

UNIT 5

Image Segmentation and Feature Extraction

(10 Hours) — Image Processing (CACS404)

⚠ Note: This may contain errors — please verify

These notes were generated with the help of AI and **may contain errors** (wrong formulas, numerical slips, missing details or syllabus mismatches). Please cross-check every important definition, equation and worked example with your prescribed textbook, class notes and lecturer before relying on them for exams or practicals.

5.1 Introduction to Image Segmentation

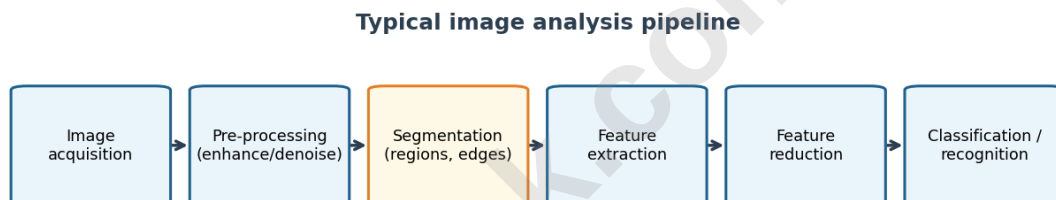


Figure 5.1 — Where segmentation and feature extraction fit in image analysis

Image segmentation subdivides an image into its constituent regions or objects. It is the first step of image analysis; the level of subdivision depends on the problem (stop when the objects of interest are isolated). Formally, segmentation partitions the image R into sub-regions $R_1, R_2, \dots, R_n$ such that:

- (a) $\cup R_i = R$ — every pixel belongs to some region (complete);
- (b) each R_i is a connected set;
- (c) $R_i \cap R_j = \emptyset$ for $i \neq j$ — regions are disjoint;
- (d) $P(R_i) = \text{TRUE}$ — pixels in a region satisfy a predicate (e.g., same gray level / texture);
- (e) $P(R_i \cup R_j) = \text{FALSE}$ for adjacent regions — different regions differ.

Two basic approaches: **discontinuity-based** (partition by abrupt intensity changes: edges, lines, points) and **similarity-based** (group pixels with similar properties: thresholding, region growing, region splitting and merging, clustering).

5.2 Thresholding

The simplest segmentation technique. If object and background pixels have distinct gray levels (a **bimodal histogram**), a threshold T separates them: $g(x, y) = 1$ if $f(x, y) > T$, else 0.

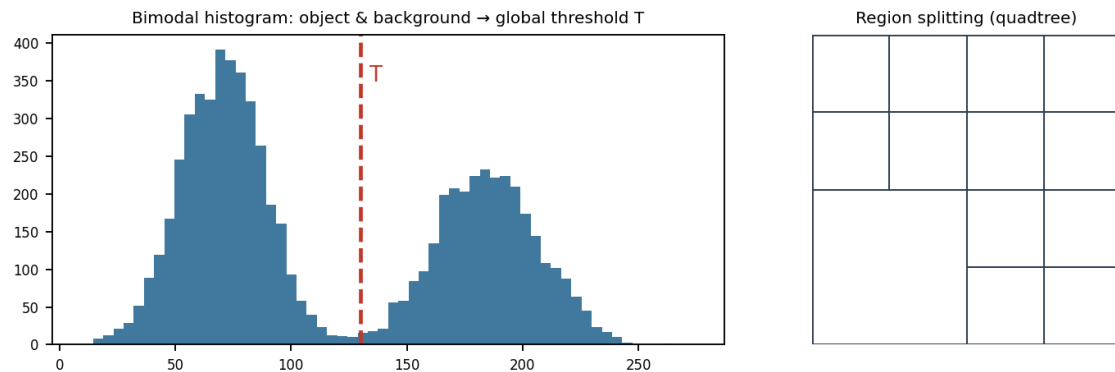


Figure 5.2 — (left) global thresholding on a bimodal histogram; (right) quadtree used in region splitting

- **Global thresholding:** one T for the whole image; works when illumination is uniform.
- **Variable / adaptive (local) thresholding:** T changes across the image — computed in sub-images, or from local mean/median; handles uneven illumination or shading.
- **Multiple thresholding:** two or more thresholds $T_1 < T_2$ separate more than two classes (multi-modal histograms).
- **Effect of noise:** noise fills the valley between histogram modes, so smooth the image first.

