Image Segmentation

- Introduction
- Strategies for image segmentation
- examples

Introduction

- Objective
  - to divide an image in regions or objects according to some criterium
    - groups of pixels sharing similar characteristics
  - it is often a crucial step in enabling the interpretation of an image
    - the segmentation accuracy often determines the success of the following interpretation steps
- Two generic strategies:
  - Discontinuity or edge-based
    - the image is partitioned based on sudden changes or discontinuities on the image characteristics
  - Similarity or region-based (spatial grouping/clustering of pixels)
    - partitioning occurs based on the similarity degree between neighbouring pixels

Discontinuity-based segmentation

- Strategy based on the assumption that between two regions there must be a boundary, frontier, discontinuity or edge
  - the idea is to identify those discontinuities and thus accurately delineate regions
- Edges thus represent significant local changes in the image intensity values
  - they are important features to analyse an image and enable separating regions, objects or to identify changes in illumination or colours
- Implemented using Local Edge Operators
  - Gradient of image intensity
    - Robert, Sobel and Prewitt operators
  - Second Derivative Operators
    - Laplacian, Laplacian of Gaussian

Similarity-based segmentation

- Region-based segmentation methods that attempt to partition or group regions according to common image properties
  - intensity values from original images
  - computed values based on an image operator
  - textures or patterns that are unique to each region
- Complex systems may use a combination of these properties to segment images
  - basically the idea is to group/cluster pixels that are similar or that have similar statistical properties
    - and/or separate pixels that are dissimilar

Discontinuity detection - image derivatives

- the concept is to compute the first or second derivatives of images
- the derivatives of an image are functions of an independent variable (coordinates/space) with infinitesimal increments
  - the 1st derivative or gradient:

    $\nabla f(x,y) \equiv \delta f/\delta x + \delta f/\delta y = f'(x) + f'(y) = f(x+1)-f(x) + f(y+1)-f(y)$

    - if there are no changes in the intensity of the image then the derivative/gradient is zero
    - before any change or immediately after a change its value is zero
    - it is only during a change of intensity that its value is non-zero
  - the 2nd derivative has non-zero value only at the instants where there is a transition from no change to change in the intensity of the image and vice-versa

Discontinuity detection - gradient

- The gradient is the first derivative of the image f(x y)
  - it provides a measure of how the function f(x y) changes as a function of changes in the arguments x and y

$$\nabla f(x,y) \equiv \nabla_x + \nabla_y = G_x(x,y) + G_y(x,y) = \delta f/\delta x + \delta f/\delta y =$$

$$f'(x) + f'(y) = f(x+1) - f(x) + f(y+1) - f(y)$$

  - the gradient vector points in the direction of maximum change
  - its length indicates the magnitude of the gradient
- in practical terms, the computation of the derivatives can be approximated to the convolution of specific kernels/masks

first derivative / gradient	second derivative
$G_x = \delta f/\delta x = [-1, 1]$	$\delta^2 f/\delta x^2 = [1, -2, 1]$
$G_y = \delta f/\delta y = [-1, 1]^T$	$\delta^2 f/\delta y^2 = [1, -2, 1]^T$

Discontinuity detection - gradient

- The gradient is the first derivative of the image f(x y)
  - it provides a measure of how the function f(x y) changes as a function of changes in the arguments x and y

$$\nabla f(x,y) \equiv \nabla_x + \nabla_y = G_x(x,y) + G_y(x,y) = \delta f/\delta x + \delta f/\delta y =$$

$$f'(x) + f'(y) = f(x+1) - f(x) + f(y+1) - f(y)$$

  - the gradient vector points in the direction of maximum change
  - its length indicates the magnitude of the gradient
- So the operations to perform using kernels H(x,y):
  - obtain horisontal edges by computing $G_x(x,y)=H_x*f(x,y)$
    - convolution with the horizontal filter kernel H_x
  - obtain vertical edges by convolving with the vertical kernel H_y

Discontinuity detection - gradient using local operators

- but normally, larger kernels are used to include smoothness and deliver coherent results in both directions
  - common operators for computing the gradient are the Roberts, Prewitt and Sobel kernels

Roberts

$H_x(x,y)$

0	1
-1	0

$H_y(x,y)$

1	0
0	-1

Prewitt

$H_x(x,y)$

-1	0	1
-1	0	1
-1	0	1

$H_y(x,y)$

1	1	1
0	0	0
-1	-1	-1

Sobel

-1	0	1
-2	0	2
-1	0	1

1	2	1
0	0	0
-1	-2	-1

- So the operations to perform using the kernels H(x,y) to compute the gradient are:
  - obtain horisontal edges by computing $G_x(x,y)=H_x*f(x,y)$
    - this is the convolution with the horizontal filter kernel H_x
  - obtain vertical edges by convolving with the vertical kernel H_y

Discontinuity detection - gradient using local operators (2)

