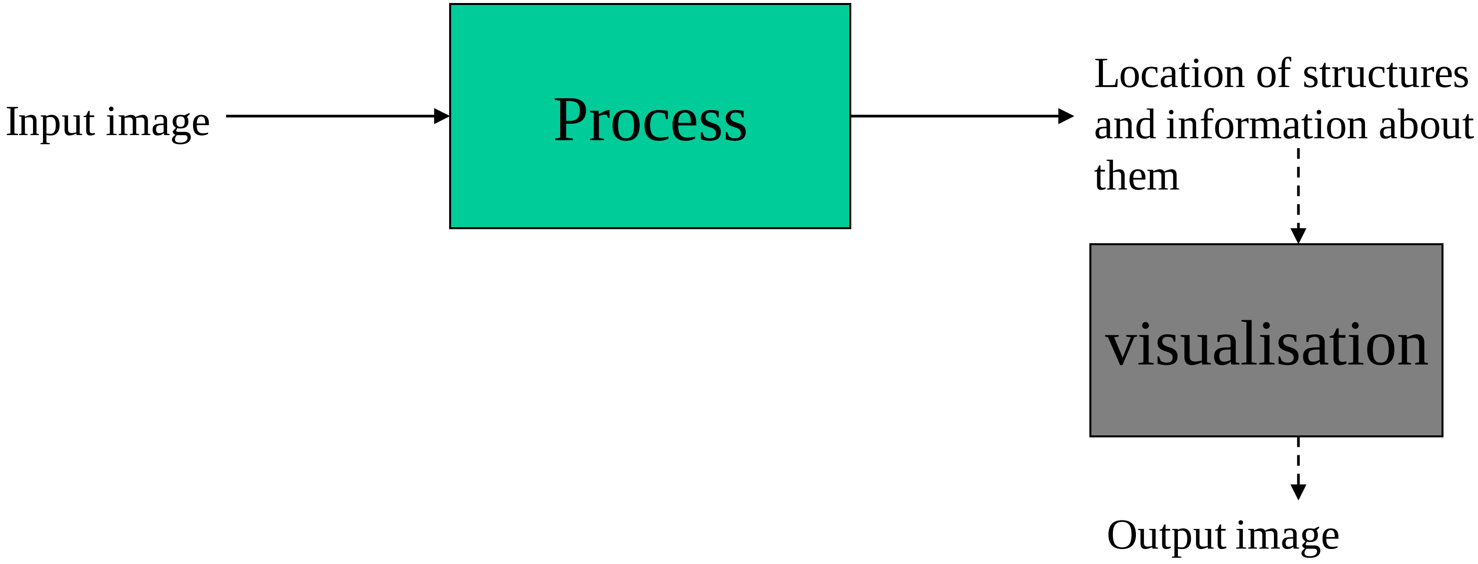
Image Analysis

What is image analysis?

- Processing images to organise the image into structures and derive information about them



Example application areas

- Image measurement
  - Area, perimeter, etc.
    - Ex. remote sensing
  - No of objects
    - Ex. medical images

- Object recognition
  - Faces, animals, buildings, license plates, etc
  - Terrain types
  - Lesions, tumors

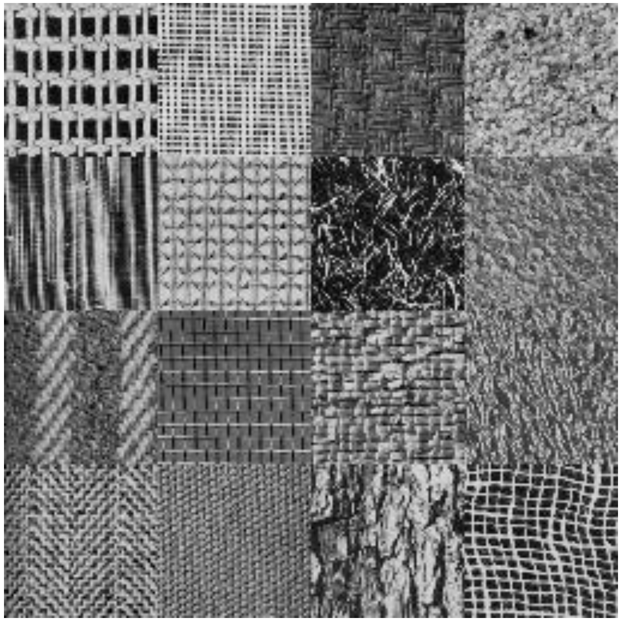
Key problems in image analysis

- Segmentation
- Feature detection
  - Feature extraction
    - Shape analysis
    - Texture analysis
- Motion analysis

Segmentation

Goal: Organise the image into <u>meaningful</u> groups/regions

- Groups are characterised by <u>content</u>

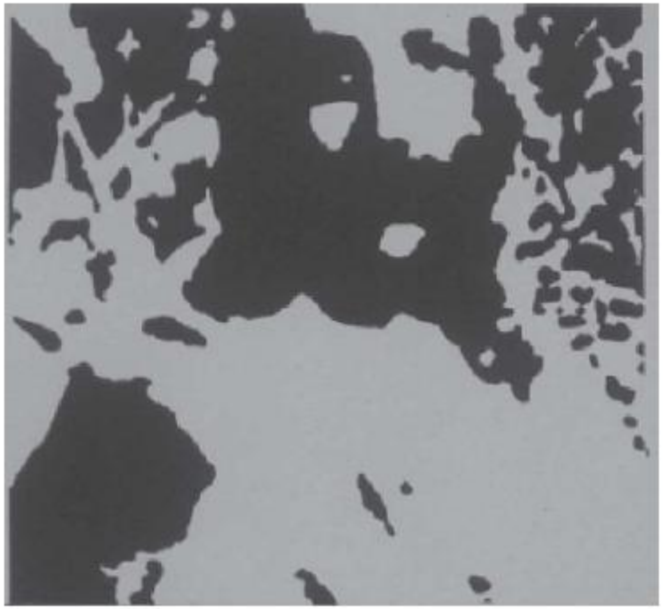


16 distinct regions/groups

How many regions/contours in this image?