Basic Iterative Global Threshold Algorithm

(1) Choose an initial estimate T (e.g., the image mean). (2) Segment with T into G_1 (pixels $> T$) and G_2 (pixels $\leq T$). (3) Compute mean gray levels μ_1 and μ_2 of the two groups. (4) New $T = \frac{1}{2}(\mu_1 + \mu_2)$. (5) Repeat steps 2–4 until T changes by less than a small ΔT .

Example 5.1 — Iterative threshold

Find the threshold for pixel values {10, 20, 30, 40, 200, 220, 240} using the iterative algorithm.

Solution

$T_0 = \text{mean} = 760/7 = 108.6$. $G_1 = \{200, 220, 240\}$: $\mu_1 = 220$; $G_2 = \{10, 20, 30, 40\}$: $\mu_2 = 25$.

$T_1 = (220 + 25)/2 = 122.5$. Re-grouping with 122.5 gives the same two groups, so T has converged: **$T = 122.5$** .

Otsu's Method

Chooses T automatically to **maximise the between-class variance** $\sigma_B^2(T) = \omega_0(T) \cdot \omega_1(T) \cdot [\mu_0(T) - \mu_1(T)]^2$, where ω_0, ω_1 are the class probabilities (fractions of pixels $\leq T$ and $> T$) and μ_0, μ_1 the class means. It needs only the normalised histogram, is fully automatic and optimal for two well-separated classes, but works poorly when classes have very different sizes or heavy noise.

5.3 Region-Based Segmentation

5.3.1 Region Growing

Groups pixels into larger regions starting from **seed** points: (1) select seeds; (2) examine the neighbours (4- or 8-connected) of each region; append a neighbour if it satisfies the similarity criterion (e.g., $|f(\text{neighbour}) - f(\text{seed})| \leq T$ or a difference from the region mean); (3) stop when no more pixels can be added. Issues: choice of seeds, similarity criterion, stopping rule, and connectivity.

5.3.2 Region Splitting and Merging

Start with the whole image as one region. **Split**: if $P(R) = \text{FALSE}$, divide R into four quadrants (quadtree) and test each recursively. **Merge**: adjacent regions R_j, R_k are merged if $P(R_j \cup R_k) = \text{TRUE}$. Repeat until no more splits or merges are possible. It needs no seeds but square-block boundaries can look blocky.

Aspect	Thresholding	Region growing	Split and merge
Based on	Intensity histogram	Seed + similarity	Homogeneity predicate
Spatial info	Ignored	Used (connectivity)	Used
Parameters	T	Seeds, tolerance	Predicate, min. block size
Weakness	Noise, illumination	Seed sensitivity	Blocky borders

Watershed segmentation (introduction): treats the gradient magnitude as a topographic surface; flooding from regional minima builds dams (watershed lines) at ridges that separate basins (regions). Prone to over-segmentation, so marker-controlled watershed is used.

5.4 Edge, Line and Curve Detection

An **edge** is a set of connected pixels on the boundary between two regions where the intensity changes abruptly. Edges are detected by derivatives: the **first derivative** (gradient) has a peak at an edge (magnitude is large); the **second derivative** has a **zero-crossing** at the centre of the edge.

