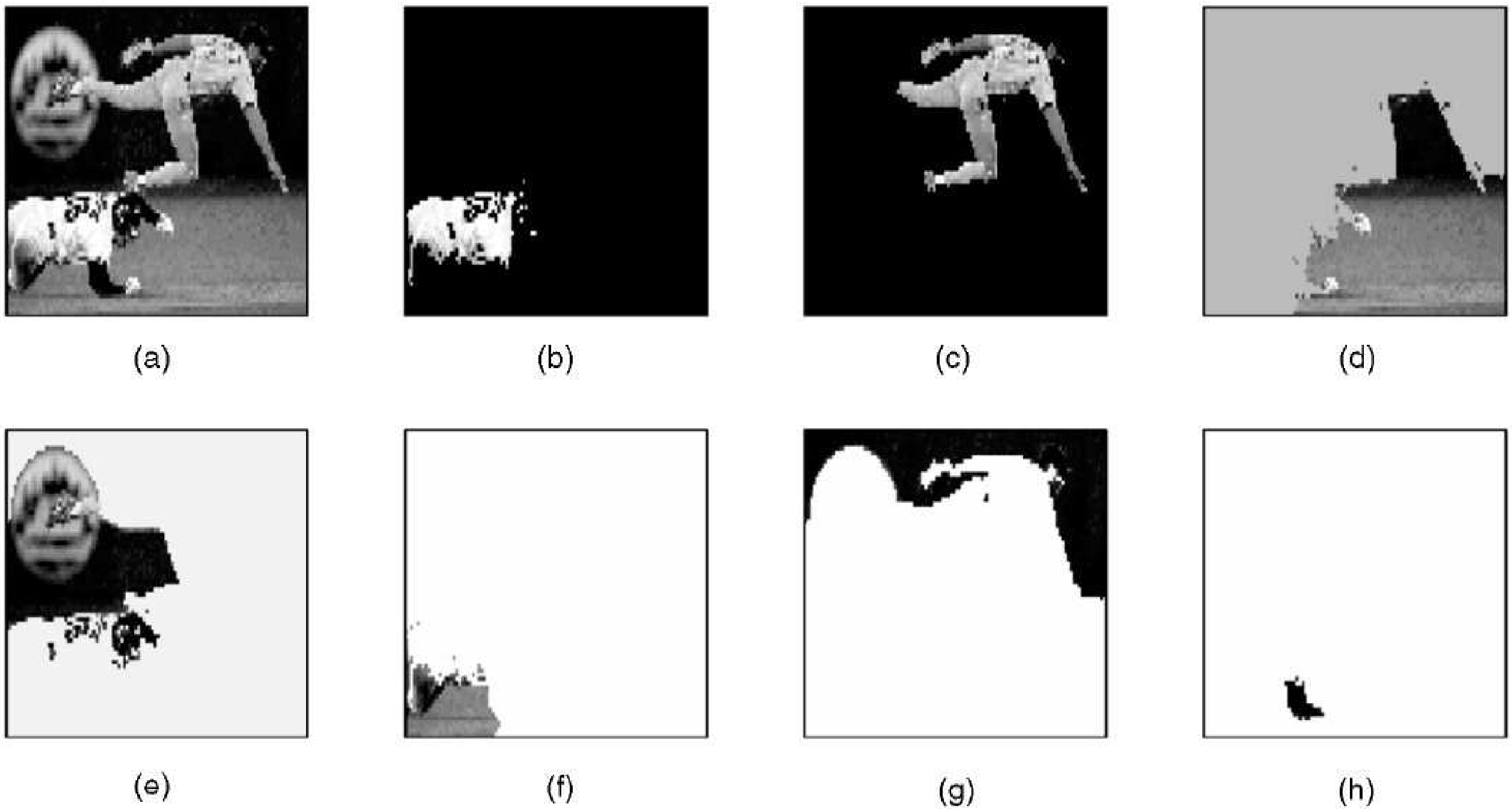
Segmentation by Clustering
Reading: Chapter 14 (skip 14.5)

- **Data reduction -** obtain a compact representation for *interesting* image data in terms of a set of components
- Find components that belong together (form **clusters**)
- Frame differencing - Background Subtraction and Shot Detection

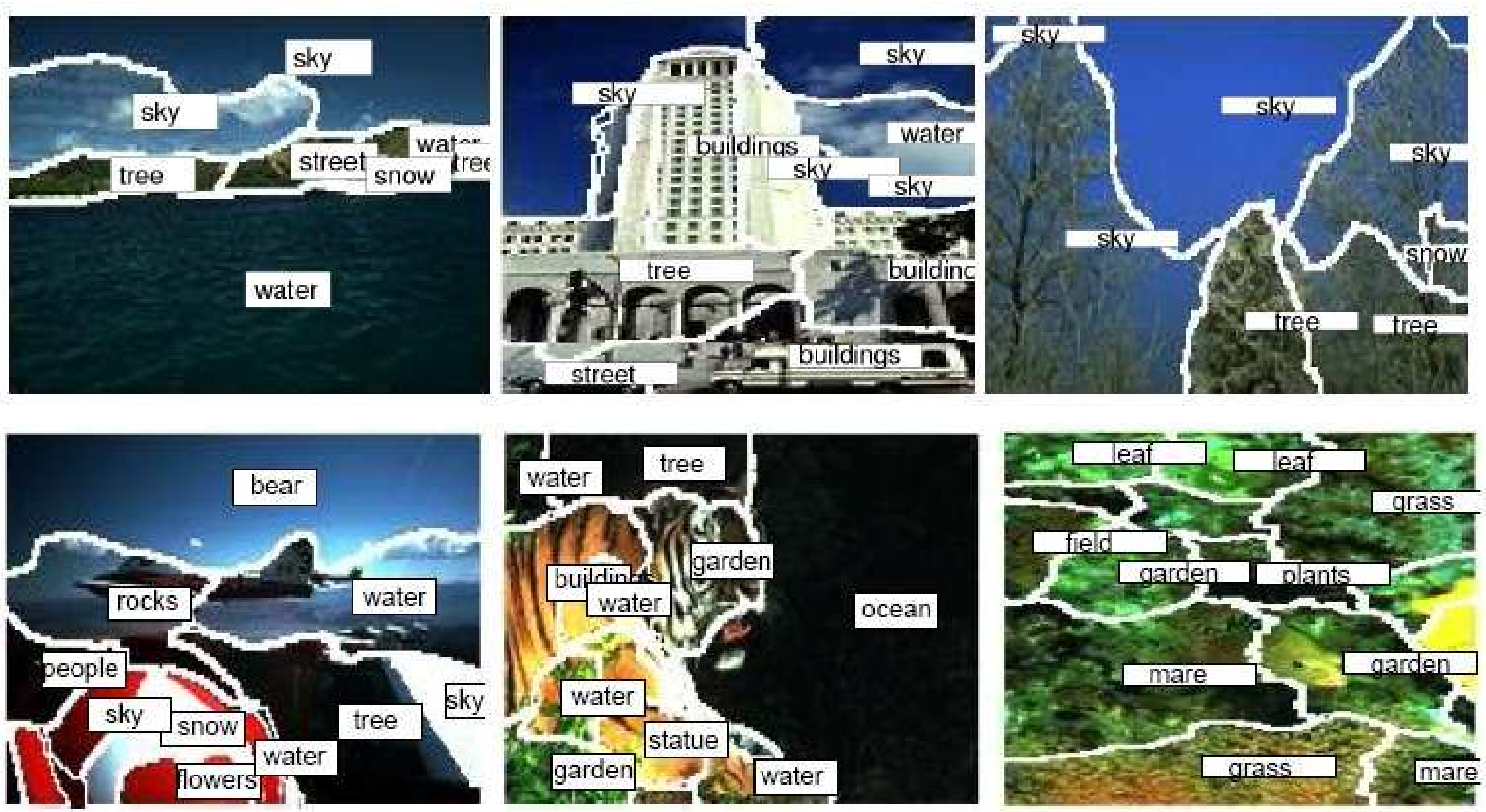
Slide credits for this chapter: David Forsyth, Christopher Rasmussen