Tough!

Object recognition



We see *'objects'* by grouping basic elements (blobs/lines)

Rule: Elements are neighbours *and* they are semantically related

- different ways of grouping ➔ different objects!

Grouping in HumanVision

How do we know which parts of visual input belong together ?

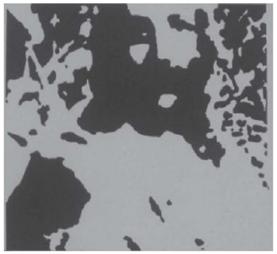
- This has been studied by perception theorists
- A set of principles have been identified:

Max Wertheimer's concept of *pragnanz*:

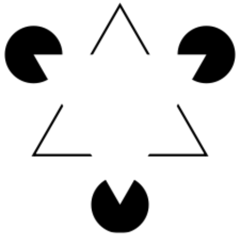
"when things are grasped as wholes, minimal amount of energy is exerted in thinking"

Key principles

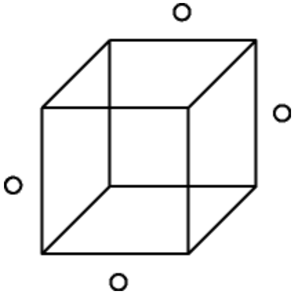
- **Emergence** – simple grouping rules lead to complex pattern formation



- **Reification** – perception is cosntructive/generative



- **Multistability** – multiple percepts can be stable and switch back and forth



- **Invariance** – recognition of simple objects is invariant to geometrical transformations

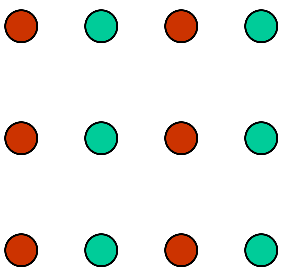
Gestalt principles of grouping

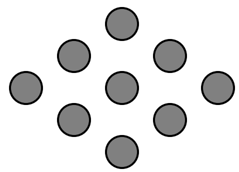
- **Proximity –** group based on neighbourliness of blobs
- **Common fate -** group if there is coherent motion
- **Parallelism-** group parallel curves/lines
- **Closure –** group if it leads to closed figures
- **Continuity –** group if continuous in space or feature
- **Similarity-** group based on some shared feature
- **Symmetry** –group if it can result in symmetric structures
- **Familiar Configuration** – if blobs when grouped, lead to a familiar object do the grouping!

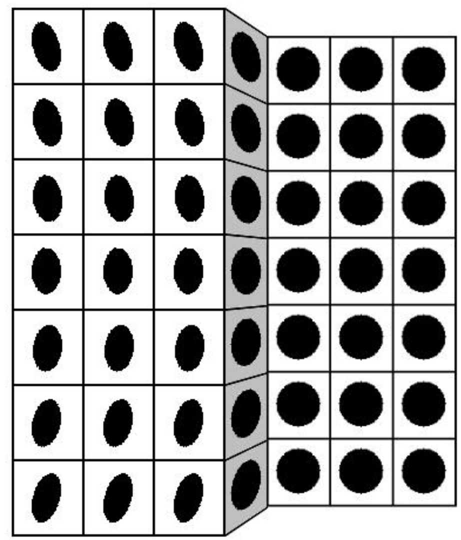
cognitive

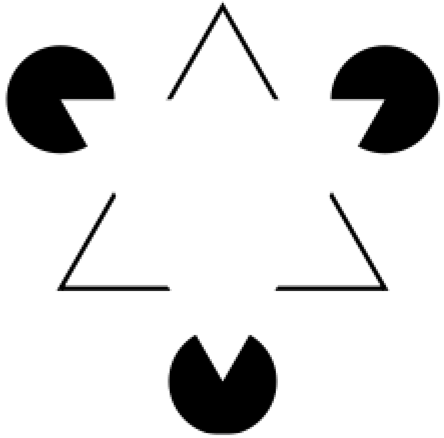
adapted from G.D. Hager

Grouping examples









Why is segmentation hard?

- It is an ill-posed problem!
- For natural images, the benchmark (ground truth) is human perception
- Need to make the problem tractable

Segmentation – towards a formulation

Goal of segmentation is to organise the image into groups/regions such that there is

1. Homogeneity within regions
   - Interiors are devoid of holes

2. Distinctness across adjacent regions

Segmentation – formulation 1

Given image *x[m,n]*,

$$x[m,n] = \bigcup_i R_i \qquad ; \bigcap_i R_i = \phi$$

<u>Find all the pixels belonging to a segment</u>:

Label each pixel as belonging to some R_i

- Membership (unique) in a region is based on shared property (homogeneity within a region)

Segmentation - formulation 2

Find all points that mark the interface between segments:

- $x[m,n] = \{c_i\}$ a set of contours or boundaries

- A contour point is a point where x is *discontinuous* in some feature space
  - Distinctness across regions

Segmentation into 2 classes

2 class case (binary output)

