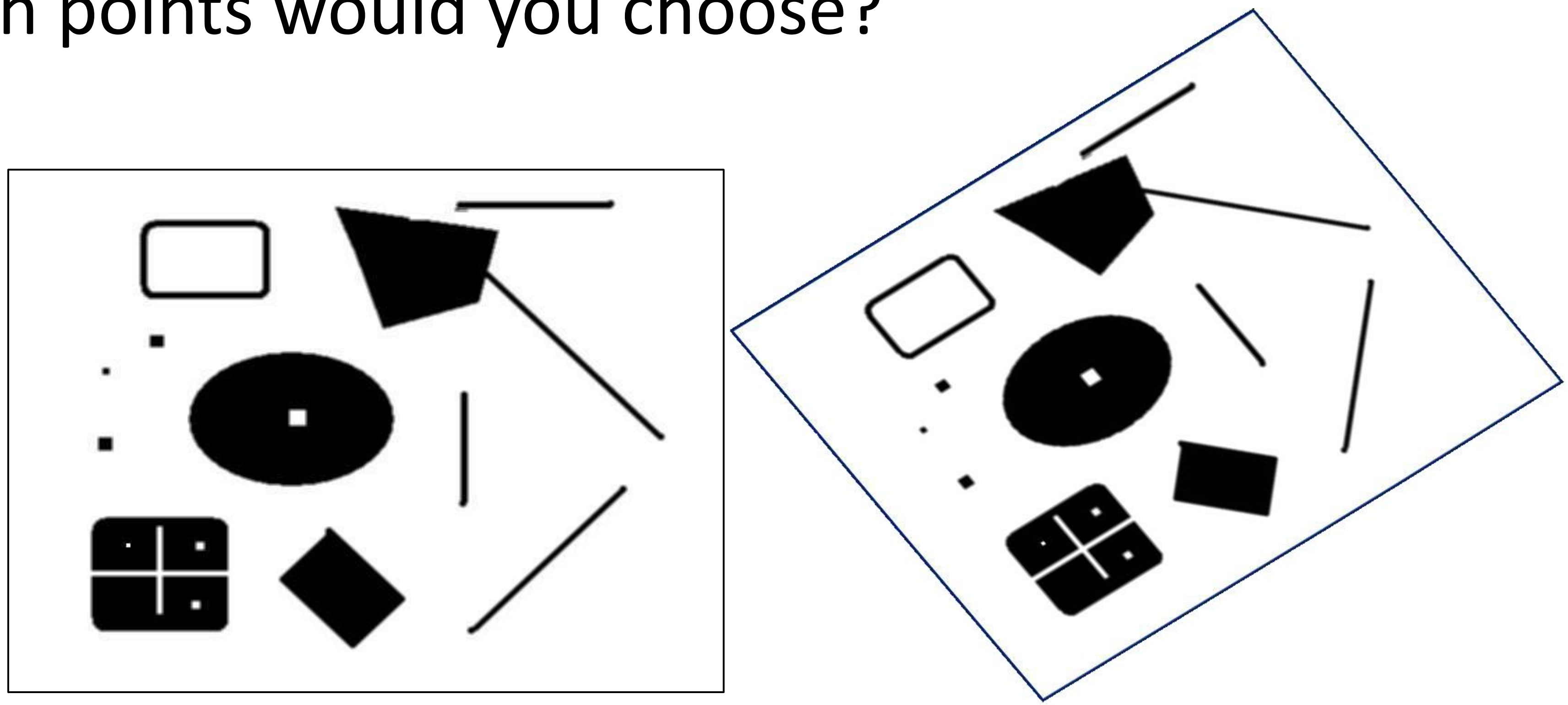
1. Features Detection

Keypoints

Keypoints = Interest Points = Features

- Suppose you have to click on the SAME points before and after the image is deformed.

Which points would you choose?

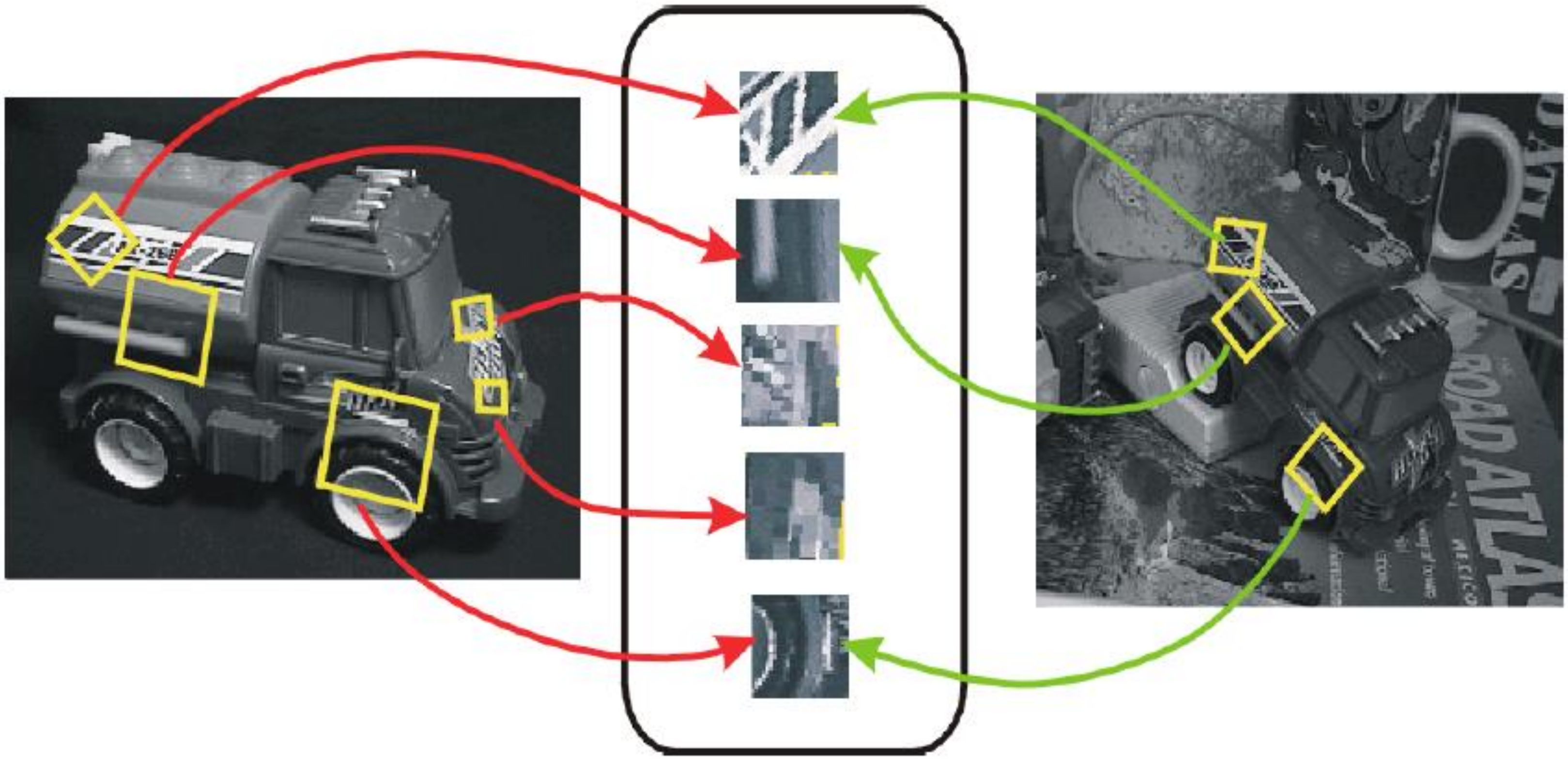


Keypoints – (cont.)

Which points would you choose?

Invariance

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



Features Descriptors

James Hayes

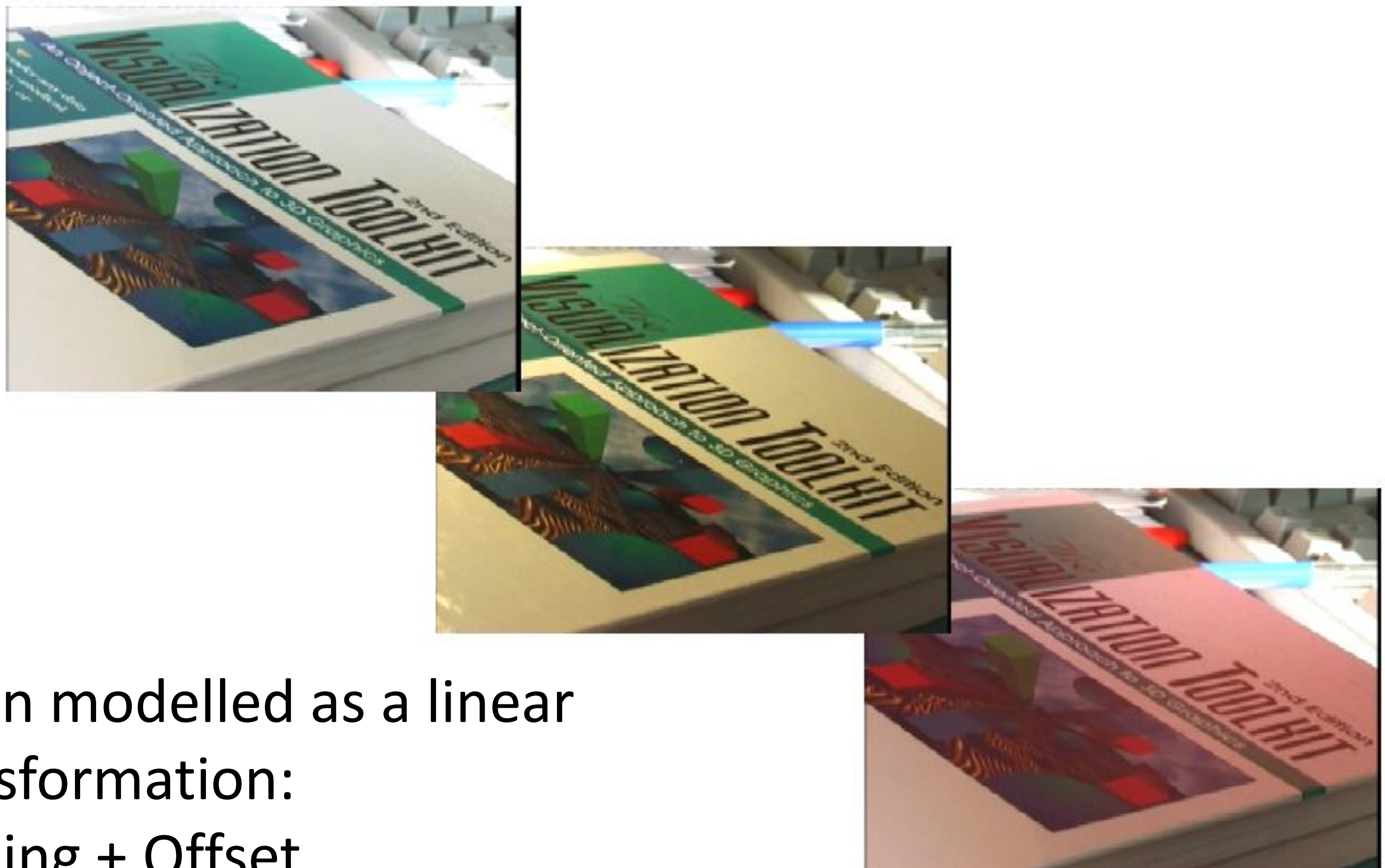
Invariance – (cont.)

- To geometric transformations

James Hayes

Invariance – (cont.)

- To photometric transformations

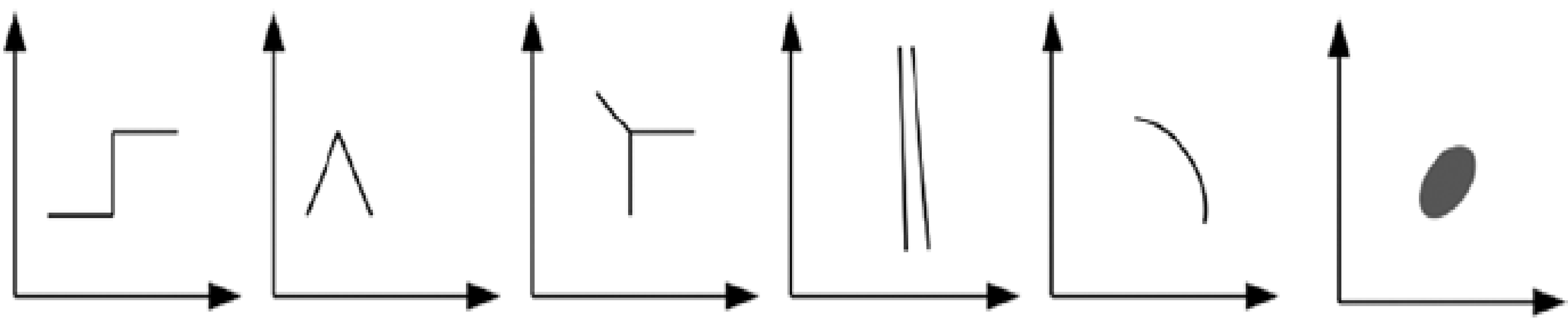


Often modelled as a linear transformation:
Scaling + Offset

Hayes

Good Features

- A good interest point must be easy to find and ideally fast to compute.
- It is hoped that the interest point is at a good location to compute a feature *descriptor*.



Types of keypoints, including corners and interest points. (Left to right) Step, roof, corner, line or edge, ridge or contour, maxima region.

Good Features – (cont.)

- The keypoint location itself may not be enough for feature matching → the need for descriptors.

- However, some methods rely on *keypoints only,* without a feature descriptor.

- There is no superior method for interest point detection for all applications.

Good Features – (cont.)

Repeatability

The same feature can be found in several images despite geometric and photometric transformations.

Saliency

Each feature is distinctive.

Compactness and efficiency

Many fewer features than image pixels.

Locality

A feature occupies a relatively small area of the image; robust to clutter and occlusion.

James Hayes

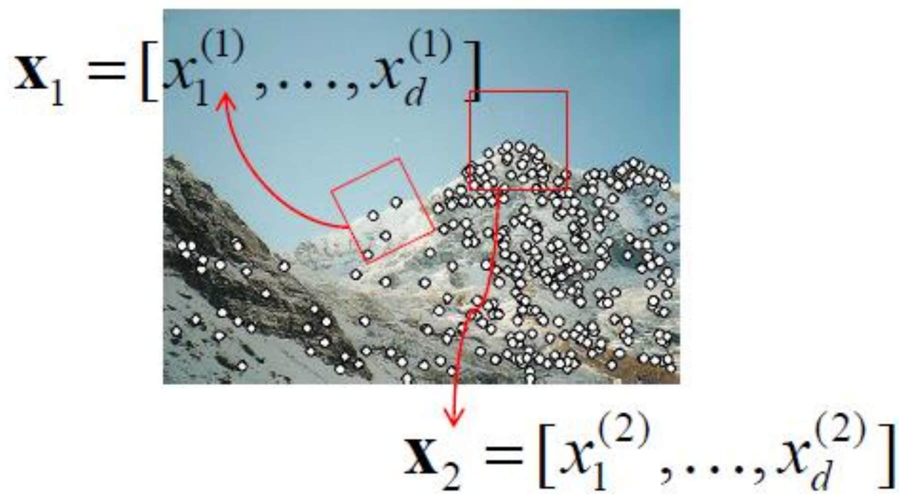
Local Feature Extraction

1. Detection:

Identify the interest points.

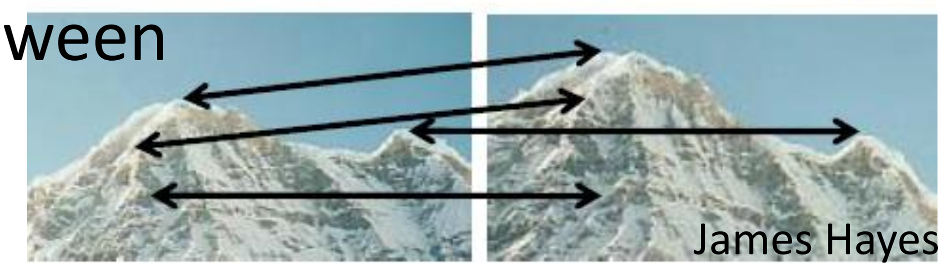
2. Description:

Extract vector feature descriptor surrounding each interest point.



3. Matching:

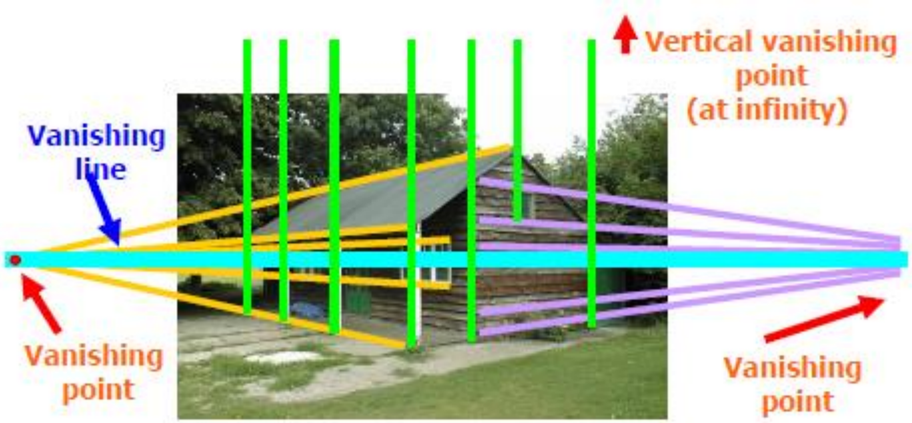
Determine correspondence between descriptors in two views.

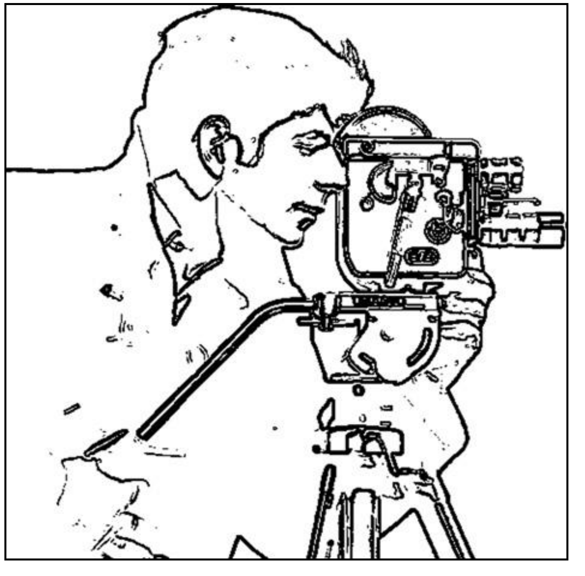


1.1 Edge Detection

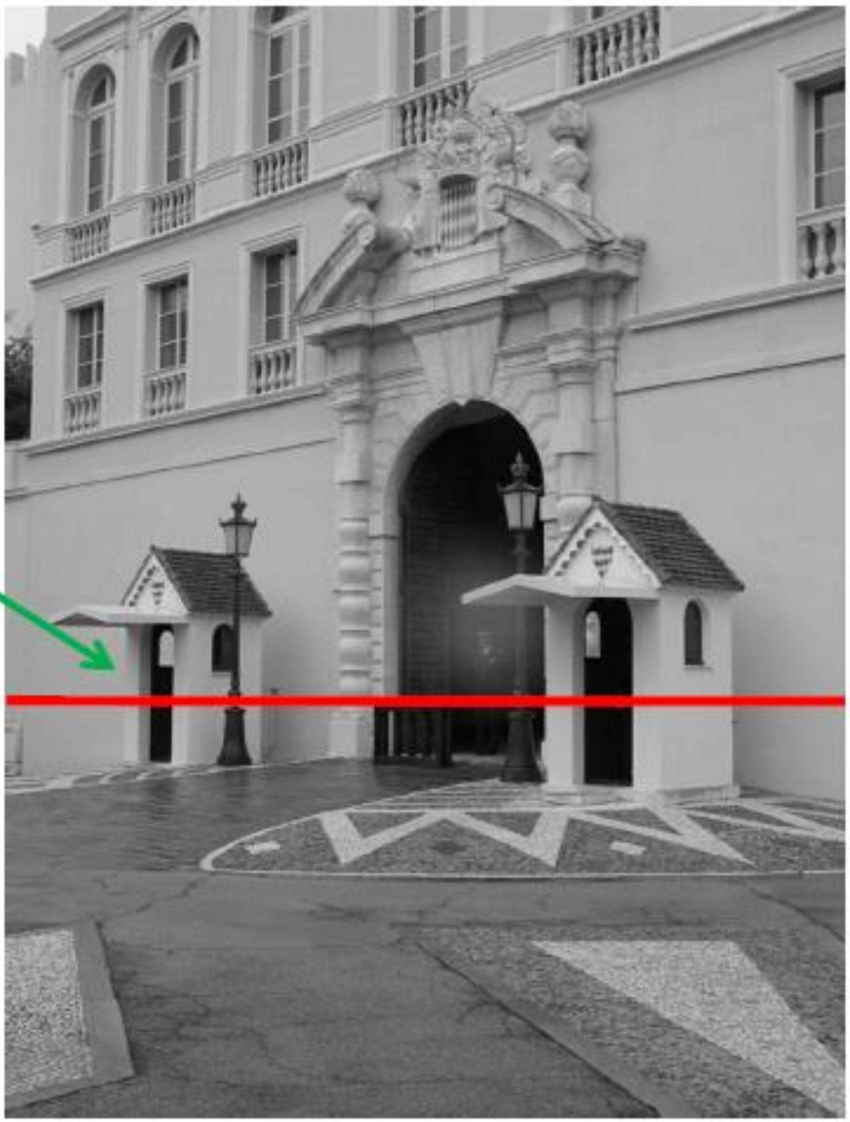
Edge Detection

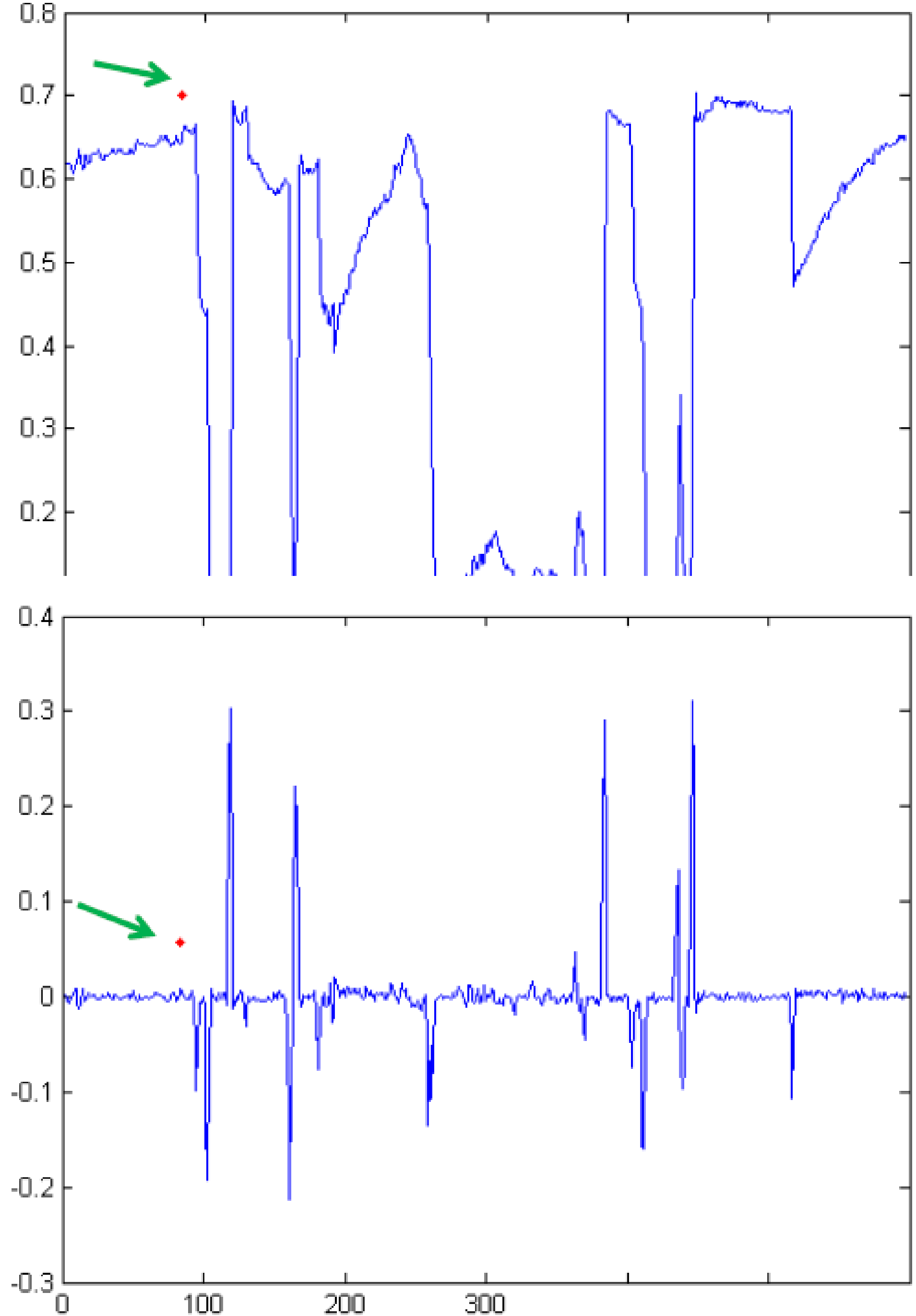
- Extract information.
- Recognize objects.
- Recover geometry and viewpoint.
- Usually derivative based.