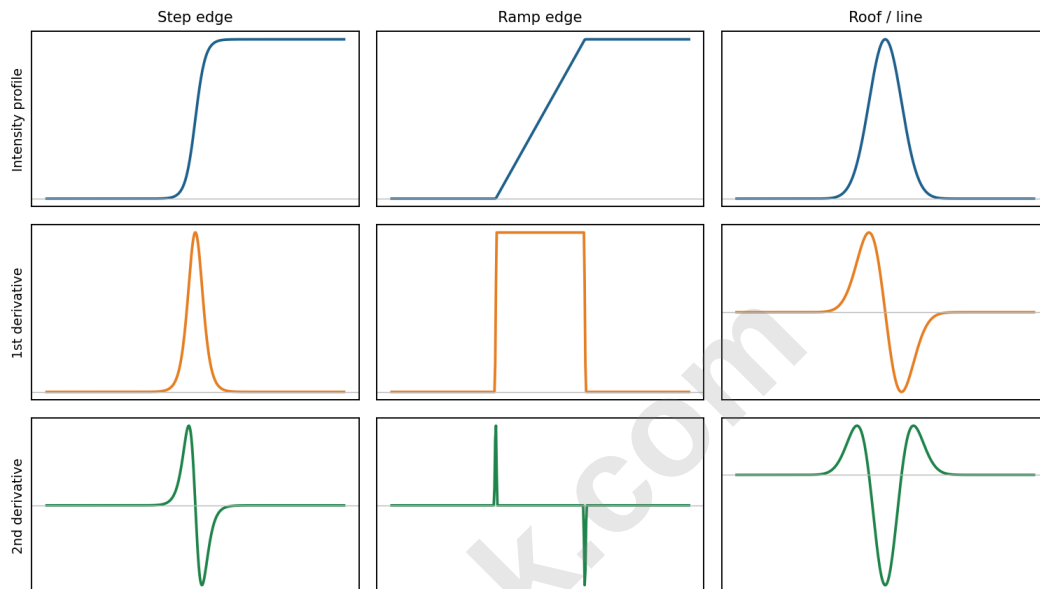


Figure 5.3 — Step, ramp and roof edge models with their first and second derivatives

Gradient

$\nabla f = [G_x, G_y]^T = [\partial f / \partial x, \partial f / \partial y]^T$. Magnitude: $M = \sqrt{G_x^2 + G_y^2} \approx |G_x| + |G_y|$. Direction: $\alpha = \tan^{-1}(G_y / G_x)$; the edge is perpendicular to α .

Edge Operators (Masks)

Operator	Mask for G_x	Mask for G_y	Notes
Roberts (2×2)	$[-1 \ 0; 0 \ 1]$	$[0 \ -1; 1 \ 0]$	Diagonal differences; simple, noise-sensitive, even-sized
Prewitt (3×3)	$[-1 \ -1 \ -1; 0 \ 0 \ 0; 1 \ 1 \ 1]$	$[-1 \ 0 \ 1; -1 \ 0 \ 1; -1 \ 0 \ 1]$	Averages over 3 pixels; some smoothing
Sobel (3×3)	$[-1 \ -2 \ -1; 0 \ 0 \ 0; 1 \ 2 \ 1]$	$[-1 \ 0 \ 1; -2 \ 0 \ 2; -1 \ 0 \ 1]$	Weights centre row/column by 2 → more smoothing; most popular

Example 5.2 — Sobel edge detection

Compute the Sobel gradient at the centre of the 3×3 region (rows): [10 10 50], [10 10 50], [10 10 50]. Use $G_x = (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)$ and $G_y = (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)$.

Solution

$$G_x = (10 + 2 \cdot 10 + 50) - (10 + 2 \cdot 10 + 50) = 80 - 80 = 0.$$

$$G_y = (50 + 2 \cdot 50 + 50) - (10 + 2 \cdot 10 + 10) = 200 - 40 = 160.$$

Magnitude $M = \sqrt{(0^2 + 160^2)} = 160$ (or $|0| + |160| = 160$); direction $\alpha = \tan^{-1}(160/0) = 90^\circ$, so the edge is perpendicular to this (a vertical edge between the 10-valued and 50-valued columns).

Laplacian and Laplacian of Gaussian (LoG)

The Laplacian $\nabla^2 f$ (Unit 2) is isotropic but very noise-sensitive and gives double edges. The **LoG** (Marr–Hildreth) first smooths with a Gaussian, then applies the Laplacian: $\nabla^2 G(x, y) = -[(x^2 + y^2 - 2\sigma^2)/\sigma^4] \cdot e^{-(x^2+y^2)/2\sigma^2}$. Edges are the **zero-crossings** of the result; σ sets the scale. Gives thin, closed contours.

Canny Edge Detector

Designed to meet three criteria: **good detection** (low false positives/negatives), **good localisation** and **single response** per edge. Steps:

- 1. Smooth the image with a Gaussian filter (removes noise).
- 2. Compute gradient magnitude and direction (Sobel).
- 3. **Non-maximum suppression**: keep a pixel only if its magnitude is a local maximum along the gradient direction $\rightarrow$ thin (1-pixel) edges.
- 4. **Hysteresis thresholding**: two thresholds $T_H > T_L$ (ratio $\approx 2:1$ or $3:1$). Pixels above T_H are strong edges; pixels between T_L and T_H are kept only if connected to a strong edge; others are discarded.