2 - class pixel classification

Thresholding

x:{object, background}

Edge Based

x:{object edge, background}

Thresholding - issues

- Division of image into 2 uniform regions
  - Object vs background
- Uniformity in intensity is popular

if $x[m,n] > t$

$x[m,n] = 255$ *object*

else

$x[m,n] = 0$ *background*

Problems:

- Thresholding (aka binarisation) is a point processing method
  - Context and spatial relations are ignored
    - Contiguous regions are not guaranteed
    - Will be sensitive to noise
- Post proc. is required for corrections

Thresholding approaches

Automatic thresholding

1. Fixed
   - limited use as it is sensitive to any change in image characteristic
   - seen earlier

2. Adaptive
   - Global
   - Local
   - Dynamic

Adaptive thresholding - Simple Methods

Adaptive thresholding

x :given image p_l : some local property

t : threshold

- **Global**: $t = f\{x[m,n]\}$ ex: $t = \alpha\mu_x$

  global property global mean

- **Regional**: $t = f\{p_l\}$ ex: $t = a\mu_l$

  local property

- **Dynamic**: $t = f\{[m,n], p_l\}$ ex: $t = a\mu_x + b\sigma_{mn}$

  location local property local standard deviation

Global thresholding

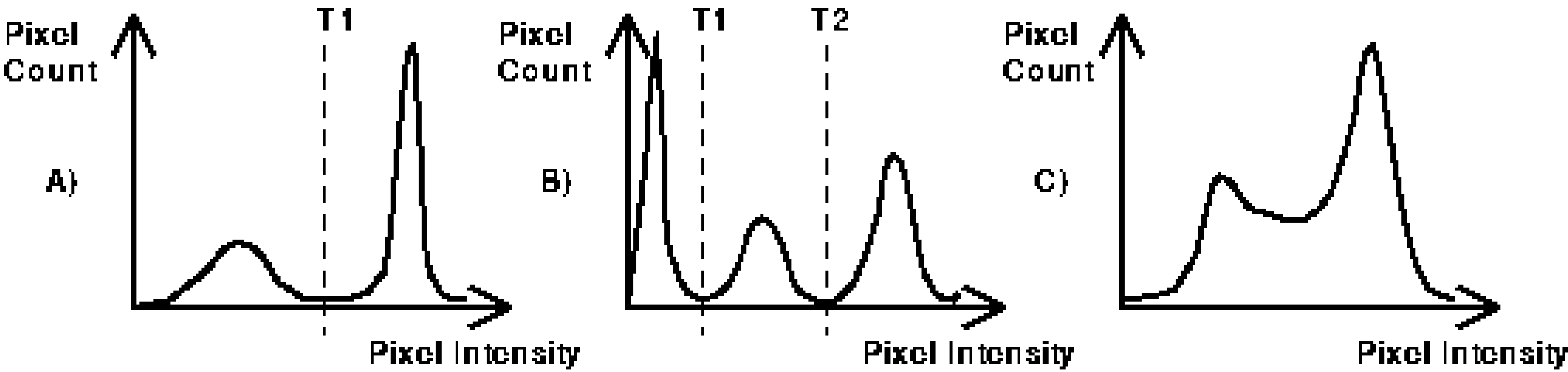
- Applied to the whole image

- Uses image data to determine the threshold
  - Brightness histograms *h[g]* are popular
  - *h* should have well defined peaks and troughs for best results
  - Single or dual thresholds {trough} or {trough 1: trough 2}