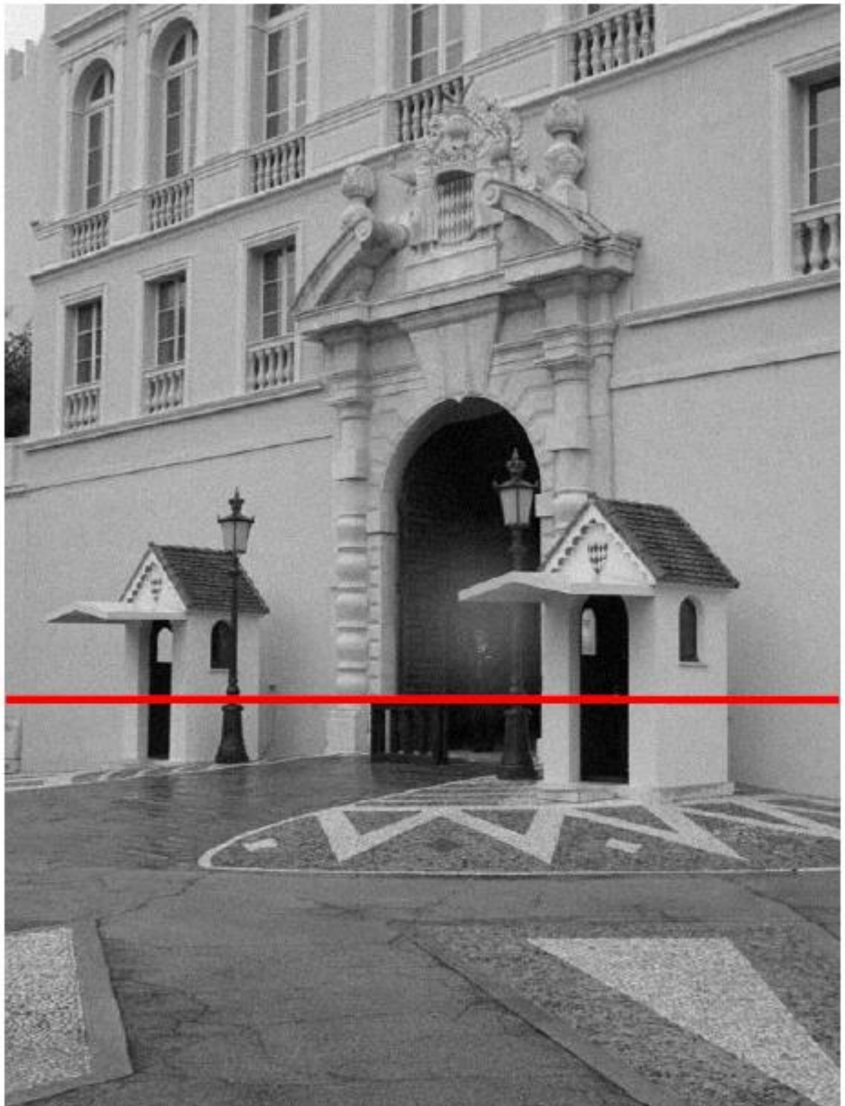
Intensity Profile



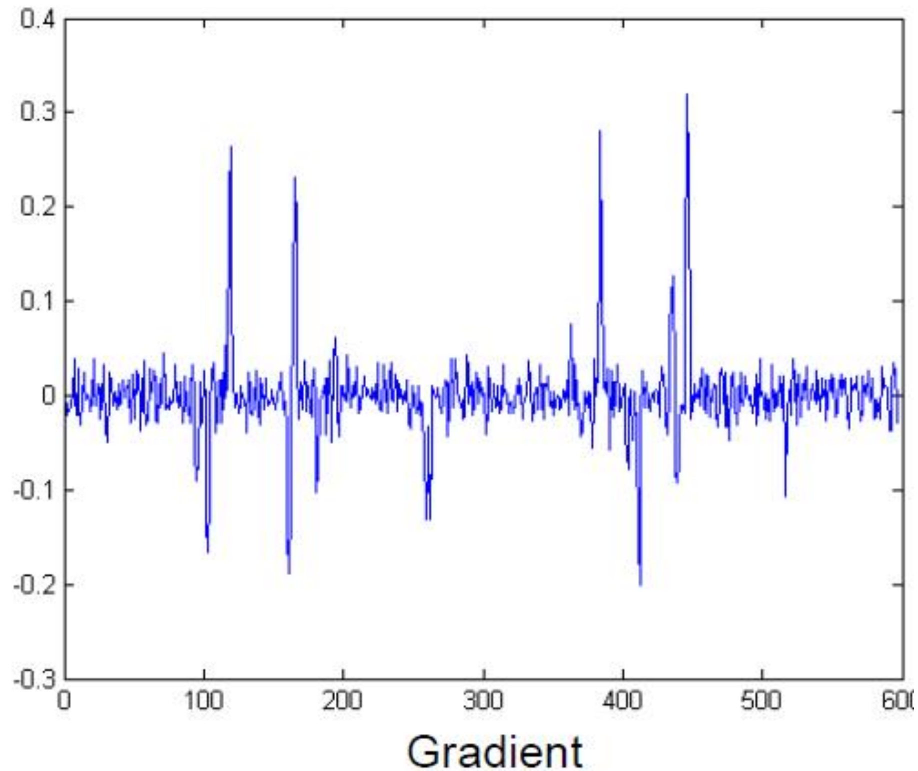


James Hayes

Intensity Profile – (cont.)



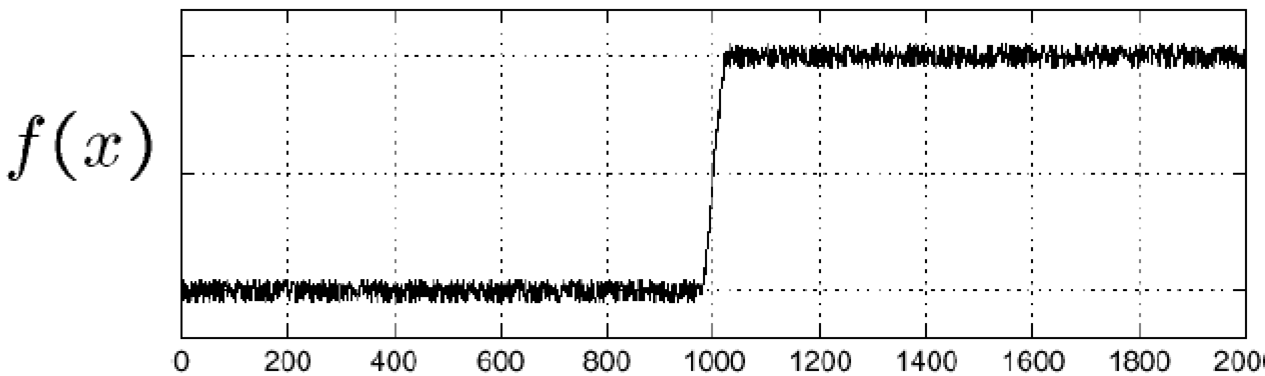
With a little Gaussian noise!

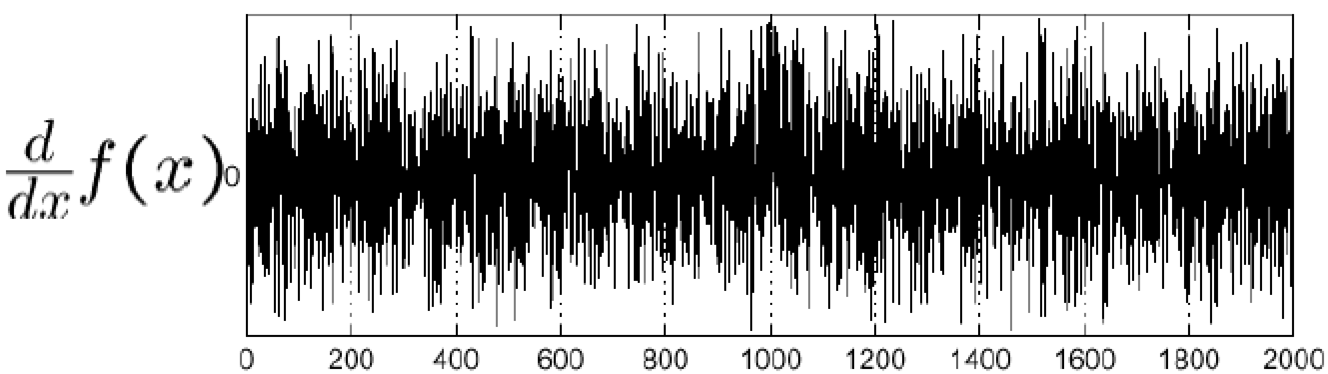


James Hayes

Intensity Profile – (cont.)

With a more noise!

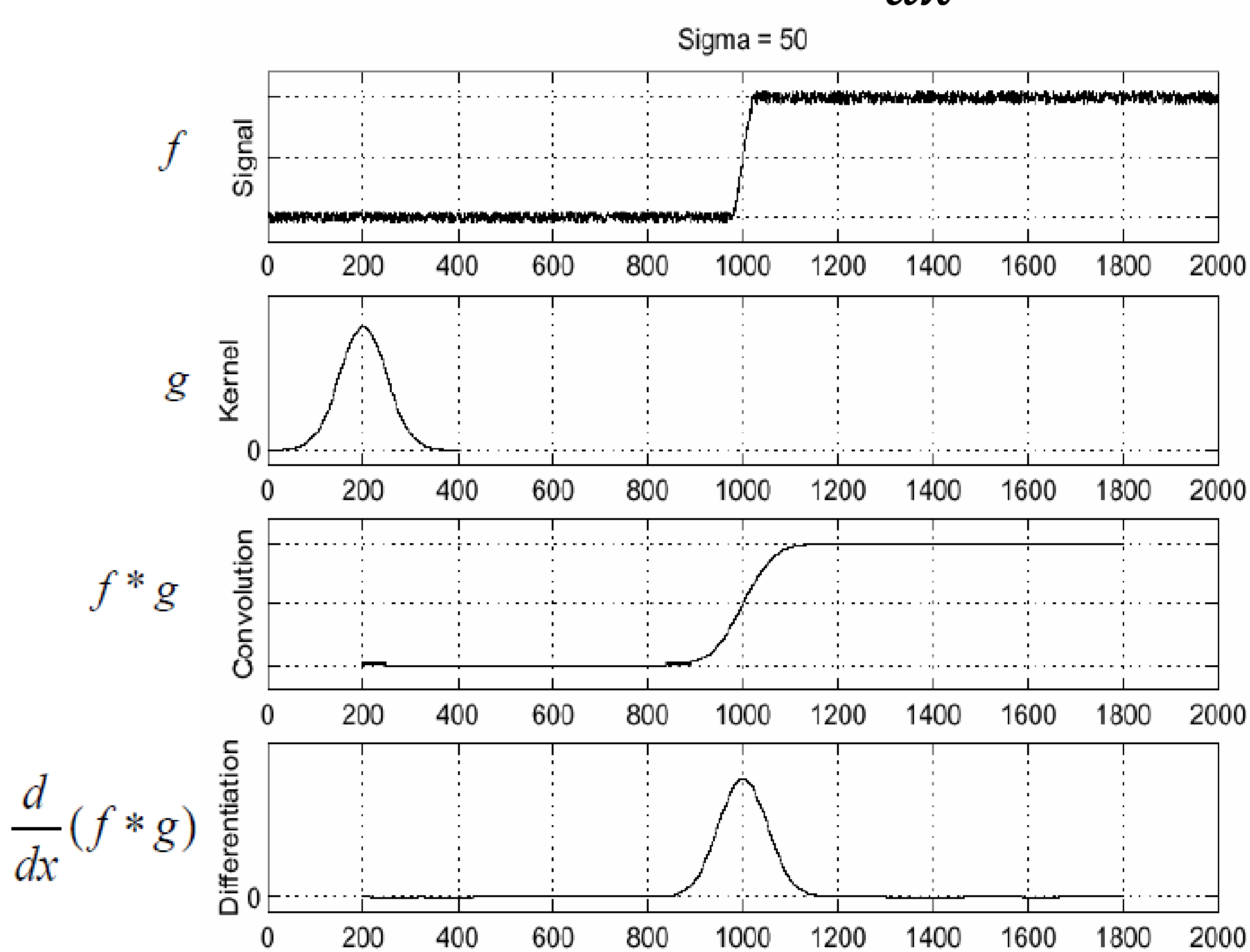




James Hayes

Smoothing then Derivative

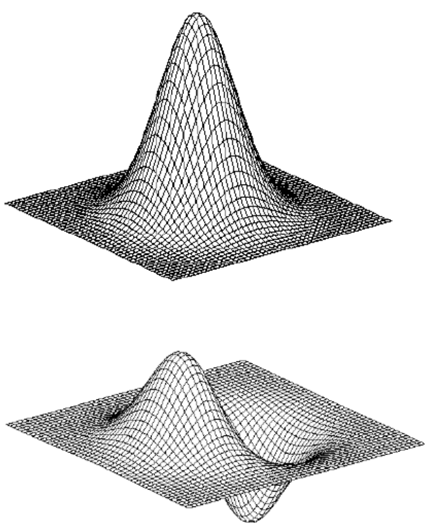
- To find edges, look for peaks in $\frac{d}{dx}(f * g)$



James Hayes

Smoothing then Derivative – (cont.)

- Differentiation is convolution, and convolution is associative: $\frac{d}{dx}(f * g) = f * \frac{d}{dx} g$

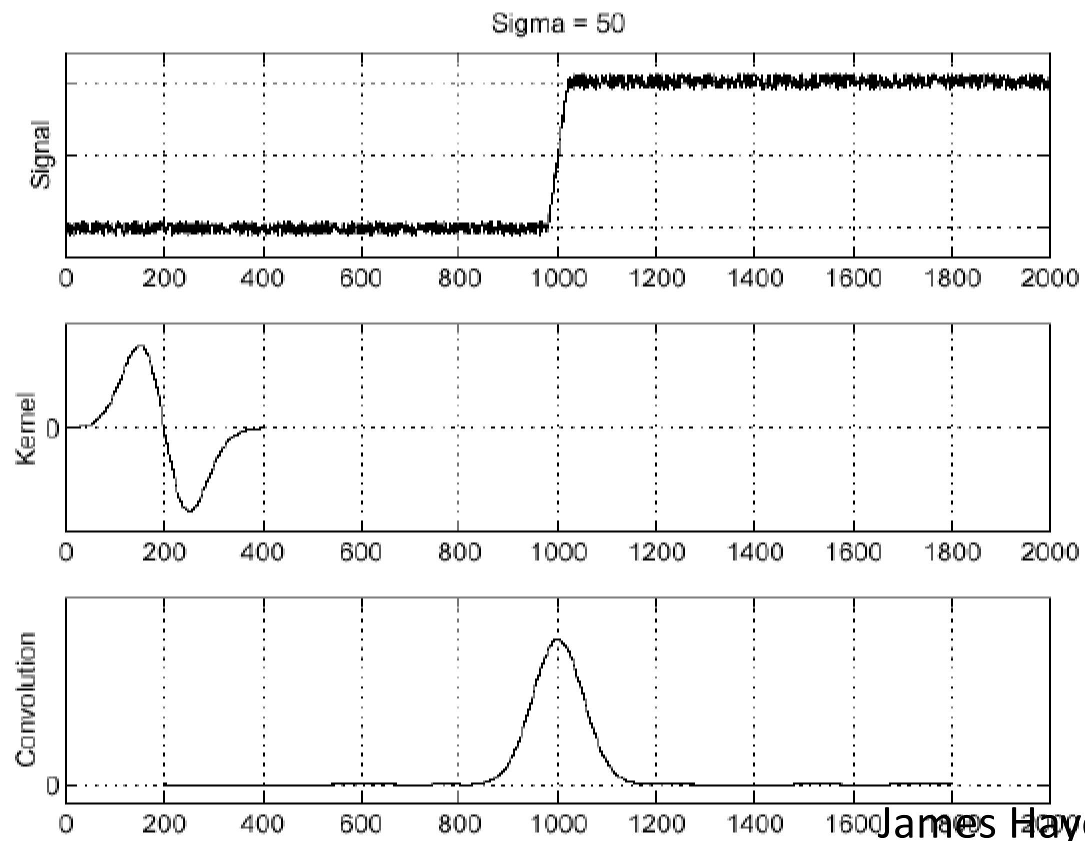


Derivative of a Gaussian

f

$\frac{d}{dx} g$

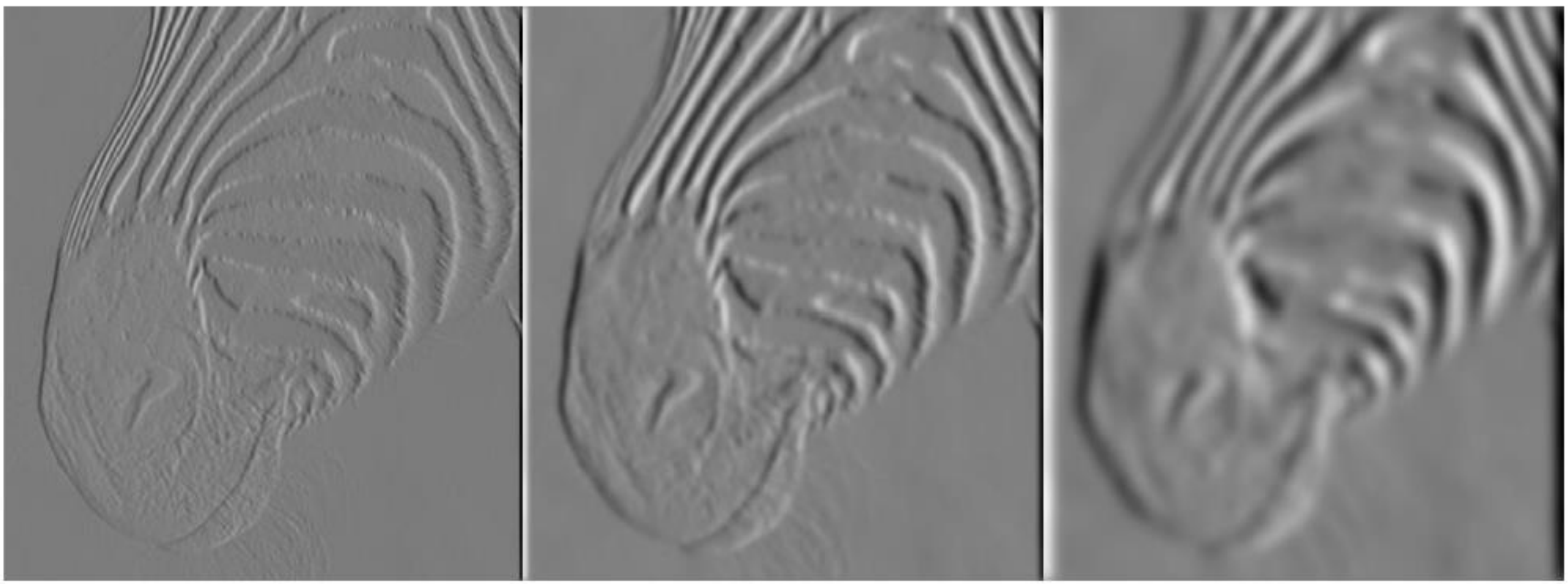
$f * \frac{d}{dx} g$



James Hayes

Smoothing then Derivative – (cont.)

- Tradeoff between smoothing and localization.



1 pixel 3 pixels 7 pixels

James Hayes

Good Edge Detector

Good detection

- The optimal detector should find all real edges, ignoring noise or other artifacts.

Good localization

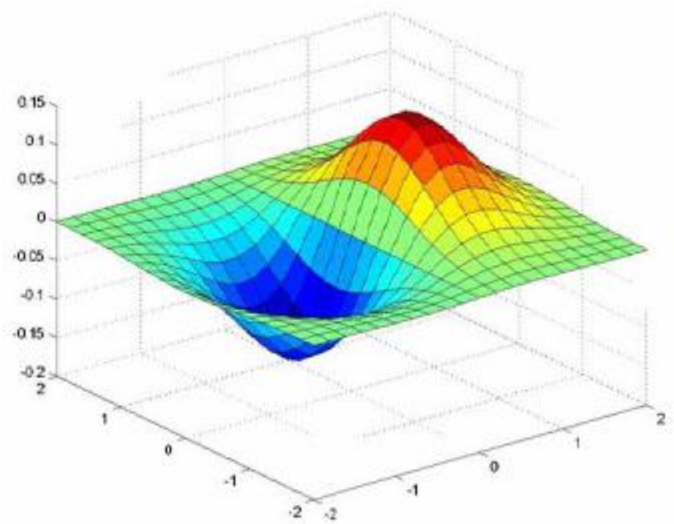
- The edges detected must be as close as possible to the true edges.
- The detector must return one point only for each true edge point.

Common Edge Detectors

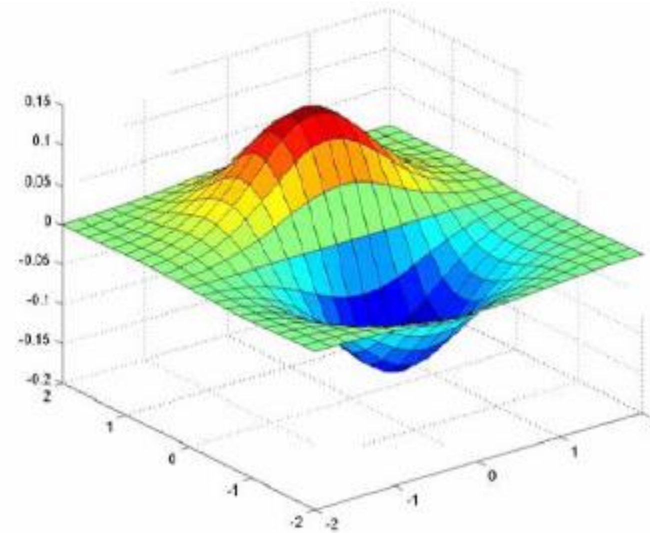
- Gradient
- Laplacian
- **Canny**
  - Theoretical model: step-edges corrupted by additive Gaussian noise.
  - Derivative of a Gaussian (Dog).
  - Most widely used edge detector in practice.