- Using the gradient or derivatives, two values are computed
  - magnitude $|\nabla f(x,y)| \equiv G$ and orientation θ

$$G = \sqrt{(G_x^2 + G_y^2)} \qquad \theta = \text{arctang}\,(G_y/G_x)$$

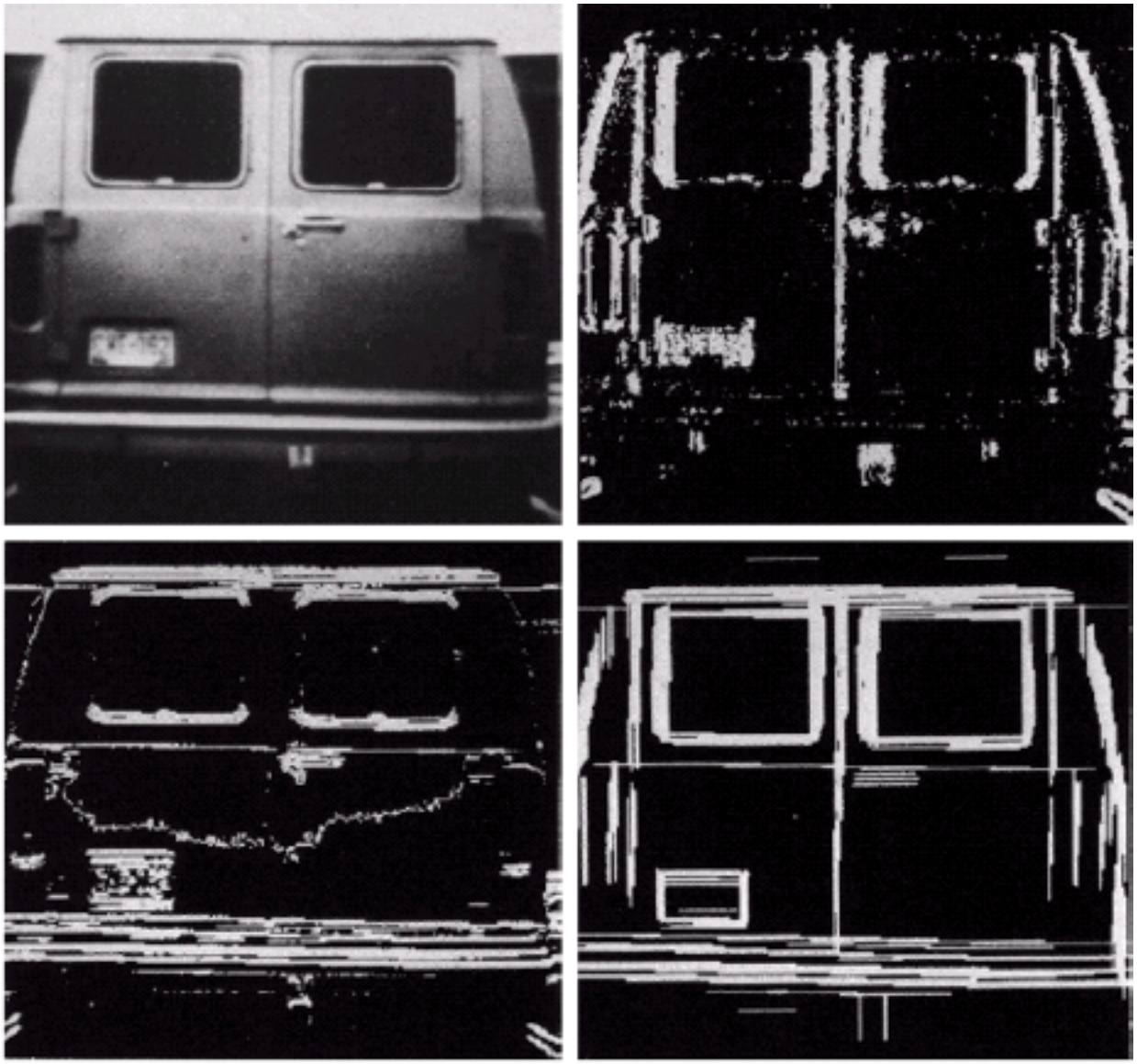
or simply $G = G_x + G_y$) or max (Gx,Gy)

  - if magnitude is above a certain threshold, the pixel is considered to be an edge
- but a further decision needs to be made
  - whether two pixels that have been identified as edges belong to the same contour/frontier
- Often, the different pixels that have been identified as edges, do not define a unified boundary/frontier due to
  - noise in the picture, breaks in the boundary caused by occlusion, etc
  - consequently the edge detection operation is followed by a linking operation to assemble edge pixels into uniform boundaries

Discontinuity detection - edge linking

- 2 alternative strategies can be used
  - Local, comparing magnitude and orientation of the gradient of both pixels
  - Global, using the Hough transform

Discontinuity detection - edge linking (2)



Discontinuity detection - second derivative

- The Laplacian operator highlights regions in an image of rapid intensity change by calculating the second spatial derivative

$$L(x,y) = \delta^2 f/\delta x^2 + \delta^2 f/\delta y^2$$

  - the operator is normally applied to a gray scale image producing another gray scale image as output