- Computationally <u>less</u> intensive

Selecting the threshold

Number of peaks and definition of troughs influence threshold selection



Difficult case

Histogram smoothing

- When a histogram is ragged, it can be smoothed before selecting thresholds

- Threshold selection is made easier

<u>Warning</u>: Smoothing can shift peaks

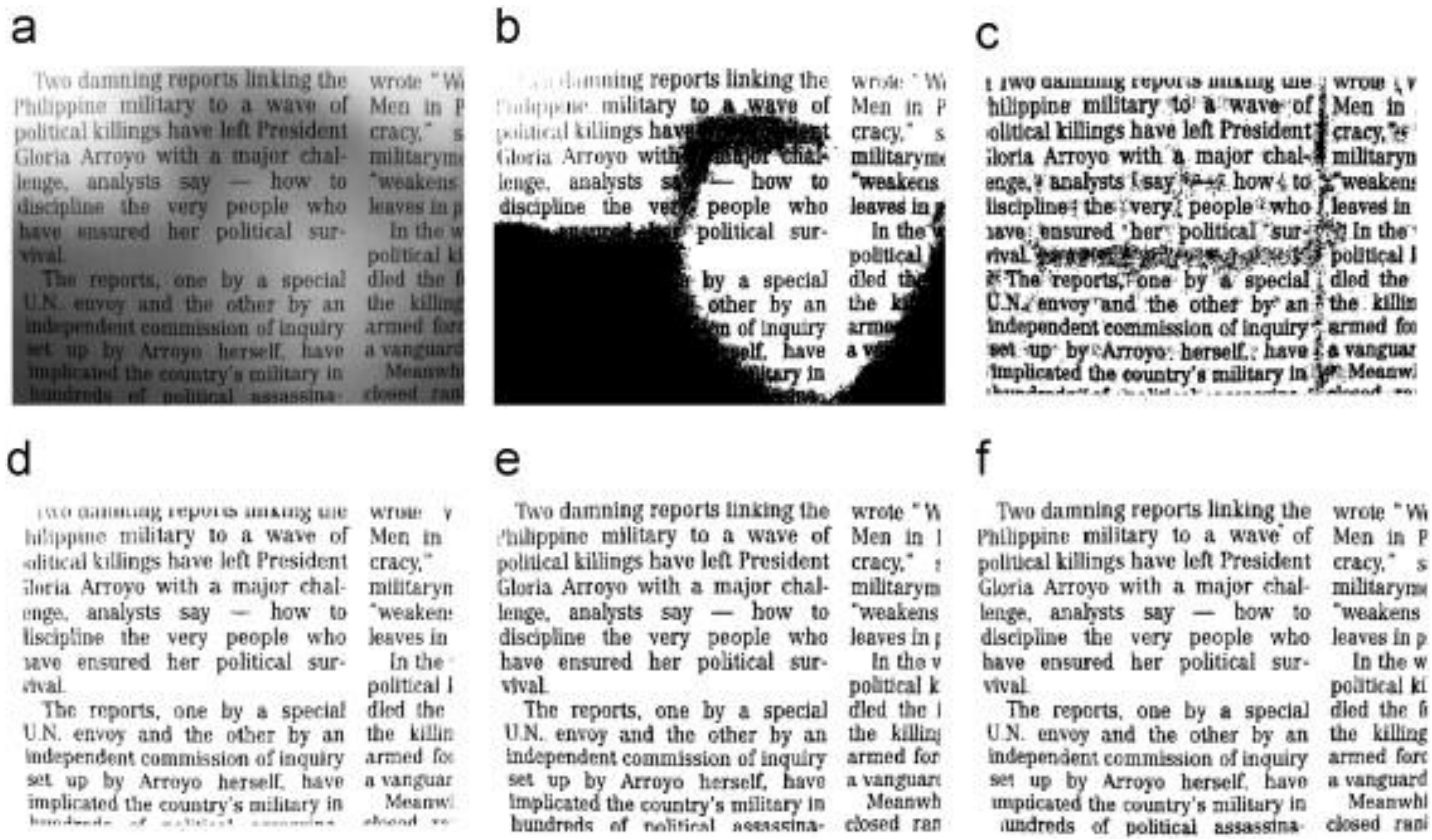
- Depends on size of smoothing kernel

Regional thresholding

- Divide the image into patches prior to thresholding
- Patch boundaries need to be post processed
- Combinations of local statistics and other derived information are used to find t
  - ex. Texture, entropy etc.
- Effective for handling non-uniform illumination
- Computationally <u>more</u> intensive

Thresholding- Example

Scanned doc



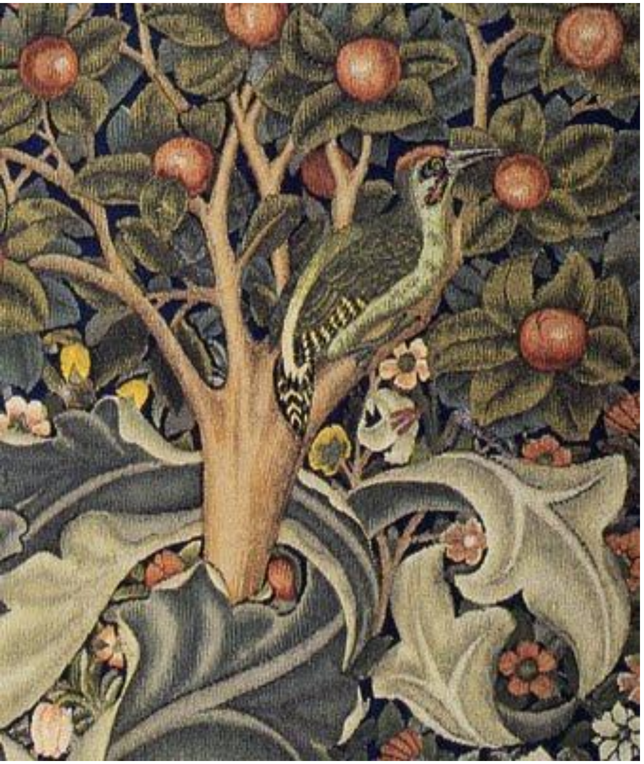
b to f. Results of binarisation with different methods

Pai et al PR 2010

Thresholding colour images $t = 127$

Where is the face?

Thresholding colour images $t = 127$



on green

on blue

Where is the bird?

Iterative and non-iterative methods

- Many traditional methods have been proposed
  - Triangle
  - Isodata
  - Otsu

- No universal solution exists!