Canny Edge Detector

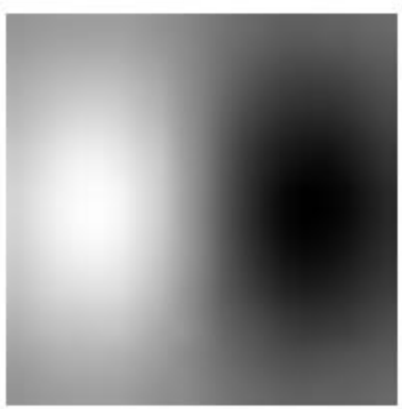
- Derivative of a Gaussian

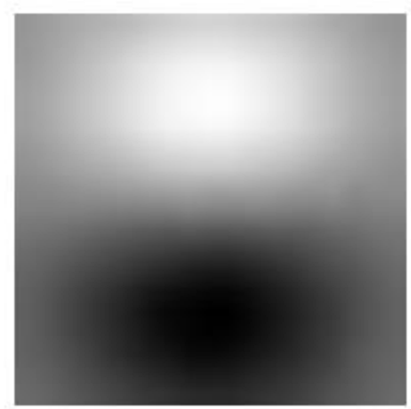


x-direction



y-direction





James Hayes

Canny Edge Detector – (cont.)

Steps

1. Image smoothing

Filter image with x, y derivatives of Gaussian.

2. Gradient Edge points detection → edge map

Find magnitude and orientation of gradient.

3. Thinning (non-maximum suppression)

Thin multi-pixel wide “ridges” down to single pixel width.

4. Thresholding + Connectivity

Define two thresholds: low and high, use the high to start edge curves and the low to continue them.

Canny Edge Detector – (cont.)

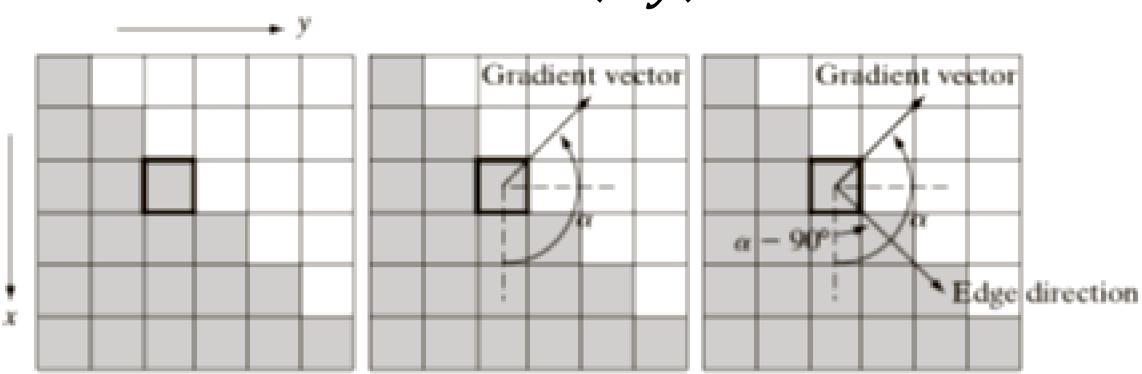
1. Image Smoothing

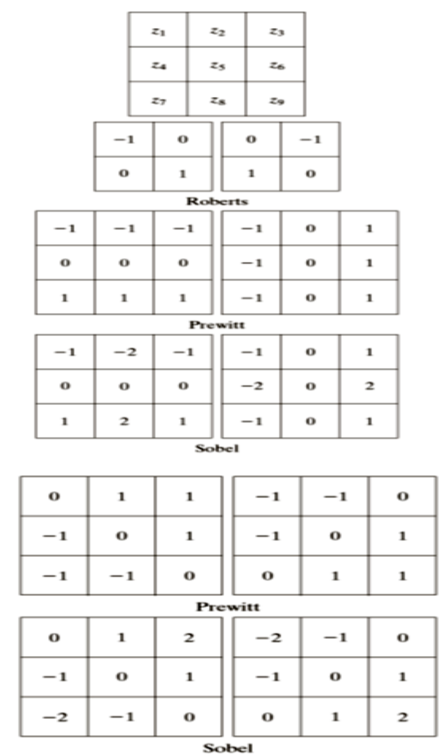
- Filtering the input image using a Gaussian smoothing function
- $G(x, y) \; = e^{-\frac{x^2}{2\sigma^2}}$
- $f_s(x, y) = G(x, y) \star f(x, y)$
- Size of mask is determined from the variance of the Gaussian function.
- size = smallest odd integer $n \geq 6\sigma$

Canny Edge Detector – (cont.)

2. Gradient Edge Map

- Computing gradient magnitude and direction of smoothed image $f_s(x, y)$
- $M(x, y) = \sqrt{{g_x}^2 + {g_y}^2}$
- $\alpha(x, y) = tan^{-1}\left(\frac{g_x}{g_y}\right)$





Canny Edge Detector – (cont.)

- Derivative of a Gaussian

X-Derivative of Gaussian

Y-Derivative of Gaussian

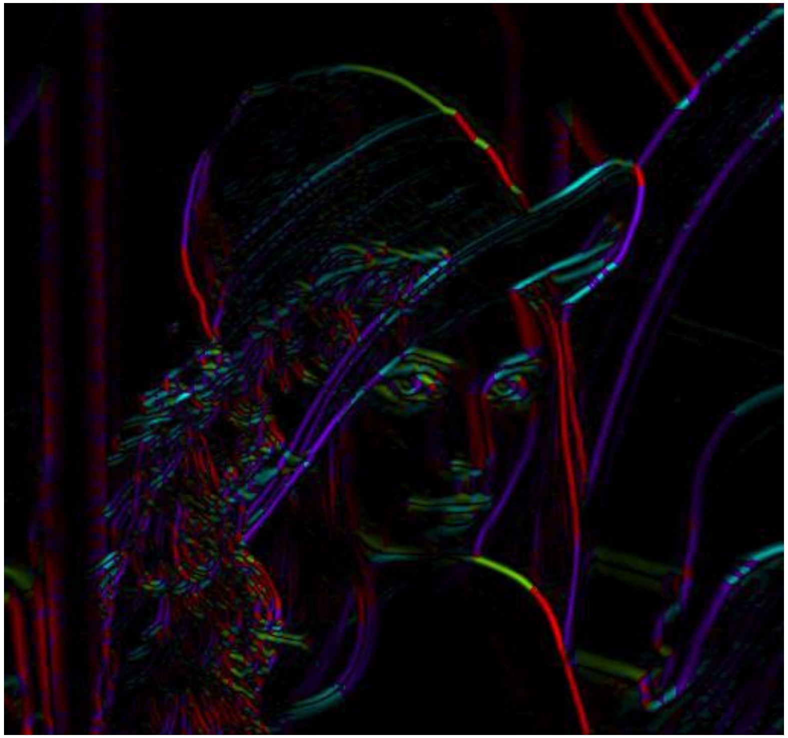
Gradient Magnitude

James Hayes

Canny Edge Detector – (cont.)

- Threshold and get orientation at each pixel.

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



James Hayes

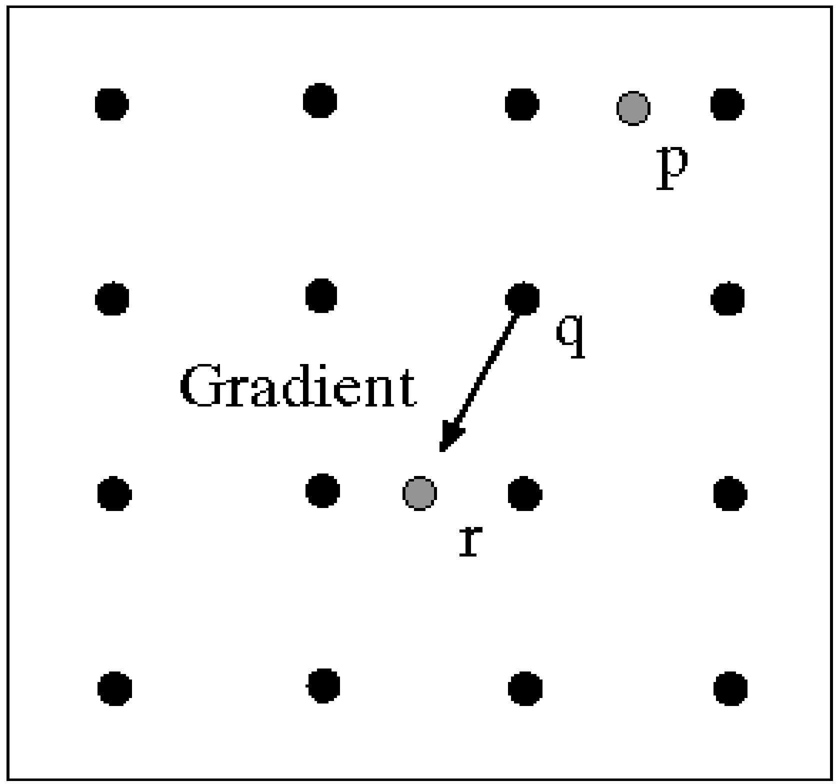
Canny Edge Detector – (cont.)

3. Non-maxima Suppression

- Gradient maps typically contain <u>wide</u> ridges around local maxima.
- <u>Thinning</u> can be done by specifying a set of discrete directions for edge normal (gradient vector).
- And selecting edge points having a “reasonable closeness” along those directions.

Canny Edge Detector – (cont.)

3. Non-maxima Suppression



At q, we have a maximum if the value is larger than those at both p and at r. Interpolate to get these values.

James Hayes

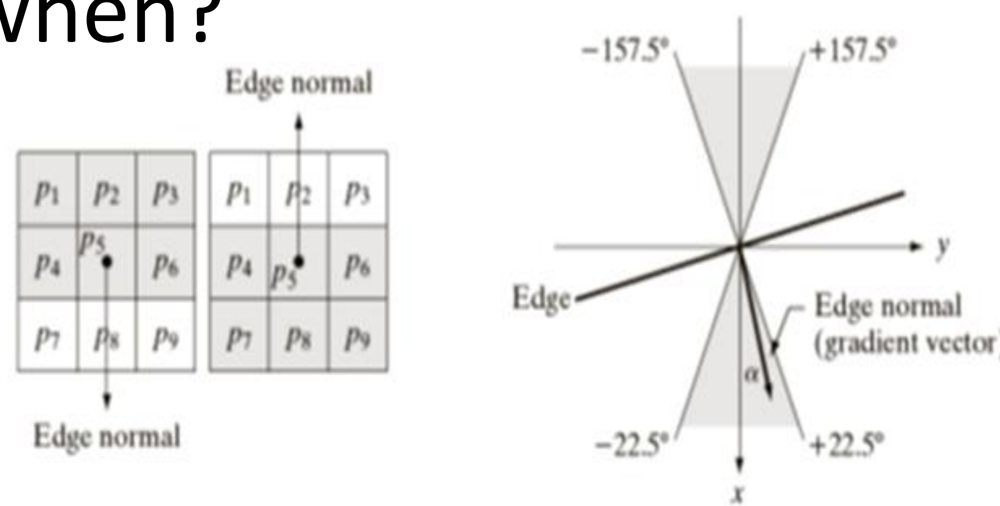
Canny Edge Detector – (cont.)

3. Non-maxima Suppression

- e.g. region of 3×3 : Four possible orientations for an edge passing through the center point of the region: horizontal, vertical, $+45^o$, and -45^o.
- Horizontal edge… when?

If $157.5 \geq \alpha \geq -157.5$

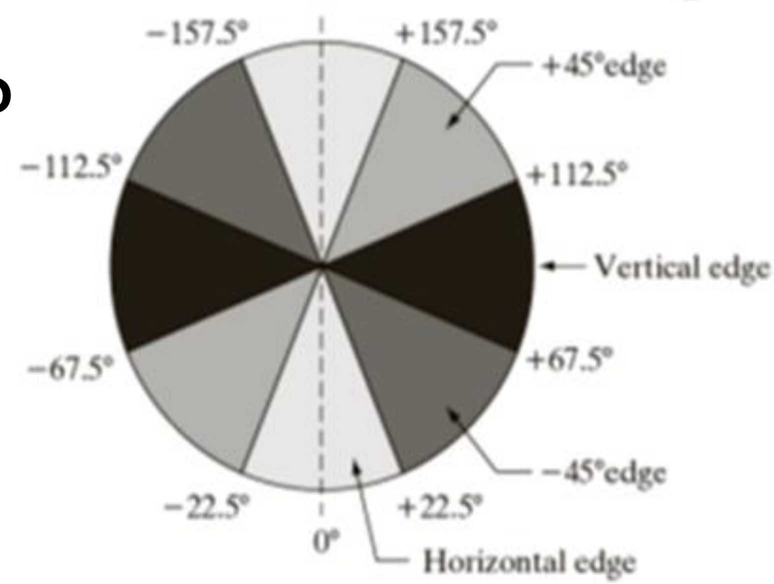
Or $22.5 \geq \alpha \geq -22.5$



Canny Edge Detector – (cont.)

3. Non-maxima Suppression

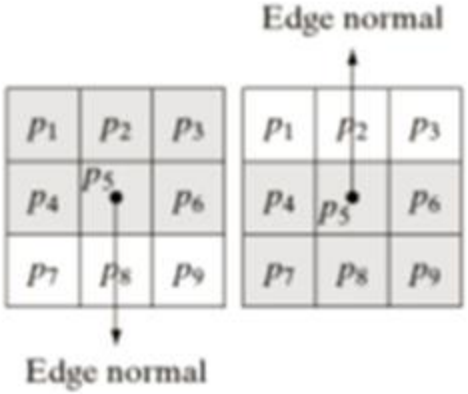
- e.g. region of 3×3 : Four possible orientations for an edge passing through the center point of the region: horizontal, vertical, $+45^o$, and -45^o.
- Other directions… when?



Canny Edge Detector – (cont.)

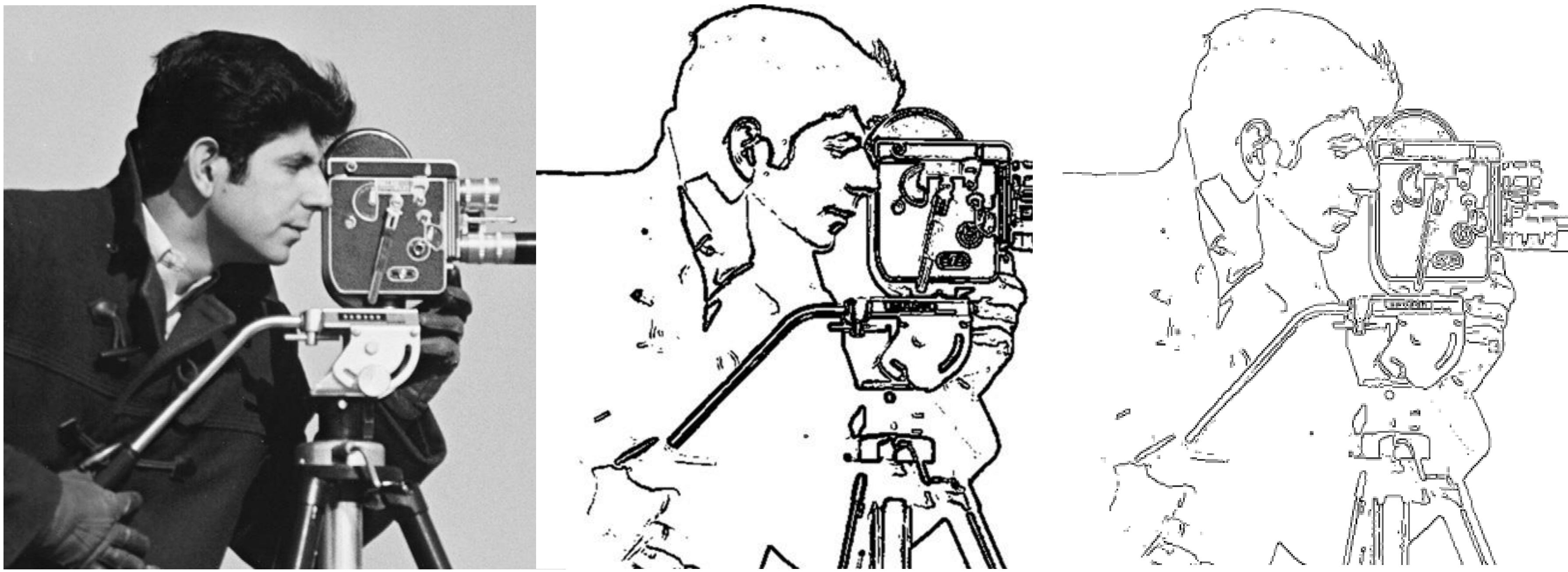
3. Non-maxima Suppression

1. Let $g_N(x, y)$ be non-maxima suppressed image.
2. Find direction d_k closest to $\alpha(x, y)$.
3. If $M(x, y) <$ at least one of its neighbors along d_k, let $g_N(x, y) = 0$ (suppression).
4. Else let $g_N(x, y) = M(x, y)$



Canny Edge Detector – (cont.)

3. Non-maxima Suppression



Gradient edge detection steps, using the Sobel operator: (a) After thresholding $|\nabla f|$; (b) after thinning (a) by finding the local maximum of $|\nabla f|$ along the gradient direction.

Canny Edge Detector – (cont.)

3. Non-maxima Suppression

Before non-maximum suppression.

After non-maximum suppression.

James Hayes

Canny Edge Detector – (cont.)

4. Thresholding

- Thresholding $g_N(x, y)$ to reduce false edge points.
- Too low → Still remains non edge points (false positives).
- Too high→ eliminate actual edge points (false negatives).
- Canny uses double thresholding.

Canny Edge Detector – (cont.)

4. Thresholding

- $g_{NH}(x, y) = g_N(x, y) \geq T_H$ **Strong edge pixels**
- $g_{NL}(x, y) = g_N(x, y) \geq T_L$
- $g_{NL}(x, y) = g_{NL}(x, y) - g_{NH}(x, y)$

Weak edge pixels

- All nonzero pixels in are marked immediately as valid actual edge points. But its edges have gaps. Use nonzero pixels in $g_{NL}(x, y)$ to connect them.

Canny Edge Detector – (cont.)

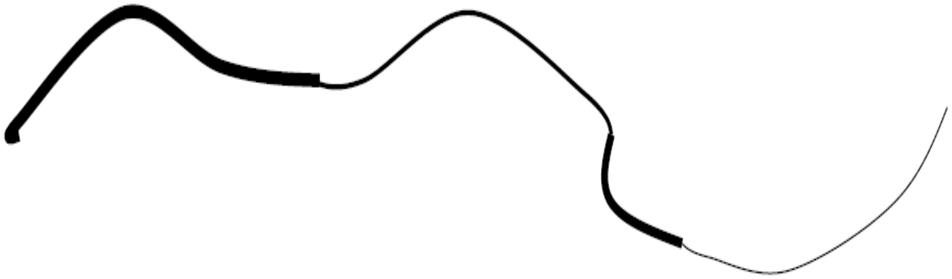
4. Thresholding

1. Locate unvisited nonzero pixel p in $g_{NH}(x, y)$.
2. Mark as valid all weak points in $g_{NL}(x, y)$ that are 4- or 8- connected to p.
3. If all nonzero visited go to step 4, else return to step 1.
4. Set to zero all pixels in $g_{NL}(x, y)$ that were not marked as valid.
5. Append to $g_{NH}(x, y)$ all nonzero pixels from $g_{NL}(x, y)$.

Canny Edge Detector – (cont.)

- Threshold at low/high levels to get weak/strong edge pixels.
- Do connected components, starting from strong edge pixels.

use a high threshold to start edge curves and a low threshold to continue them.



James Hayes

Canny Edge Detector – (cont.)

- Final result

James Hayes

Canny Edge Detector – (cont.)

Example

a b
c d

FIGURE 10.25
(a) Original image of size 834 × 1114 pixels, with intensity values scaled to the range [0, 1].
(b) Thresholded gradient of smoothed image.
(c) Image obtained using the Marr-Hildreth algorithm.
(d) Image obtained using the Canny algorithm. Note the significant improvement of the Canny image compared to the other two.

Canny Edge Detector – (cont.)

- The choice of σ depends on desired behavior
  - large σ detects large scale edges.
  - small σ detects fine features.

original

Canny with $\sigma = 1$

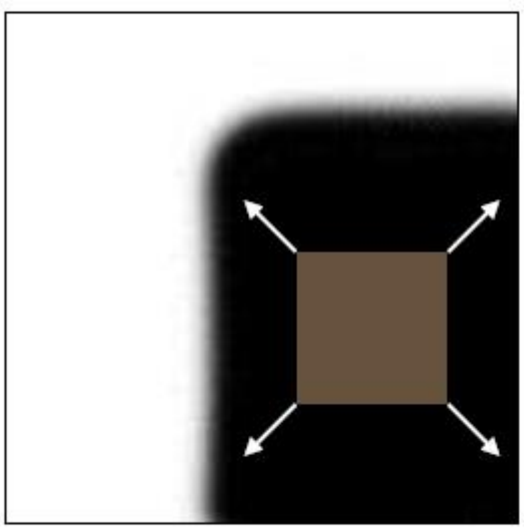
Canny with $\sigma = 2$

James Hayes

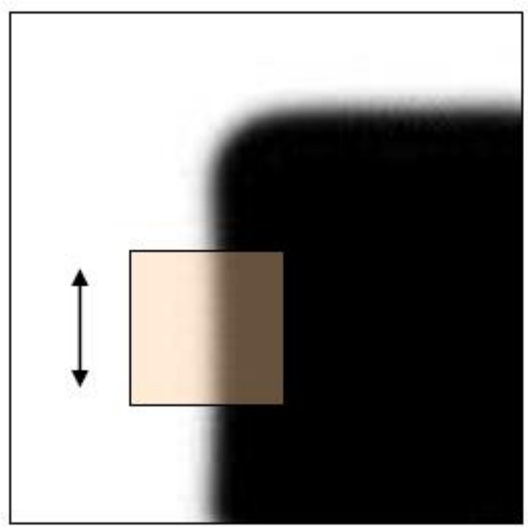
1.2 Corner Detection

Basic Idea

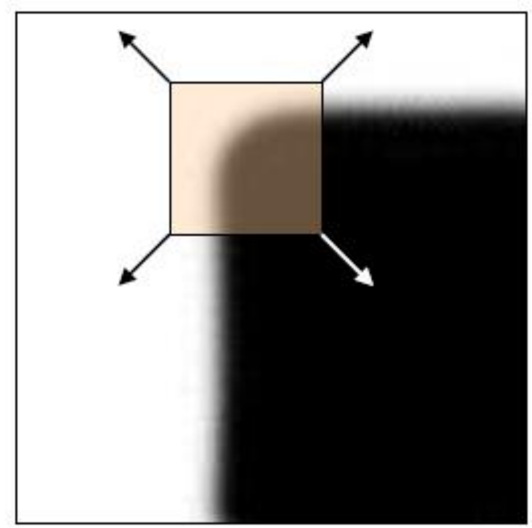
- We should easily recognize the point by looking through a small window.
- Shifting a window in any direction should give a large change in intensity.