Edge Linking

Detected edge pixels are linked into boundaries by **local processing** (link neighbours with similar gradient magnitude and direction) or **global processing** (Hough transform).

Line and Curve Detection — Hough Transform

Detects parametric shapes by **voting** in a parameter (accumulator) space. A line $y = mx + b$ has unbounded slope, so the normal form is used: $\rho = x \cos \theta + y \sin \theta$ (ρ = perpendicular distance from the origin, θ = angle of the normal). Each edge point (x, y) maps to a **sinusoid** in (θ, ρ) space; collinear points give sinusoids that intersect at one point — a peak in the accumulator.

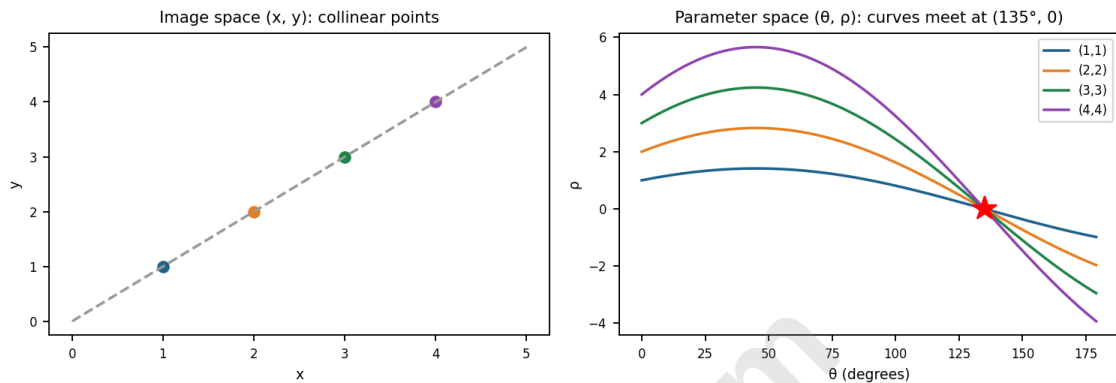


Figure 5.4 — Hough transform: collinear points in image space become curves crossing at one point in (θ, ρ) space

- **Algorithm:** (1) edge-detect the image; (2) quantize ρ and θ and initialise accumulator $A(\rho, \theta) = 0$; (3) for every edge pixel and every θ , compute ρ and increment $A(\rho, \theta)$; (4) local maxima of A above a threshold are the detected lines.
- **Circles:** $(x - a)^2 + (y - b)^2 = r^2$ has three parameters $(a, b, r) \rightarrow 3$ -D accumulator.
Generalised Hough transform detects arbitrary shapes using an R-table.
- Advantages: robust to gaps and noise. Drawback: memory and time grow with the number of parameters.

Example 5.3 — Hough transform

Points $(1,1)$, $(2,2)$, $(3,3)$ lie on $y = x$. Show that they give the same (ρ, θ) .

Solution

At $\theta = 135^\circ$: $\cos \theta = -0.707$, $\sin \theta = 0.707$. $\rho(1,1) = -0.707 + 0.707 = 0$; $\rho(2,2) = -1.414 + 1.414 = 0$; $\rho(3,3) = 0$. All three vote for the same cell ($\rho = 0$, $\theta = 135^\circ$), which is the line $y = x$ (normal at 135° , passing through the origin).

5.5 Image Features and Feature Extraction

A **feature** is a measurable property of a region or image used to describe it (shape, colour, texture, size). Features should be discriminating, invariant to translation, rotation and scale, and compact. **Feature extraction** converts the segmented image into a numeric **feature vector** $x = [x_1, x_2, \dots, x_n]^T$ for classification.

5.5.1 Boundary (Shape) Descriptors

- **Chain code:** represents the boundary as a sequence of direction numbers (4- or 8-directional) from a starting point. Normalised for starting point (circular shift to the smallest number) and rotation (use the **first difference**, counting direction changes counter-clockwise).
- **Signature:** 1-D function (distance from the centroid vs. angle) that represents the boundary.
- **Shape number:** the first-difference chain code of the smallest magnitude (a normalised form); its length is the *order*.
- **Fourier descriptors:** treat boundary points as complex numbers $s(k) = x(k) + jy(k)$; take the DFT; keep the first few coefficients — a compact, invariant description (low coefficients = overall shape).
- **Polygonal approximation, convex hull and skeleton (medial axis)** are other descriptors.