- Newer more complex methods continue being proposed

Triangle algorithm

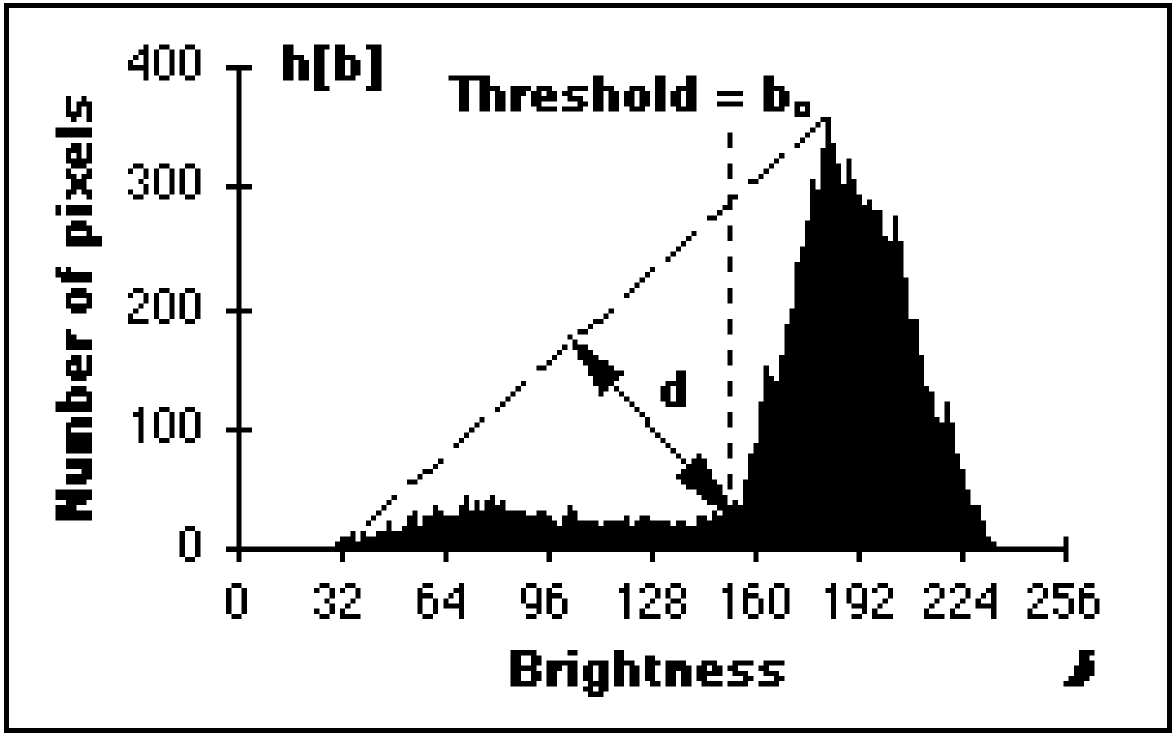
- A non-iterative method

Algorithm

From a given image with histogram $h[g]$

1. Find the line l between the highest peak i.e. $\max\{h[g]\}$ and g_{min}
2. For every g find distance $d(g)$ from l to $h[g]$
3. Desired $t = argmax\ d(g)$

Triangle algorithm



Effective when objects produce weak peaks

Isodata algorithm

- An iterative method

Algorithm:

1. Start with threshold $T_0 = 2^b - 1$ (for a b-bit grey scale image) to get 2 regions, R_1 and R_2

For $k > 0$

1. $T_{k+1} = 0.5\,(m_{k,1} + m_{k,2});$

   $m_{k,1}$ = mean grey value of R_1

2. Iterate until $T_{k+1} = T_k$

Note: No assumption is made about the distribution of pixel values in object/background

Otsu's method

$h_x(g)$: the histogram of image $x[m,n] \rightarrow p_x(g)$ pdf

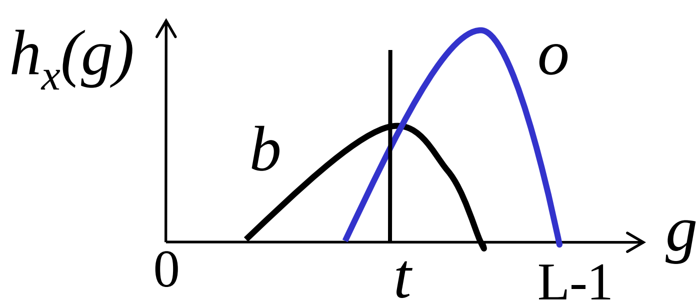
$g \in [0, L-1]$

o: object ***b***: background

Task: need to find a threshold $t = g_0$ from h_x

Let $g < t$ belong to b class and $g \geq t$ belong to o class

Need to minimise error probability in classification – MAP detector



Otsu's solution:
optimal t is one which minimises the variance <u>within</u> each class

$$\sigma^2_{within}(t) + \sigma^2_{between}(t) = \sigma^2$$

Otsu's solution ..contd.

$$\sigma^2_{between}(t) = n_b(t)\sigma_b^2(t) + n_o(t)\sigma_o^2(t)$$ Weighted sum of class variances

$$n_b(t) = \sum_{g=0}^{t-1} p(g); \quad n_o(t) = \sum_{g=t}^{L-1} p(g)$$

Similarly $\mu = n_b(t)\mu_b(t) + n_o(t)\mu_o(t)$

We can show that

$$\sigma^2{}_{between}(t) = n_b(t)n_o(t)[\mu_b(t) - \mu_o(t)]^2$$ Weighted squared difference of class means

Derivation for inter-class variance

global

$$\sigma^2_{between}(t) = \sigma^2 - \sigma^2_{within}(t)$$

$$= [\frac{1}{N}\sum_{m,n} x^2[m,n] - \mu^2] - [n_b(t)\sigma_b^2(t) + n_o(t)\sigma_o^2(t)]$$

$$= [\frac{1}{N}\sum_{m,n} x^2[m,n] - \mu^2] - [n_b(t)\sum_{m,n} x^2[m,n] - \mu_b{}^2] - [n_o(t)\sum_{m,n} x^2[m,n] - \mu_o{}^2]$$

N: number of pixels in x

$$= n_b(t)[\mu_b(t) - \mu]^2 + n_o(t)[\mu_o(t) - \mu]^2;$$

$$= n_b(t)n_o(t)[\mu_b(t) - \mu_o(t)]^2 \text{ since } \quad \mu = n_b(t)\mu_b(t) + n_o(t)\mu_o(t)$$