"flat" region: no change in all directions



"edge": no change along the edge direction



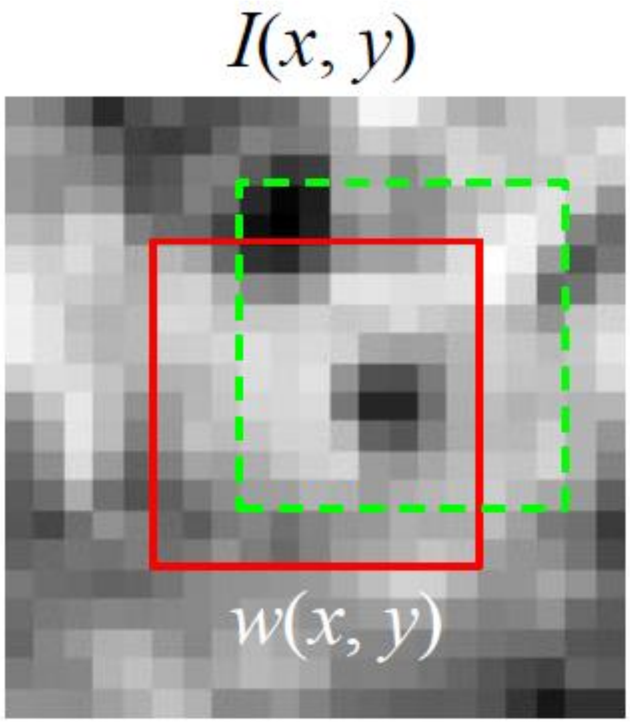
"corner": significant change in all directions

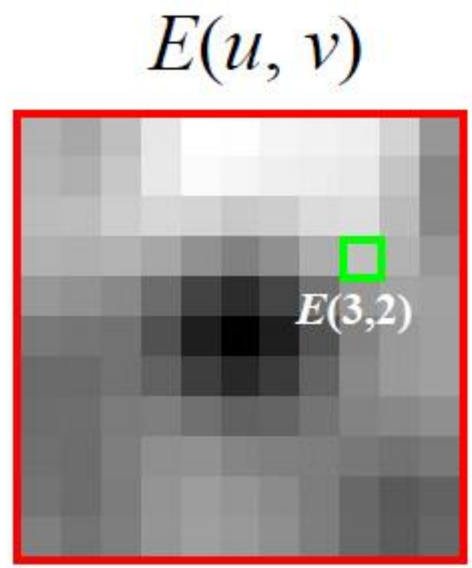
James Hayes

Mathematics

- Change in appearance of a window $w(x, y)$ for the shift $[u, v]$.
- $E(u, v) = \sum_{x,y} w(x, y)[I(x + u, y + v) - I(x, y)]^2$

  Sum is over image region (the area we are checking for corner)





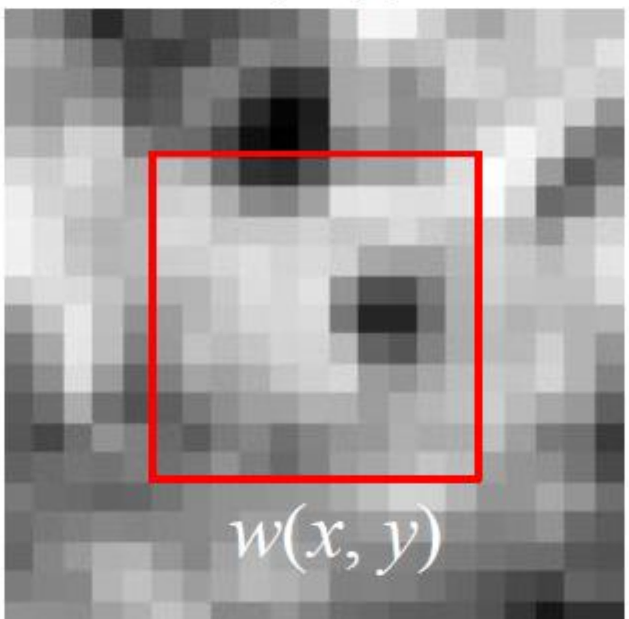
James Hayes

Mathematics – (cont.)

- Change in appearance of a window $w(x, y)$ for the shift $[u, v]$.
- $E(u, v) = \sum_{x,y} w(x, y)[I(x + u, y + v) - I(x, y)]^2$

Sum is over image region (the area we are checking for corner)

$I(x, y)$



$E(u, v)$

James Hayes

Mathematics – (cont.)

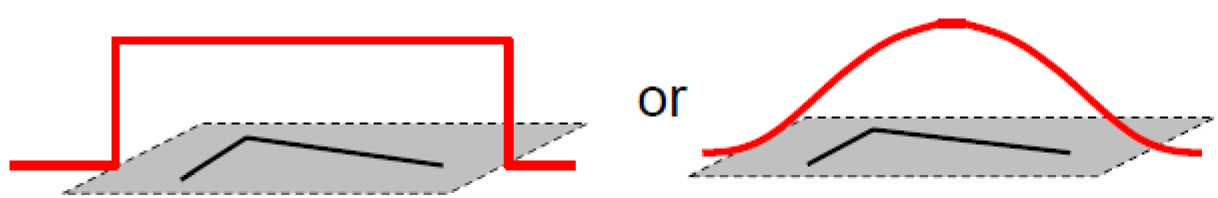
- Change in appearance of a window $w(x,y)$ for the shift $[u,v]$.
- $E(u,v) = \sum_{x,y} w(x,y)[I(x+u, y+v) - I(x,y)]^2$

Window function

Shifted intensity

Intensity

Window function $w(x,y)$ =



1 in window, 0 outside

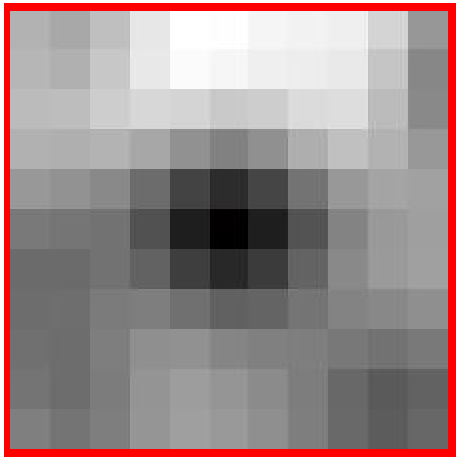
Gaussian

James Hayes

Mathematics – (cont.)

- Change in appearance of a window $w(x, y)$ for the shift $[u, v]$.
- $E(u, v) = \sum_{x,y} w(x, y)[I(x+u, y+v) - I(x, y)]^2$
- We want to find out how this function behaves for small shifts.

$E(u, v)$



But this is very slow to compute naively.
O(window_width^2 * shift_range^2 * image_width^2)
O(11^2 * 11^2 * 600^2) = 5.2 billion of these
14.6 thousand per pixel in your image

James Hayes

Mathematics – (cont.)

- Change in appearance of a window $w(x, y)$ for the shift $[u, v]$.
- $E(u, v) = \sum_{x,y} w(x, y)[I(x + u, y + v) - I(x, y)]^2$
- We want to find out how this function behaves for small shifts.

Recall Taylor series expansion. A function f can be approximated at point a as

$$\begin{aligned} f(x) &\approx \sum_{k=0}^{n} \frac{f^{(k)}(a)}{k!}(x-a)^k \\ &= f(a) + f'(a)(x-a) + \frac{1}{2}f''(a)(x-a)^2 + \ldots \end{aligned}$$

James Hayes

Mathematics – (cont.)

- $E(u,v) = \sum_{x,y} w(x,y)[I(x+u,y+v) - I(x,y)]^2$
- We want to find out how this function behaves for small shifts.
- Local quadratic approximation of $E(u,v)$ in the neighborhood of *(0,0)* is given by the *second-order Taylor expansion*:

$$E(u,v) \approx \underbrace{E(0,0)}_{\text{Always zero}} + [u \;\; v]\underbrace{\begin{bmatrix} E_u(0,0) \\ E_v(0,0) \end{bmatrix}}_{\text{First derivative is zeros}} + \frac{1}{2}[u \;\; v]\begin{bmatrix} E_{uu}(0,0) & E_{uv}(0,0) \\ E_{uv}(0,0) & E_{vv}(0,0) \end{bmatrix}\begin{bmatrix} u \\ v \end{bmatrix}$$

James Hayes

Mathematics – (cont.)

- The quadratic approximation simplifies to

$$E(u,v) \approx [u\ \ v]\ M \begin{bmatrix} u \\ v \end{bmatrix}$$

- Where M is a second moment matrix computed from image derivatives:

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

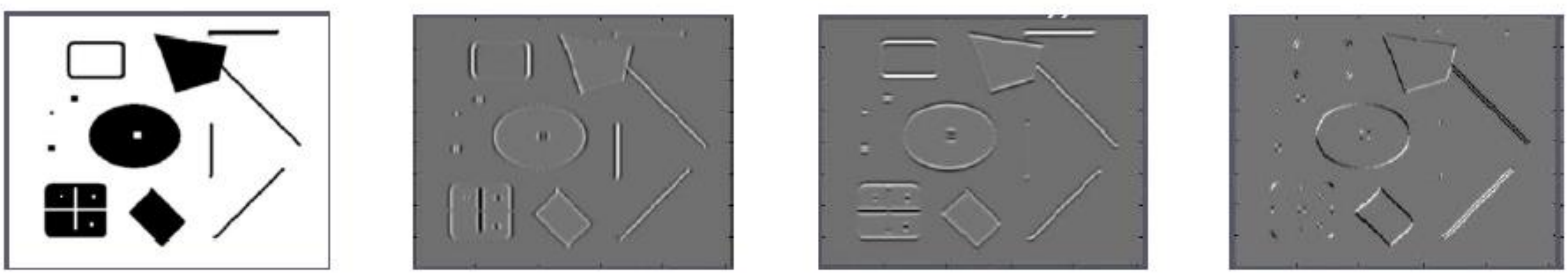
$$M = \begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} = \sum \begin{bmatrix} I_x \\ I_y \end{bmatrix} [I_x\ I_y] = \sum \nabla I (\nabla I)^T$$

James Hayes

Corners

- 2 x 2 matrix of image derivatives (averaged in neighborhood of a point).

$$M = \sum w(x,y) \begin{bmatrix} I_x I_x & I_x I_y \\ I_x I_y & I_y I_y \end{bmatrix}$$



Notation: $I_x \Leftrightarrow \frac{\partial I}{\partial x}$ $\quad I_y \Leftrightarrow \frac{\partial I}{\partial y}$ $\quad I_x I_y \Leftrightarrow \frac{\partial I}{\partial x}\frac{\partial I}{\partial y}$

James Hayes

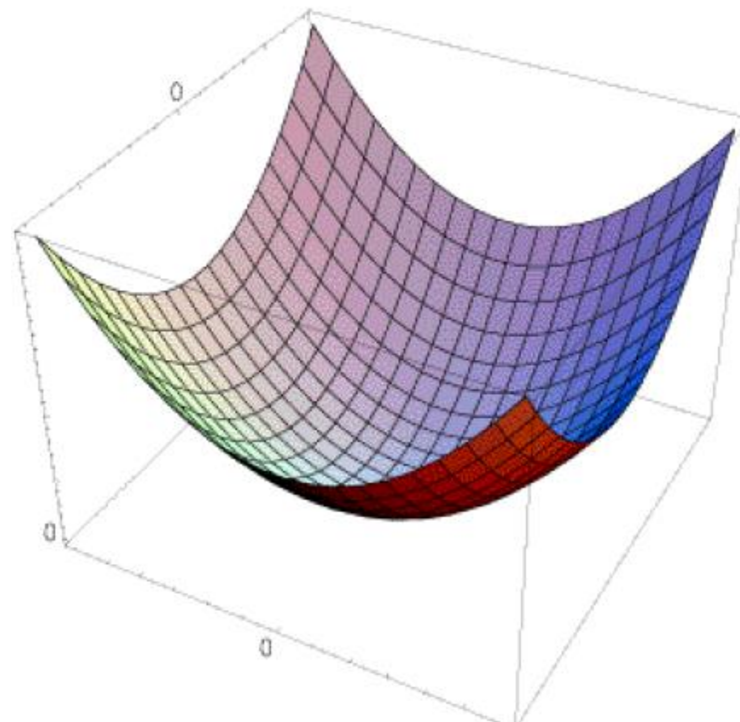
Corners – (cont.)

Interpreting the Second Moment Matrix

- The surface $E(u,v)$ is locally approximated by a quadratic form. Let's try to understand its shape.

$$E(u,v) \approx [u \;\; v] \; M \begin{bmatrix} u \\ v \end{bmatrix}$$

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

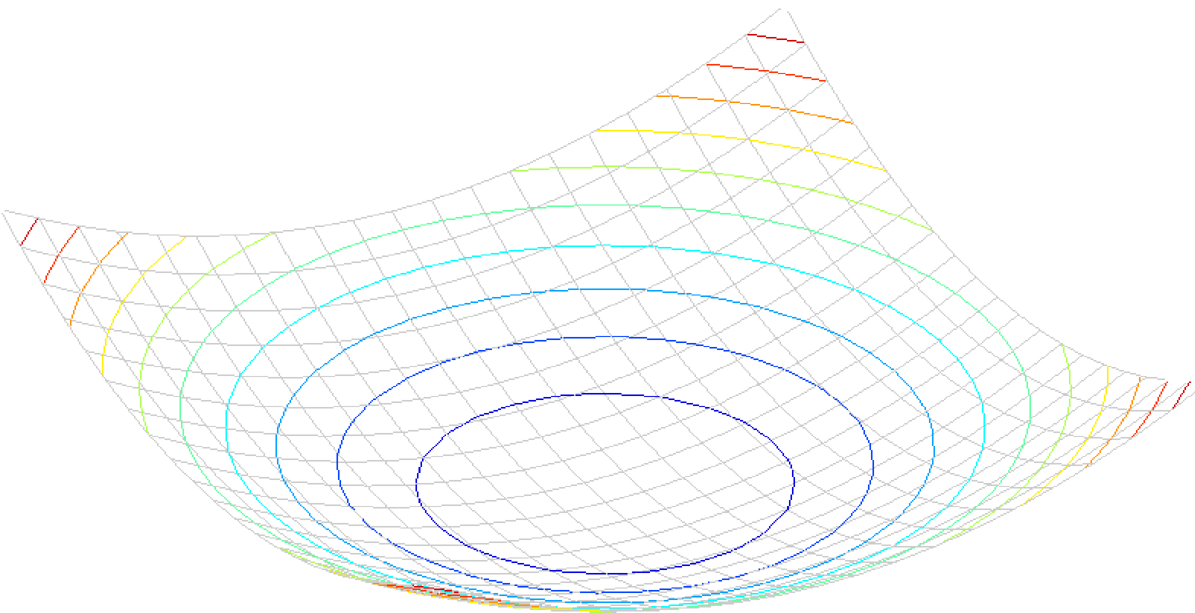


James Hayes

Corners – (cont.)

Interpreting the Second Moment Matrix

- Consider a horizontal "slice" of $E(u, v)$; this is the equation of an ellipse.

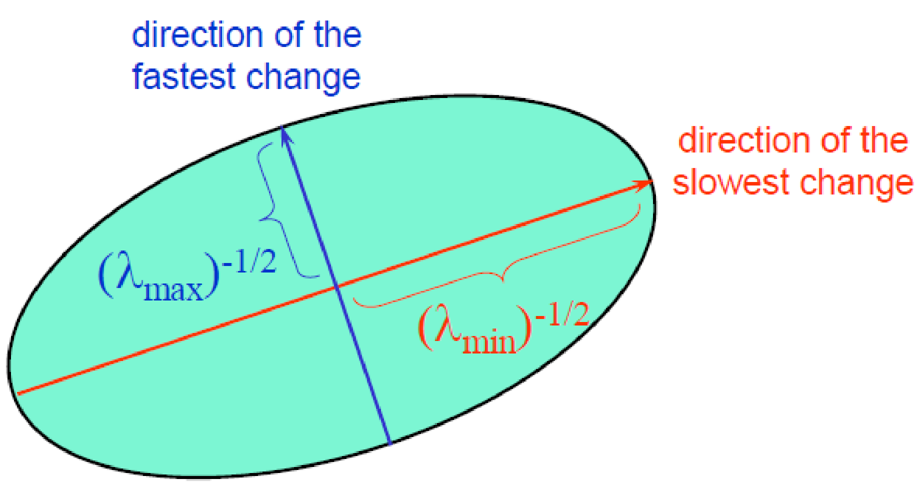


James Hayes

Corners – (cont.)

Interpreting the Second Moment Matrix

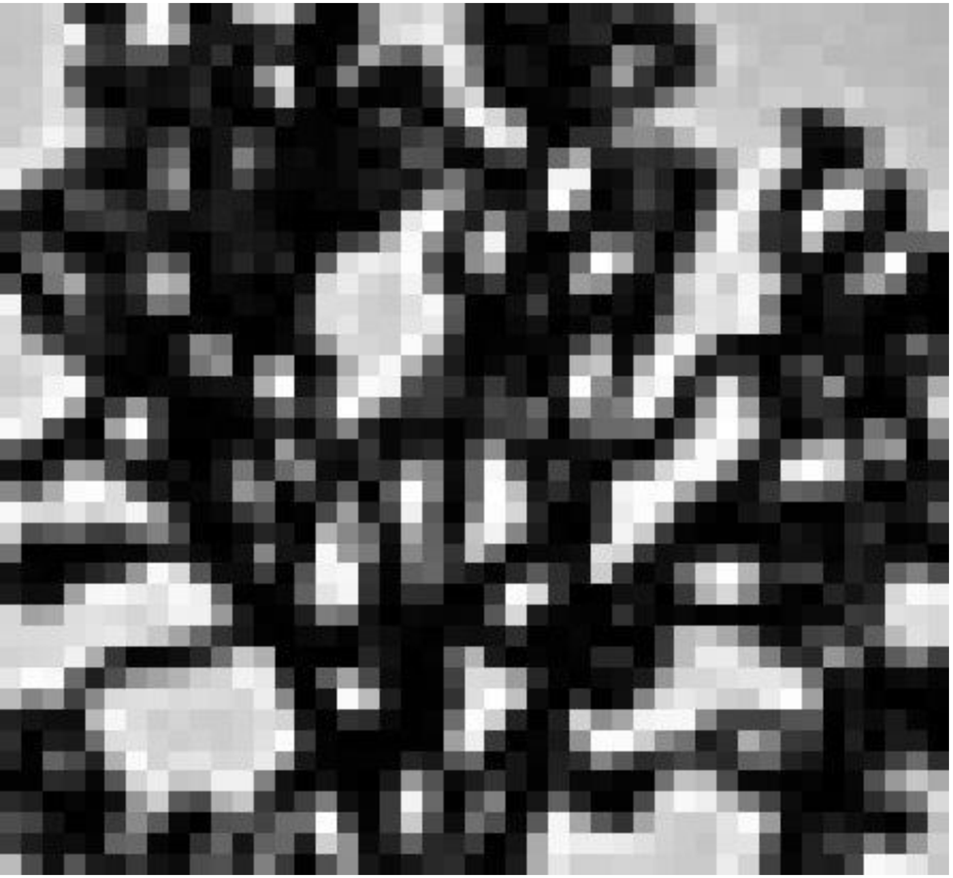
- Consider a horizontal "slice" of $E(u, v)$; this is the equation of an ellipse.
- Diagonalization of M results $M = R^{-1}\begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R$
- The axis lengths of the ellipse are determined by the eigenvalues and the orientation is determined by R.

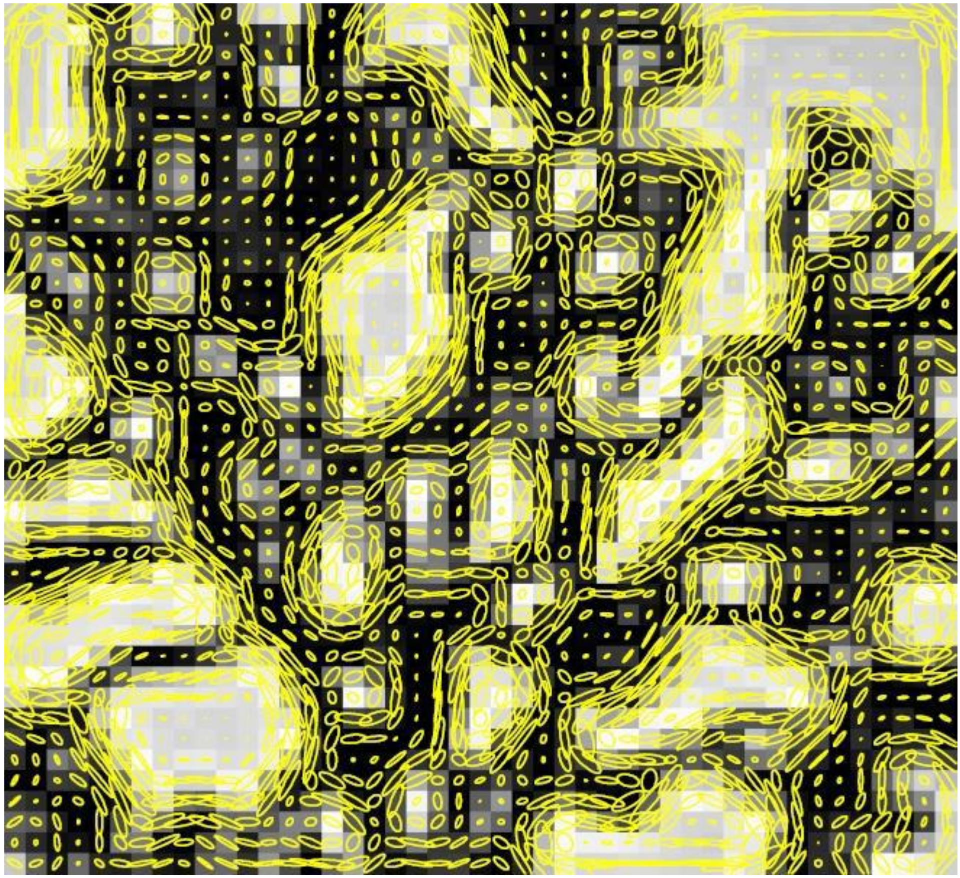


James Hayes

Corners – (cont.)

Visualization of the Second Moment Matrices



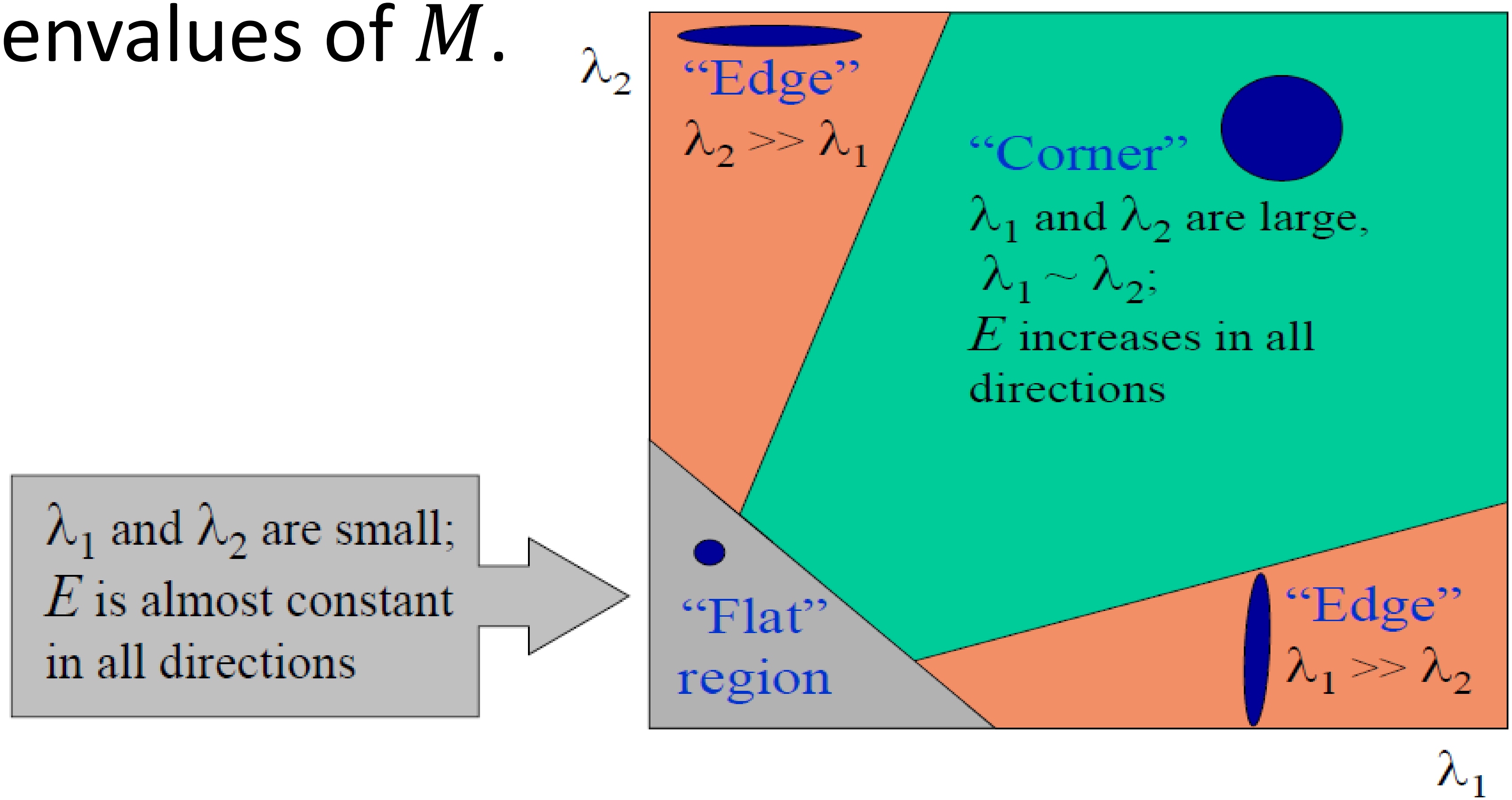


James Hayes

Corners – (cont.)