Segmentation by Clustering

Segmentation by Clustering



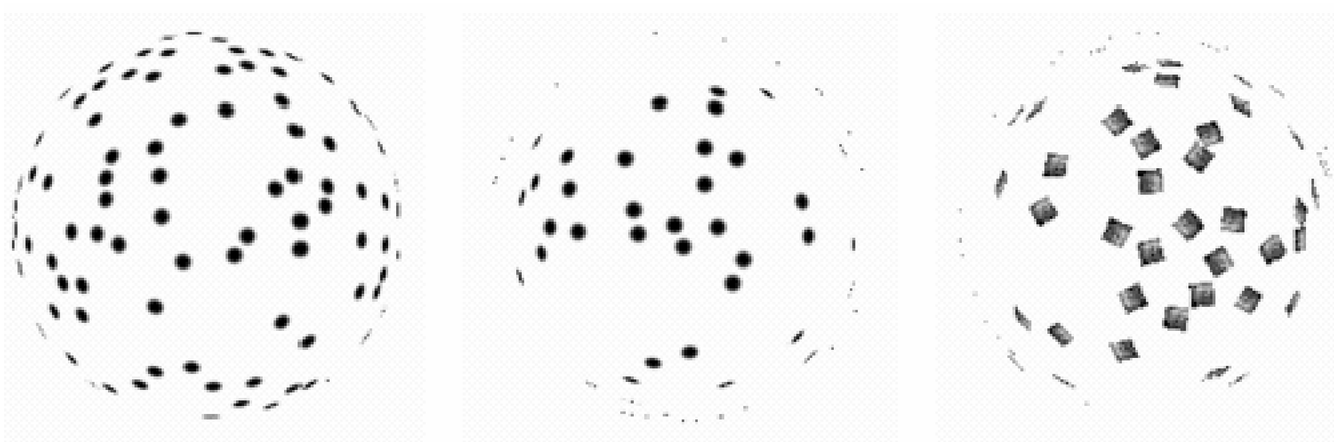
Segmentation by Clustering



From: Object Recognition as Machine Translation, Duygulu, Barnard, **de Freitas**, Forsyth, ECCV02

General ideas

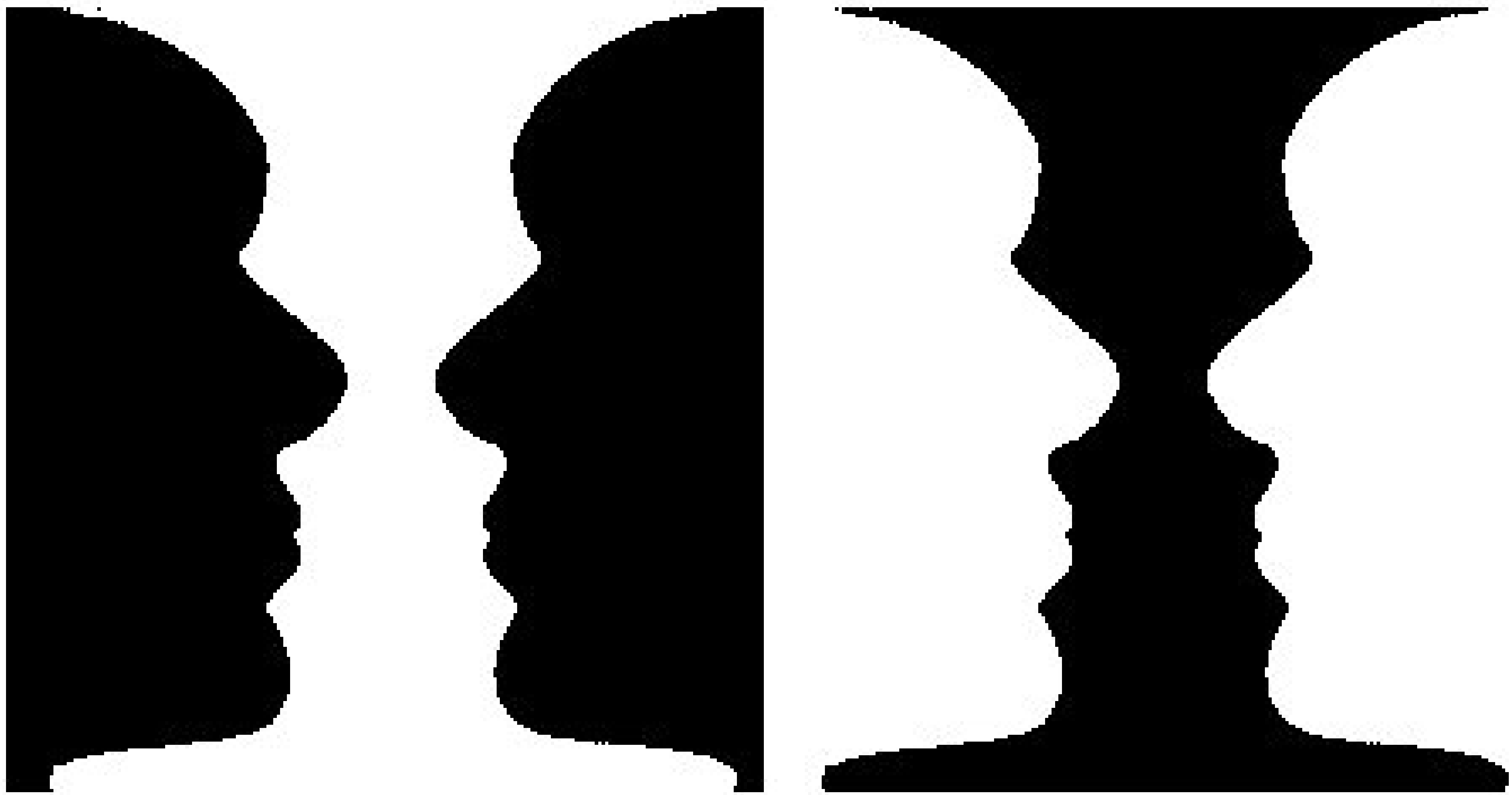
- **Tokens**
  - whatever we need to group (pixels, points, surface elements, etc., etc.)
- **Top down segmentation**
  - tokens belong together because they lie on the same object
- **Bottom up segmentation**
  - tokens belong together because they are locally coherent
- These two are not mutually exclusive



Why do these tokens belong together?