Otsu's algorithm

1. For every t_k , threshold x and bin the results into 2 bins.

2. Compute $\sigma^2_{between}(t_k) = n_b(t_k)n_o(t_k)[\mu_b(t_k) - \mu_o(t_k)]^2$

n_b and n_o are number of pixels in the 2 bins

3. Required $t = \arg\max \sigma^2_{between}(t_k)$

Can be implemented recursively

Comparison of thresholding methods

- There are many more methods to find threshold
- For a survey check

Sezgin, M and Sankur, B (2004), "Survey over Image Thresholding Techniques and Quantitative Performance Evaluation", Journal of Electronic Imaging 13(1): 146-165

- For a comparison check

  http://www.fmwconcepts.com/imagemagick/threshold_comparison/index.php

Segmentation– region based approaches

Region-based approaches

- Grouping by similarity and spatial proximity
- Suitable for segmenting non-uniform regions that are perceived to be uniform
- Region growing algorithms

Pixel aggregation **Split & merge**

Pixel aggregation

- A bottom-up approach to segmentation

Algorithm:

1. Start with a seed pixel
2. Add neighbouring pixels if they are "similar"
   - Similarity in grey value, texture, moments, colour etc
3. Stop when no new pixels are added

Issues in pixel aggregation

Seed selection

- Manual or automatic
- *A priori* knowledge is essential for good results

Criteria for aggregation

- Pixel value, local statistics, connectivity
- Too small vs large area for testing

Stopping the growth

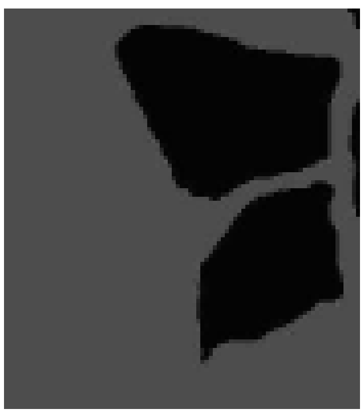
- Change in size, shape, some knowledge about boundary

❖ Inefficient for detecting large number of regions in a complex image
❖ Results are sensitive to *seed choice* for inhomogeneous regions
❖ To find K segments need to do K region growing operations

Region growing example

Input image

methods



Human marked segments

Computed segments

Quad tree (1971)

Given an image f of size (2^kx2^k), recursively divide it into smaller regions

Method:

Define a measure χ of intensity variation

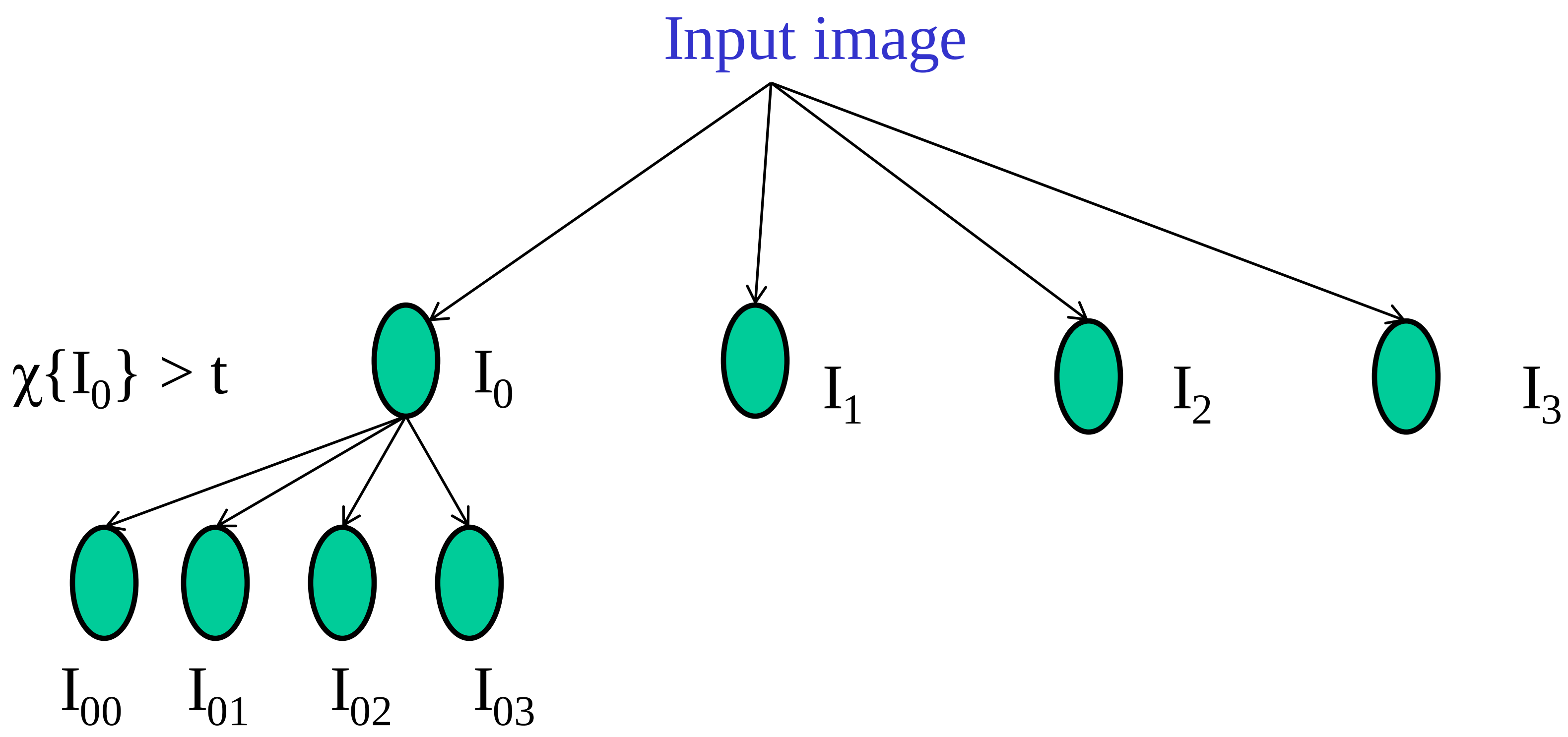
Set $f = f^k$

1. If $\chi(f^k) > \alpha$ then split f^k into subimages $f_j^{k-1}; j = 1,..n$

2. Repeat previous step on f_j^{k-1}

Final result: a tree of degree n, leaves are homogeneous subimages

Splitting using quad tree



Merge

Merging criterion: homogeneity

Method: Scan the split results and check for homogeneity of adjacent regions and merge them