- The calculation of the second derivatives of an image can be approximate by standard convolutional filters represented by their kernels/masks as for the gradient case

- Two commonly used small kernels using a negative peak are

0	-1	0
-1	4	-1
0	-1	0

-1	-1	-1
-1	8	-1
-1	-1	-1

- Using one of these kernels, the Laplacian can be calculated using standard convolution methods

Discontinuity detection - second derivative (2)

- kernels that approximate a second derivative measurement on the image are very sensitive to noise (they detect "instantaneous" transitions)
- as such the image is often filtered (Gaussian filter) to reduce high frequency noise, prior to applying the Laplacian masks
  - given that spatial filtering can be implemented through convolution and because convolution is associative
    - the Gaussian smoothing filter is first convolved with the Laplacian filter and only then this hybrid filter is convolved with the image
      - the Laplacian of Gaussian approach, LoG

Discontinuity detection - LoG operator

- Given that both the Gaussian and the Laplacian kernels are usually much smaller than the image, this method leads to fewer arithmetic operations
- The LoG kernel can be calculated beforehand
  - thus, only one convolution needs to be performed at run-time on the image
- the usual 2-D LoG function is centred on zero with Gaussian standard deviation σ has the form:

$$LoG(x,y) = -\frac{1}{\pi\sigma^4}\left[1 - \frac{x^2+y^2}{2\sigma^2}\right]e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Discontinuity detection - LoG operator (2)

- discrete kernel that approximates this function (using a Gaussian filter with σ = 1.4)

0	1	1	2	2	2	1	1	0
1	2	4	5	5	5	4	2	1
1	4	5	3	0	3	5	4	1
2	5	3	-12	-24	-12	3	5	2
2	5	0	-24	-40	-24	0	5	2
2	5	3	-12	-24	-12	3	5	2
1	4	5	3	0	3	5	4	1
1	2	4	5	5	5	4	2	1
0	1	1	2	2	2	1	1	0

- when the Gaussian is made increasingly narrower, the LoG kernel becomes the same as the simple Laplacian kernels shown previously
  - because smoothing a discrete grid/matrix with a very narrow Gaussian (σ < 0.5 pixels) has no effect
    - the simple Laplacian can be seen as a limiting case of the LoG for narrow Gaussians

Discontinuity detection - kernels for convolution

- Different kernels/masks may be used with different aims

-1	-1	-1
2	2	2
-1	-1	-1

horizontal

-1	-1	2
-1	2	-1
2	-1	-1

+45°

-1	2	-1
-1	2	-1
-1	2	-1

vertical

2	-1	-1
-1	2	-1
-1	-1	2

-45°

- the above represented masks are suitable for detecting bright lines above a dark background
  - for dark lines above a bright background, symmetrical values should be used
  - to detect both types, then the absolute values of the different convolutions' results should be used

-2	-1	0
-1	0	1
0	1	2

+45°

0	1	2
-1	0	1
-2	-1	0

-45°

Discontinuity detection - kernels for convolution (2)

- line detection example:

diagonal mask

0	-1	0
-1	4	-1
0	-1	0

-1	-1	-1
-1	8	-1
-1	-1	-1

Laplacian filter

Discontinuity detection - kernels for convolution (3)