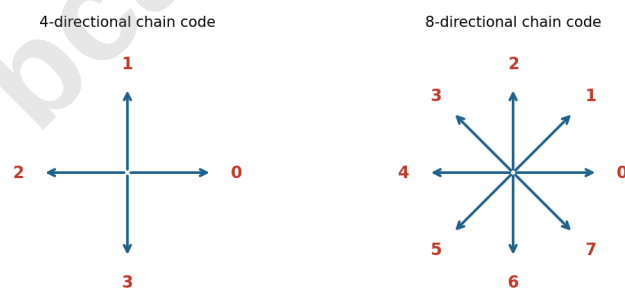


Figure 5.5 — 4- and 8-directional chain code numbering

Example 5.4 — Chain code

A 2×2 square boundary traced clockwise from its top-left corner has the 4-directional chain code 0 0 3 3 2 2 1 1 (0 = right, 1 = up, 2 = left, 3 = down). Find its first difference and shape number.

Solution

First difference = (current – previous) mod 4, treating the code as circular: (0–1) mod 4 = 3, 0–0 = 0, 3–0 = 3, 3–3 = 0, 2–3 = 3, 2–2 = 0, 1–2 = 3, 1–1 = 0 → **3 0 3 0 3 0 3 0**.

Smallest circular rotation (shape number) = **0 3 0 3 0 3 0 3** (order 8).

5.5.2 Region (Regional) Descriptors

- **Area** A (pixel count), **perimeter** P (boundary length), **centroid** $(\bar{x}, \bar{y})$.
- **Compactness** = P^2 / A (min. for a circle, 4π); **circularity** = $4\pi A / P^2$; **eccentricity** (ratio of principal axes); **Euler number** = components – holes.
- **Moments**: $m_{pq} = \sum \sum x^p \cdot y^q \cdot f(x, y)$; central moments μ_{pq} are translation-invariant; **Hu's seven invariant moments** are invariant to translation, scale and rotation.
- **Topological descriptors**: number of holes, connected components.

5.5.3 Texture

Texture describes the spatial arrangement of intensity (smooth, coarse, regular, grainy) over a region. Three approaches:

- **Statistical**: moments of the gray-level histogram: mean $m = \sum z_i p(z_i)$; variance σ^2 ; **smoothness** $R = 1 - 1/(1 + \sigma^2)$ (0 for constant regions, → 1 for rough); **uniformity** $U = \sum p^2(z_i)$; **entropy** $e = -\sum p(z_i) \log_2 p(z_i)$ (randomness).
- **Gray-Level Co-occurrence Matrix (GLCM)**: $G(i, j)$ = count of pixel pairs with levels i and j at a given offset (distance and direction). Haralick features computed from the normalised matrix $P(i, j)$: **contrast** $\sum (i-j)^2 P$, **energy** $\sum P^2$, **homogeneity** $\sum P/(1+|i-j|)$, correlation, entropy.
- **Structural** (texture primitives (texels) + placement rules) and **spectral** (Fourier spectrum peaks give periodicity and direction); also **Local Binary Patterns (LBP)** and Gabor/wavelet features.

Example 5.5 — GLCM

For the 4×4 image with 4 gray levels (rows) [0 0 1 1], [0 0 1 1], [0 2 2 2], [2 2 3 3], find the GLCM for the offset 'one pixel to the right' and compute contrast, energy and homogeneity.

Solution

Pairs (i, j) counted (12 in total): (0,0)×2, (0,1)×2, (1,1)×2, (0,2)×1, (2,2)×3, (2,3)×1, (3,3)×1.

GLCM rows (i = 0..3): [2 2 1 0], [0 2 0 0], [0 0 3 1], [0 0 0 1]; $P = G/12$.

Contrast = $\sum (i-j)^2 P = (2 \cdot 1 + 1 \cdot 4 + 1 \cdot 1) / 12 = 7/12 = \mathbf{0.583}$ [pairs (0,1)×2, (0,2)×1, (2,3)×1].

Energy = $\sum P^2 = (4+4+4+1+9+1+1)/144 = 24/144 = \mathbf{0.167}$. Homogeneity = $\sum P / (1+|i-j|) = \mathbf{0.819}$ (high → smooth texture).