Issues in split and merge

Splitting method

- Use of quadtrees - Can result in blocky boundaries

Criterion for splitting

- Test for inhomogeneity – local statistics
  - Standard statistical tests assume normal distributions which is rarely true

❖ Quality of result depends on testing criteria

❖ All segments can be found in one run

❖ No. of iterations depend on image content and criterion choice

Example

9	10	10	10	10	10	10	10
5	50	55	60	50	50	50	9
10	55	52	55	200	10	55	9
10	50	200	50	54	5	55	10
10	60	200	200	54	57	60	10
10	58	10	10	10	50	58	10
10	52	55	60	55	60	60	9
10	9	9	9	9	9	9	10

$g_{mean} = 55$; delta = ± 5 Red colour – ideal segment

Segmentation: Probabilistic approaches

Relaxation

- An iterative pixel labeling approach

- Unlike histogram-based approach, it takes into account both greyvalue and context of a pixel
  - Can incorporate local constraints in labelling

Uses a probabilistic reasoning for assigning labels

- p_{xl} : probability that pixel x has a label l
  - p_{xl} also depends on labels of neighbouring pixels
    - To enforce homogeneity criterion

Probabilistic relaxation - method

pixel values f_i ; $i = 1,2..N$ **classes** l_m ; $m = 1,2..M$

For every pixel pair f_i, f_h with labels l_j, l_k

Define

1. a <u>compatibility</u> measure $C(f_i,l_j\,;\,f_h,l_k)$

   $C > 0$ ➔ labels are compatible and vice versa

   $C \sim 0$ ➔ uncertain compatibility

2. p_{ij} : probability that f_i has label l_j

Strategy:

- initialise the probabilities
- if p_{hk} is high and $C(f_i,l_j; f_h,l_k) > 0$ then increment p_{ij}
- if p_{hk} is high and $C < 0$ then decrease p_{ij}
- if p_{hk} is low OR $C(f_i,l_j; f_h,l_k) \sim 0$ then do nothing to p_{ij}

Compatibility function

Practical assumptions:

- Defined only for neighbouring pixels
- It is spatially invariant
  - Requires $8M^2$ computations for a 3x3 neighbourhood

Example – detect smooth curves in an image

- Use the slope (θ_j) at every pixel f_i to define the initial p_{ij} ;
- p_{im} is the probability there is no curve at f_i
- Definition for C

$$c(i,j;h,k) = |\cos(\theta_j - \theta_{ih})||\cos(\theta_k - \theta_{ih})|$$

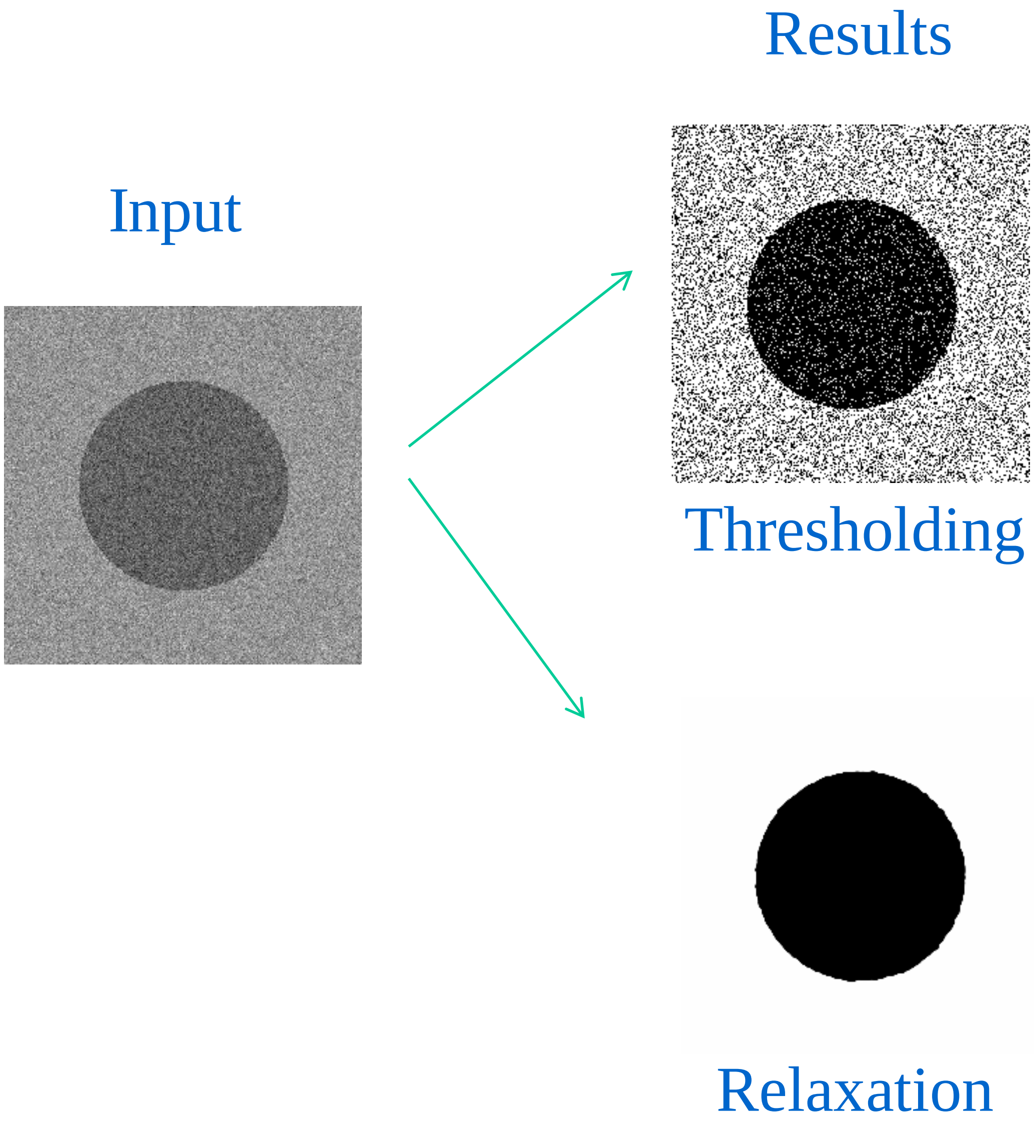
$$c(i,m;h,k) = -\cos 2(\theta_k - \theta_{ih})$$

