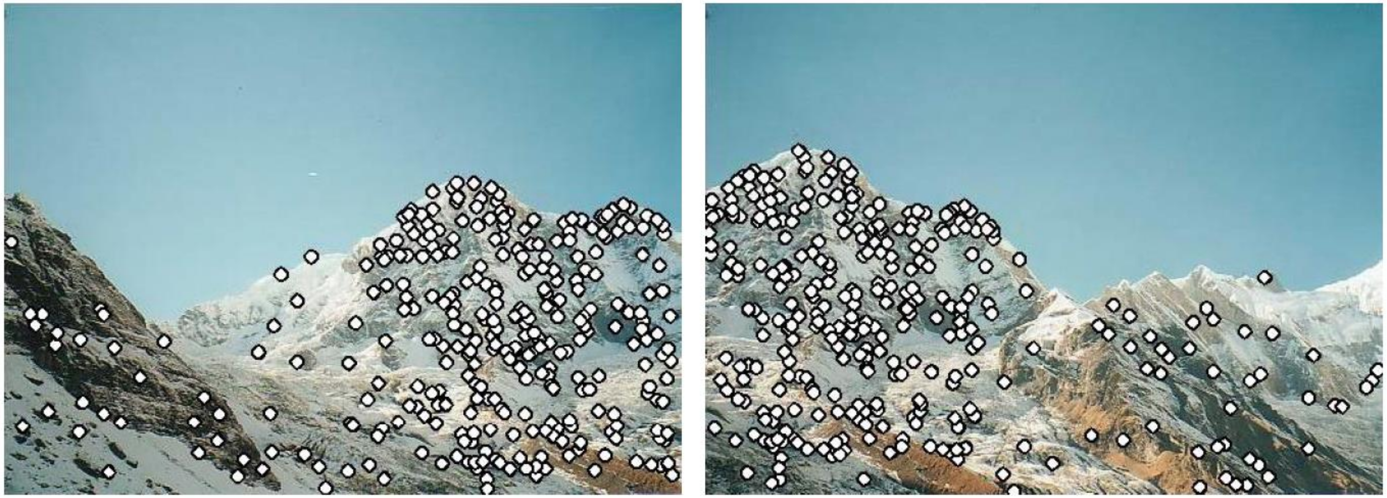


COMP9517: Computer Vision

Feature Representation

Part 2

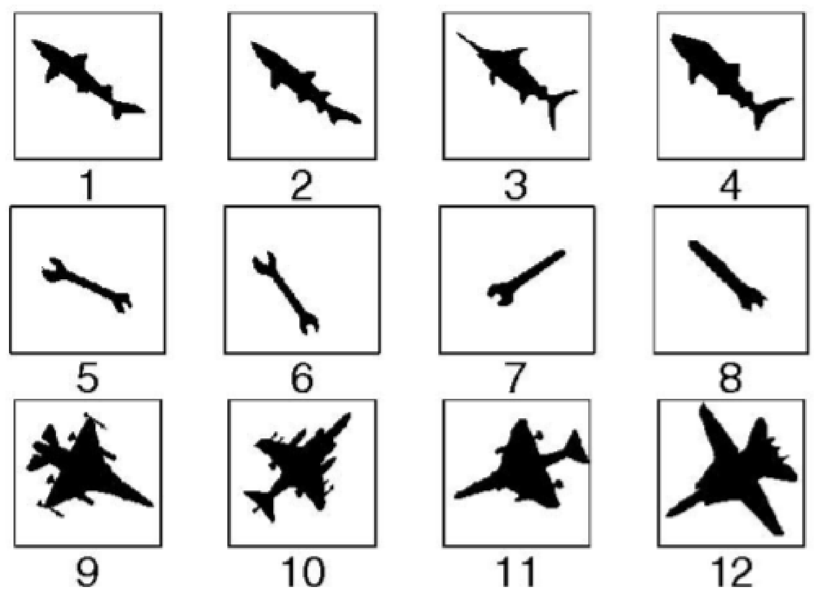
Feature Types

- Colour features (Part 1)
  - Colour moments
  - Colour histogram
- Texture features (Part 1)
  - Haralick texture features
  - Local binary patterns (LBP)
  - Scale-invariant feature transform (SIFT)
  - Texture feature encoding
- Shape features (Part 2)
  - Basic shape features
  - Shape context
  - Histogram of oriented gradients (HOG)

Shape Features

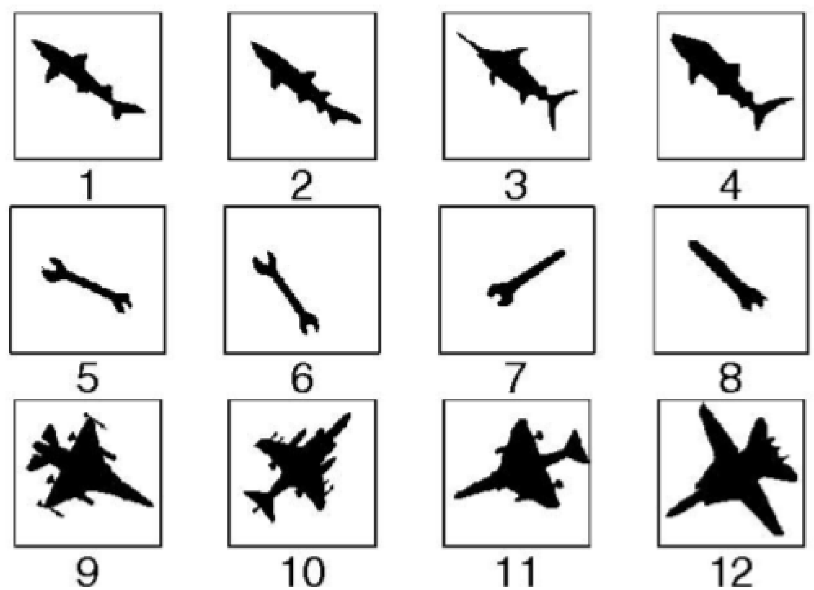
- **Shape** is an essential feature of material objects that can be used to identify and classify them

- Example: object recognition



Shape Features

- Human perception of an object or region involves capturing prominent / salient aspects of shape
- Shape features in an image are normally extracted after the image has been segmented into object regions

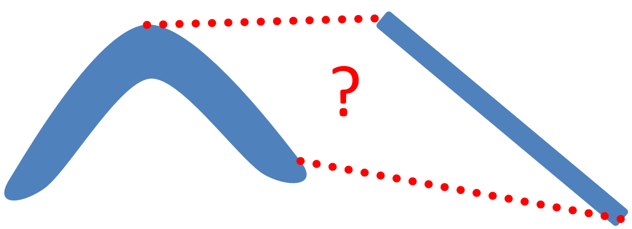


Shape Features

- Challenges
  - Invariance to rigid transformations

  - Tolerance to non-rigid deformations

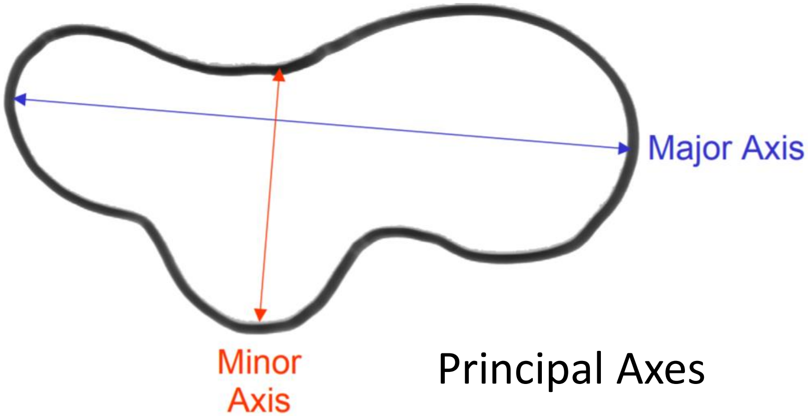
  - Correspondence unknown



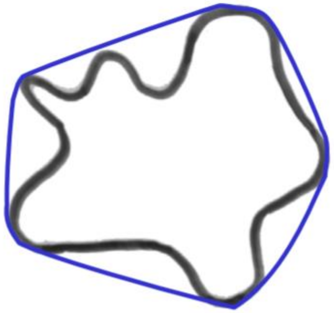
Basic Shape Features

- Simple geometrical shape descriptors

Net Area



Principal Axes



Convex Area:

Area of the convex hull that encloses the object

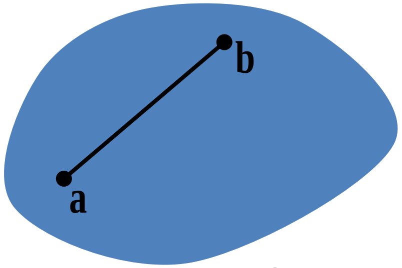
Basic Shape Features

- Convexity versus concavity of an object

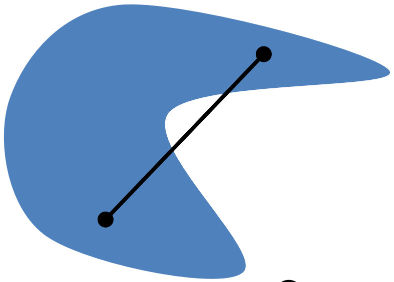
  An object O is called convex (or concave) if the straight line between any two points in the object is (or is not) contained in the object

$$\forall \mathbf{a}, \mathbf{b} \in O, \quad \forall\, 0 \le \alpha \le 1$$

$$(1-\alpha)\mathbf{a} + \alpha\mathbf{b} \in O$$



Convex



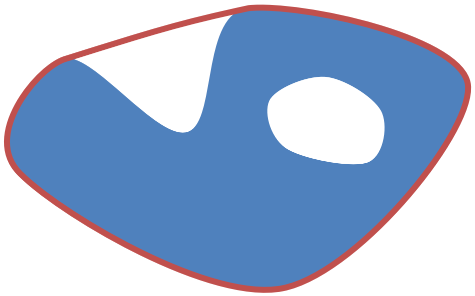
Concave

- Convex hull of an object

  The smallest convex set that contains the object

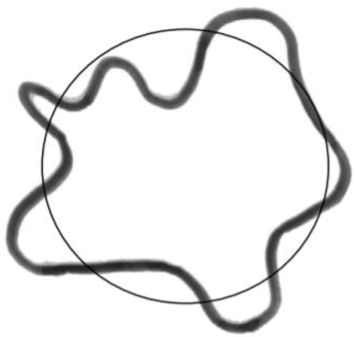
- Convex deficiency of an object

  Set difference between the convex hull and the object



Basic Shape Features

- Simple geometrical shape descriptors



Compactness:

Ratio of the area of an object to the area of a circle with the same perimeter

Circularity:

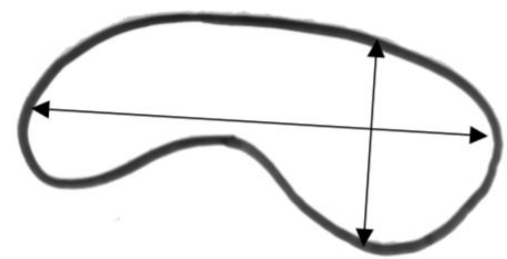
Ratio of 4π times the area of an object to the second power of its perimeter ($4\pi A/P^2$ equals 1 for a circle)

Basic Shape Features

- Simple geometrical shape descriptors

Elongation:

Ratio between the length and width of the object’s bounding box

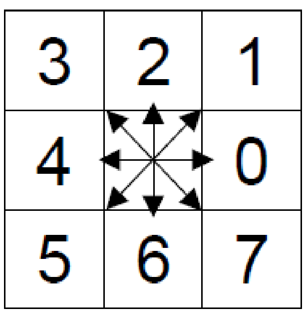


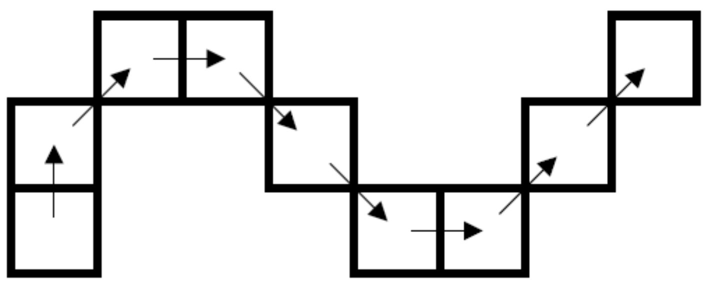
Eccentricity:

Ratio of the length of the minor axis to the length of the major axis

Boundary Descriptors

- Chain code descriptor
  - The shape of a region can be represented by labelling the relative position of consecutive points on its boundary
  - A chain code consists of a list of directions from a starting point and provides a compact boundary representation



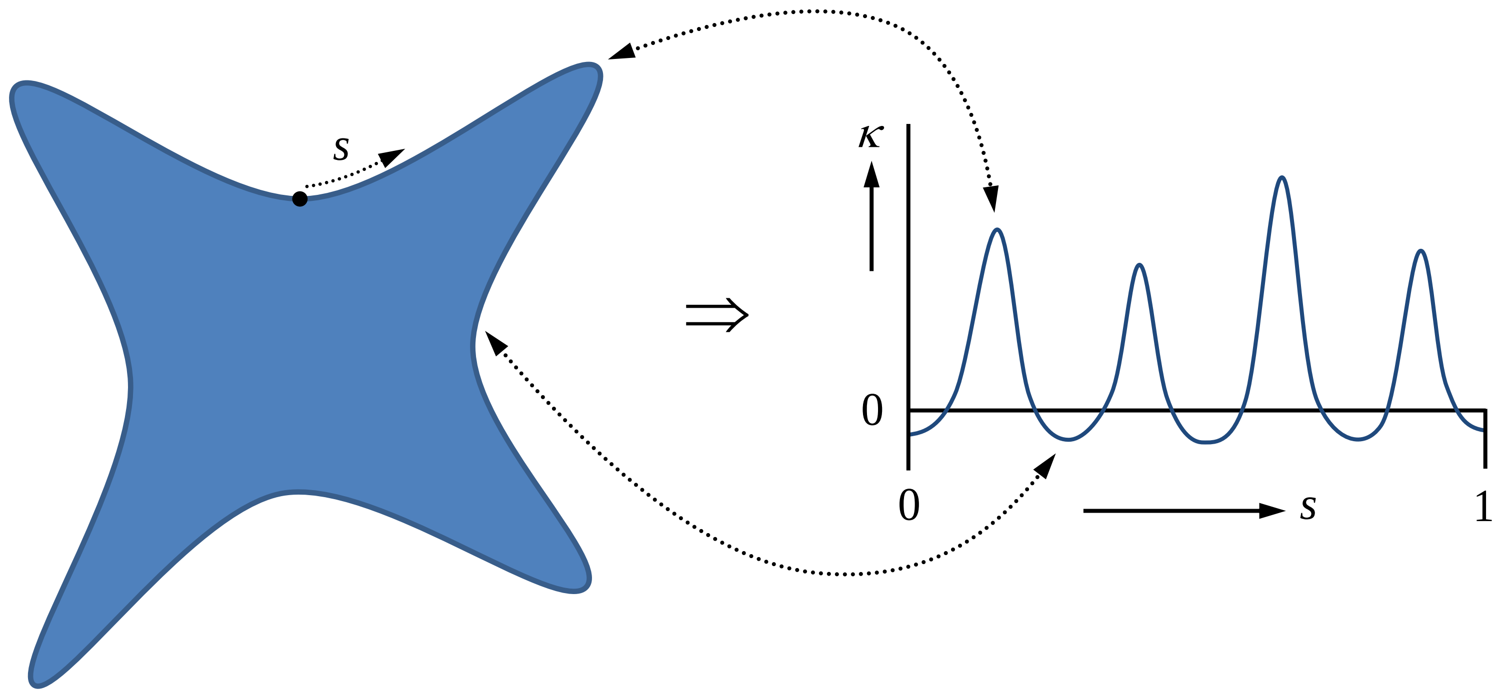


Example:

2,1,0,7,7,0,1,1

Boundary Descriptors

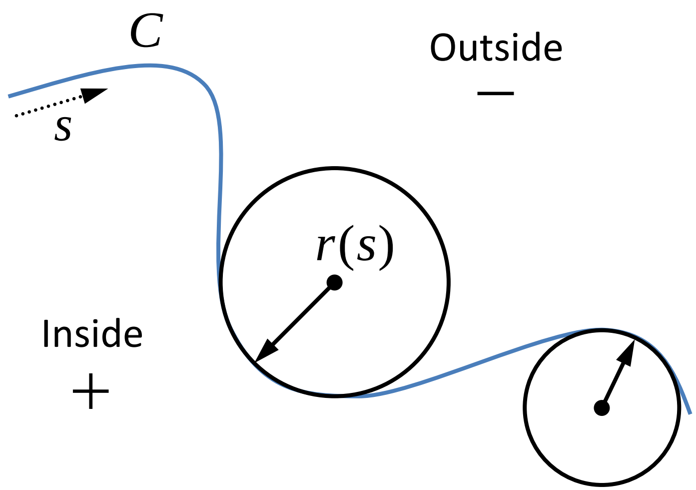
- Local curvature descriptor
  - The curvature of an object is a local shape attribute
  - Convex (versus concave) parts have positive (versus negative) curvature



Boundary Descriptors

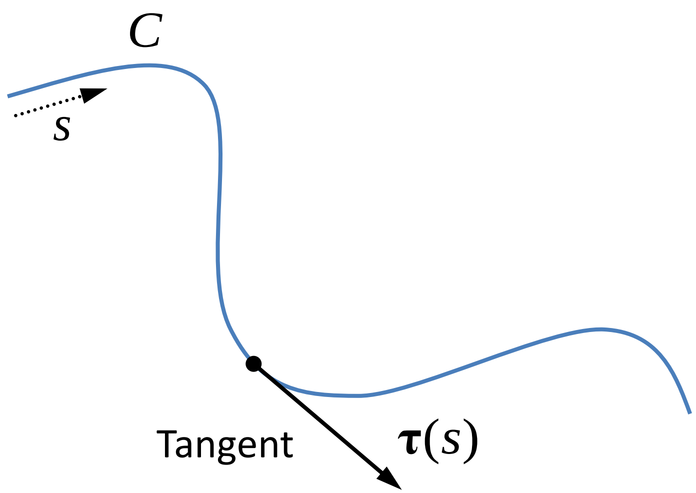
- Two interpretations of local curvature

Suppose the boundary is parameterized as $C:[0,1] \to \mathbf{R}^2 \Rightarrow C(s) = \left[x(s), y(s)\right]$



Geometrical interpretation

$$\kappa(s) = \pm \frac{1}{r(s)}$$



Physical interpretation

$$\kappa(s) = \pm \left\| \frac{d\boldsymbol{\tau}}{ds}(s) \right\|$$

Boundary Descriptors

- Global curvature descriptors
  - Total bending energy

$$B = \oint_C \kappa^2(s)ds$$

    - Amount of physical energy stored in a rod bent to the contour
    - Circular objects have the smallest contour bending energy $B = 2\pi/r$

  - Total absolute curvature

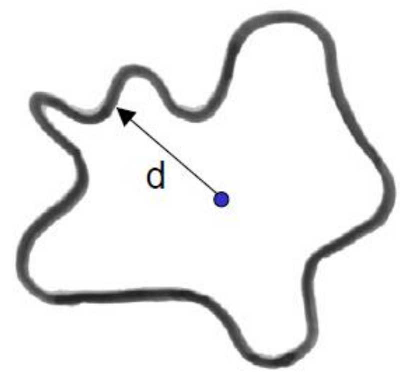
$$K = \oint_C |\kappa(s)|ds$$

    - Absolute value of the curvature integrated along the object contour
    - Convex objects have the smallest total absolute curvature $K = 2\pi$

Boundary Descriptors

- Radial distance descriptor
  - Use the centroid of the shape as the reference point and compute the radial distance for all N pixels along its boundary

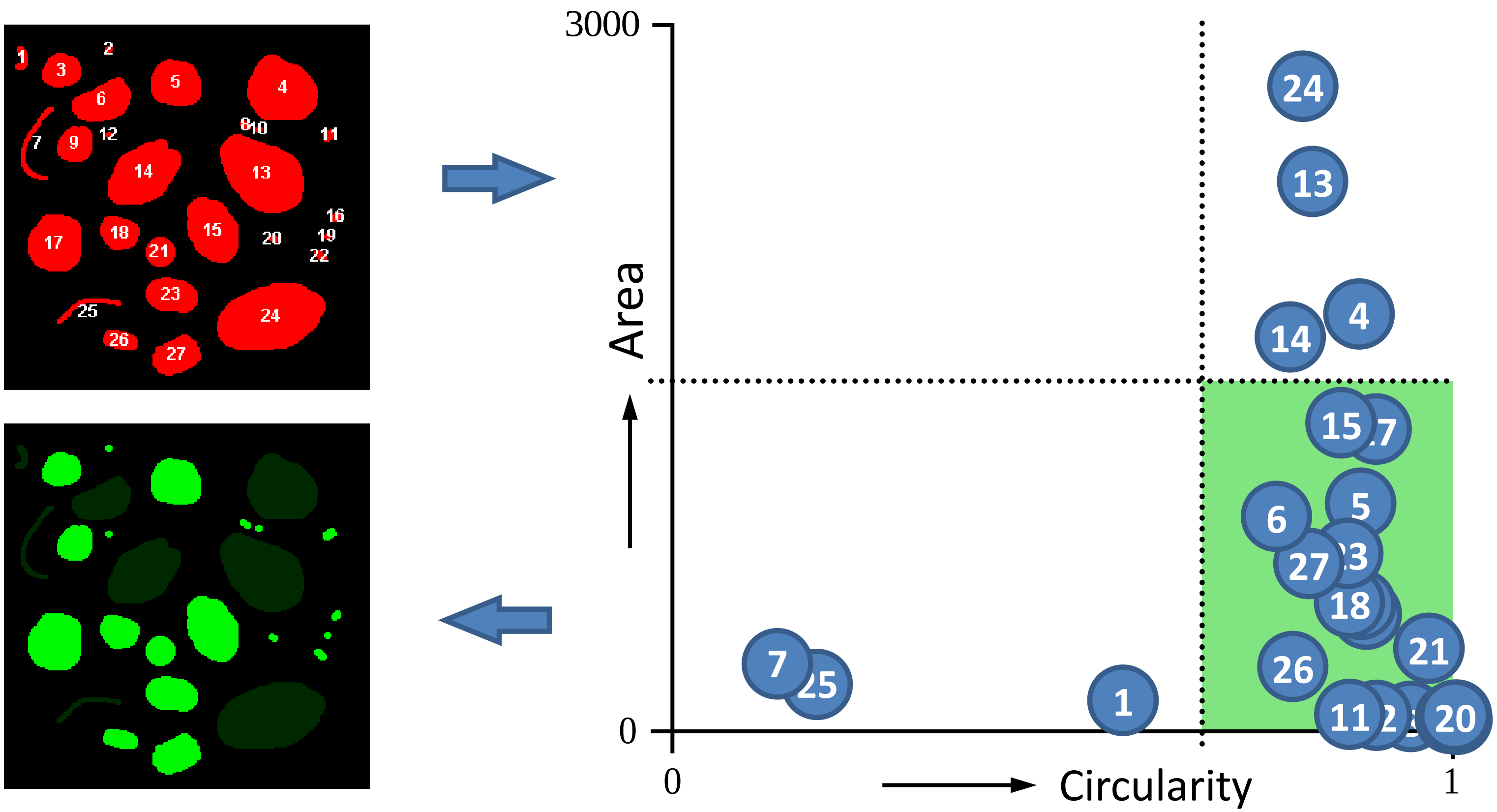
$$d(n) = \sqrt{\left(x(n) - \bar{x}\right)^2 + \left(y(n) - \bar{y}\right)^2}$$
$$\text{for } n = 0, 1, \ldots, N-1$$