Interpreting the Eigenvalues

- Classification of image points using eigenvalues of M.



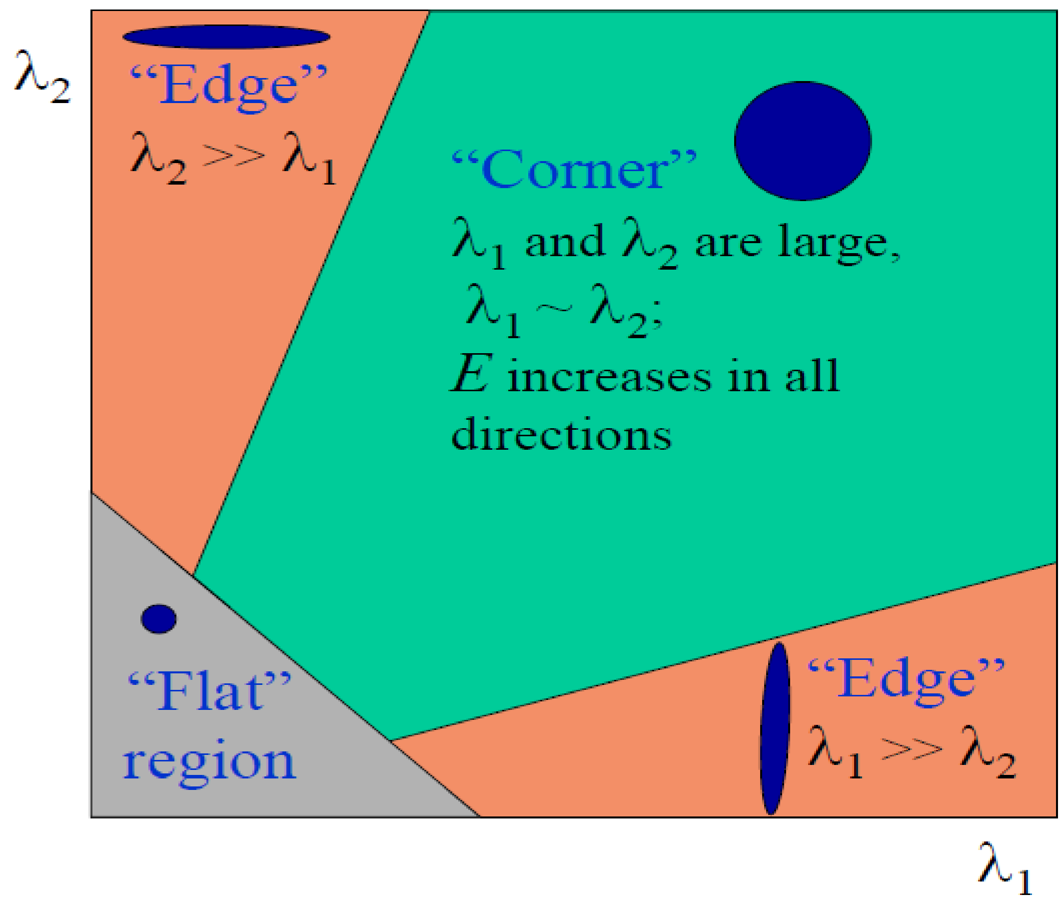
Corners – (cont.)

Corner Response Function

- $R = \det(M) - \alpha\, trace(M)^2$

$= \lambda_1\lambda_2 - \alpha(\lambda_1 + \lambda_2)^2.$

α: constant (0.04 to 0.06)



Harris Corner Detector

Steps

1. Compute M matrix for each image window to get their cornerness scores.

2. Find points whose surrounding window gave large corner response (f > threshold)

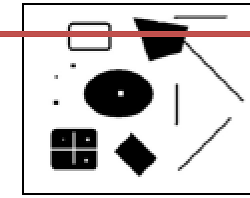
3. Take the points of local maxima, i.e., perform non-maxima suppression.

C.Harris and M.Stephens. "A Combined Corner and Edge Detector." *Proceedings of the 4th Alvey Vision Conference*: pages 147—151, 1988.

James Hayes

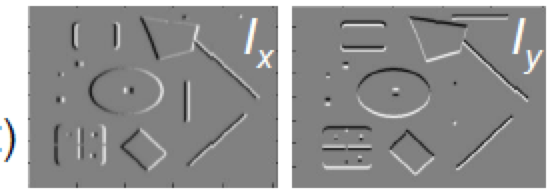
Harris Corner Detector – (cont.)

- Second moment matrix



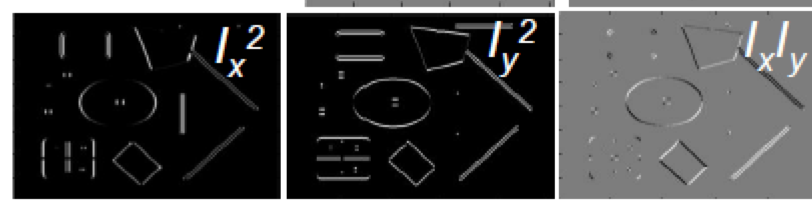
$$\mu(\sigma_I,\sigma_D) = g(\sigma_I) * \begin{bmatrix} I_x^2(\sigma_D) & I_xI_y(\sigma_D) \\ I_xI_y(\sigma_D) & I_y^2(\sigma_D) \end{bmatrix}$$

1. Image derivatives (optionally, blur first)



$$\det M = \lambda_1\lambda_2$$
$$\operatorname{trace} M = \lambda_1 + \lambda_2$$

2. Square of derivatives

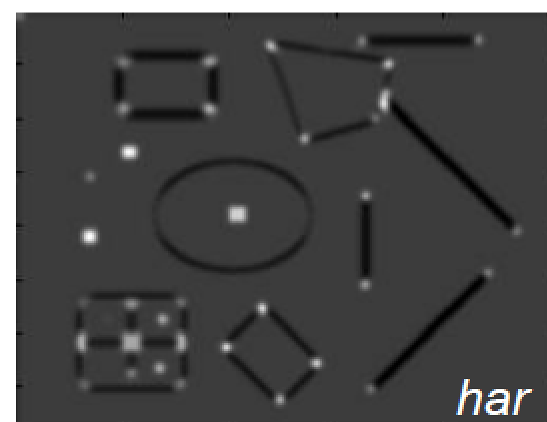


3. Gaussian filter $g(\sigma_I)$

4. Cornerness function – both eigenvalues are strong

$$har = \det[\mu(\sigma_I,\sigma_D)] - \alpha[\operatorname{trace}(\mu(\sigma_I,\sigma_D))^2] =$$
$$g(I_x^2)g(I_y^2) - [g(I_xI_y)]^2 - \alpha[g(I_x^2)+g(I_y^2)]^2$$

5. Non-maxima suppression

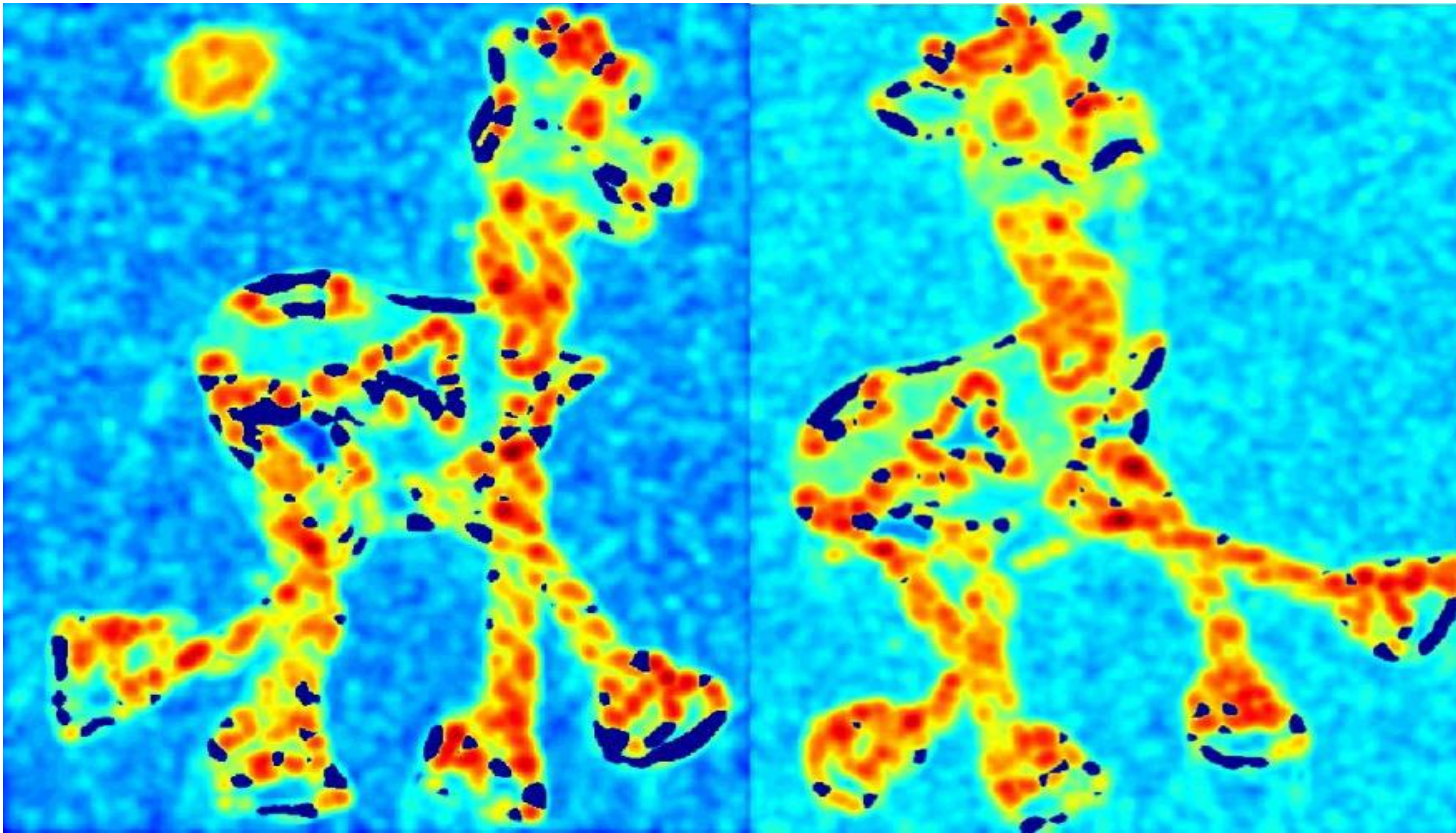


Harris Corner Detector – (cont.)

Example

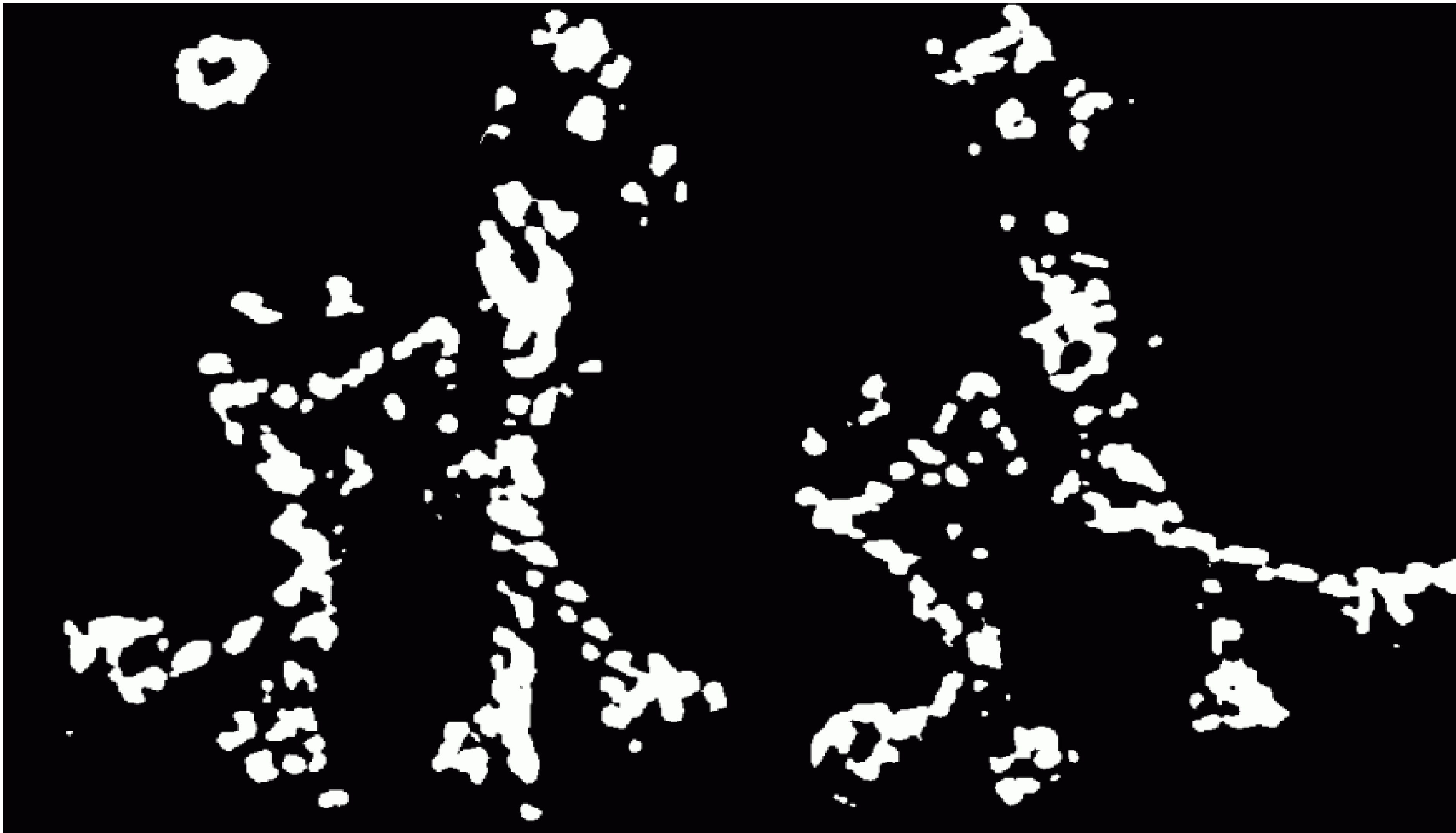
Harris Corner Detector – (cont.)

Example: corner response R



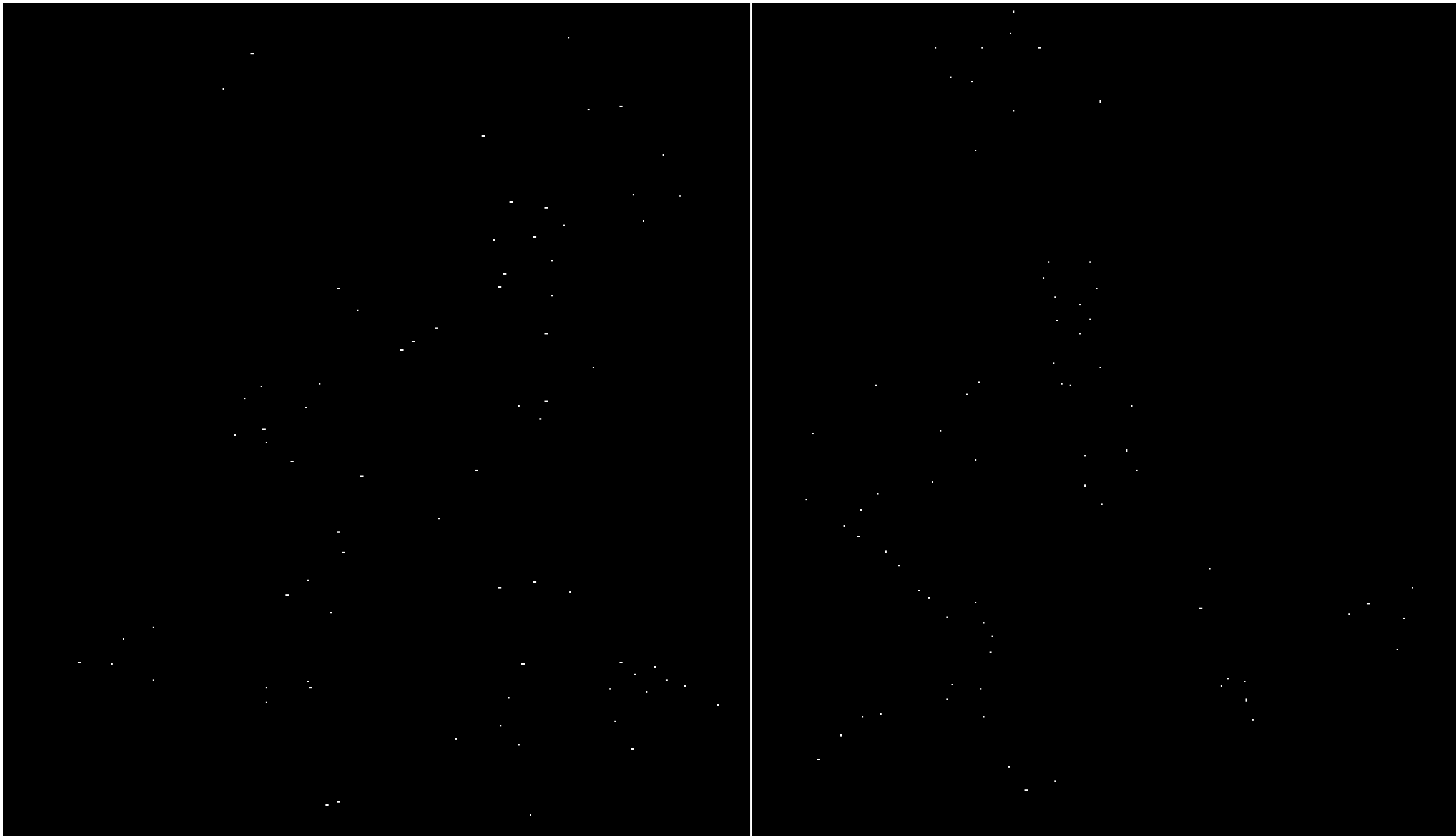
Harris Corner Detector – (cont.)

Example: points with large corner response: R>threshold



Harris Corner Detector – (cont.)

Example: only the points of local maxima of R

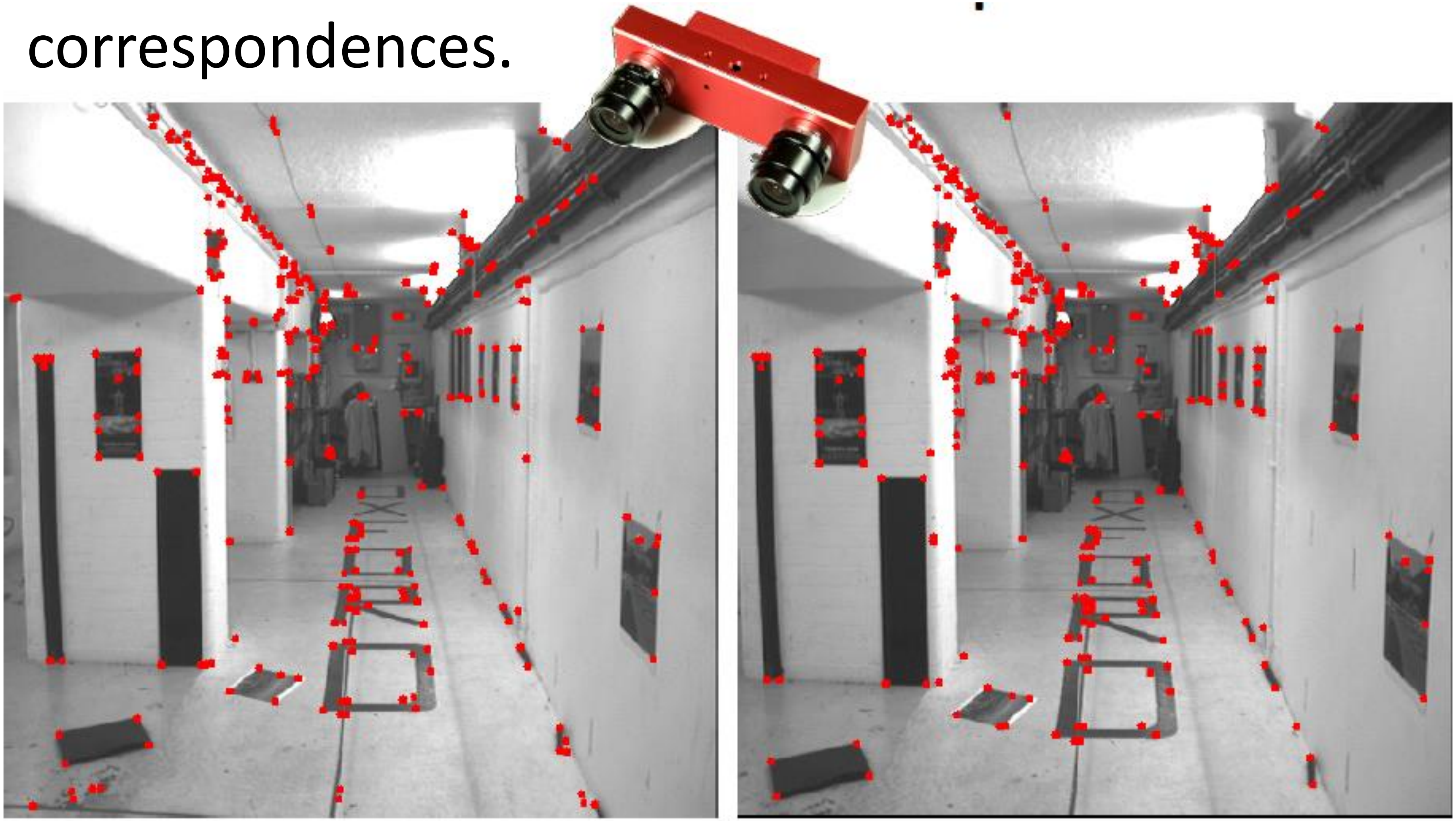


Harris Corner Detector – (cont.)

Example

Harris Corner Detector – (cont.)

- Results are well suited for finding stereo correspondences.



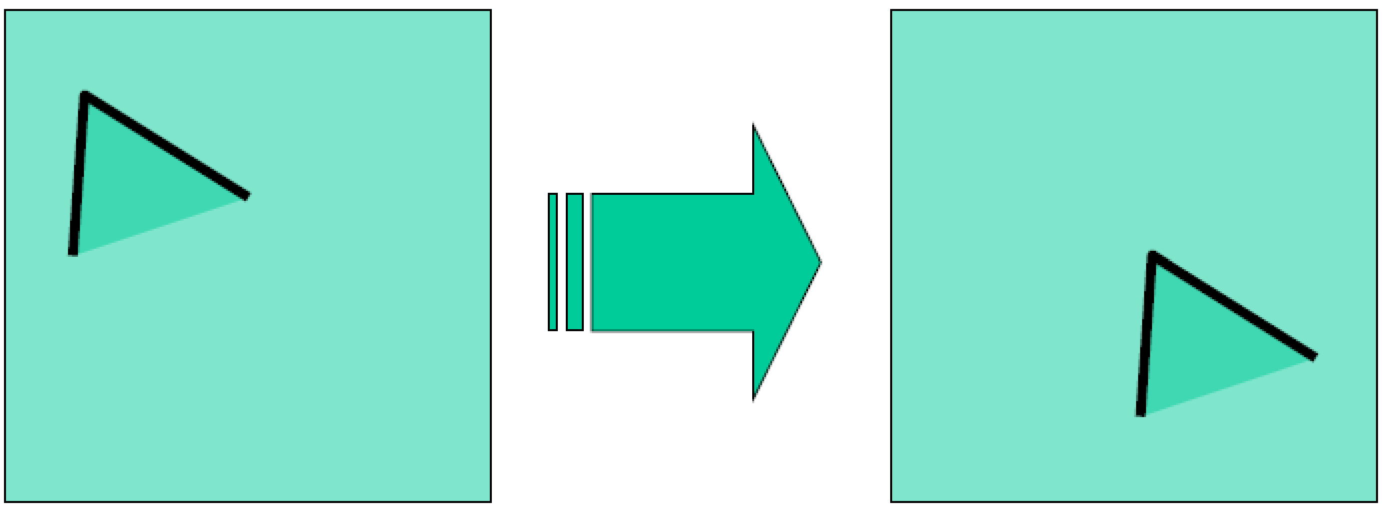
Fei-Fei Li

Invariance

- Translation?
- Rotation?
- Scale?