Original image

Prewitt

Canny

Sobel

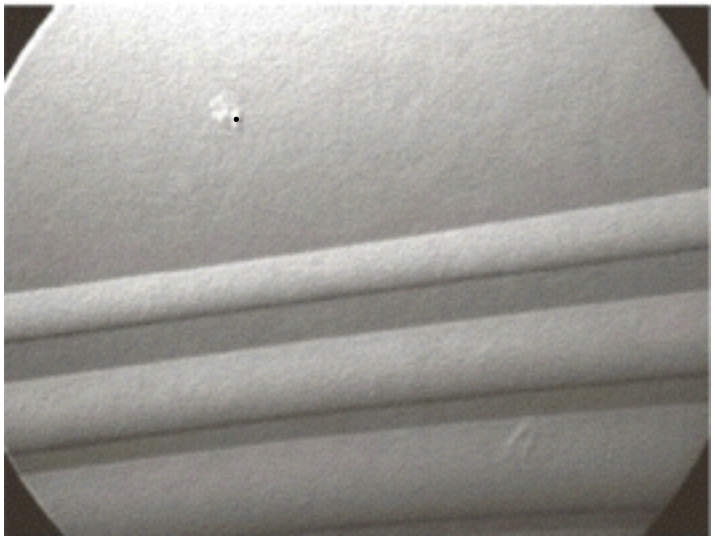
Log

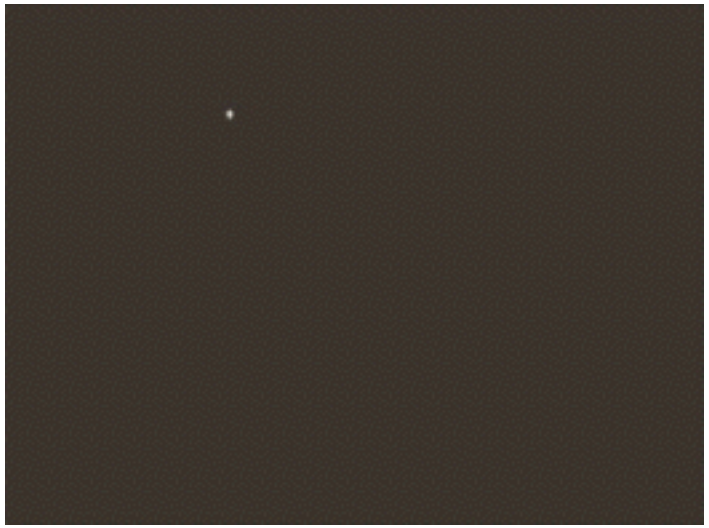
Discontinuity detection - kernels for convolution (4)

- detecting isolated points

$$f(x-1,y-1) + f(x-1,y) + f(x-1,y+1) + f(x,y-1) - 8f(x,y) + f(x,y+1) + f(x+1,y-1) + f(x+1,y) + f(x+1,y+1)$$

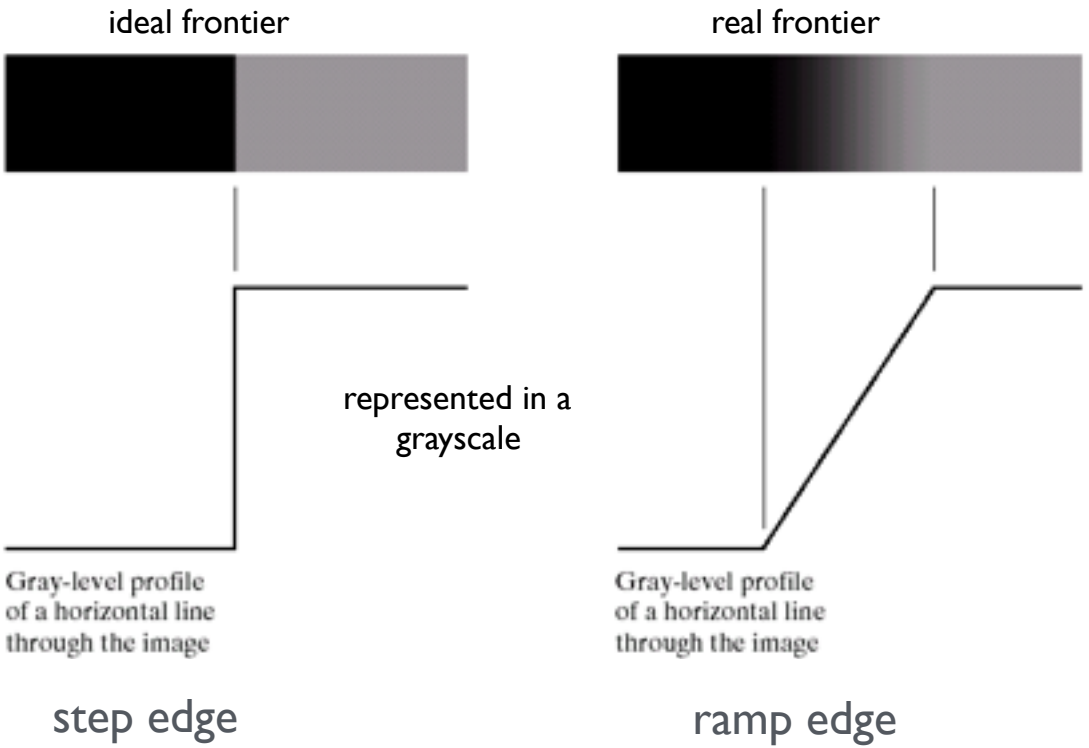
1	1	1
1	-8	1
1	1	1





Discontinuity detection - challenges

- frontiers between objects in real-world images are usually not abrupt but rather gradual in terms of brightness
  - may be noisy or naturally smooth



Discontinuity detection - challenges (2)

- 4 different types of edges are normally considered

step

the intensity change is instantaneous (ideal)

ramp

the intensity change is not instantaneous but rather occurs over a finite distance