  - Scale invariance is achieved by normalising $d(n)$ by the maximum distance to obtain the radial distance $r(n)$
  - The number of times the signal $r(n)$ crosses its mean can be used as a measure of boundary roughness

Application Example

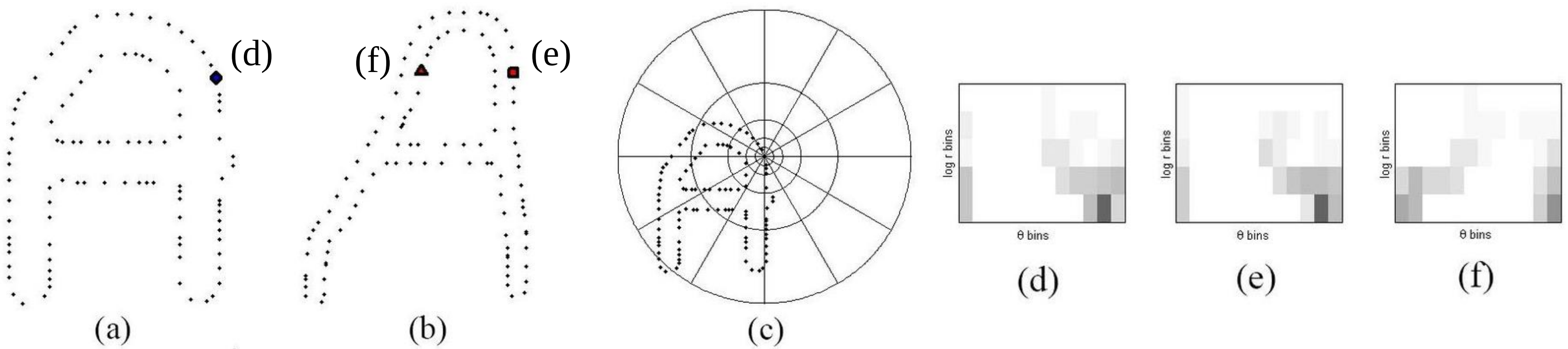
- Combining feature descriptors to classify objects



Shape Context

- **Shape context** is a point-wise local feature descriptor
  - Pick n points on the contour of a shape
  - For each point p_i construct a histogram h_i of the relative coordinates of the other $n-1$ points => this is the shape context of p_i

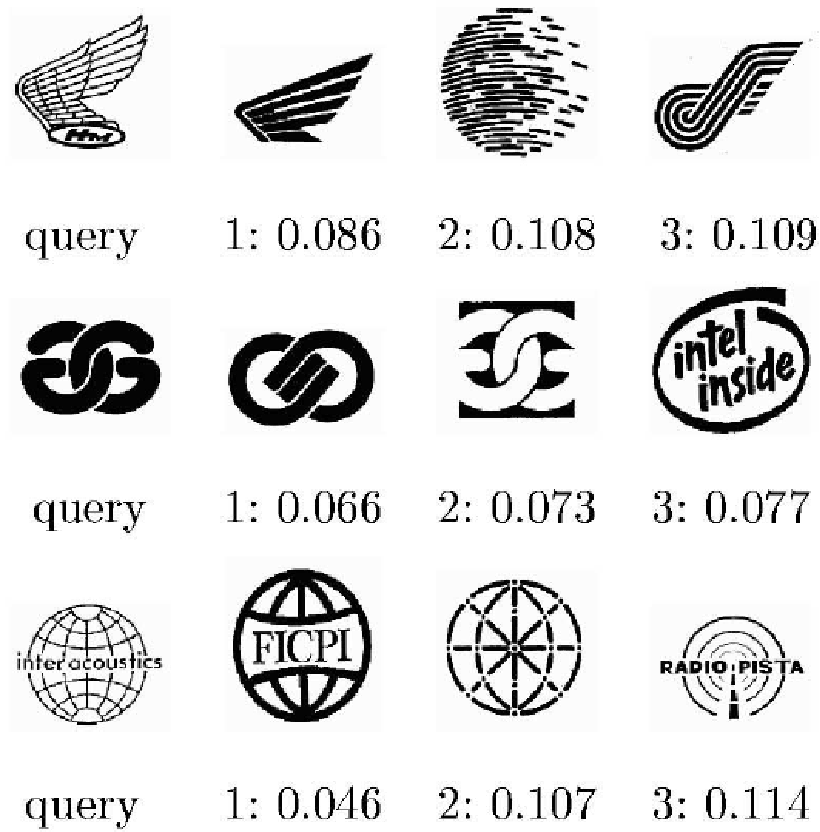
$$h_i(k) = \#\{q \neq p_i \ : \ (q - p_i) \in \text{bin}(k)\}$$



S. Belongie, J. Malik, J. Puzicha, "Shape matching and object recognition using shape contexts," IEEE Transactions on Pattern Analysis and Machine Intelligence 24(4):509-522. https://doi.org/10.1109/34.993558

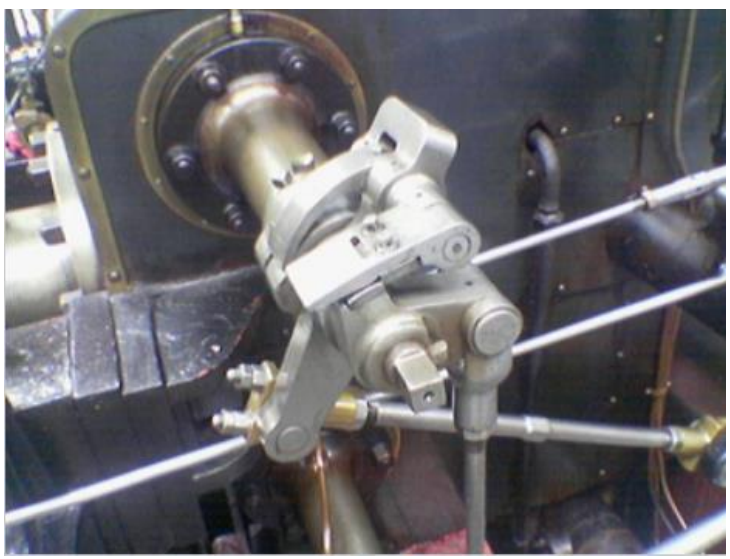
Application Example

- Shape matching

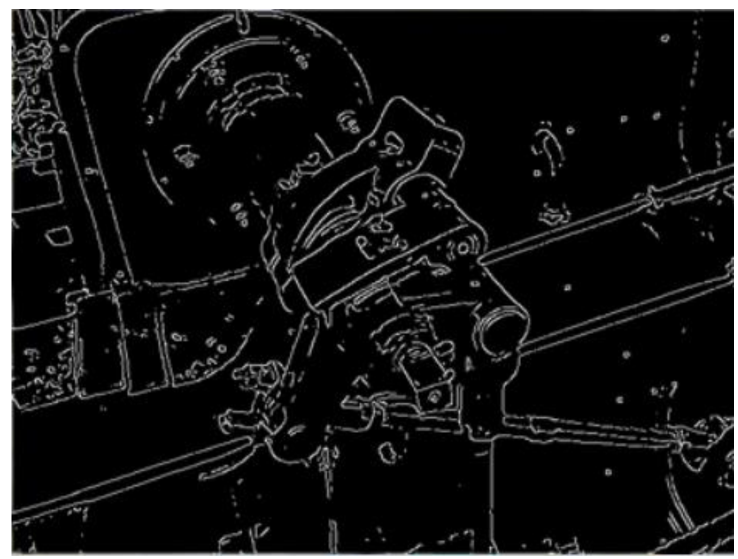


Shape Context

- Shape matching
  - Step 1: Sample a list of points on shape edges
    For example from Canny edge detector (Gaussian filtering, intensity gradient, non-maximum suppression, hysteresis thresholding, edge tracking)