- *c(i,j:h,k)* ➔ C =1 for collinear points and C=0 for ?
- *c(i,m:h,k)* ➔ a curve with slope θ_k at f_h is incompatible with no curve at f_i.

Probabilistic relaxation

- **Adv**: handles noisy conditions well
- **Disadv**: can suffer when structures have non-uniform shapes

Example



Relaxation - extensions

Relaxation labelling technique has evolved over time

- a pixel is represented by a host of features (not just intensity)

Newer techniques are based on :

- Expectation maximisation (EM)
- Gaussian mixture modelling (GMM)

Naïve Bayes theorem

x: pixel   c: class

$$p(c \mid x) = \frac{p(x \mid c)\, p(c)}{p(x)}$$

$p(c|x)$ *a posteriori* probability

$p(x|c)$ likelihood

$p(x)$, $p(c)$ are *prior* probabilities or 'Priors'

Segmentation as Expectation Maximisation

Assumption: Pixels in class *l* can have different values
→ represent it with a feature vector

θ_l : a vector of parameters (mean, variance of greyvalues, texture, etc) associated with pixels in class *l*

Define

- $p_l(x|\theta_l); l = 1,2..L$ the probability distribution of a pixel given a class label

EM formulation: Segmentation is an incomplete data estimation problem

- *incomplete* data are the measured feature vectors θ_l

EM algorithm

- Helps find the Maximum likelihood (ML) or Maximum a posteriori (MAP) estimate

EM algorithm

Initialize an estimate of the parameter vector $\theta_{l;}\ l = 1,2..L$

Repeat

1. <u>E step</u>

Estimate the labels based on the current parameter estimates

2. <u>M step</u>

Update the parameter estimates based on the current labelling

Until Convergence

Gaussian Mixture Modelling and EM

- Assume the probability distribution of the pixels in different classes to be Gaussians
- For L classes we have a mixture of L Gaussian distributions

$$p(x \mid \Theta) = \sum_{l=1}^{L} \alpha_l \, p_l(x \mid \theta_l)$$ Mixture of Gaussians

$$\Theta = \{\mu_l, \Sigma_l, \alpha_l\}$$

$$p_l(x \mid \theta_l) = \frac{1}{\sqrt{2\pi} \, \det(\Sigma_l)^{1/2}} e^{-\frac{1}{2}(x-\mu_l)^T \Sigma_l^{-1} (x-\mu_l)}$$

E step now has a linear solution

GMM based segmentation

<u>The E step</u> :

Given a pixel x_j and its parameter vector θ_j find its label l as

$$P(l \mid x_j, \theta_l) = \frac{\alpha_l p_l(x_j \mid \theta_l)}{\sum_{k=1}^{L} \alpha_k p_k(x \mid \theta_k)}$$

Posterior probability

GMM based segmentation

- The M Step - parameter updates

weight $$\alpha_l^{(m+1)} = \frac{1}{n}\sum_{j=1}^{n} P(l \mid x_j, \theta_l^{(m)})$$

Mean $$\mu_l^{(m+1)} = \frac{\sum_{j=1}^{n} x_j P(l \mid x_j, \theta_l^{(m)})}{\sum_{j=1}^{n} P(l \mid x_j, \theta_l^{(m)})}$$

Covariance $$\Sigma_l^{(m+1)} = \frac{\sum_{j=1}^{n} P(l \mid x_j, \theta_l^{(m)}) \left\{ (x_j - \mu_l^{(m)})(x_j - \mu_l^{(m)})^T \right\}}{\sum_{j=1}^{n} P(l \mid x_j, \theta_l^{(m)})}$$

Summary of segmentation

We have seen simple to complex methods

- Pixel and region based approaches
- Thresholding, clustering
- Iterative and non-iterative methods

All of the above use a bottom-up approach

Recent approaches are based on

- Partial differential equations (snakes, active contours, level sets..)
- Graph partitioning (graph cuts..)

These are covered in CV, MIP courses*!*