ridge / line

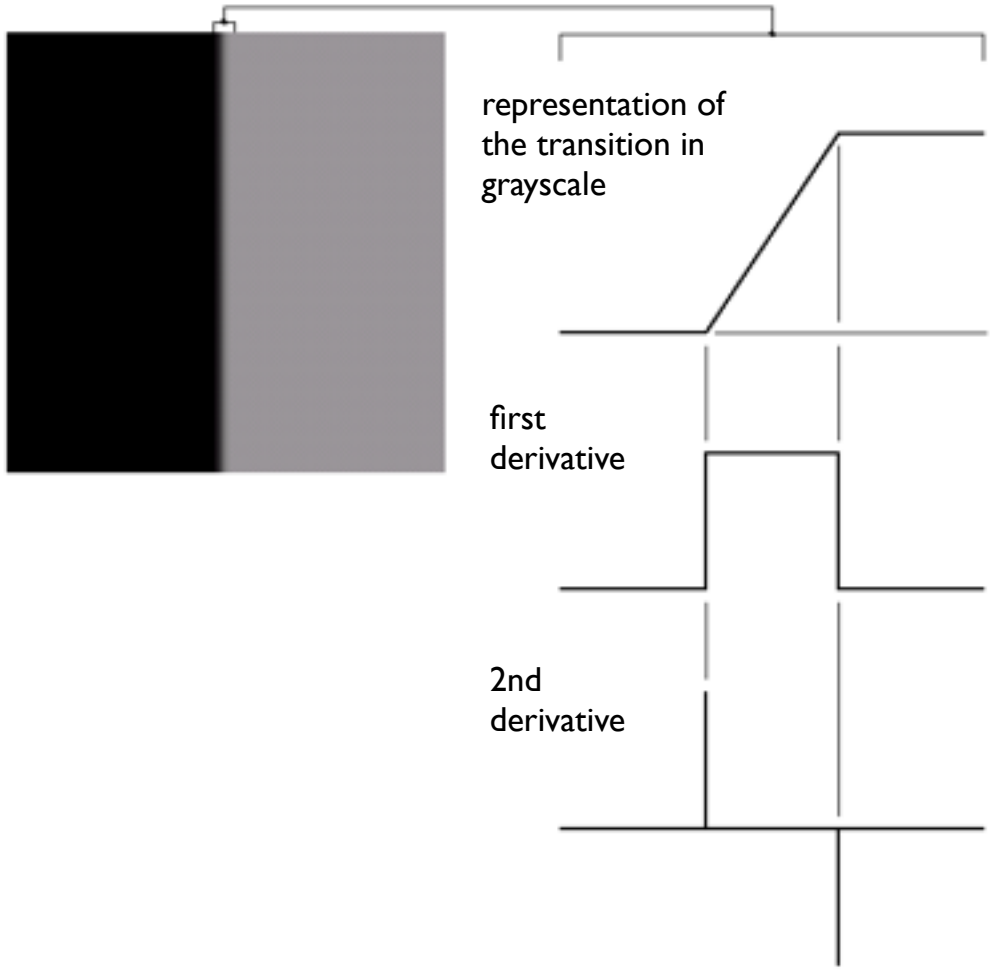
the intensity change is instantaneous and within a short distance it returns to its previous value

roof

the intensity change occurs over a finite distance and then it returns back to its previous value over a finite distance

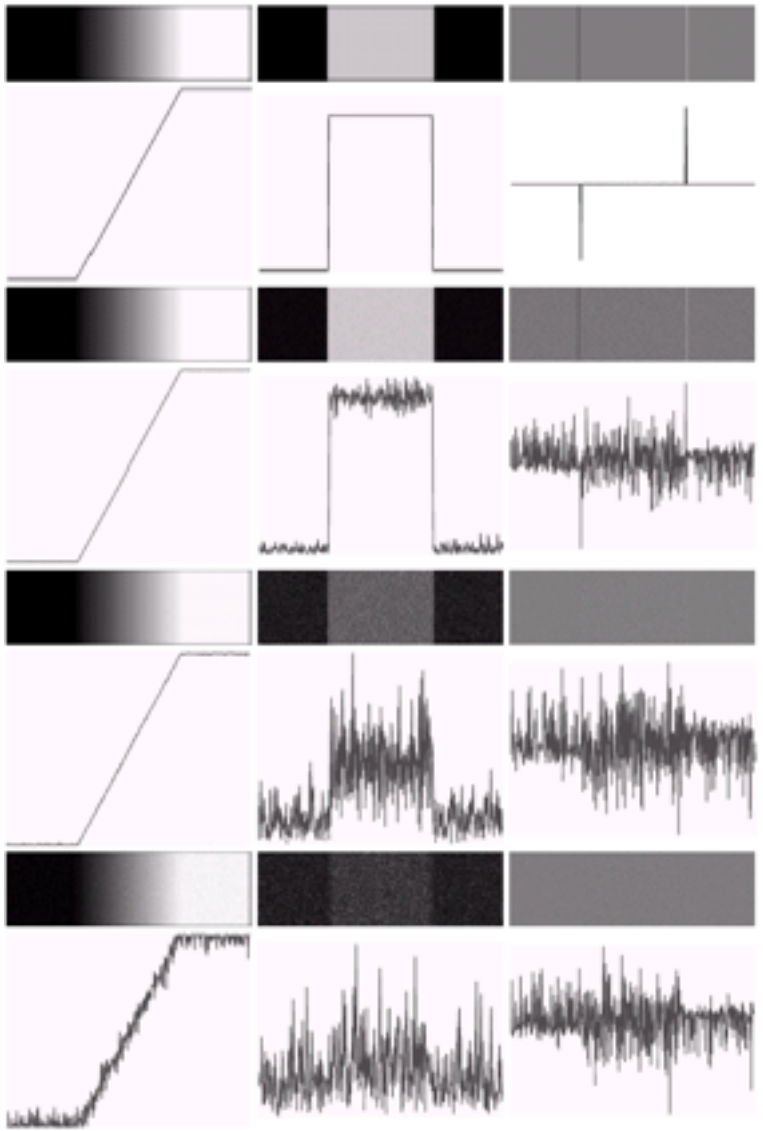
Discontinuity detection - challenges (3)