$\Rightarrow$



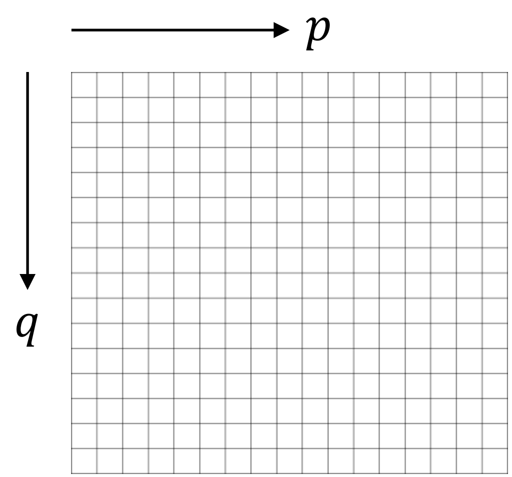
Shape Context

- Shape matching
  - Step 2: Compute the shape context for each point

$$h_i(k) = \#\{q \neq p_i \;:\; (q - p_i) \in \text{bin}(k)\}$$

  - Step 3: Compute the cost matrix between two shapes P and Q
    The cost between any two points $p \in P$ and $q \in Q$ with corresponding shape contexts g and h is defined as

$$C_S = \frac{1}{2} \sum_{k=1}^{K} \frac{[g(k) - h(k)]^2}{g(k) + h(k)} \quad \Rightarrow$$



Shape Context

- Shape matching
  - Step 4: Find the one-to-one matching that minimises the total cost between pairs of points on the two shapes

$$H(\pi) = \sum_i C\left(p_i, q_{\pi(i)}\right)$$

  - Step 5: Transform or deform one shape to the other based on the previous one-to-one point matching
    - Choose the desired transformation (for example affine)
    - Apply least-squares or RANSAC fitting

Shape Context

- Shape matching
  - Step 6: Compute the shape distance

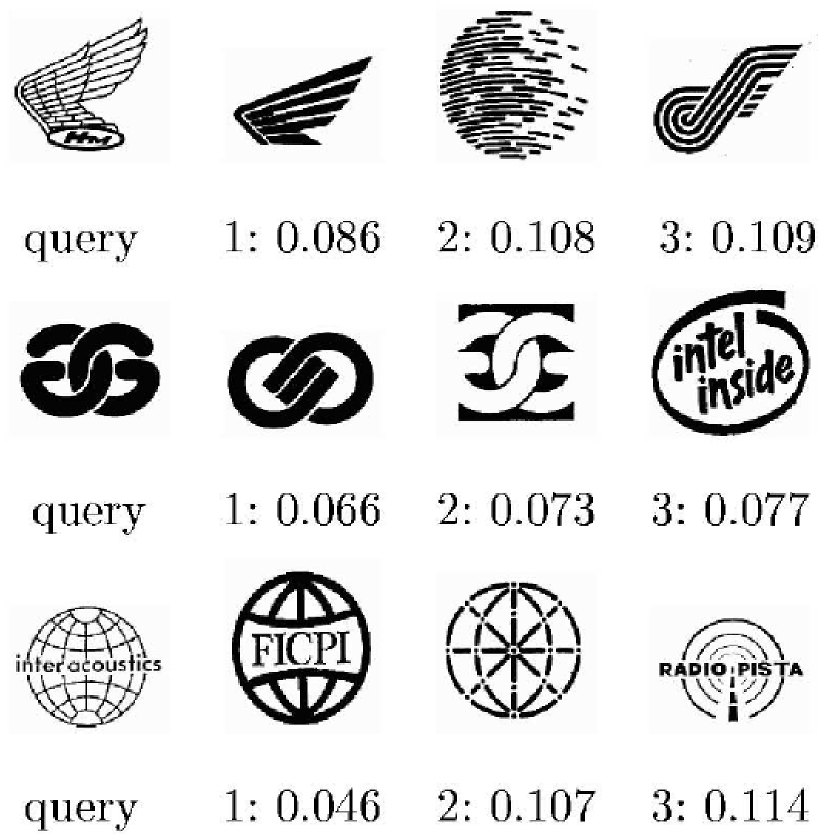
$$D_{sc}(P,Q) = \frac{1}{n}\sum_{p\in P}\min_{q\in Q} C(p,T(q)) + \frac{1}{m}\sum_{q\in Q}\min_{p\in P} C(p,T(q))$$