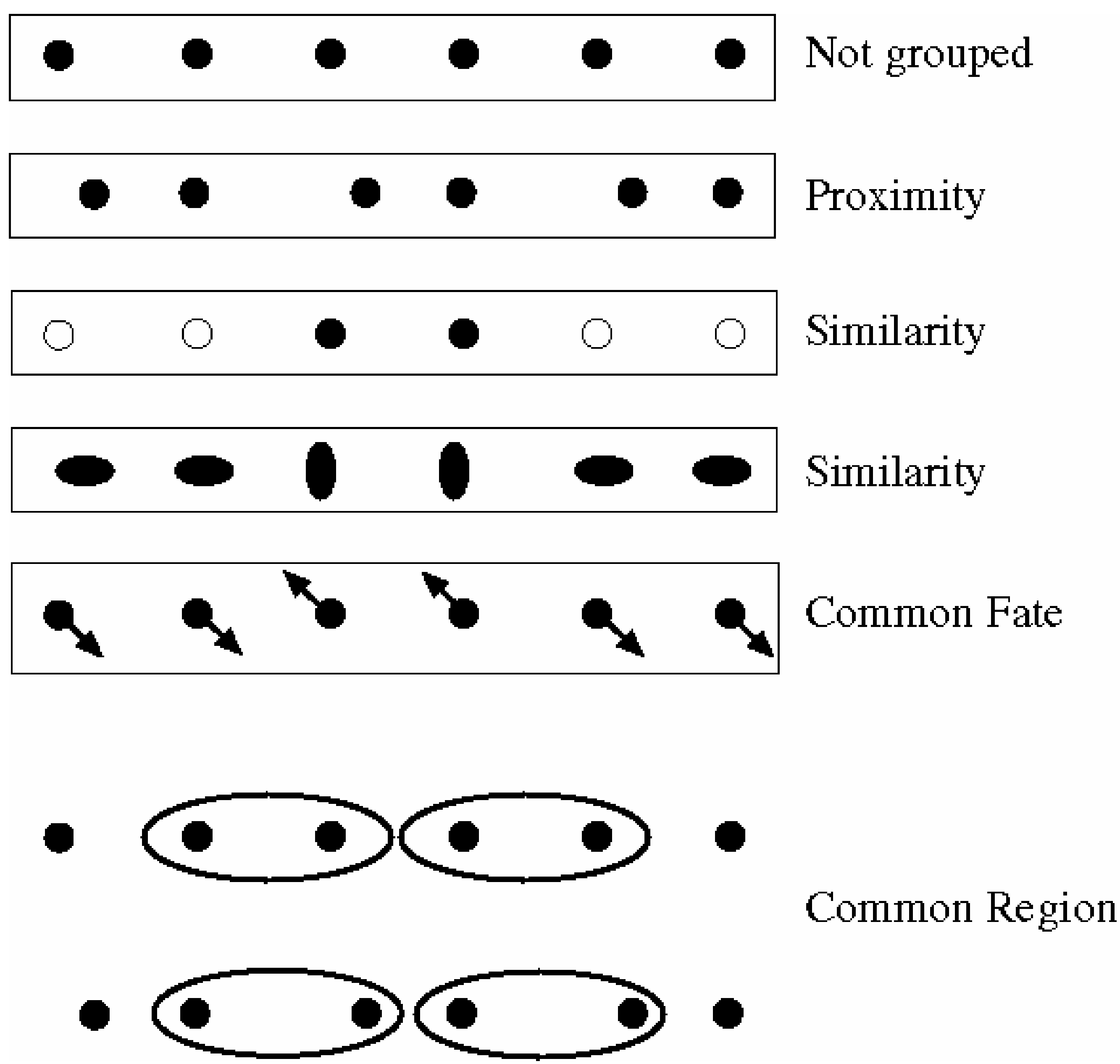
Top-down segmentation

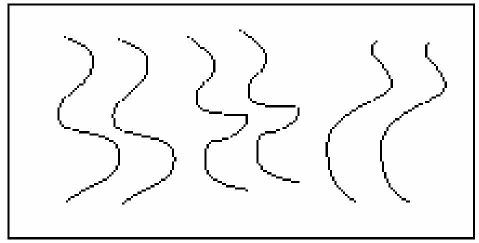
Credit: D. Marr, “Vision,” W.H. Freeman, 1982



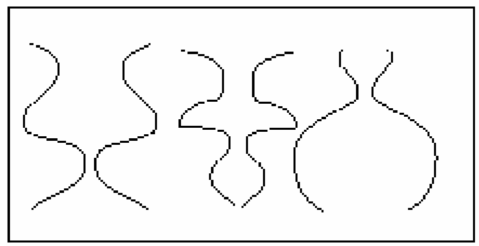
Basic ideas of grouping in human vision

- **Figure-ground discrimination**
  - grouping can be seen in terms of allocating some elements to a figure, some to ground
  - Can be based on local bottom-up cues or high level recognition
- **Gestalt properties**
  - Psychologists have studies a series of factors that affect whether elements should be grouped together
    - Gestalt properties