- it is easier to calculate the derivatives using a grayscale representation



discontinuity detection - challenges (4)

- the values of the derivatives may become too noisy to enable distinguishing an edge
  - thus not allowing a decision to be taken
- the solution passes by performing a pre low-pass filtering to eliminate noise in the image
  - needs to be performed in such a way as not to eliminate real edges



Detection by similarity (*clustering*)

- base concept is to identify regions in an image, to partition or group regions according to common image properties
- a region is defined as
  - an homogeneous part of the image in relation to some characteristic such as
    - intensity levels in a try scale
    - color values
    - texture
- basically the idea is to group/cluster pixels that are similar or that have similar statistical properties
  - and/or separate pixels that are dissimilar

Detection by similarity (*clustering*) (2)

- the inside of a region must be simple without many "holes" even if small in size ("hole" ≡ small aggregations of pixels that do not share the same homogeneous characteristic as the other pixels in the region)
- regions that are labelled as disjoint in a segmentation by similarity process, must present significantly distinct in relation to the characteristic in which one of them is homogeneous

Detection by similarity - binarization

- An import step in this segmentation approach is the binarization of the image
  - identification of pixels that are in the foreground and thus belong to objects from those that are in the background
    - it can be seen as a basic way of segmenting the image in only two groups
      - a black background and white objects in the foreground
  - binarization by threshold is one of the most common ways of achieving it
- the aim in binarization is to group pixels in only two classes, Q_1 and Q_2 (the background B and the foreground F)
  - with respect to some similar characteristic
    - necessary to find the value T that distinguishes those classes

Detection by similarity - binarization (2)

- there are three possible strategies to implement threshold binarization
  - global
    - uses only one threshold value for all pixels depending only on the overall intensity values
  - local
    - divides the image in several sub-images, applying a specific threshold value for each sub-image depending on their own characteristics
  - dynamic
    - the threshold value depends on the pixel position

Binarization by global thresholding

- regions/objects are separated using the pixel intensity
  - Global simple
    - one color is assigned to the background and another unique color to all the objects in the foreground

$$g(x,y)=\begin{cases}0 & se \quad f(x,y)<T\\ 1 & se \quad f(x,y)\geq T\end{cases}$$

  - Global multiple
    - one color is assigned to the background and different colours to each object in the foreground

$$g(x,y)=\begin{cases}0 & se \quad f(x,y)<T_1\\ 1 & se \quad T_1\leq f(x,y)<T_2\\ 2 & se \quad T_2\leq f(x,y)<T\\ \ldots\end{cases}$$

General approach to Global Binarization

1. estimate an initial global threshold T
2. segment the image by binarization using T
   - two groups are obtained: G1 with pixels > T an G2 with pixels < = T
3. calculate the average intensity for each group m1 and m2
4. calculate new T = (m1+m2)/2
5. repete 2 to 4, until the T between successive interactions is below a defined ΔT

Note 2: Matlab has an implementation for binarization in the function imbinarization()

General approach to Global Binarization (2)

- the threshold *T* is often estimated based on the average background intensity value
- which can can be identified by several means
  - through the histogram
    - in principle the background is the largest region in the image so, it will correspond to the largest number of occurrences in the histogram
  - by subtracting two consecutive, or close in time, images in a video sequence
  - by the texture or color
  - etc

Note: a frequently used approach is the Otsu algorithm (next slides); Matlab has implementations for computing the threshold: graythres(), otsuthresh(), adaptthresh()

Automated approaches to estimate the threshold

- five ways of using the image histogram and identifying two distinctive modes, one for the background and one for the foreground

  (1) image characteristics known - cumulative histogram

  - if the objects in the foreground occupy a fraction $1/p$ of the image and are brighter than the background

$$c(a) = \sum_{a=0}^{A} p(a)$$ *T* is chosen such that $c(T) = 1/p$

```
//input the value of p and the image dimensions NxM
for (i=0;i<M,i++)
for (j=0;j<N,j++)
 hist[I[i,j]]++;
// in MATLAB: imhist(I)
sum=0; a=0;
while(sum<(N*M)/p && a<256)
  { sum+=hist(a); a++; }
T = a-1; // desired threshold
```

  - if the objects are darker than the background, choose T such that $c(T) = 1 - 1/p$

Automated approaches to estimate the threshold (2)

(2) finding peaks and valleys in the histogram

- find the local maxima and then the minimum between them
  - because there may be noise, first the histogram is smoothed

(3) clustering by K-means variation (next slide)

Automated approaches to estimate the threshold (3)