Other costs may also be taken into consideration

- Appearance of the image at the points
- Bending energy of the transformation

Application Example

- Shape matching



1) Sample points

2) Compute shape context

3) Compute cost matrix

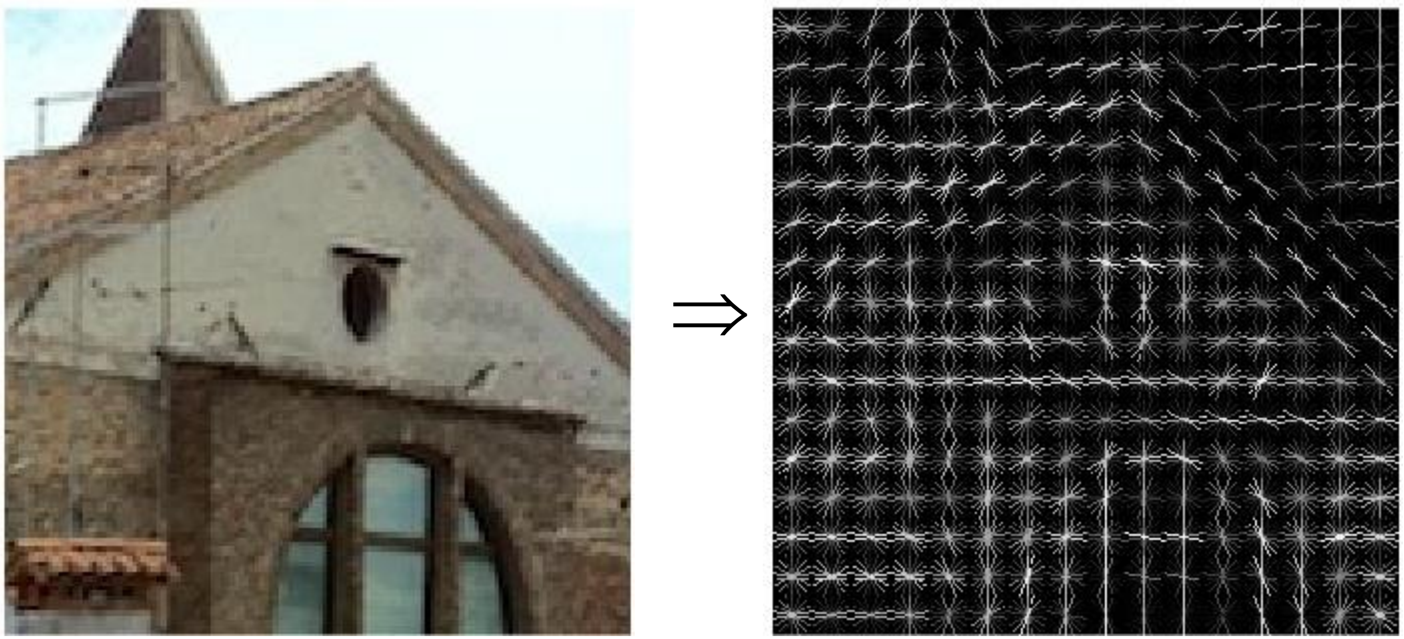
4) Find point matching

5) Perform transformation

6) Compute distance

Histogram of Oriented Gradients

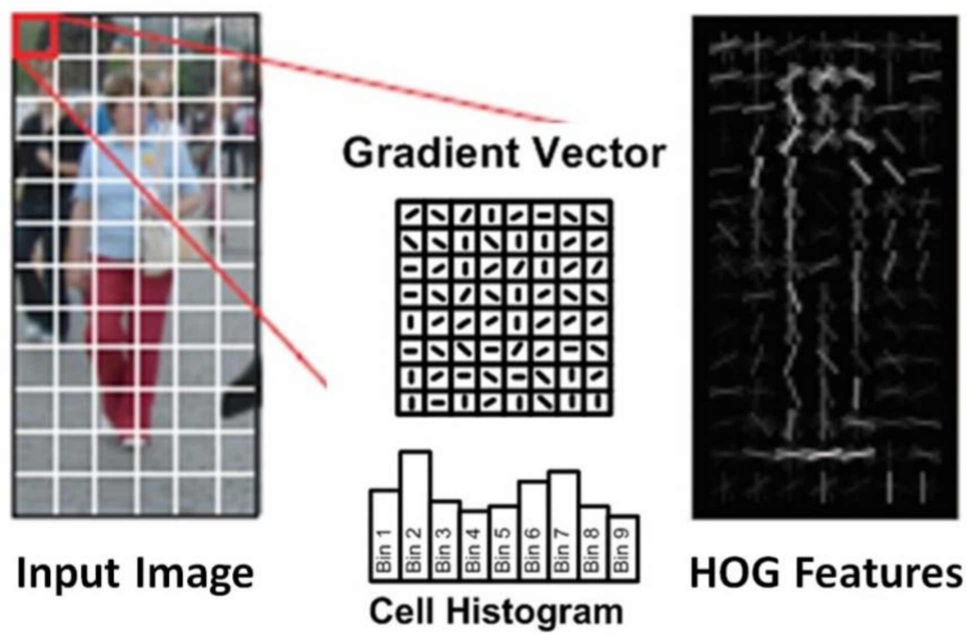
- HOG describes the distributions of gradient orientations in localized areas and does not require initial segmentation



N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” Computer Vision and Pattern Recognition 2005. https://doi.org/10.1109/CVPR.2005.177

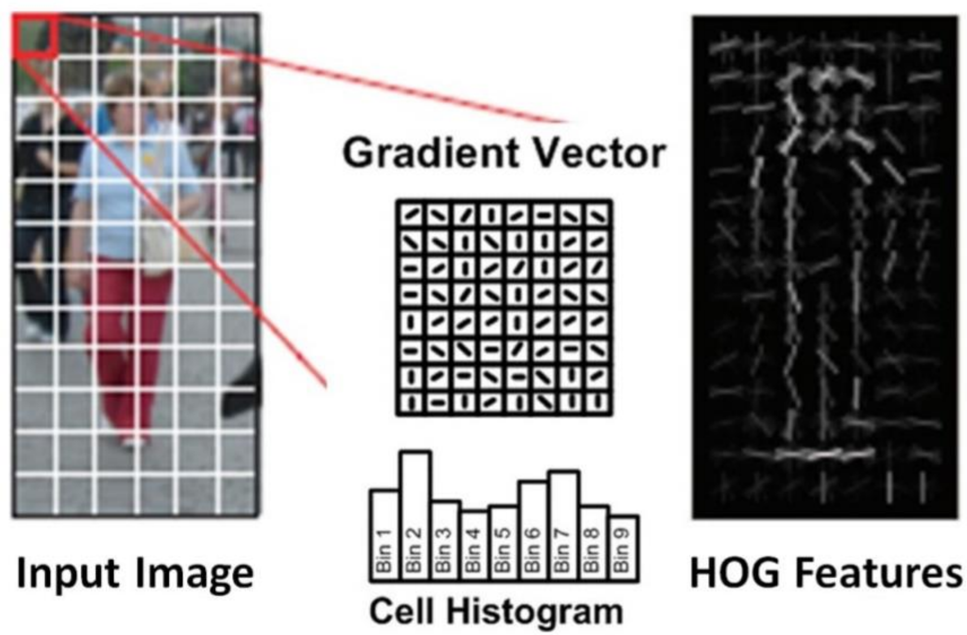
Histogram of Oriented Gradients

- Step 1: Calculate gradient magnitude and orientation at each pixel with a gradient operator => gradient vector



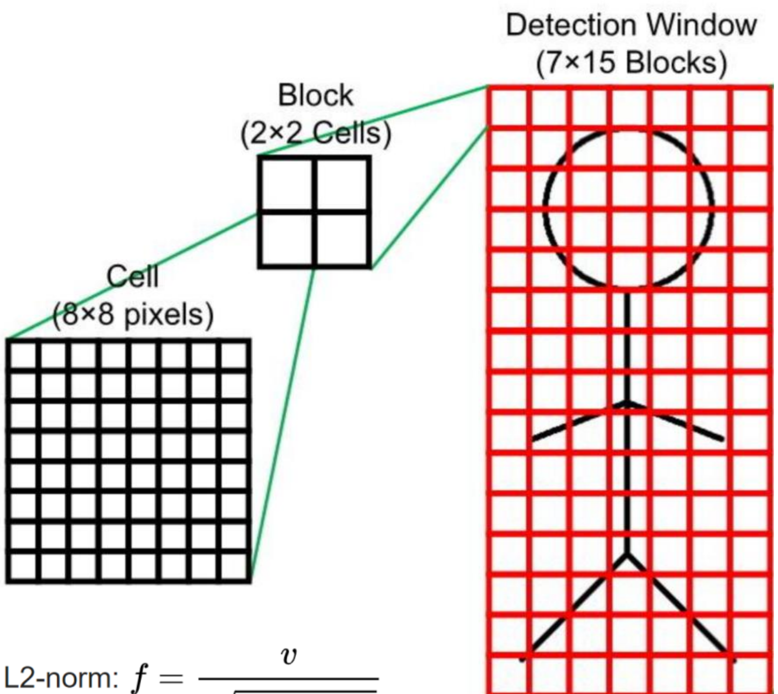
Histogram of Oriented Gradients

- Step 2: Divide orientations into N bins and assign the gradient magnitude of each pixel to the bin corresponding to its orientation => cell histogram
  - For example 9 bins evenly divided from 0 to 180 degrees



Histogram of Oriented Gradients

- Step 3: Concatenate and block-normalise cell histograms to generate detection-window level HOG descriptor