5.6 Feature Reduction Algorithms

High-dimensional feature vectors cause the **curse of dimensionality**, redundancy, slow training and over-fitting. **Feature reduction** lowers the number of features while keeping discriminating information:

- **Feature selection** — choose the best subset of the original features (filter methods, wrapper methods, forward selection / backward elimination).
- **Feature extraction (transformation)** — construct a smaller set of new features from combinations of the originals: **PCA**, **LDA**, **ICA**.

Principal Component Analysis (PCA, Karhunen–Loève / Hotelling transform)

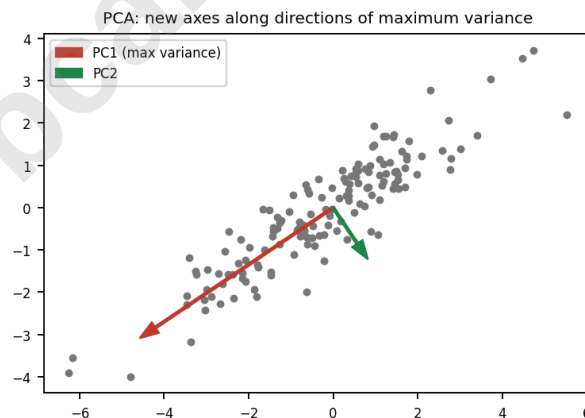


Figure 5.6 — PCA finds the axis of maximum variance (PC1)

1. Form the data vectors x_i ($i = 1 \dots K$), n -dimensional. Compute the mean vector $m = (1/K) \sum x_i$.
2. Compute the covariance matrix $C = (1/K) \sum (x_i - m)(x_i - m)^T$.
3. Find the eigenvalues $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n$ and eigenvectors $e_1 \dots e_n$ of C .
4. Keep the k eigenvectors with the largest eigenvalues (retaining e.g., 95 % of total variance $\sum \lambda$).

- 5. Project: $y_i = A_k(x_i - m)$, where the rows of A_k are the chosen eigenvectors. The new features are **uncorrelated**.

LDA (Linear Discriminant Analysis) is supervised: it finds the projection that maximises the ratio of between-class scatter to within-class scatter, so it focuses on class separability rather than just variance.

5.7 Image Classification

Classification assigns each object (feature vector x) to one of W known classes $\omega_1 \dots \omega_W$.

Supervised classification uses labelled training samples; **unsupervised** classification (clustering) discovers the groups automatically.

- **Minimum-distance classifier:** assign x to the class whose mean vector m_j is nearest:
 $D_j(x) = \|x - m_j\|$; equivalently the decision function $d_j(x) = x^T m_j - \frac{1}{2} m_j^T m_j$, choose the largest.
- **k-Nearest Neighbour (k-NN):** majority vote of the k closest training samples.
- **Bayes classifier:** choose the class with maximum posterior probability $P(\omega_j | x) \propto p(x | \omega_j)P(\omega_j)$; optimal in minimum-error sense when densities are known (often assumed Gaussian).
- **Others:** Support Vector Machine (maximum-margin hyper-plane), decision trees, neural networks / CNNs (learn features automatically).

Example 5.6 — Minimum-distance classifier

Class means: $m_1 = (1, 1)$, $m_2 = (5, 5)$. Classify $x = (2, 3)$.

Solution

$D_1 = \sqrt{[(2-1)^2 + (3-1)^2]} = \sqrt{5} = 2.236$; $D_2 = \sqrt{[(2-5)^2 + (3-5)^2]} = \sqrt{13} = 3.606$. $D_1 < D_2 \rightarrow x$ belongs to **class ω_1** .

5.8 Clustering Techniques

Clustering groups unlabelled data so that samples in a cluster are similar and those in different clusters are dissimilar. In images it is used for colour quantization and segmentation (cluster pixels by colour/position).

5.8.1 K-Means Clustering

- 1. Choose the number of clusters K and K initial centres (random or K samples).

- 2. **Assign** each sample to the nearest centre (Euclidean distance).
- 3. **Update** each centre to the mean of its assigned samples.
- 4. Repeat steps 2–3 until the assignments (or centres) no longer change. It minimises $J = \sum_k \sum_{x \in C_k} \|x - \mu_k\|^2$.