(3) clustering by K-means variation

- starts by defining a threshold T and group pixels in 2 clusters
  - T is set to be equidistant to the averages of the intensities of the background and the foreground, respectively μ_B and μ_F
    - the background is assumed as being formed only by the four corner pixels, everything else being part of the foreground
- then a recursive cycle is initiated to minimize the distance between each pixel in a cluster and the average of that cluster
  - moving the pixel when necessary and recomputing the cluster intensity averages as well as the threshold
- so the basic idea is to cluster pixels according to how close their intensities are from the clusters' average intensities, μ_B and μ_F

$$\forall p \geq T : |p-\mu B(T)| > |p-\mu F(T)| \quad \&\& \quad \forall p < T : |p-\mu B(T)| < |p-\mu F(T)|$$

Automated approaches to estimate the threshold (4)

(4) clustering by the Otsu method

- the aim in binarization is to group pixels in only two classes, Q_1 and Q_2 obtaining the best value T that distinguishes those classes
- the Otsu algorithm selects T trying to minimize the variance inside each class (*intra-class*)
  - probabilities $q_1(t) = \sum_{i=1}^{t} P(x)$ $\quad q_2(t) = \sum_{i=t+1}^{I} P(x)$
  - variances $\sigma_1^2(t) = \sum_{i=1}^{t} [i - \mu_1(t)]^2 \frac{P(i)}{q_1(t)}$ $\quad \sigma_2^2(t) = \sum_{i=t+1}^{I} [i - \mu_2(t)]^2 \frac{iP(i)}{q_2(t)}$
  - averages $\mu_1(t) = \sum_{i=1}^{t} \frac{iP(i)}{q_1(t)}$ $\quad \mu_2(t) = \sum_{i=t+1}^{I} \frac{iP(i)}{q_2(t)}$
- which is equivalent to maximising the inter-class variance

$$\sigma_b^2 = \sigma^2 - \sigma_w^2(t) = q_1(t)[1 - q_1(t)][\mu_1(t) - \mu_2(t)]^2$$

$$T: \textit{máx}\ \sigma_b^2(t) = \sigma_b^2(T)$$

Automated approaches to estimate the threshold (4-2)

(4) clustering by the Otsu method (cont.)

- for each new computed value of T, $T: máx\ \sigma_b^2(t) = \sigma_b^2(T)$
  - separate the pixels into two clusters according to T
  - find the mean of each cluster
  - square the difference between the means
  - multiply by the number of pixels in one cluster times the number of pixels in the other (or the probabilities)
  - compute new threshold

$$n_B(T+1) = n_B(T) + n_T$$

$$n_F(T+1) = n_F(T) - n_T$$

$$\mu_B(T+1) = \frac{\mu_B(T)n_B(T) + n_T\,T}{n_B(T+1)}$$

$$\mu_F(T+1) = \frac{\mu_F(T)n_F(T) - n_T\,T}{n_F(T+1)}$$

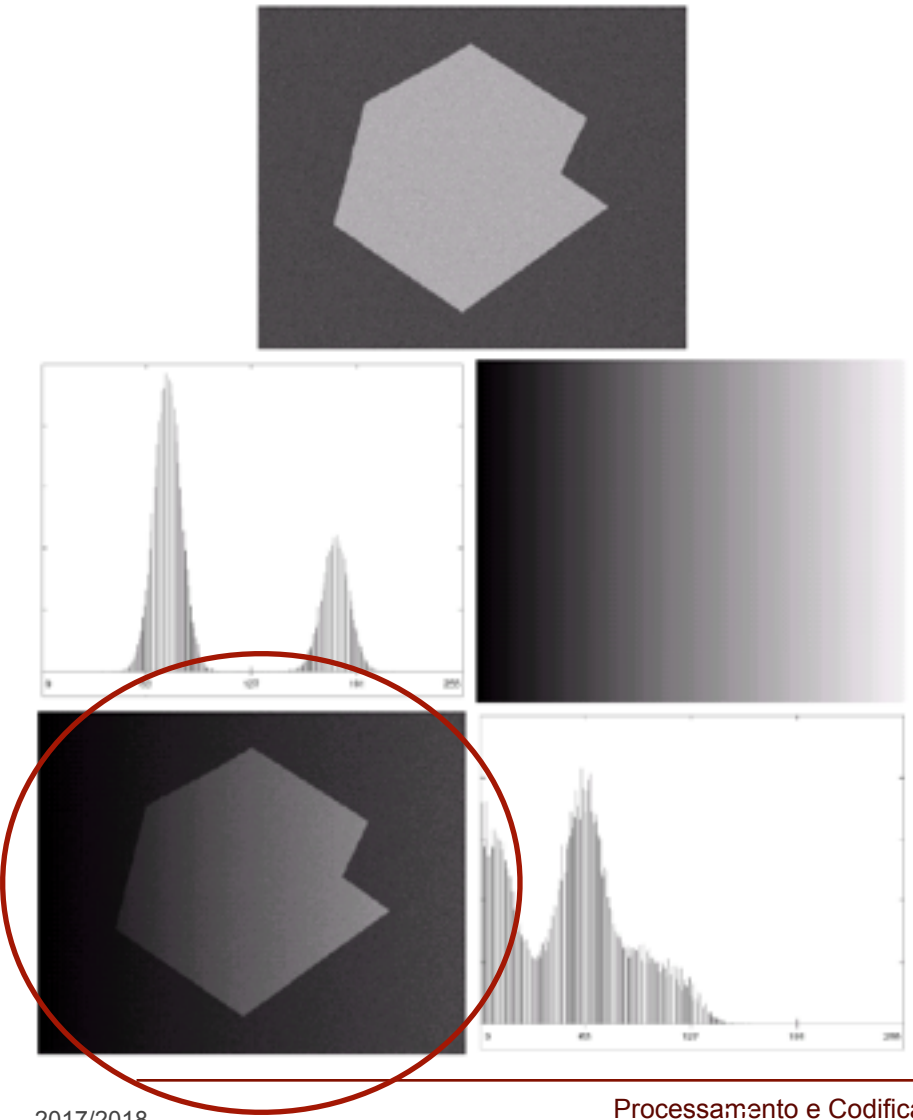
Automated approaches to estimate the threshold (5)