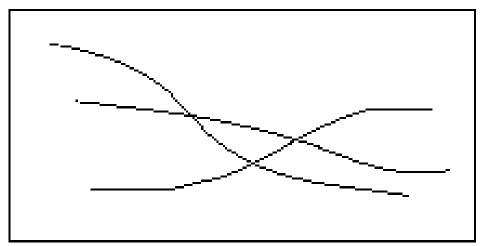

Not grouped
Proximity
Similarity
Similarity
Common Fate
Common Region



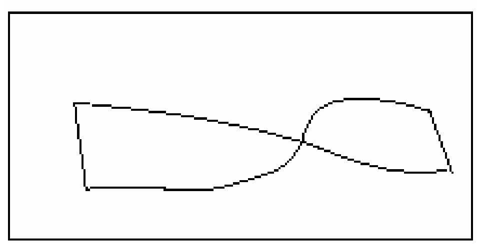
Parallelism



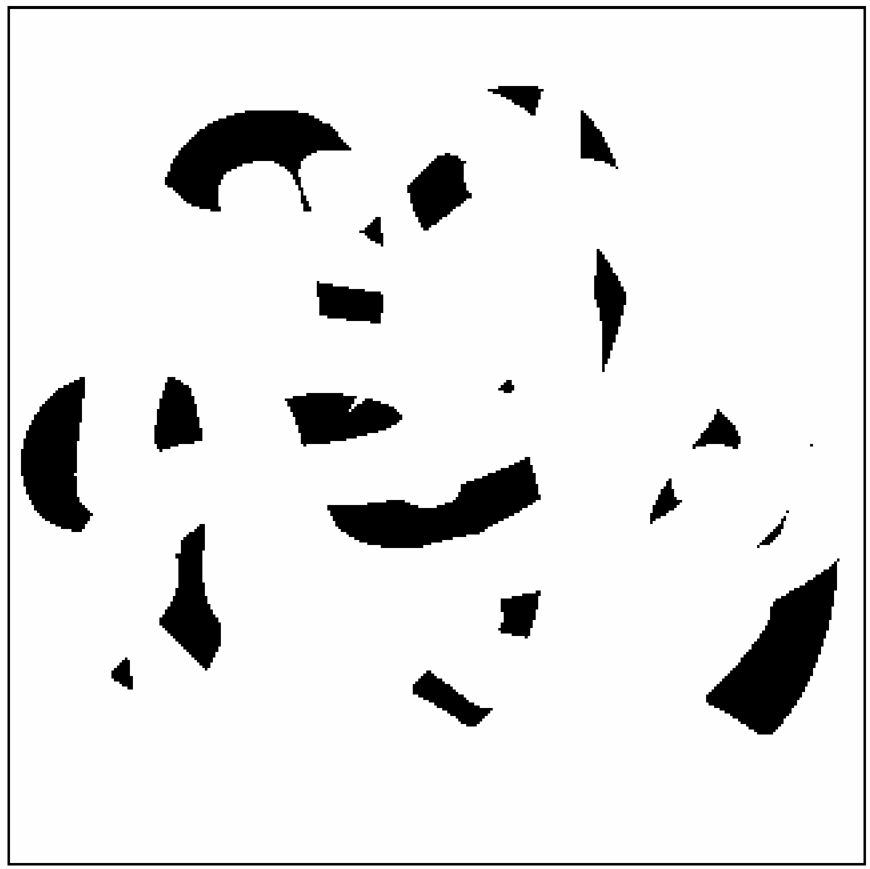
Symmetry

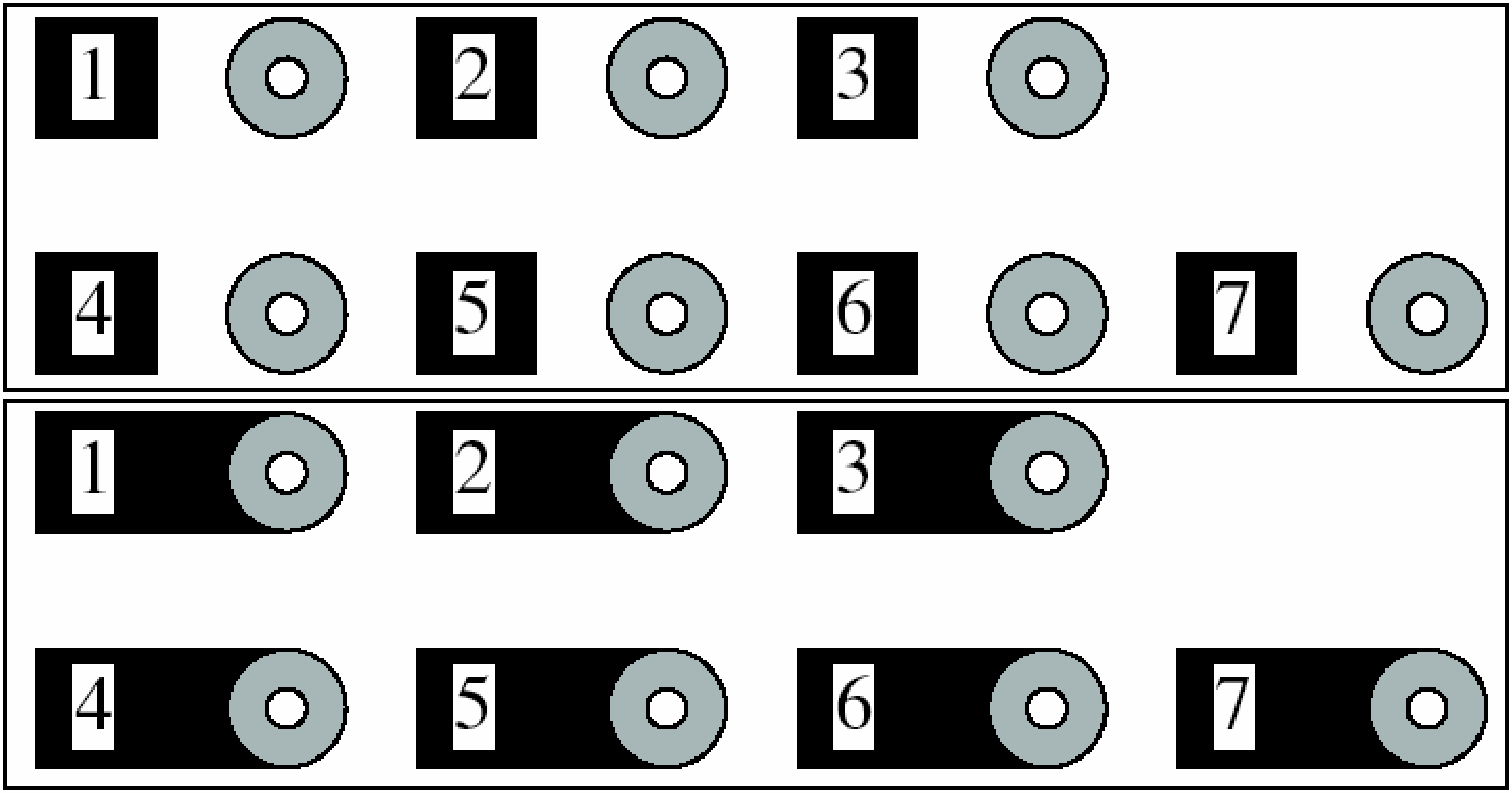


Continuity



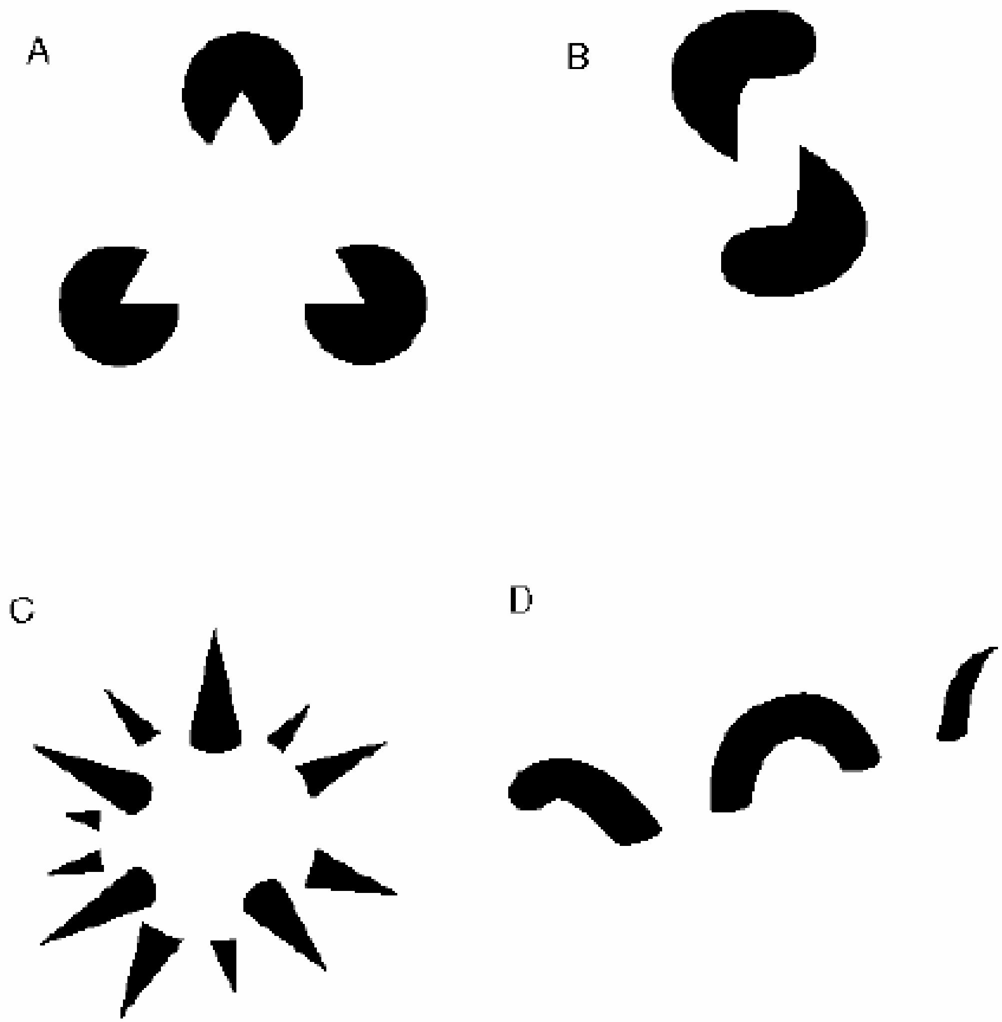
Closure





Elevator buttons in Berkeley Computer Science Building

Groupings by Invisible Completions



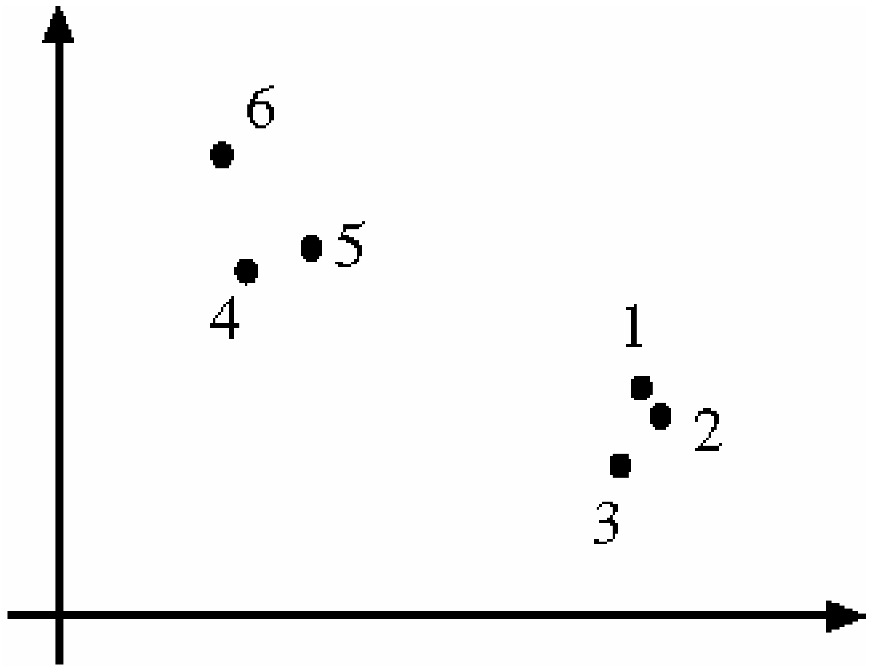
"Illusory Contours"

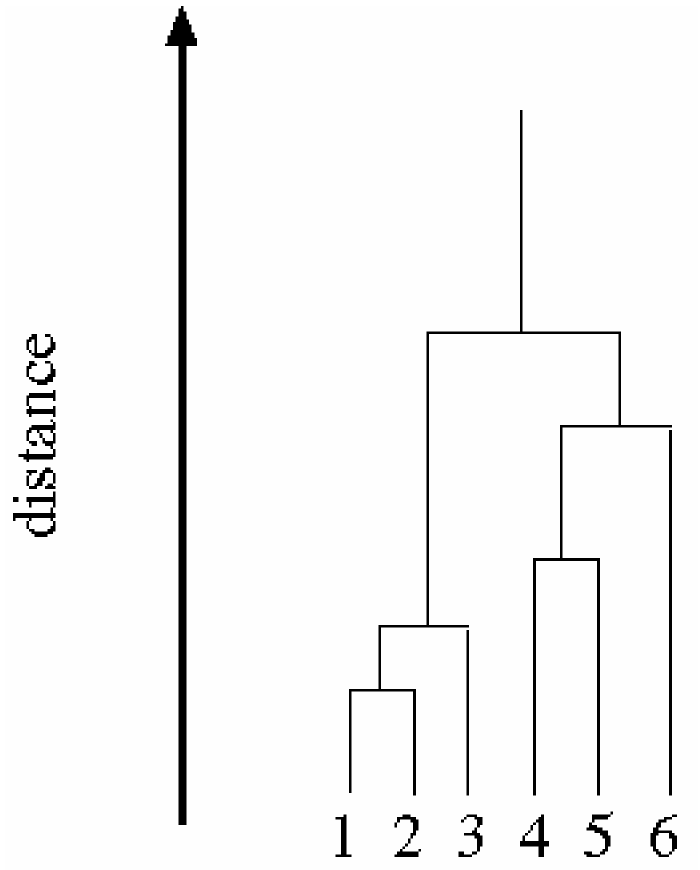
* Images from Steve Lehar's Gestalt papers: http://cns-alumni.bu.edu/pub/slehar/Lehar.html

Segmentation as clustering

- Cluster together (pixels, tokens, etc.) that belong together
- **Agglomerative clustering**
  - merge closest clusters
  - repeat
- **Divisive clustering**
  - split cluster along best boundary
  - repeat
- **Point-Cluster distance**
  - single-link clustering
  - complete-link clustering
  - group-average clustering
- **Dendrograms**
  - yield a picture of output as clustering process continues

Dendrogram from Agglomerative Clustering





Instead of a fixed number of clusters, the dendrogram represents a hierarchy of clusters

Feature Space

- Every token is identified by a set of salient visual characteristics called ***features***. For example:
  - Position
  - Color
  - Texture
  - Motion vector
  - Size, orientation (if token is larger than a pixel)

- The choice of features and how they are quantified implies a ***feature space*** in which each token is represented by a point

- Token similarity is thus measured by distance between points **("feature vectors")** in feature space

Slide credit: Christopher Rasmussen

K-Means Clustering

- Initialization: Given K categories, N points in feature space. Pick K points randomly; these are initial cluster centers (means) $m_1, \ldots, m_K$. Repeat the following:
  1. Assign each of the N points, $\boldsymbol{x}_j$, to clusters by nearest m_i (make sure no cluster is empty)
  2. Recompute mean m_i of each cluster from its member points
  3. If no mean has changed, stop