Invariance - Translation

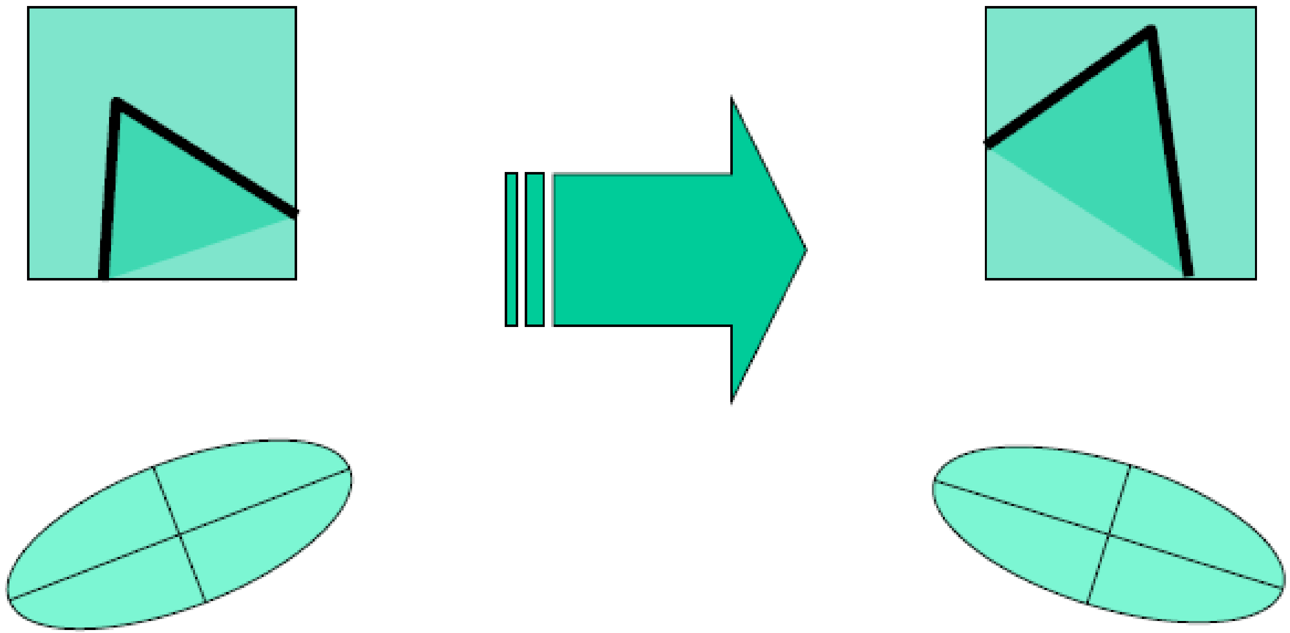
- Derivatives and window function are shift-invariant.
- Corner location is covariant w.r.t. translation



James Hayes

Invariance - Rotation

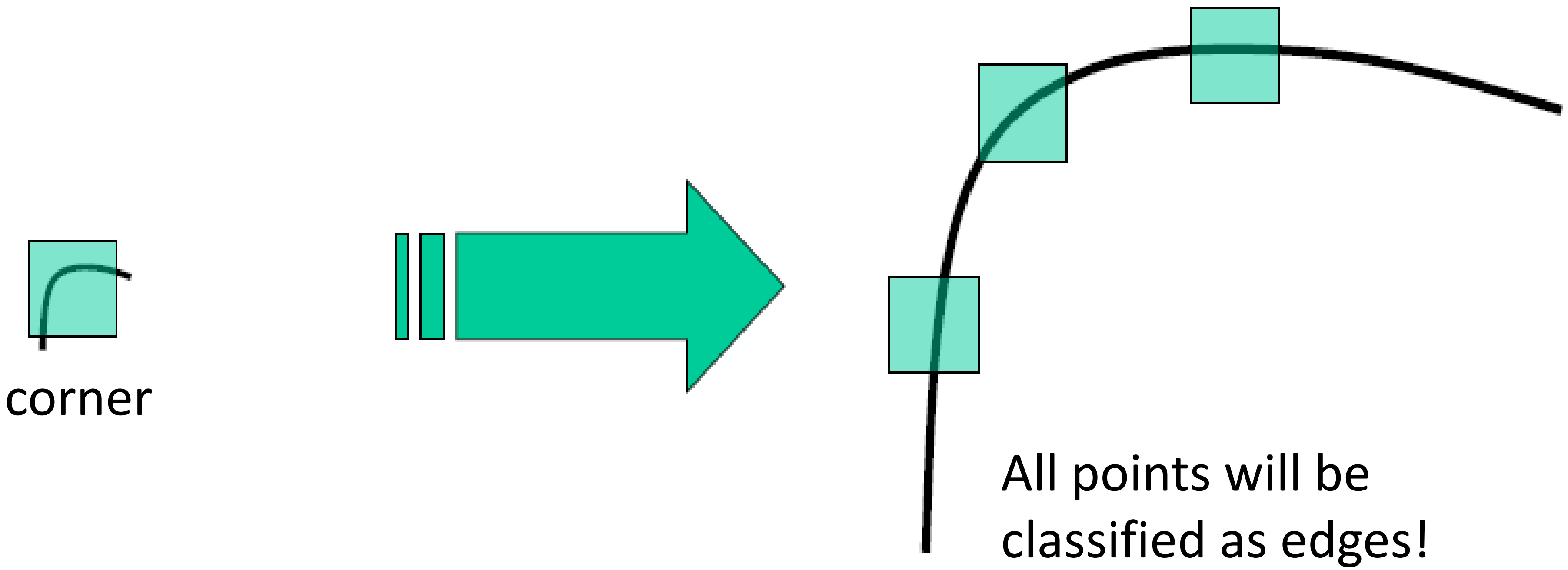
- Second moment ellipse rotates but its shape (i.e. eigenvalues) remains the same.
- Corner response R is invariant to image rotation.



James Hayes

Invariance - Scale

- Not invariant to image scale.



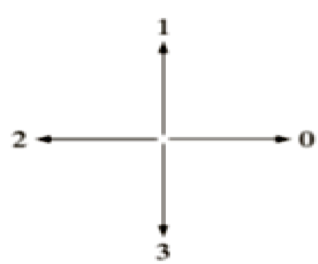
James Hayes

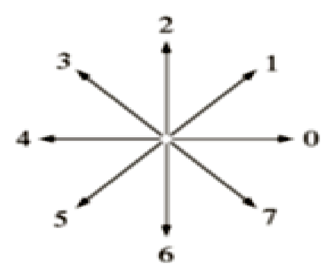
2. Descriptors

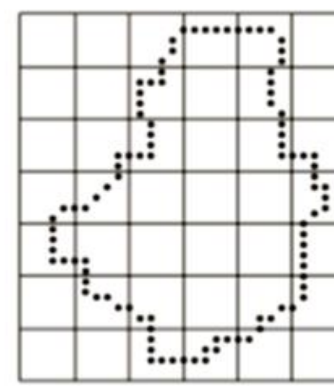
Chain Code (11.1.2)

- Used to represent a boundary by a connected sequence of straight-line segments of specific length and direction.
- Based on 4- and 8-connectivity.
- Direction is coded by a numbering scheme in a clockwise/counterclockwise orientation

→ Freeman Chain Code



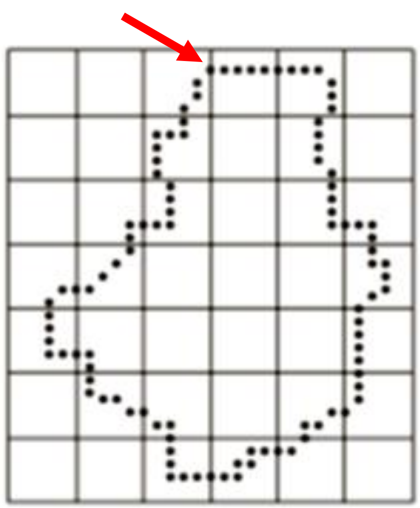


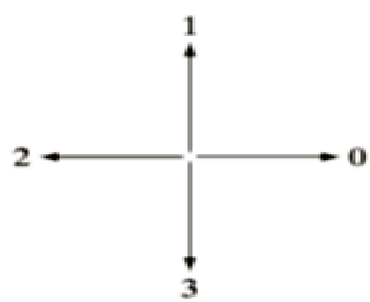


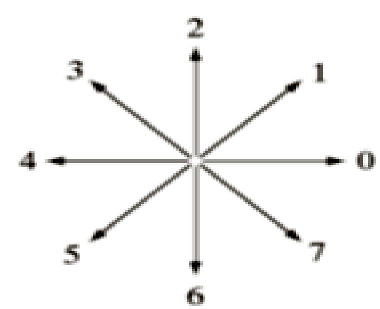
Chain Code – (cont.)

(11.1.2) Chain Code (Freeman)

- Starting point: uppermost leftmost.







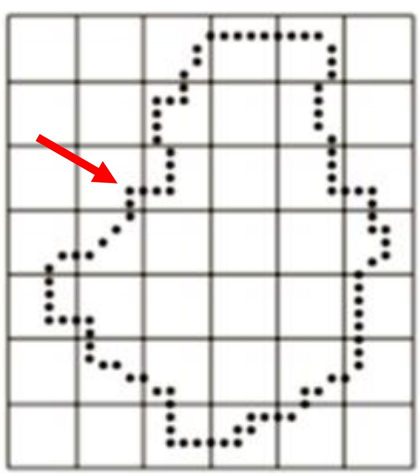
000000000 03333 2 …
Or
000000000 30333 2…

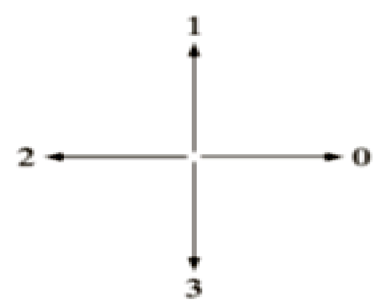
000000000766 …

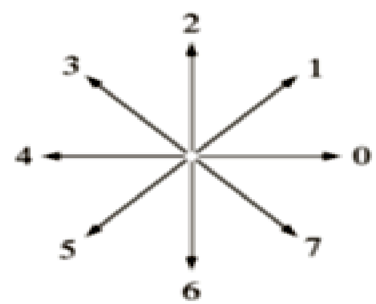
Chain Code – (cont.)

(11.1.2) Chain Code (Freeman) – Problems

1. Too long if the boundary is followed exactly.
2. Depends on the starting point.
3. Too sensitive to noise and small changes.







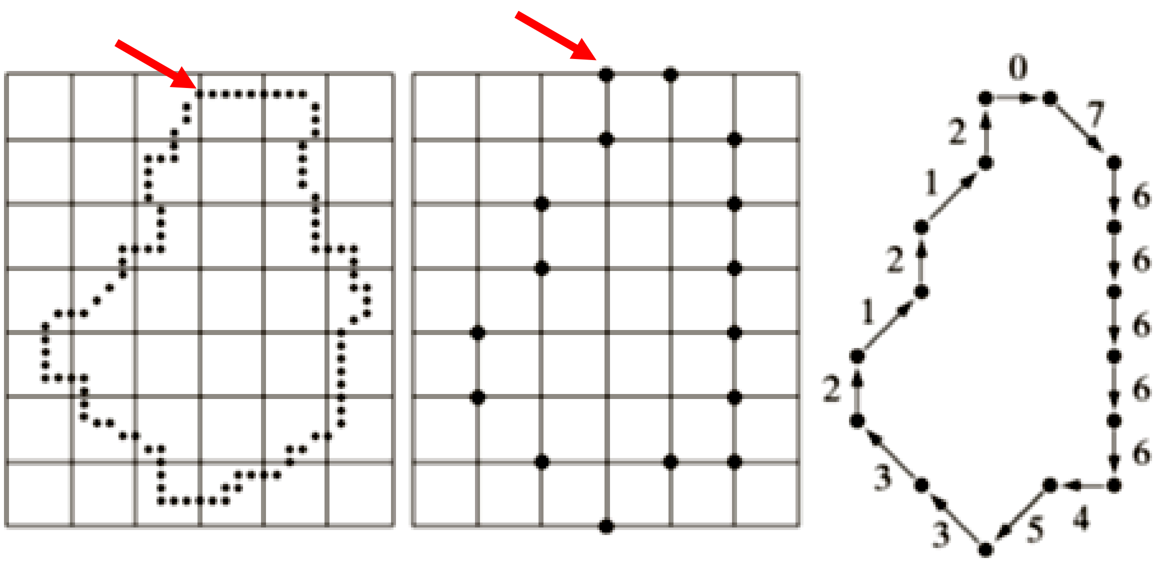
000000000 03333 2 …
Or
000000000 30333 2…

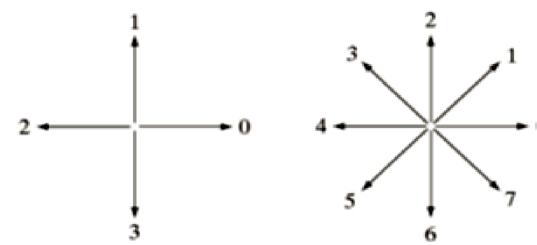
000000000766 …

Chain Code – (cont.)

(11.1.2) Chain Code – Solutions (1)

- Re-sample the boundary on a larger grid and assign a boundary point to each node.
- Accuracy of code depends on the sampling spacing.





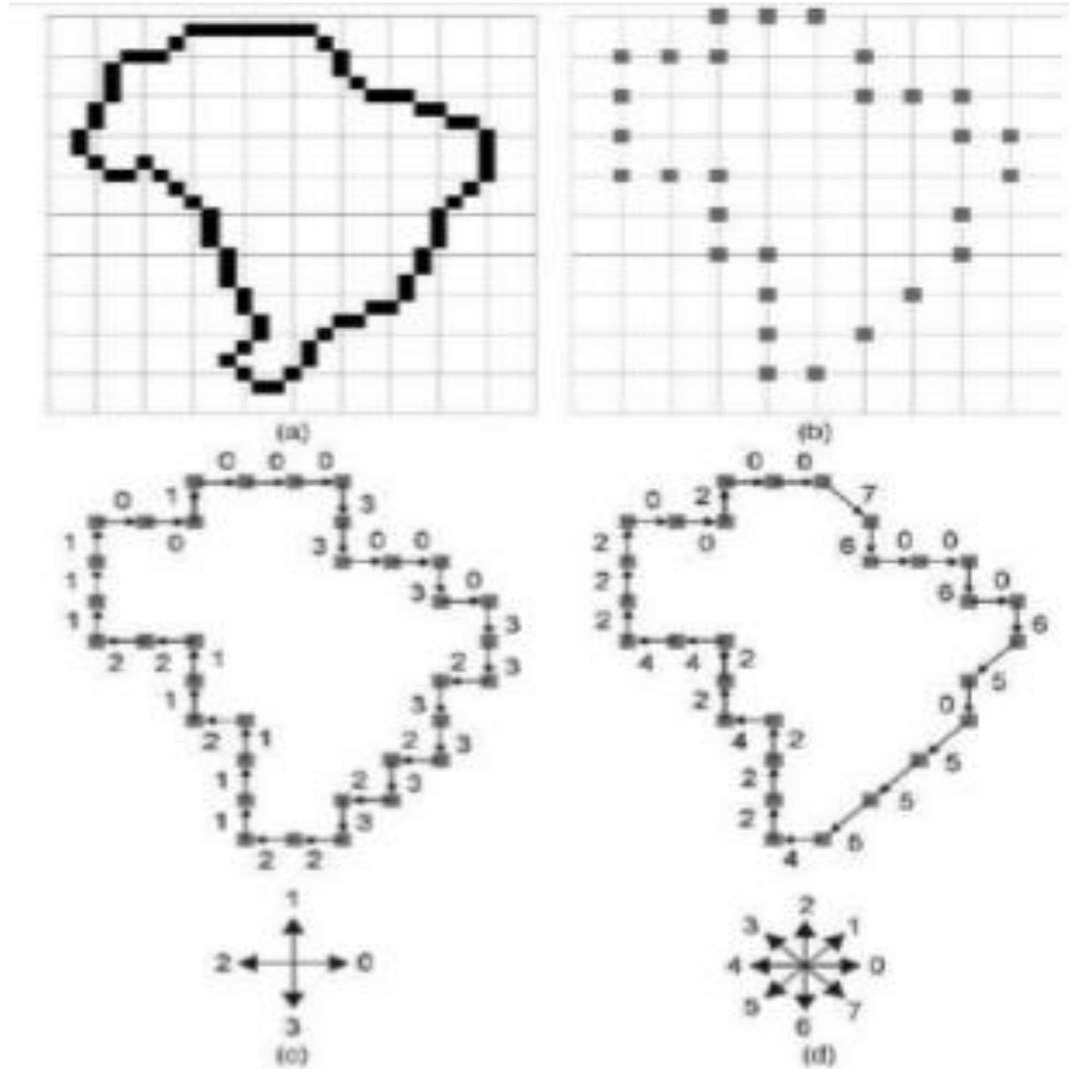
076666453321212

8-connectivity

Chain Code – (cont.)

(11.1.2) Chain Code

- Example



Chain Code – (cont.)

(11.1.2) Chain Code – Example

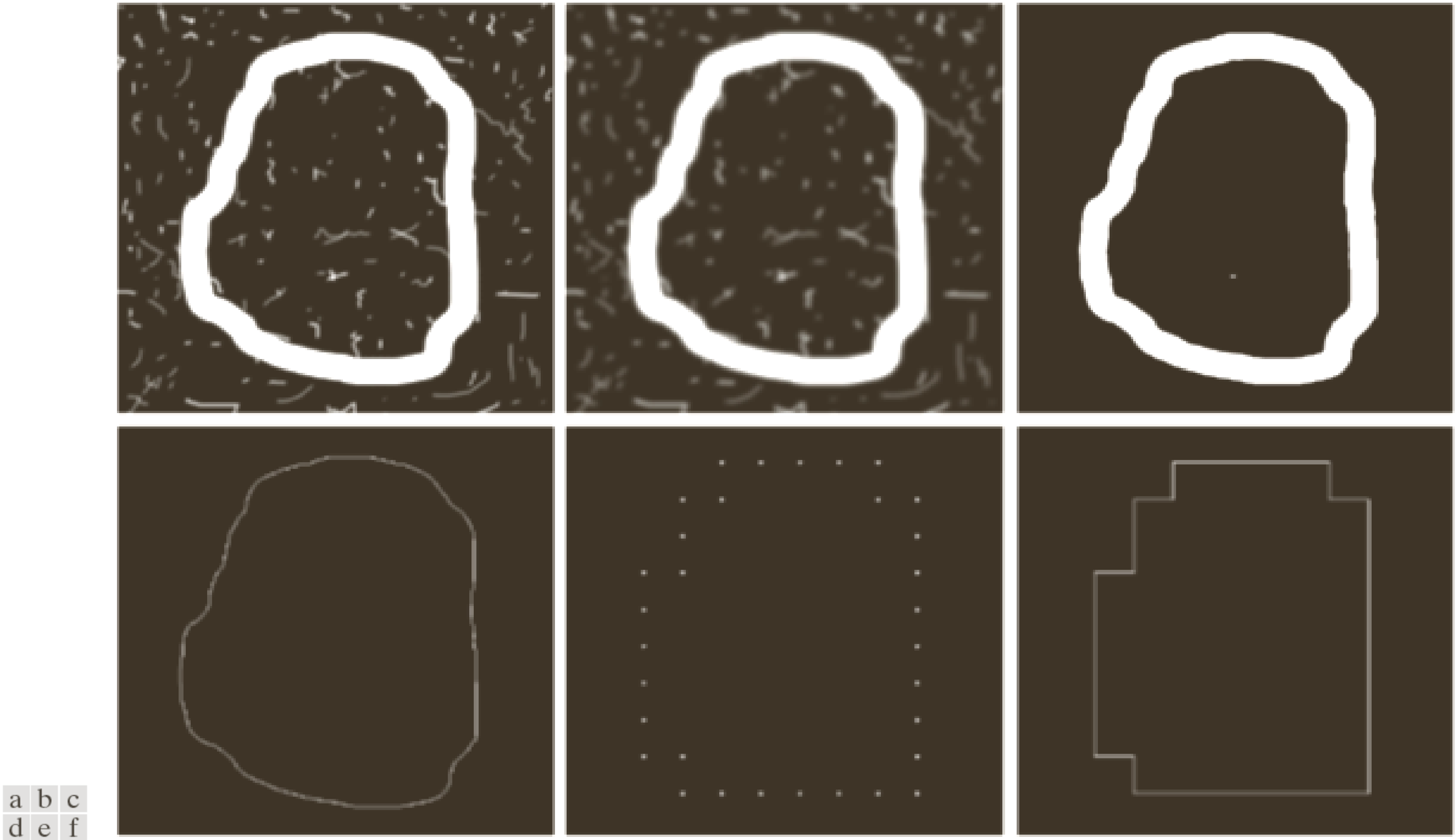


FIGURE 11.5 (a) Noisy image. (b) Image smoothed with a 9 × 9 averaging mask. (c) Smoothed image, thresholded using Otsu's method. (d) Longest outer boundary of (c). (e) Subsampled boundary (the points are shown enlarged for clarity). (f) Connected points from (e).

Chain Code – (cont.)

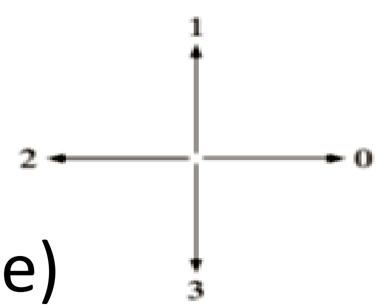
(11.1.2) Chain Code – Solutions (2,3)

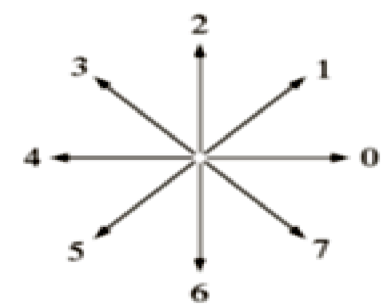
- **Normalize to start point:** redefine start point by circulating sequence → smallest integer.
- **Normalizing to scale:** re-sizing sampling grid.
- **Normalize to rotation (not all angles):** use first difference of code instead of code itself.
  - **In a 4-directional counterclockwise code:**

10103322 becomes

3133030 (difference)

0303133 (minimum magnitude)





Chain Code – (cont.)

(11.1.2) Chain Code – Example

- 8-directional counterclockwise Freeman chain code:

000060666666664444442422222202202

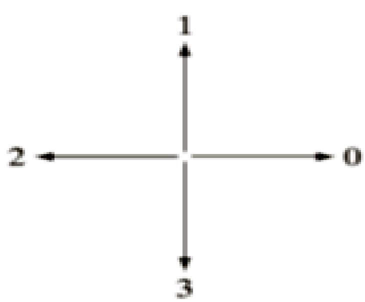
(already minimum magnitude)

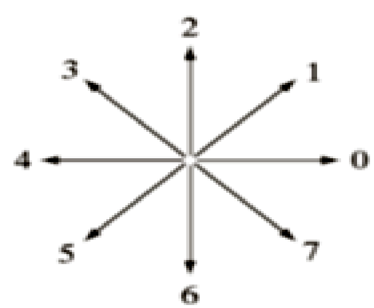
- First difference code:

00062600000006000006260000062062**6**

Often the diff of first and last digits is put in the start.

6**00062600000006000006260000062062**

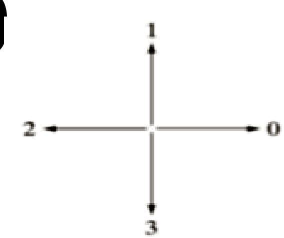




Shape Number – (11.2.2)

- The Shape number = first difference of smallest magnitude of chain code.

e.g. 10103322 → 3133030 → 33133030



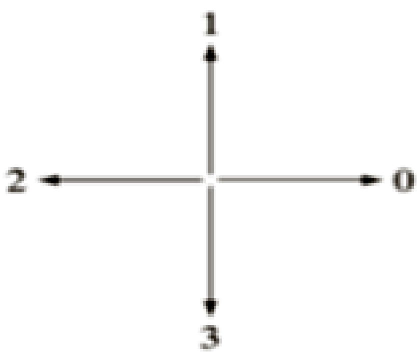
Shape number = 03033133

- Order n of shape number = number of digits.