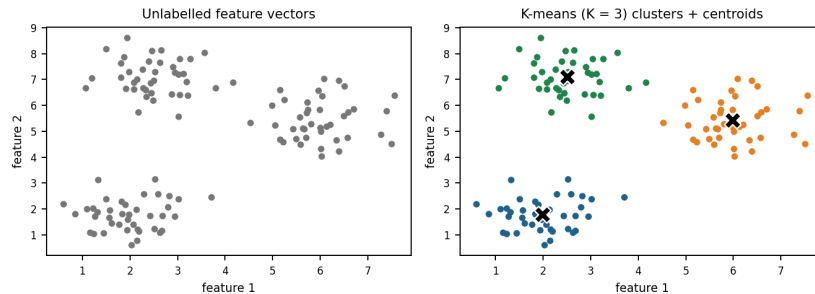


Figure 5.7 — K-means clustering of 2-D feature vectors ($K = 3$)

Example 5.7 — K-means (1-D)

Cluster the data $\{2, 4, 10, 12, 3, 20, 30, 11, 25\}$ into $K = 2$ clusters with initial centres 2 and 4.

Solution

Iteration 1: $C_1 = \{2, 3\} \rightarrow \text{mean } 2.50$; $C_2 = \{4, 10, 12, 20, 30, 11, 25\} \rightarrow \text{mean } 16.00$.

Iteration 2: $C_1 = \{2, 4, 3\} \rightarrow \text{mean } 3.00$; $C_2 = \{10, 12, 20, 30, 11, 25\} \rightarrow \text{mean } 18.00$.

Iteration 3: $C_1 = \{2, 4, 10, 3\} \rightarrow \text{mean } 4.75$; $C_2 = \{12, 20, 30, 11, 25\} \rightarrow \text{mean } 19.60$.

Iteration 4: $C_1 = \{2, 4, 10, 12, 3, 11\} \rightarrow \text{mean } 7.00$; $C_2 = \{20, 30, 25\} \rightarrow \text{mean } 25.00$.

Iteration 5: $C_1 = \{2, 4, 10, 12, 3, 11\} \rightarrow \text{mean } 7.00$; $C_2 = \{20, 30, 25\} \rightarrow \text{mean } 25.00$.

Centres stop changing: final clusters $C_1 = \{2, 4, 10, 12, 3, 11\}$, $C_2 = \{20, 30, 25\}$.

Note: K-means needs K in advance, is sensitive to initial centres and outliers, and finds only roughly spherical clusters; try several initialisations.

5.8.2 Other Clustering Techniques

- **Hierarchical clustering:** agglomerative (start with each sample as a cluster and repeatedly merge the closest pair; displayed as a **dendrogram**) or divisive. Linkage: single, complete, average.
- **Fuzzy C-means (FCM):** each sample has a membership degree in every cluster (soft assignment), useful for overlapping classes (e.g., MRI tissue).
- **Mean-shift / ISODATA / DBSCAN:** do not need K to be fixed (density- or mode-based).

Method	Supervised?	Needs K?	Output
K-means	No	Yes	Hard clusters
Hierarchical	No	No (cut dendrogram)	Tree of clusters
Fuzzy C-means	No	Yes	Soft memberships
k-NN / Min-distance / Bayes	Yes	—	Class labels

Key Takeaways

- ✓ Segmentation partitions an image by discontinuity (edges) or similarity (threshold, regions, clusters).
- ✓ Thresholding: global, adaptive, Otsu (maximise σ_B^2); iterative $T = \frac{1}{2}(\mu_1 + \mu_2)$.
- ✓ Region growing uses seeds; split-and-merge uses a quadtree and a homogeneity predicate.
- ✓ Edges: first derivative peaks / second derivative zero-crossing; Roberts, Prewitt, Sobel, LoG, Canny (NMS + hysteresis).
- ✓ Hough transform: $\rho = x \cos \theta + y \sin \theta$ voting for lines; extends to circles.
- ✓ Features: chain code, signatures, Fourier descriptors, moments, area/perimeter, texture (statistical, GLCM).
- ✓ PCA reduces dimensions via eigenvectors of the covariance matrix; LDA maximises class separability.
- ✓ Classification: minimum-distance, k-NN, Bayes; clustering: K-means, hierarchical, fuzzy C-means.

Reminder: these notes may contain errors — verify with your textbook and lecturer.

— End of Unit 5 —