- Effectively carries out gradient descent to minimize:

$$\sum_{i \in \text{clusters}} \left\{ \sum_{j \in \text{elements of i'th cluster}} \left\| x_j - \mu_i \right\|^2 \right\}$$

Slide credit: Christopher Rasmussen

K-Means

Minimizing squared distances to the center implies that the center is at the mean:

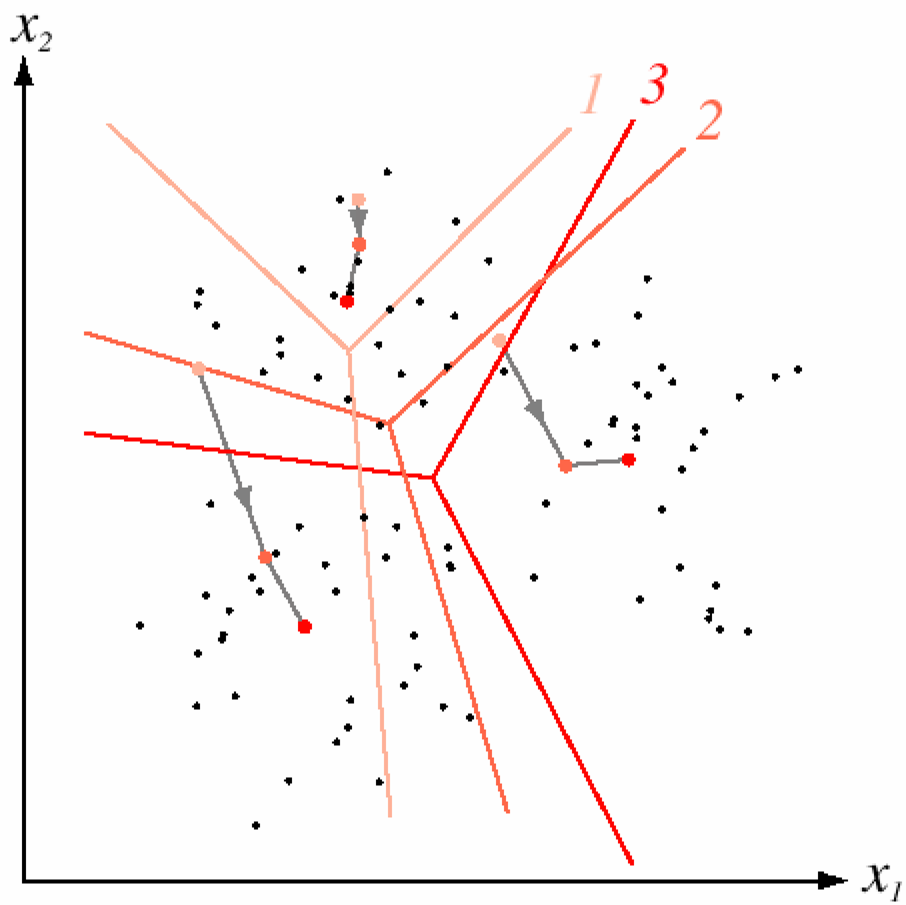
$$e(\mathbf{m}_i) = \sum_{i=1}^{n_c} \sum_{j;c_j=i} |\mathbf{x}_j - \mathbf{m}_i|^2$$

$$\frac{\partial e}{\partial \mathbf{m}_k} = \sum_{j;c_j=k} -2(\mathbf{x}_j - \mathbf{m}_k) = 0$$

Derivative of error is zero at the minimum

$$\mathbf{m}_k = \frac{\sum_{j;c_j=k} \mathbf{x}_j}{\sum_{j;c_j=k} 1} = \frac{1}{n_k} \sum_{j;c_j=k} \mathbf{x}_j$$

Example: 3-means Clustering



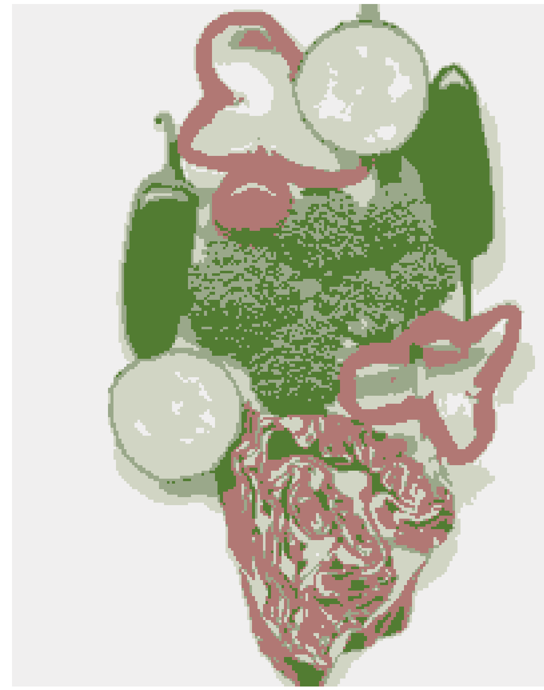
from
Duda et al.