Shape Number – (cont.)

(11.2.2) Shape Number

- Examples:



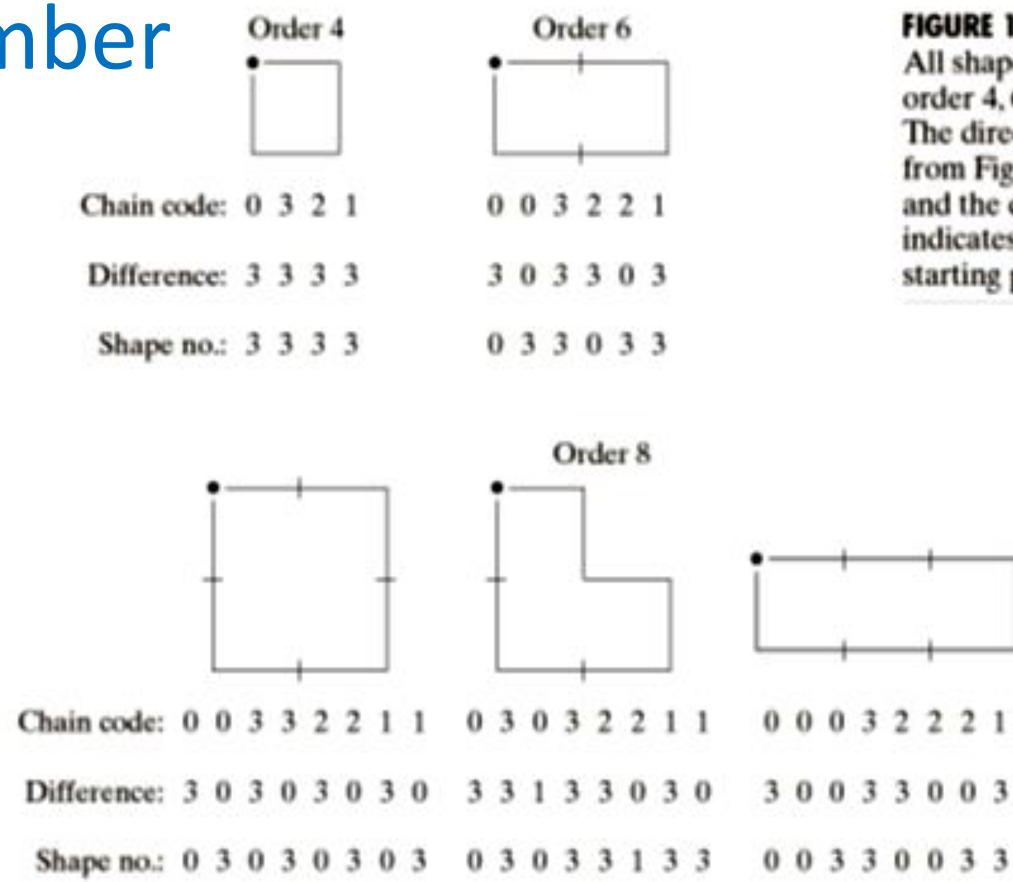


FIGURE 11.17 All shapes of order 4, 6, and 8. The directions are from Fig. 11.3(a), and the dot indicates the starting point.

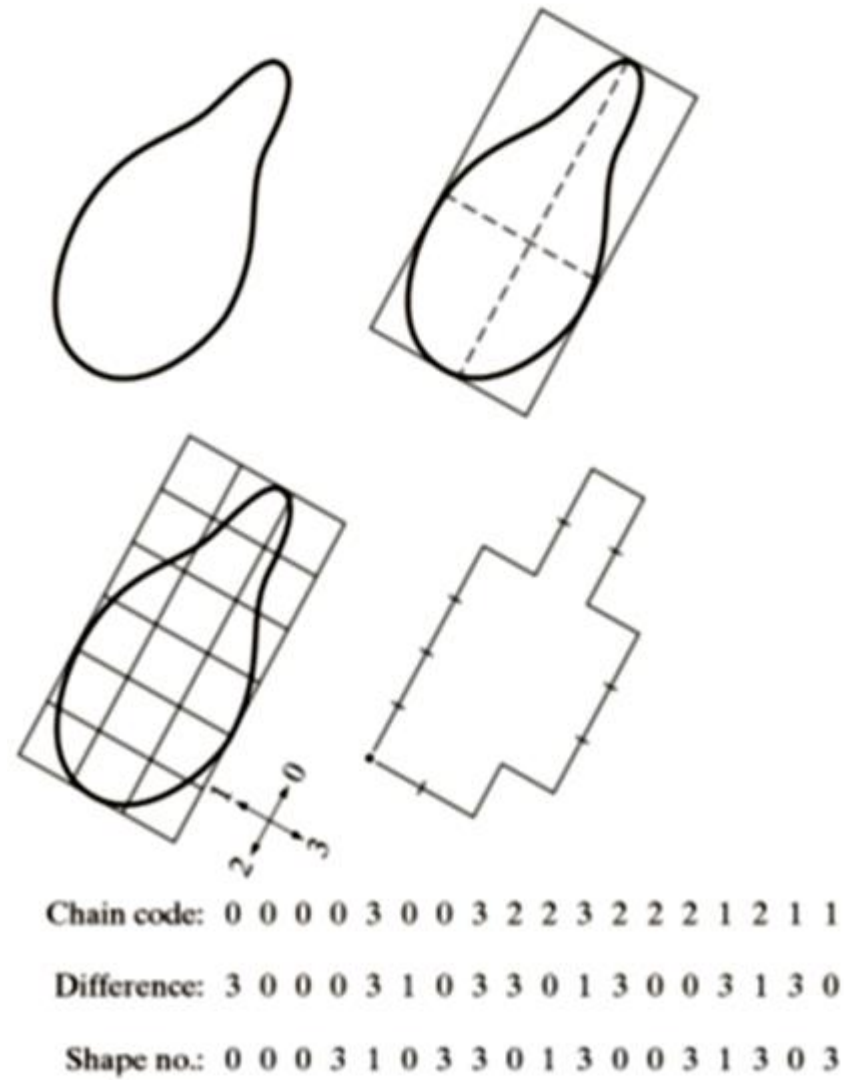
- In practice, align the chain code grid with the basic rectangle of the shape to normalize to rotation.

Shape Number – (cont.)

(11.2.2) Shape Number

- Obtain shape number of order $n = 18$

1. Find basic rectangle.
2. Define grid of size n.
3. Align direction axis to grid.
4. Obtain chain code.
5. Obtain first diff of this code.



Shape Number – (cont.)

(11.2.2) Shape Number

- High dimension → Reduction in data needed to store boundary.
- Sensitive to noise.
- Unified way to analyze boundary shape.
- Subsampled boundary can be recovered from code.

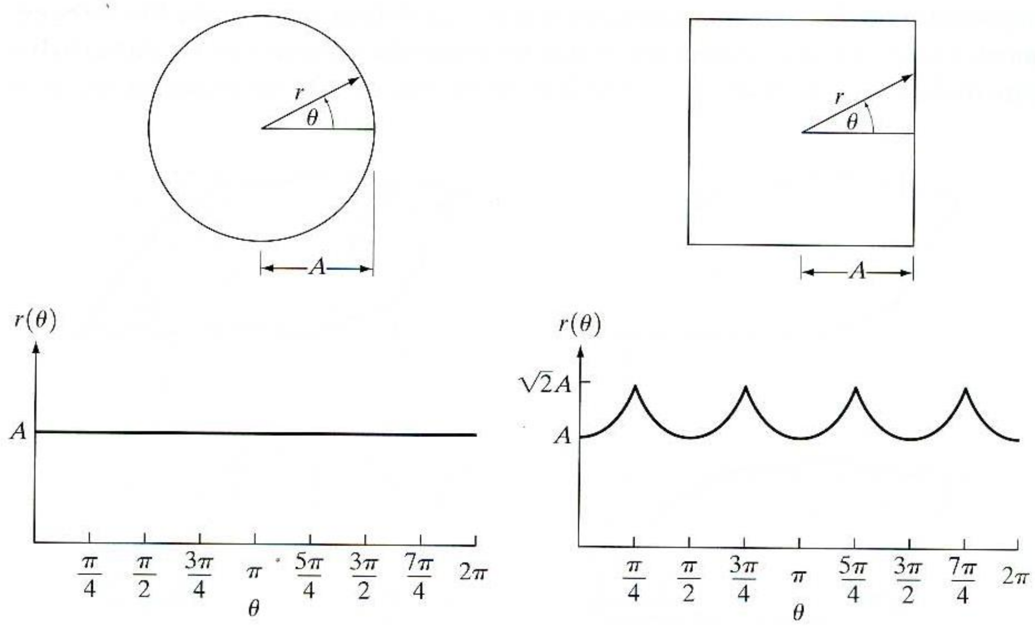
Signature (11.1.5)

- A 1D function of a boundary.
- Reduces the boundary representation to a 1D function that is easier to describe than the original 2D boundary.
- Examples: centroid distance, slope density function, cumulative angle, curvature, area, etc.

Signature – (cont.)

(11.1.5) Signature

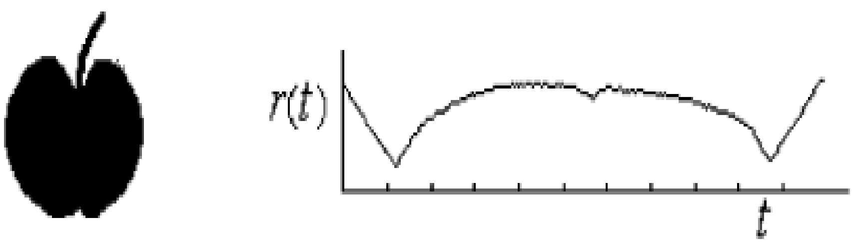
- Examples:
  - Centroid distance: Plotting distance from centroid to boundary as function of angle.



Signature – (cont.)

(11.1.5) Signature

- Examples:



An apple shape and its centroid distance signature.

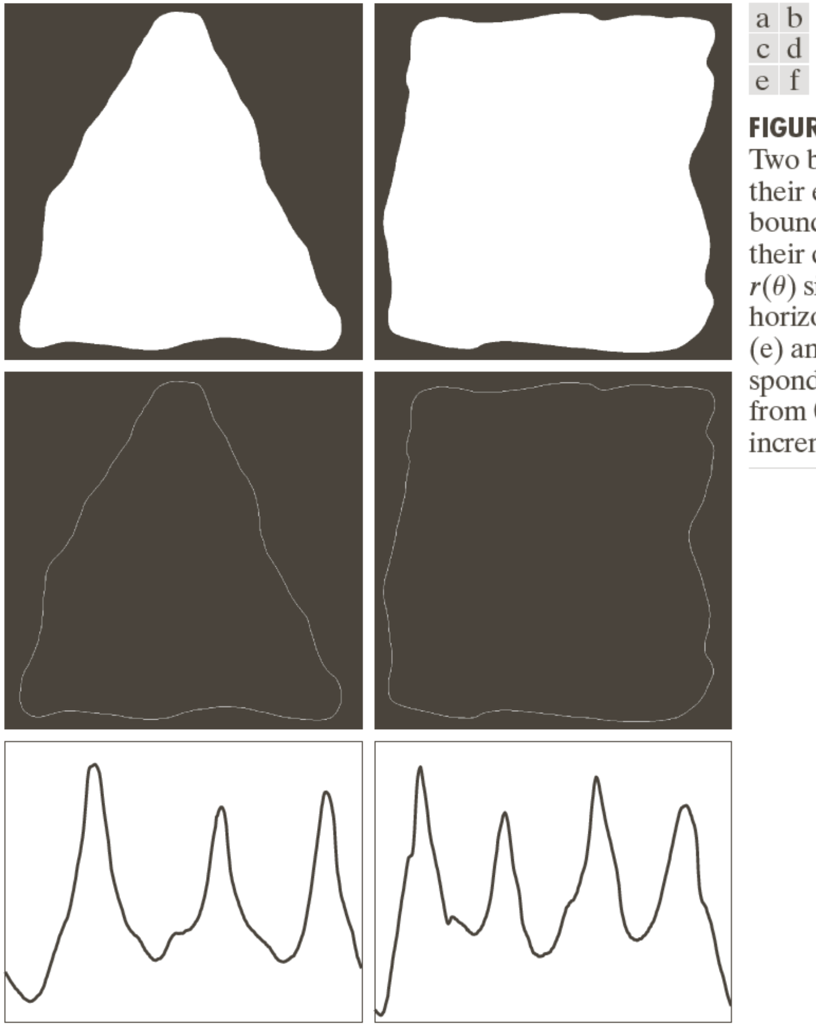


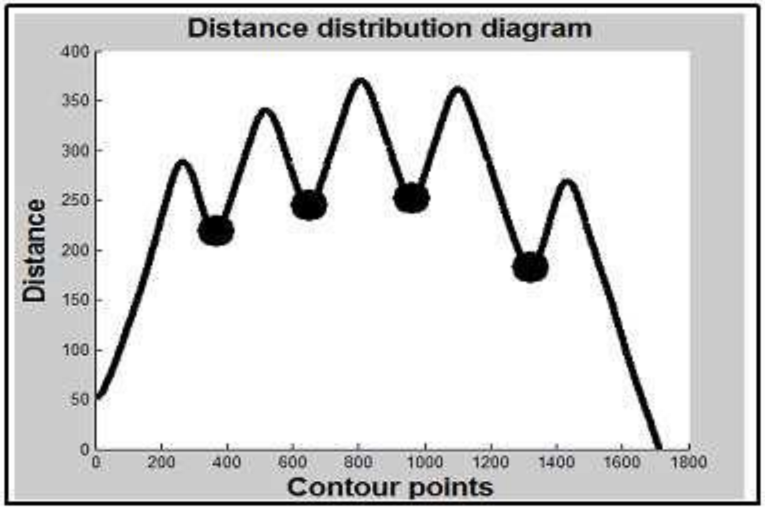
FIGURE 11.11 Two binary regions, their external boundaries, and their corresponding $r(\theta)$ signatures. The horizontal axes in (e) and (f) correspond to angles from 0° to 360°, in increments of 1°.

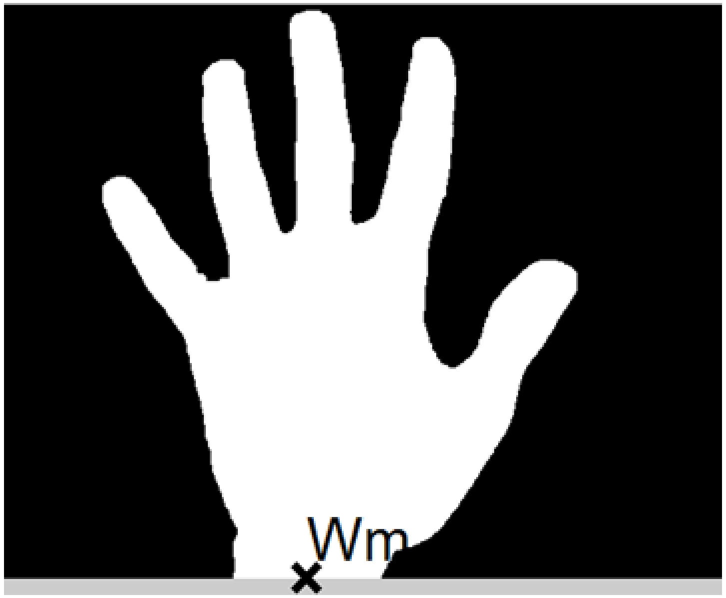
D. Zhang, and G. Lu, "*Review of Shape Representation and Description Techniques*", The Journal of Pattern Recognition, 2003.

Signature – (cont.)

(11.1.5) Signature

- Examples:





S. Ben Jemaa, M. Hammami, and H. Ben-Abdallah, "*Biometric Identification Using a New Direction in Contactless Palmprint Imaging*", IPCV12, 2012.

Signature – (cont.)

(11.1.5) Signature

- Centroid distance is translation invariant.
- **Normalizing to rotation:** select same criteria for starting point regardless of orientation
  - Select starting point farthest from centroid (assuming its unique).
  - Select a point on Eigen axis farthest from centroid.
- **Normalizing to scaling:**
  - Scale all functions to span same range, e.g. [0, 1] (noise sensitive).

Problem from Final 2012-2013

  - Divide each sample by signature variance (computation).

Signature – (cont.)

(11.1.5) Signature

- Usually normalized to be translation and scale invariant.
- To compensate for orientation changes, *shift matching* is needed to find the best matching between two shapes.
- High matching cost.
- Sensitive to noise.
- Signature histogram is rotation invariant.

More Descriptors

Representation – Textbook Outline

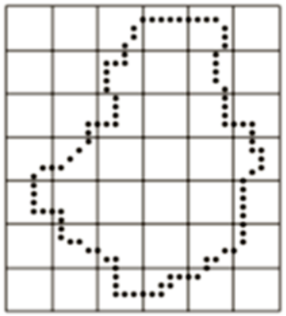
- **11.1 Representation**
  - 1. Boundary Following
  - 2. Chain Codes
  - 3. & 4. Polygon Approximation
  - 5. Signatures
  - 6. Boundary Segments
  - 7. Skeletons
- **11.2 Boundary Descriptors**
  - 1. Simple Descriptors
  - 2. Shape Number
  - 3. Fourier Descriptor
  - 4. Statistical Moments
- **11.3 Regional Descriptors**
  - 1. Simple Descriptors
  - 2. Topological
  - 3. Texture
  - 4. Moment Invariants
- **11.5 Relational Descriptors**
  - Strings, directed line segments, trees, …

Boundary-Based Descriptors

(11.2.1) Simple Descriptors

- **Perimeter**

$\boldsymbol{P}$, the number of pixels in the boundary of the shape (usually sampled).



- **Diameter**

Length and orientation of the line segment connecting the two extreme points (major axis).

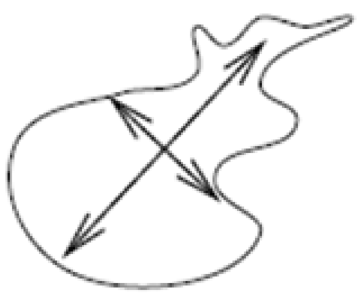
$$\boldsymbol{Diam(B) = max_{i,j}(p_i, p_j)}$$

Boundary-Based Descriptors – (cont.)

(11.2.1) Simple Descriptors

- **Eccentricity**

The ratio of the length of the longest chord (major axis) of the shape to the longest chord perpendicular to it (minor axis).



Principal axes

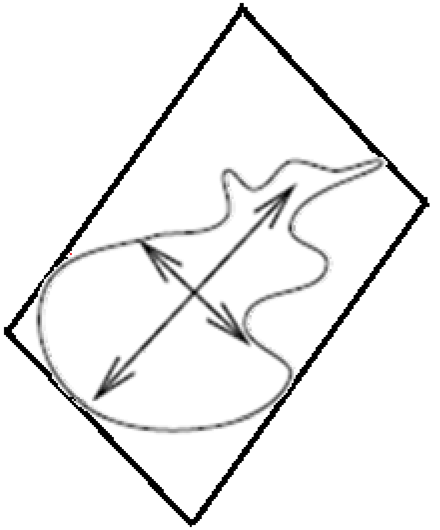
Boundary-Based Descriptors – (cont.)

(11.2.1) Simple Descriptors

- **Rectangularity**

How rectangular a shape is (how much it fills its minimal bounding box).

$$A_{object}/A_{box}$$



Boundary-Based Descriptors – (cont.)

(11.2.1) Simple Descriptors

- **Curvature**

Rate of change of slope.

Difference of slopes at segment intersections.

- **Convexity:**

Ratio of perimeter of convex hull

to perimeter of boundary: P_{Hull}/P.

Convexity

Boundary-Based Descriptors – (cont.)

(11.2.3) Fourier Descriptors

- Spectral features based on the Fourier transform of the boundary points.
- Sometimes the FT of the shape signature is used.

Boundary-Based Descriptors – (cont.)

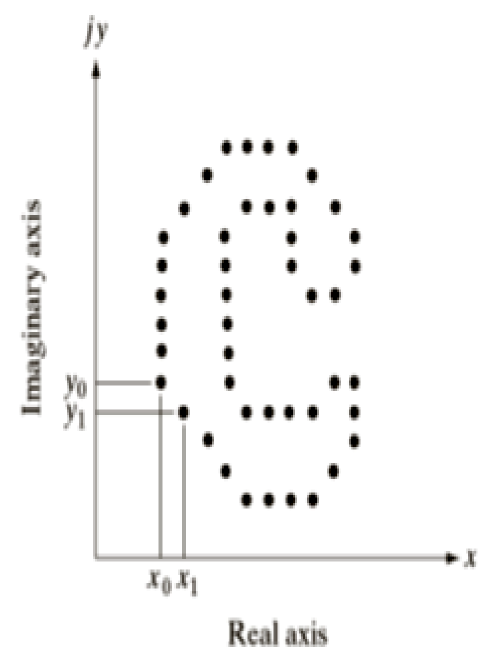
(11.2.3) Fourier Descriptors

- Consider a K-point boundary s in the xy-plane.
- s consists of

$$(x_o, y_o), (x_1, y_1), \ldots, (x_{K-1}, y_{K-1})$$

- Or we can write:

$$s(k) = x(k) + jy(k), k = 0, \ldots, K-1$$



Boundary-Based Descriptors – (cont.)

(11.2.3) Fourier Descriptors

- $DFT\{s(k)\} = a(u) = \sum_{k=0}^{K-1} s(k)e^{-j2\pi ku/K}$
  $u = 0, \dots, K-1$
- $a(u)$ are called *Fourier descriptors* of the boundary.
- s can be fully reconstructed from a

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

$k = 0, \dots, K-1$

Boundary-Based Descriptors – (cont.)

(11.2.3) Fourier Descriptors

- Describing a boundary can be done without ALL the Fourier coefficients.
- Global shape ← low frequencies
- Finer details ← high frequencies
- So we can use only a number of low frequency descriptors.

Boundary-Based Descriptors – (cont.)

(11.2.3) Fourier Descriptors

- Using first P coefficients, $s(k)$ approximated:

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

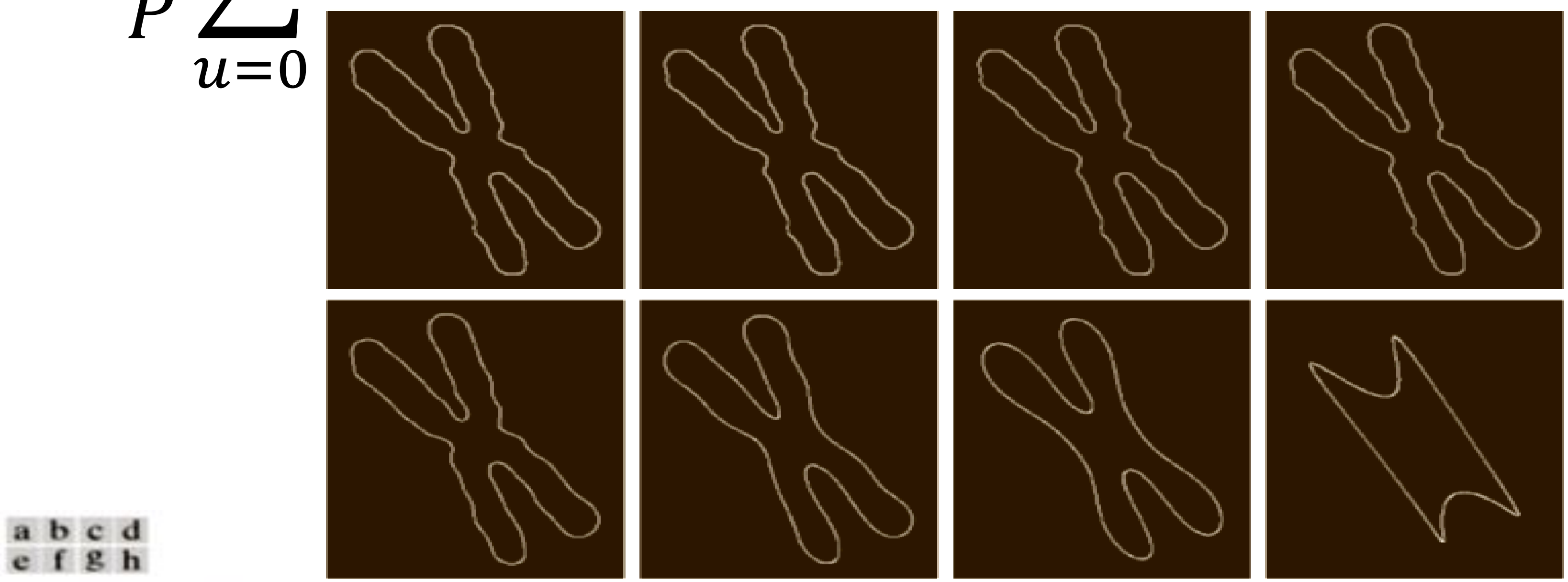


FIGURE 11.20 (a) Boundary of human chromosome (2868 points). (b)–(h) Boundaries reconstructed using 1434, 286, 144, 72, 36, 18, and 8 Fourier descriptors, respectively. These numbers are approximately 50%, 10%, 5%, 2.5%, 1.25%, 0.63%, and 0.28% of 2868, respectively.