L2-norm: $f = \dfrac{v}{\sqrt{\|v\|_2^2 + e^2}}$

orientations/cell

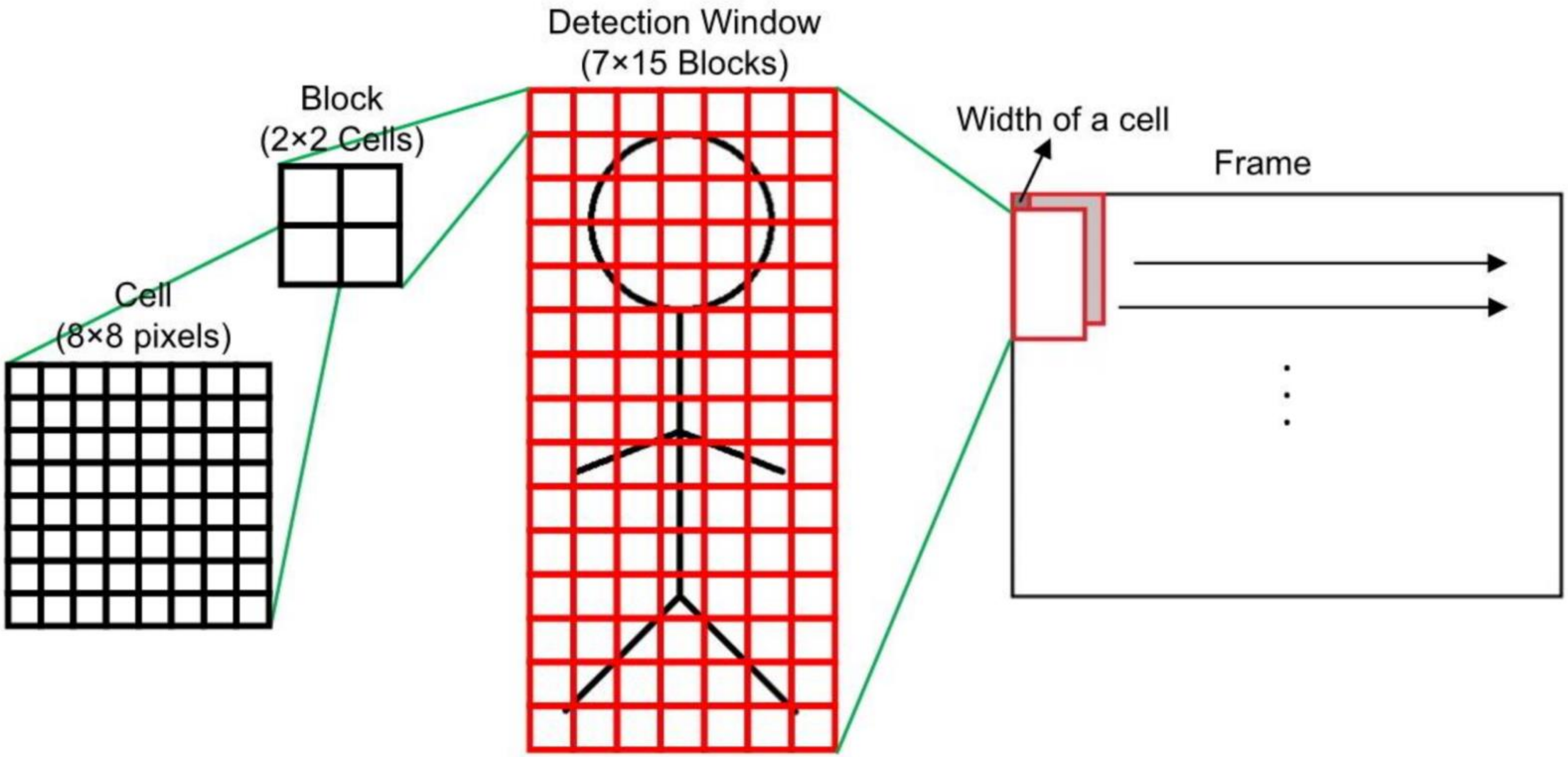
\# features = (7 x 15) x 9 x 4 = 3780

\# blocks

\# cells/block

Histogram of Oriented Gradients

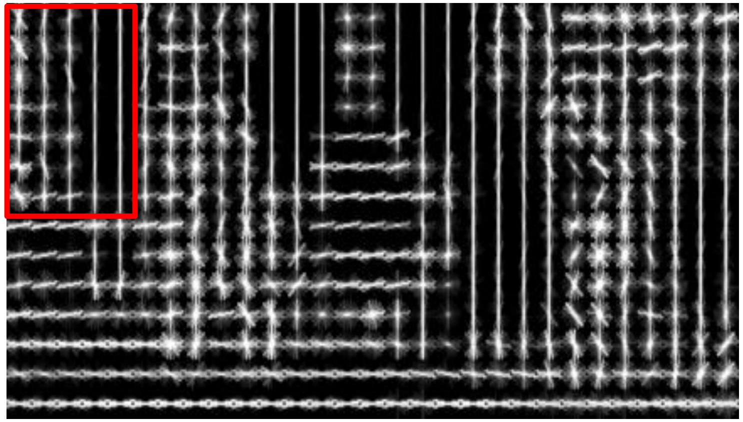
- Detection via sliding window on the image