Convergence in 3 steps

Image

Clusters on intensity

Clusters on color

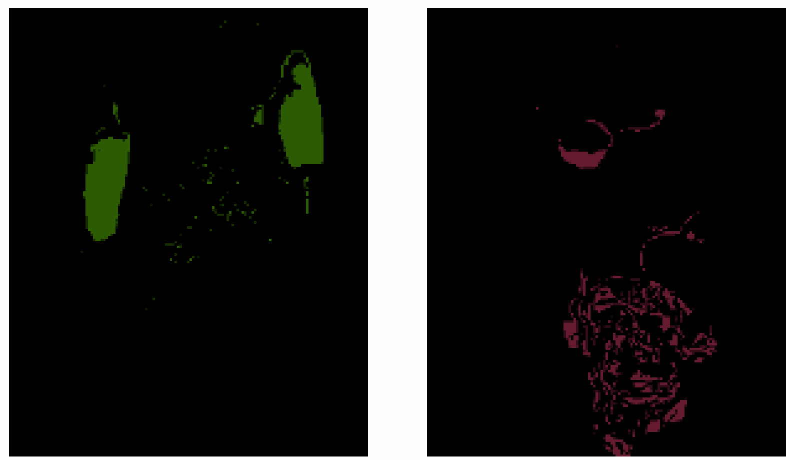


K-means clustering using intensity alone and color alone

Original Image

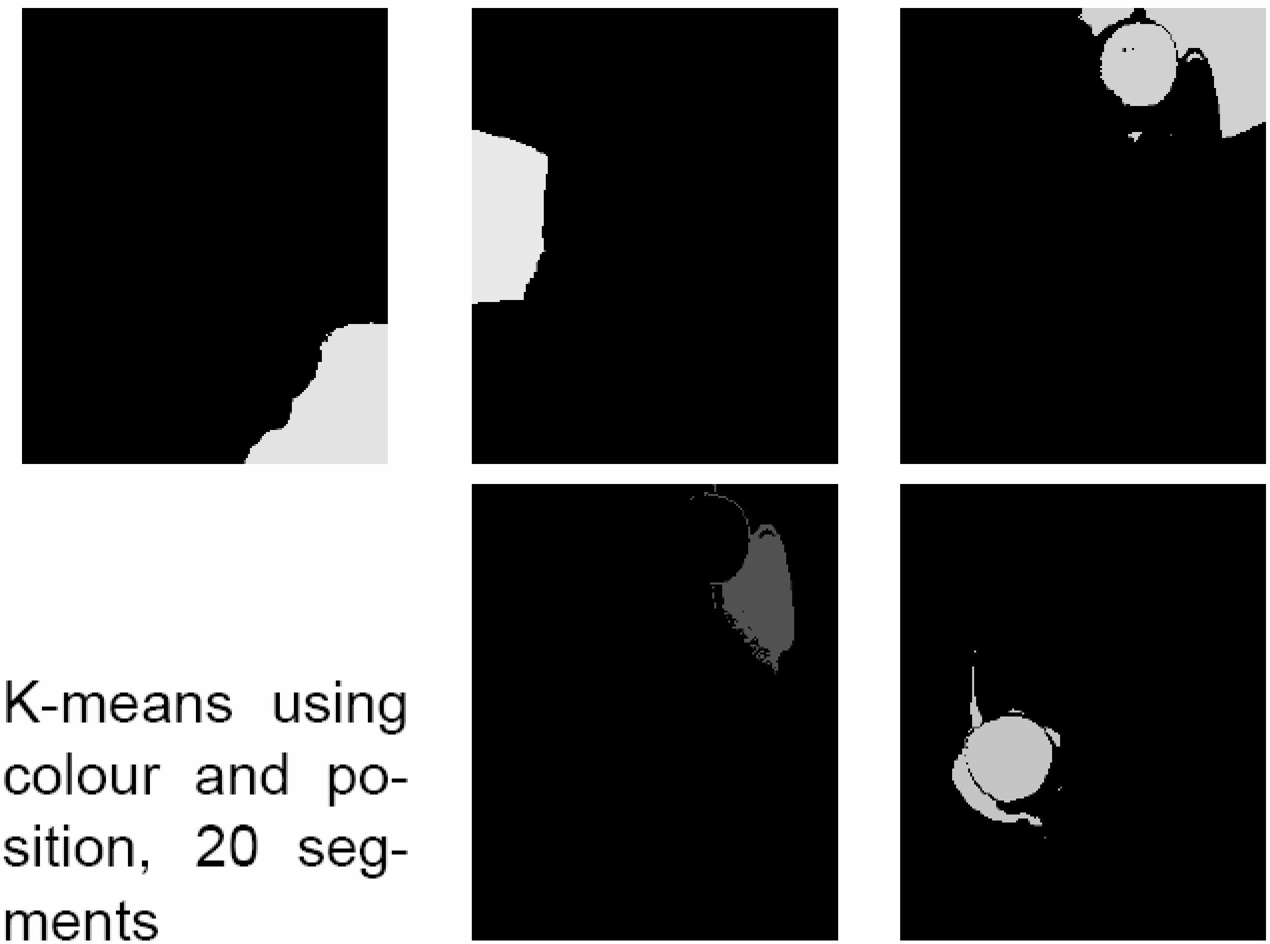
Segmentation Using Colour

K-means using colour alone, 11 segments



K-means using colour alone, 11 segments

Forsyth & Ponce Figure 14.14



K-means using colour and position, 20 segments

Forsyth & Ponce Figure 14.15

Technique: Background Subtraction

- If we know what the background looks like, it is easy to segment out new regions
- **Applications**
  - Person in an office
  - Tracking cars on a road
  - Surveillance
  - Video game interfaces

- **Approach:**
  - use a moving average to estimate background image
  - subtract from current frame
  - large absolute values are interesting pixels

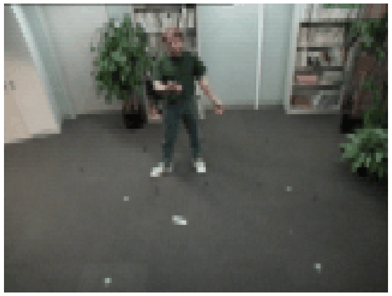
Background Subtraction

- The problem: Segment moving foreground objects from static


from C. Stauffer and W. Grimson
Current image


Background image


Foreground pixels


courtesy of C. Wren


Pfinder

Slide credit: Christopher Rasmussen

Algorithm

video sequence $I(\mathbf{x}, t)$ background $I_0(\mathbf{x}, t)$

frame difference $d(\mathbf{x}, t)$ thresholded frame diff $d_T(\mathbf{x}, t)$

for t = 1:N

  Update background model $I_0(\mathbf{x}, t)$

  Compute frame difference $d(\mathbf{x}, t) = |I(\mathbf{x}, t) - I_0(\mathbf{x}, t)|$

  Threshold frame difference $d_T(\mathbf{x}, t) = d(\mathbf{x}, t) > thresh$

  Noise removal $d_T(\mathbf{x}, t) = imerode(d_T(\mathbf{x}, t))$

end

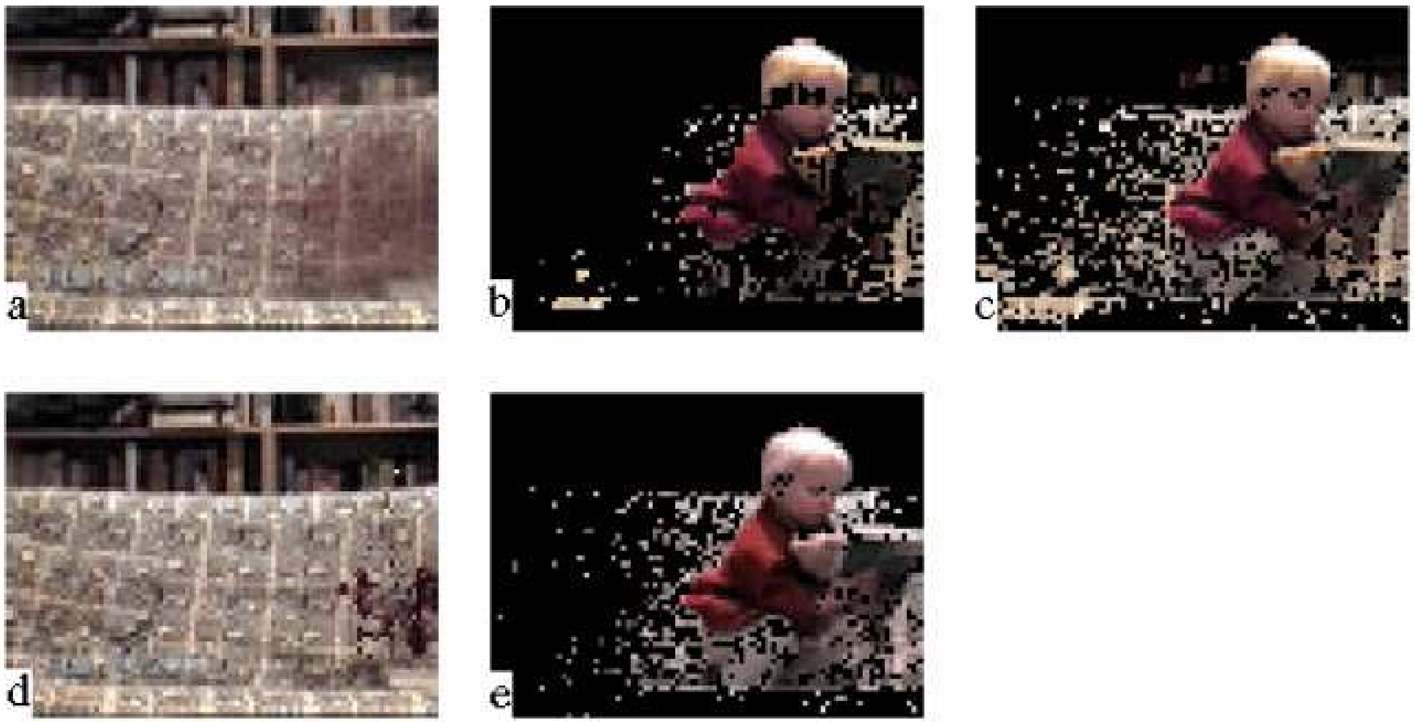
Objects are detected where $d_T(\mathbf{x}, t)$ is non-zero