Boundary-Based Descriptors – (cont.)

(11.2.3) Fourier Descriptors

- Most FD based work is dedicated to character recognition and object classification.
- Not directly invariant to transformations, but can be done simply.

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

http://fourier.eng.hmc.edu/e161/lectures/fd/node1.html

Boundary-Based Descriptors – (cont.)

(11.2.4) Statistical Moments

- The shape of a boundary segment (or its signature) can be described quantitatively using statistical moments, such as the mean, variance, and higher order moments.

- These describe expected value, variance (distribution around centroid), lopsidedness, skeweness, squatness, … etc.

Boundary-Based Descriptors – (cont.)

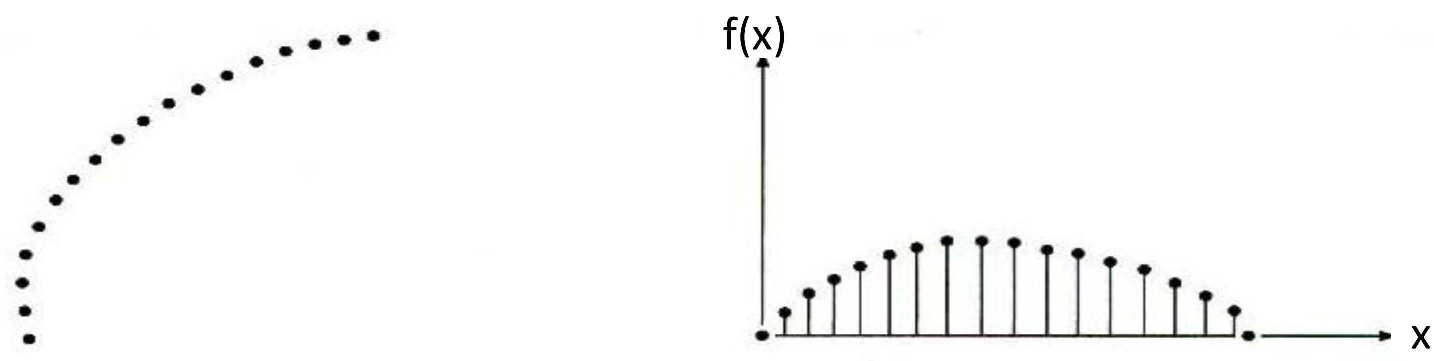
(11.2.4) Statistical Moments

- Nth moments about zero (raw moments)

$$m_n = \frac{\sum_{x=1}^{N} x^n f(x)}{\sum_{x=1}^{N} f(x)}$$

- First raw moment is mean.

$$m_1 = \mu = \frac{\sum_{x=1}^{N} x f(x)}{\sum_{x=1}^{N} f(x)}$$



a b

FIGURE 11.15
(a) Boundary segment.
(b) Representation as a 1-D function.

Boundary-Based Descriptors – (cont.)

(11.2.4) Statistical Moments

- Nth moments about mean (central moments)

$$\mu_n = \frac{\sum_{x=1}^{N}(x-\mu)^n f(x)}{\sum_{x=1}^{N} f(x)}$$

- **2nd central moment is variance**

$$\mu_2 = \sigma^2 = \frac{\sum_{x=1}^{N}(x-\mu)^2 f(x)}{\sum_{x=1}^{N} f(x)}$$

- **3rd central moment is skew.**

$$\mu_3 = \text{skew} = \frac{\sum_{x=1}^{N}(x-\mu)^3 f(x)}{\sum_{x=1}^{N} f(x)}$$

(measure of the lopsidedness of the distribution; any symmetric distribution will have a third central moment, if defined, of zero).

- **4th central moments is kurtosis.**

(measure of whether the distribution is tall and skinny or short and squat).

Boundary-Based Descriptors – (cont.)

(11.2.4) Statistical Moments

- If we have an infinite number of central moments can completely describe the shape of a function.
- Most popular and easy to implement.
- Invariant to rotation, and can be scaled for size normalization.
- It is difficult to associate physical interpretation to higher order moments.

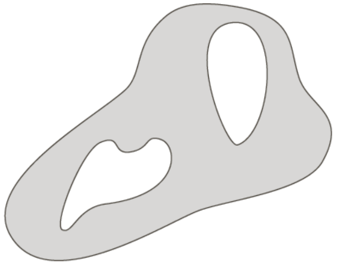
Representation – Textbook Outline

- **11.1 Representation**
  - 1. Boundary Following
  - 2. Chain Codes
  - 3. & 4. Polygon Approximation
  - 5. Signatures
  - 6. Boundary Segments
  - 7. Skeletons
- **11.2 Boundary Descriptors**
  - 1. Simple Descriptors
  - 2. Shape Number
  - 3. Fourier Descriptor
  - 4. Statistical Moments
- **11.3 Regional Descriptors**
  - 1. Simple Descriptors
  - 2. Topological
  - 3. Texture
  - 4. Moment Invariants
- **11.5 Relational Descriptors**
  - Strings, directed line segments, trees, …

Region-Based Descriptors – (cont.)

(11.3.1) Simple Descriptors

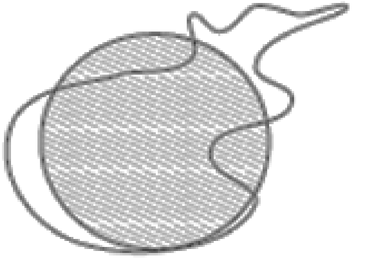
- **Area:**



Number of pixels in a region.

- **Compactness:**

How closely-packed the shape is: $\boldsymbol{P}^2/\boldsymbol{A}$. The most compact shape is a circle (4π). All other shapes have a compactness larger than 4π.



Compactness

Region-Based Descriptors – (cont.)

(11.3.1) Simple Descriptors

- **Circularity Ratio**

How circular a shape is.

$$\boldsymbol{R_c = 4\pi A/P^2}$$

$$\boldsymbol{R_c(circle) = 1}$$

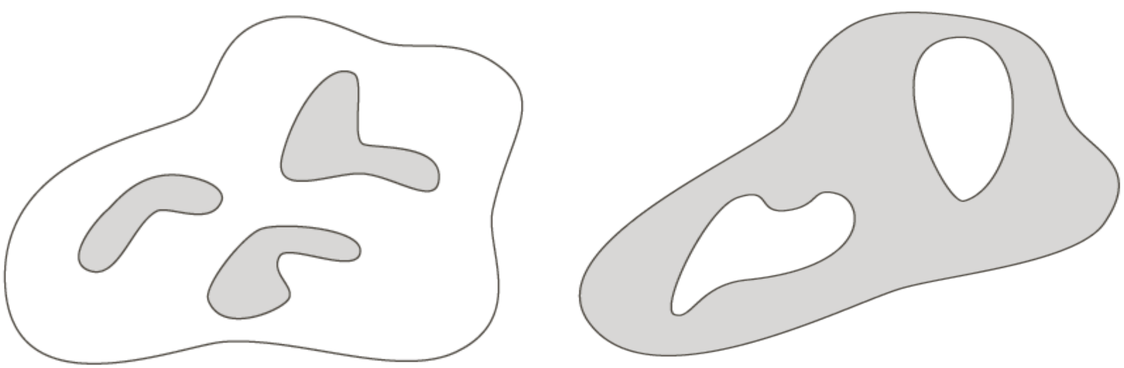
$$\boldsymbol{R_c(square) = 4\pi}$$

$\boldsymbol{R_c}$ is invariant to uniform scaling and orientation.

Region-Based Descriptors – (cont.)

(11.3.2) Topological Descriptors

- Topology: study of properties of a figure that are unaffected by any deformation (as long as there are no rubber sheet transforms).
- e.g. number of holes H, number of faces F, number of connected components C, ...



Region-Based Descriptors – (cont.)

(11.3.2) Topological Descriptors

- Euler Number $E = C - H$
- Euler formula $E = C - H = V - Q + F$

V: vertices = 7
Q: edges = 11
F: faces = 2

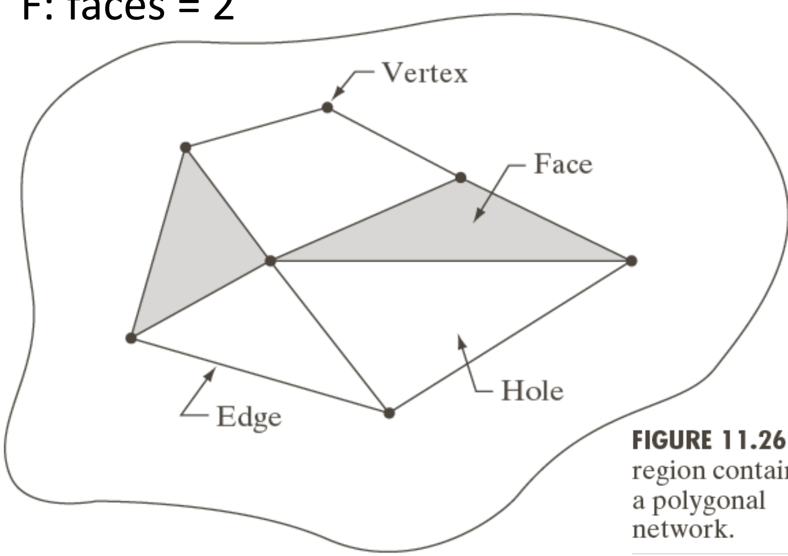
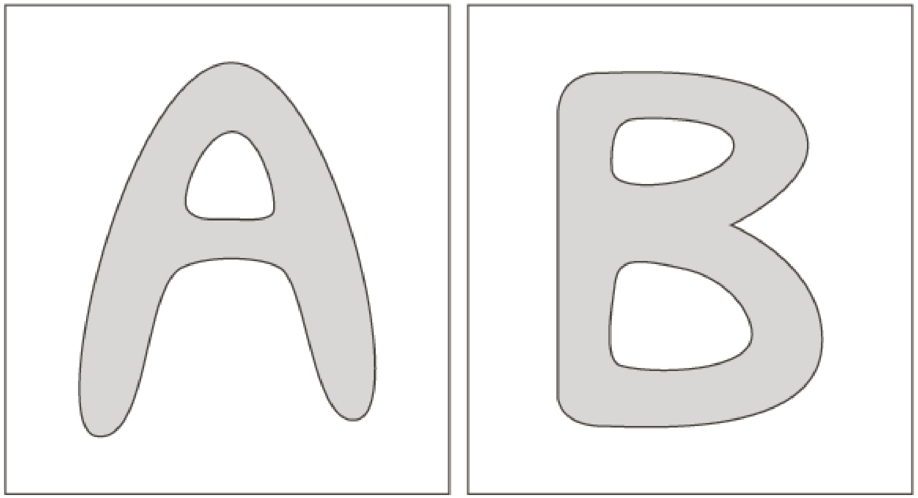


FIGURE 11.26 A region containing a polygonal network.

a b

FIGURE 11.25 Regions with Euler numbers equal to 0 and −1, respectively.



Region-Based Descriptors – (cont.)

(11.3.3) Texture

- “The notion of texture appears to depend upon three ingredients:

(1) some local ‘order’ is repeated over a region which is large in comparison to the order’s size.

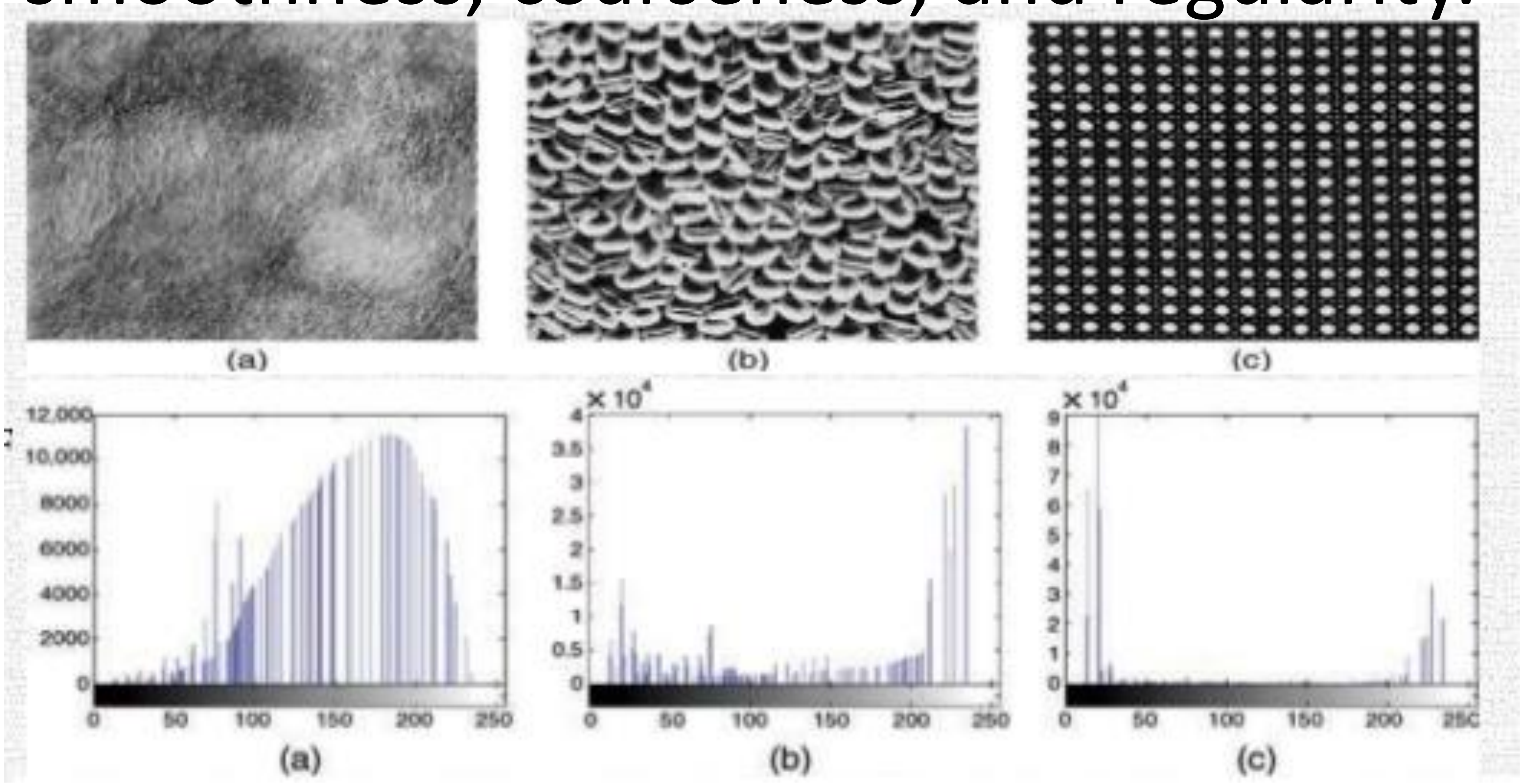
(2) the order consists in the nonrandom arrangement of elementary parts.

(3) the parts are roughly uniform entities having approximately the same dimensions everywhere within the textured region.” [12]

Region-Based Descriptors – (cont.)

(11.3.3) Texture

- Associated with image/object property such as smoothness, coarseness, and regularity.



Region-Based Descriptors – (cont.)

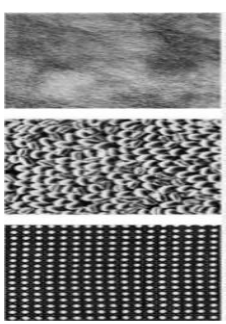
(11.3.3) Texture

- Three approaches to describe texture:
  - Structural: arrangements of image primitives such as description based on regular space parallel lines.
  - Statistical: most common and easy to implement. Characterize smooth, coarse, grainy, etc.
  - Spectral: detect global periodicity (Fourier spectrum and energy distribution).

Region-Based Descriptors – (cont.)

(11.3.3) Texture - Statistical

- Moments: mean, variance.
- Variance is used as a normalized measure of roughness: $R(z) = 1 - \frac{1}{1+\sigma^2(z)}$



Texture	Mean	Standard deviation	Roughness R	Skew	Uniformity	Entropy
Smooth	147.1459	47.9172	0.0341	−0.4999	0.0190	5.9223
Coarse	138.8249	81.1479	0.0920	−1.9095	0.0306	5.8405
Regular	79.9275	89.7844	0.1103	10.0278	0.1100	4.1181

Region-Based Descriptors – (cont.)

(11.3.4) Moments Invariants

- A set of functions commonly used to describe shape.
- A nonlinear combinations of the lower order moments → geometric moment.
- Variations: Algebraic moments, orthogonal moments, Legendre moments and Zernike moments.
- Invariant to translation, scaling and rotation.

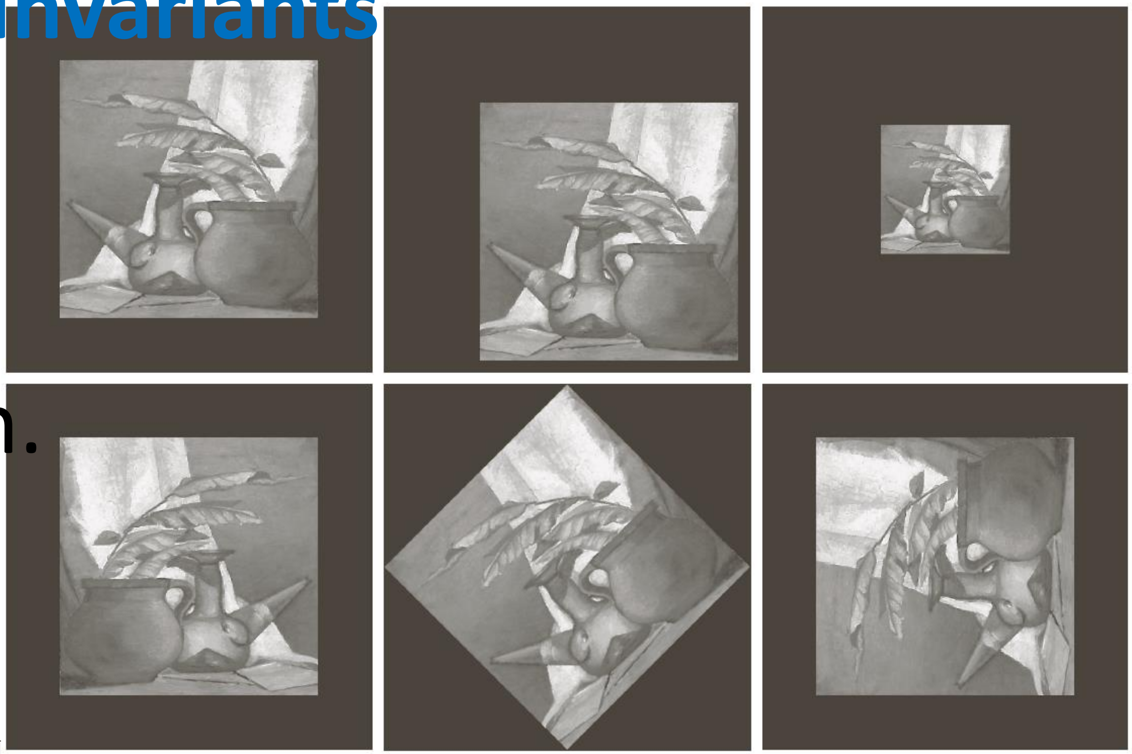
Region-Based Descriptors – (cont.)

(11.3.4) Moments Invariants

- Invariant

 to translation,

scaling and rotation.



Moment Invariant	Original Image	Translated	Half Size	Mirrored	Rotated 45°	Rotated 90°
ϕ_1	2.8662	2.8662	2.8664	2.8662	2.8661	2.8662
ϕ_2	7.1265	7.1265	7.1257	7.1265	7.1266	7.1265
ϕ_3	10.4109	10.4109	10.4047	10.4109	10.4115	10.4109
ϕ_4	10.3742	10.3742	10.3719	10.3742	10.3742	10.3742
ϕ_5	21.3674	21.3674	21.3924	21.3674	21.3663	21.3674
ϕ_6	13.9417	13.9417	13.9383	13.9417	13.9417	13.9417
ϕ_7	−20.7809	−20.7809	−20.7724	20.7809	−20.7813	−20.7809

More Advanced Features and Descriptors

- Matlab Computer Vision System Toolbox™.
- Detectors vs descriptors.
- SIFT, SURF, GIST, Haar, Hessian, HOG, LBP, CNN...

http://www.mathworks.com/help/vision/feature-detection-and-extraction.html
http://www.mathworks.com/help/vision/ug/local-feature-detection-and-extraction.html

Matching and Classification

- Euclidian distance
- Nearest neighbor
- Least squared difference
- ANN (MLP)
- SVM
- Statistical (Bayesian)

etc.

Next Lecture

Revision + MT and Quiz Compensation

Assignment

Check book sections and associated problems

Chapter 11	1.1, 1.2, 1.5, 2.1, 2.2, 2.3, 2.4, 3.1, 3.2
Associated problems	1, 2, 6, 7, 11, 12, 13, 14, 15, 16, 25, 26, 27

References

[1] Gonzalez and Woods, *Digital Image Processing*, 2008.

[2] *Image Processing, Analysis and Machine Vision*, 3rd edition, Milan Sonka, Vaclav Hlavac, Roger Boyle, Thomson, 2008.

[3] D. Zhang, and G. Lu, "*Review of Shape Representation and Description Techniques*", The Journal of Pattern Recognition, 2003.

[4] M. Peura, J. Iivarinen, Efficiency of simple shape descriptors, in: Proceedings of the Third International Workshop on Visual Form, Capri, Italy, May, 1997, pp. 443–451.

[5] A. Amanatiadis, V. G. Kaburlasos, A. Gasteratos, and S.E. Papadakis, "Evaluation of shape descriptors for shape-based image retrieval", 2010.

[6] S. Ben Jemaa, M. Hammami, and H. Ben-Abdallah, "*Biometric Identification Using a New Direction in Contactless Palmprint Imaging*", IPCV12, 2012.

[7]http://www.imageprocessingplace.com/downloads_V3/root_downloads/tutorials/contour_tracing_Abeer_George_Ghuneim/square.html

[8] http://www.slideshare.net/Jaddu44/image-feature-extraction

[9] http://www.mathworks.com/products/computer-vision/features.html

[10] http://www.mathworks.com/help/vision/ug/local-feature-detection-and-extraction.html

[11] http://en.wikipedia.org/wiki/Distance

[12] J. K. Hawkins, "Textural Properties for Pattern Recognition," in *Picture Processing and Psychopictorics*, B. C. Lipkin and A. Rosenfeld, Eds., Academic Press, New York, 1970, 347–370.

References

- Scott Krig, *Computer Vision Metrics*, Apress Open, 2014. (ch. 4, 6)
- J. Canny, A Computational Approach To Edge Detection, IEEE Trans. Pattern Analysis and Machine Intelligence, 8:679-714, 1986.
- Richard Szeliski, Computer Vision Algorithms and Applications, Springer 2010. (ch. 4)
- James Hayes presentations. http://www.cc.gatech.edu/~hays/compvision/