(5) Mixture Modelling

- assumes that the two classes - background and foreground - are Gaussian-distributed with certain mean and standard deviation, μ and σ, which are independent of the selected threshold
  - requires to compute an initial estimate of the Gaussian distributions (of the quantities n_B, n_F, μ_B, μ_F, σ_B, and σ_F)
  - the optimal threshold is the one that causes the mixture of the two estimated Gaussian distributions to best approximate the actual image histogram

Challenges in Global binarization

- local variations in light intensity (originated by shadows, etc)



- in this case, a global threshold is not adequate
  - a possible solution is to pre-process the image to compensate such lightning variations
  - equivalent to using a local thresholding technique in which the threshold value varies according to the position in the image

Challenges in binarization/thresholding

- only the individual pixel intensity is considered
  - not any relationship between pixels
  - no guarantee that the pixels identified by the thresholding process are contiguous (connected/adjacent)
- extraneous pixels are often included in regions to which they do not belong
  - on the other hand, especially near the boundaries of a region, isolated pixels that belong to the region may be missed
- in the presence of noise these undesired effects are aggravated
  - because there will be a higher probability that the pixel intensity is wrong (noisy), moving away from the normal intensity in the region
- lightning conditions/shadows also aggravate these errors

Challenges in binarization/thresholding (2)

- when using thresholding it is necessary to estimate the value of the threshold to distinguish between classes
  - it may be necessary to "play" with different values
    - due to noise, lightning conditions but also due to the actual pixel intensities and real similarity of values between pixels in the foreground and background
    - some values may lead to excessive loss of pixels in a region, others to the inclusion of extraneous background pixels in the object/foreground

Region-based segmentation approaches

- region growing
  - the process starts with the selection of pixel "seeds", to which more pixels are added when certain established criteria is observed
    - the criteria is specific to each problem
      - it can be (again) the simplest property that pixels in a region share - the intensity or the color
      - or higher order statistics of the region
    - necessary to define a stop criterium
    - necessary a careful selection of "seeds" as it may influence the final result as well as the processing delay

Region-based segmentation approaches (2)

- region growing
  - it may start considering the image as a single region, thus with only one seed
    - in which case it is normally the upper left corner
  - or with a set of regions represented by a set of seed points
  - analyses the values of neighbour pixels
    - those considered connected/adjacent are added to the region, making the region to grow
    - if a pixel is not connected it becomes the seed of a new region and the analysis process re-starts
  - the condition for pixel to be added to a region could be
    - neighbour pixels that are connected/adjacent due to similar values on average and/or variance, standard deviation

Region-based segmentation approaches (3)

- Split-and-Merge
  - it is a recursive method with an initial split step and a final merging step
  - the first step starts by dividing randomly the image into a random number of regions
    - then, each of those regions is recursively analysed and divided until homogeneous regions are obtained
      - according to some specified criteria
  - in the second step, contiguous regions, identified in the first step as being disjoint, are analysed and merged if together they form an homogeneous region
    - according to some specified criteria

References

- https://courses.cs.washington.edu/courses/cse576/book/ch6.pdf
- http://www.uio.no/studier/emner/matnat/ifi/INF4300/h09/undervisningsmateriale/hough09.pdf
- https://web.fe.up.pt/~campilho/PDI/NOTES/EdgeDetection.pdf
- http://www.inf.ufg.br/~fabrizzio/mestrado/pdi/aulas/aula10.pdf
- http://gec.di.uminho.pt/lesi/vpc0304/Aula07Segmentação.pdf
- http://iris.sel.eesc.usp.br/sel886/Aula_6.pdf
- http://mtc-m18.sid.inpe.br/col/sid.inpe.br/mtc-m18%4080/2010/06.22.18.13/doc/106003.pdf?metadatarepository=&mirror=iconet.com.br/banon/2005/09.28.12.40
- https://www.mathworks.com/help/images/functionlist.html