Background Modeling

- **Offline average** $$I_0(\mathbf{x},t) = \frac{1}{T}\sum_{t=1}^{T} I(\mathbf{x},t)$$
  - Pixel-wise mean values are computed during training phase (also called Mean and Threshold)

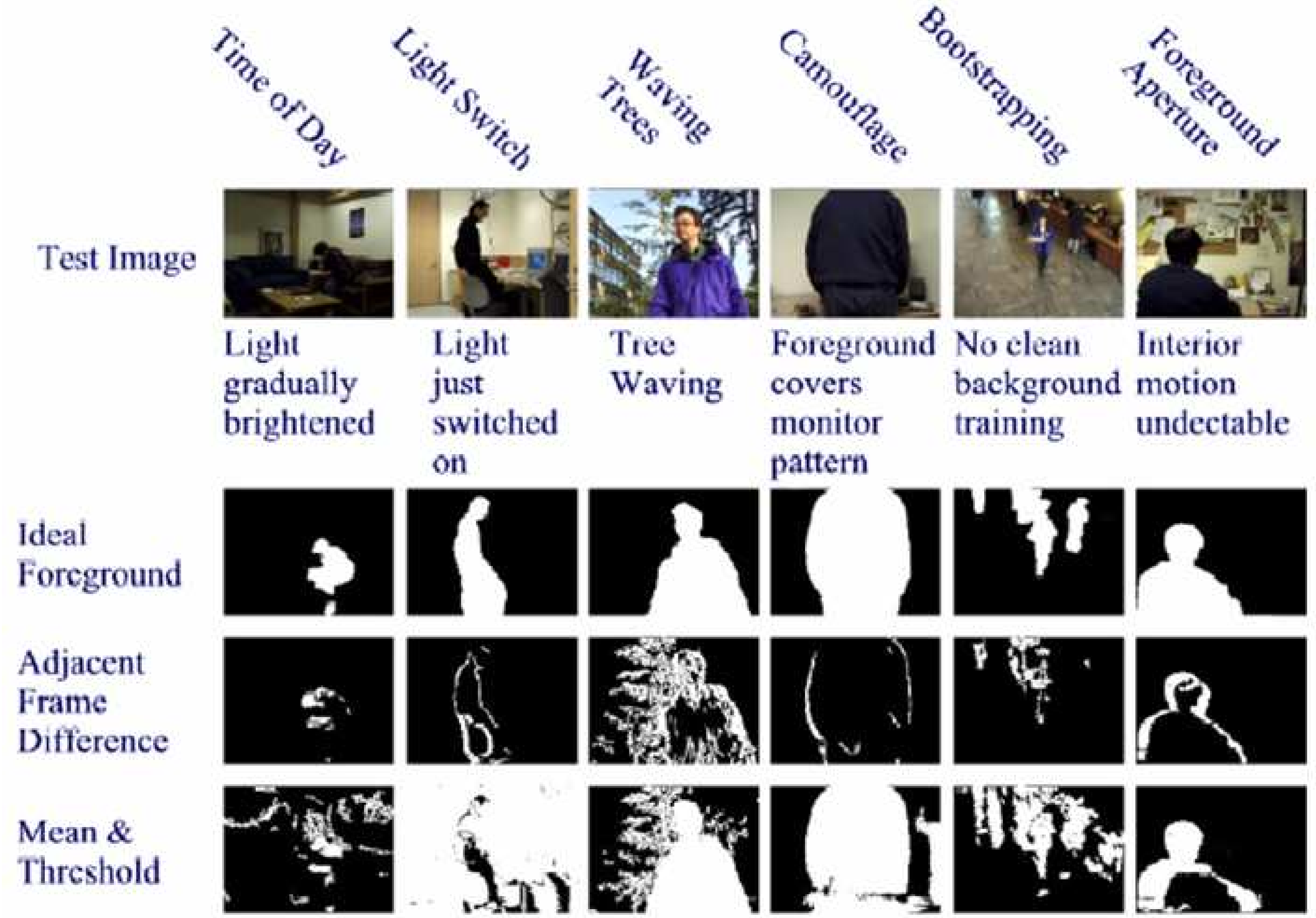
- **Adjacent Frame Difference** $$I_0(\mathbf{x},t) = I(\mathbf{x},t-1)$$
  - Each image is subtracted from previous image in sequence

- **Moving average** $$I_0(\mathbf{x},t) = \frac{w_a I(\mathbf{x},t) + \sum_{i=1}^{N} w_i I(\mathbf{x},t-i)}{w_c}$$
  - Background model is linear weighted sum of previous frames



Forsyth & Ponce Figure 14.10

Results & Problems for Simple Approaches



from K. Toyama et al.

Background Subtraction: Issues

- Noise models
  - **Unimodal**: Pixel values vary over time even for static scenes
  - **Multimodal**: Features in background can “oscillate”, requiring models which can represent disjoint sets of pixel values (e.g., waving trees against sky)

- Gross illumination changes
  - **Continuous:** Gradual illumination changes alter the appearance of the background (e.g., time of day)
  - **Discontinuous:** Sudden changes in illumination and other scene parameters alter the appearance of the background (e.g., flipping a light switch

- Bootstrapping
  - Is a training phase with “no foreground” necessary, or can the system learn what’s static vs. dynamic online?

Slide credit: Christopher Rasmussen

Application: Sony Eyetoy

- For most games, this apparently uses simple frame differencing to detect regions of motion
- However, some applications use background subtraction to cut out an image of the user to insert in video
- Over 4 million units sold

Technique: Shot Boundary Detection

- Find the **shots** in a sequence of video
  - shot boundaries usually result in big differences between succeeding frames
- **Strategy**
  - compute interframe distances
  - declare a boundary where these are big
- **Distance measures**
  - frame differences
  - histogram differences
  - block comparisons
  - edge differences
- **Applications**
  - representation for movies, or video sequences
    - obtain "most representative" frame
  - supports search