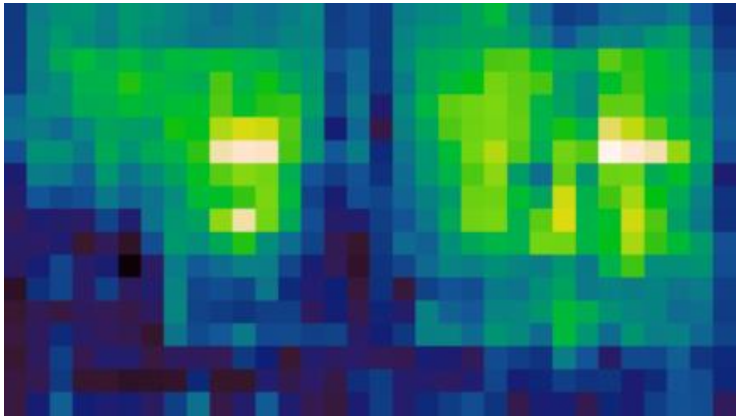
Histogram of Oriented Gradients

- Detection via sliding window on the image

HOG feature map

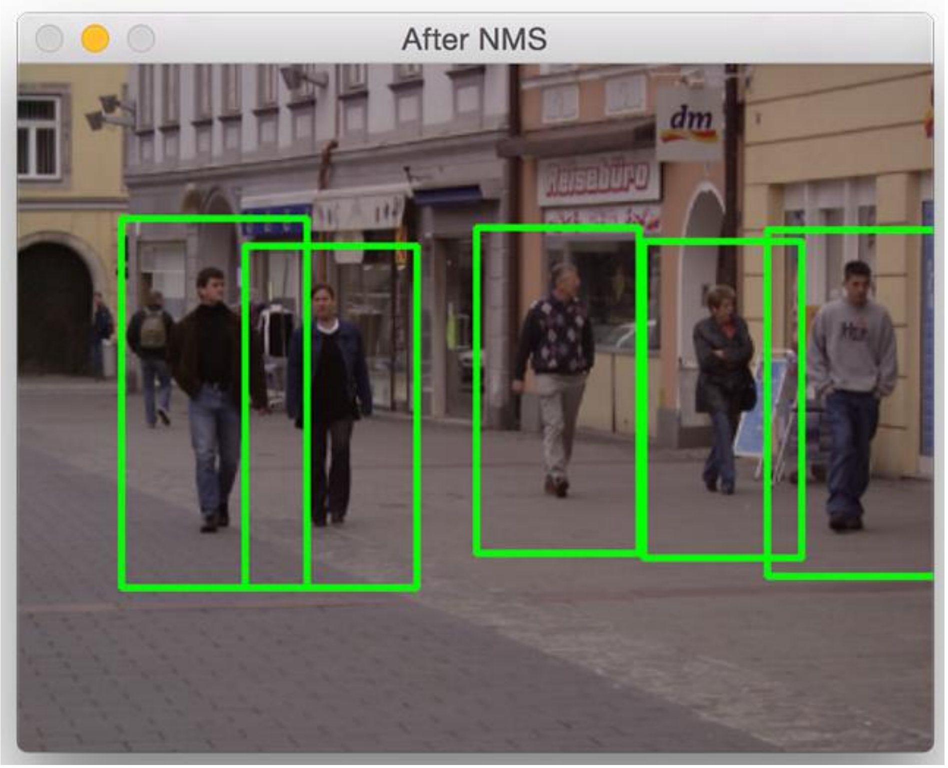


Detector response map



Application Example

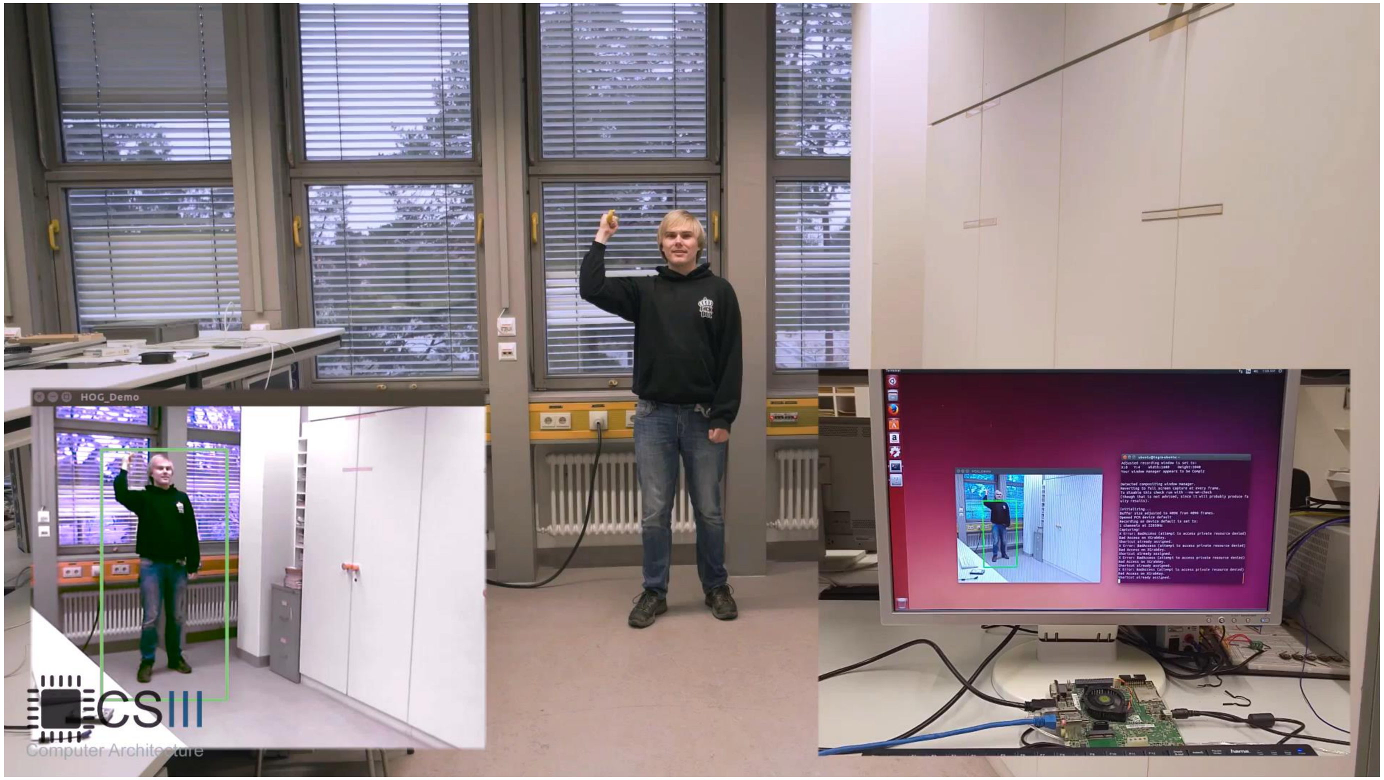
- Human detection



https://www.pyimagesearch.com/2015/11/09/pedestrian-detection-opencv/

Application Example

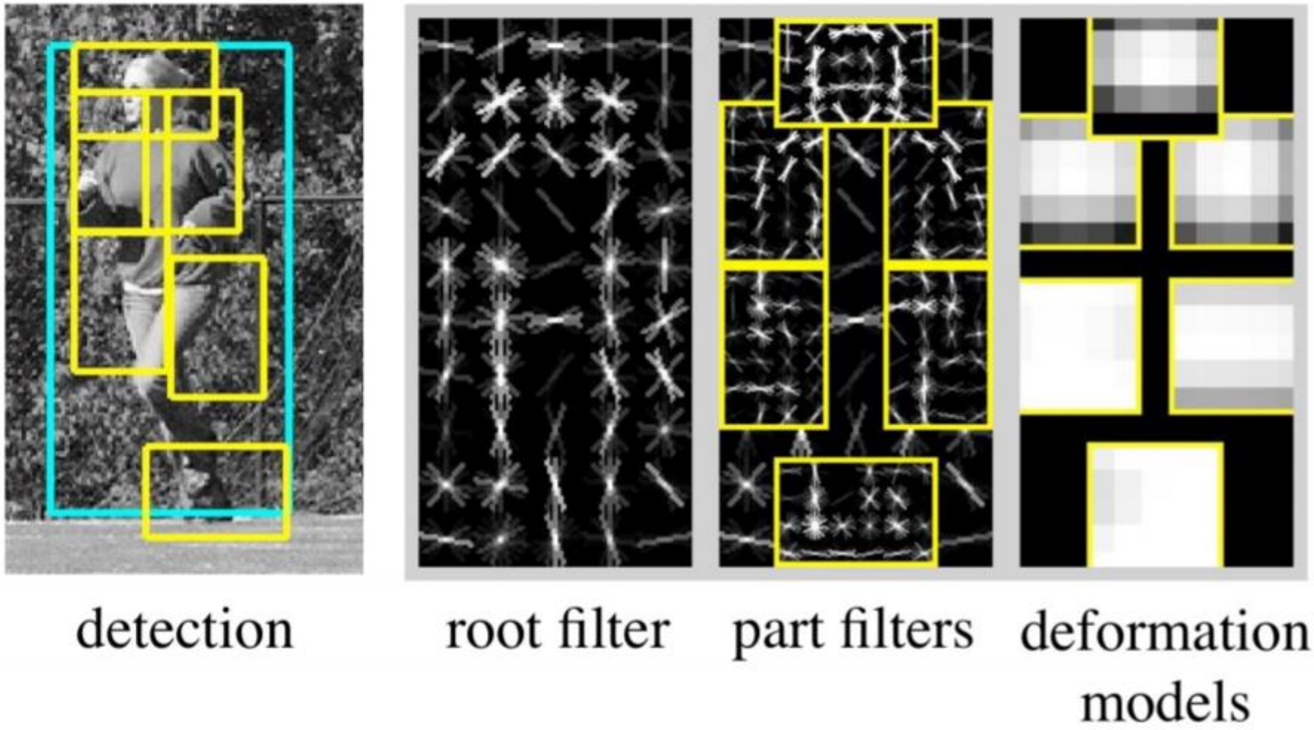
- Human detection



https://www.youtube.com/watch?v=0hMMRlB9DUc

Application Example

- Deformable part model



P. Felzenszwalb, D. McAllester, D. Ramanan, “A discriminatively trained, multiscale, deformable part model,” Computer Vision and Pattern Recognition 2008. https://doi.org/10.1109/CVPR.2008.4587597

Summary

- Feature representation is essential in solving almost all types of computer vision problems
- Most commonly used image features:
  – Colour features (Part 1)
    - Colour moments and histogram
  – Texture features (Part 1)
    - Haralick, LBP, SIFT
  – Shape features (Part 2)
    - Basic, shape context, HOG

Summary

- Other techniques described
  - Descriptor matching
  - Feature encoding (Bag-of-Words)
  - k-means clustering
  - Alignment and RANSAC
  - Spatial transformations
  - Shape features
  - Shape matching
  - Sliding window detection

References and Acknowledgements

- Szeliski, Chapter 4 (in particular Sections 4.1.1 to 4.1.3 and 4.3.2), Chapter 6 (in particular Sections 6.1.1 to 6.1.4)
- Some content are extracted from the above resource, James Hays slides, and slides from Michael A. Wirth
- L. Liu et al., From BoW to CNN: two decades of texture representation for texture classification, International Journal of Computer Vision, 2019
- And other resources as indicated by the hyperlinks