Image segmentation

Image analysis:

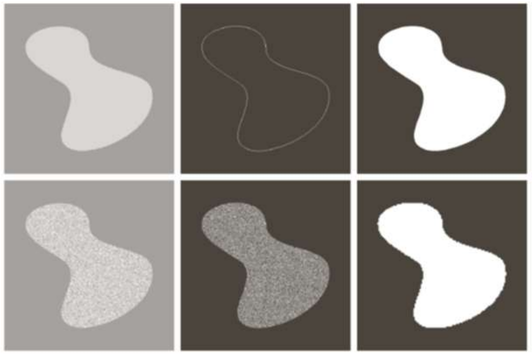
First step: **Segmentation**, i.e. subdivision of the image into its constituent parts or objects. Autonomous segmentation is one of the most difficult tasks in image processing!

Segmentation algorithms are based on two basic properties of gray-level values:

• **Discontinuity**: the image is partitioned based on *abrupt changes* in gray level. Main approach is **edge detection**.

• **Similarity**: the image is partitioned into *homogeneous* regions. Main approaches are **thresholding**, **region growing**, and **region splitting and merging**.

Toy problems & kids problems



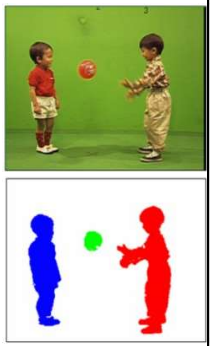


FIGURE 10.1 (a) Image containing a region of constant intensity. (b) Image showing the boundary of the inner region, obtained from intensity discontinuities. (c) Result of segmenting the image into two regions. (d) Image containing a textured region. (e) Result of edge computations. Note the large number of small edges that are connected to the original boundary, making it difficult to find a unique boundary using only edge information. (f) Result of segmentation based on region properties.

Image segmentation: discontinuities

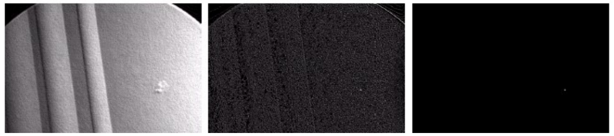
3 basic types of discontinuities in digital images: **Points, Lines, Edges**.

SNR-optimal *linear* filter in i.i.d. Gaussian noise: matched filter, a.k.a. template matching, a.k.a. cross-correlation approach

Point detection

e.g. detect a tiny hole in a turbine blade (dark pixel within the bright zone below)

–1	–1	–1
–1	8	–1
–1	–1	–1



a
b c d

FIGURE 10.2 (a) Point detection mask. (b) X-ray image of a turbine blade with a porosity. (c) Result of point detection. (d) Result of using Eq. (10.1-2). (Original image courtesy of X-TEK Systems

(c): $g = |filter(f)|$ (d): $|g|>T$, with $T= 0.9*max(|g|)$

Image segmentation: discontinuities

Thin line detection

The output of the convolution will be stronger where a one-pixel-wide line is present in the corresponding direction.

Note: zero-sum masks (~ second-order directional derivative)

–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

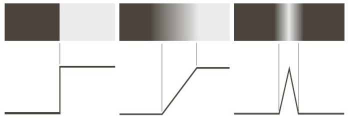
2	–1	–1
–1	2	–1
–1	–1	2

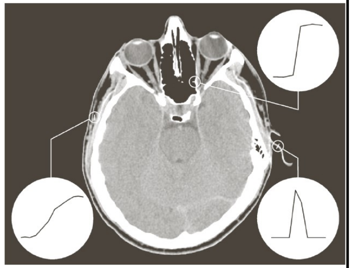
–45°

Image segmentation: discontinuities

Edge: boundary between two regions with significantly distinct gray levels

Edge models, (also for *roof* edge):





Typical real-world problems:

- edges with different slopes
- objects with different sizes
  → *scale-space* operators?
- uneven illumination
- not significant (?) details
- noise

Gianni Ramponi
http://www.units.it/ramponi
Digital Image Processing

Image segmentation: discontinuities

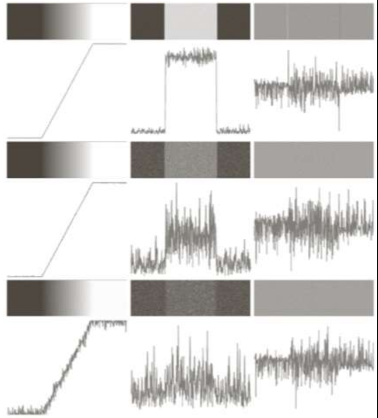
1-D case

$$\frac{\partial f}{\partial x} = f(x+1) - f(x)$$

First-order derivative has nonzero phase response

$$\frac{\partial^2 f}{\partial x^2} = [f(x+1) - f(x)] - [f(x) - f(x-1)]$$
$$= f(x+1) + f(x-1) - 2f(x)$$

Ideal ramp edge plus noise having std = 0.1, 1, 10 gray levels (out of 256); first- and second-order derivatives →



Gianni Ramponi
http://www.units.it/ramponi
Digital Image Processing

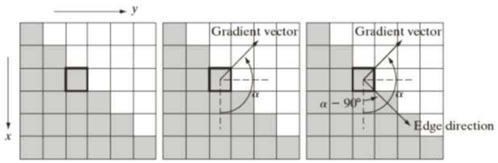
Image segmentation: discontinuities

2-D case

Gradient

$$\nabla \mathbf{f} = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

$$|\nabla \mathbf{f}| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2} \; ; \quad \alpha = \tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$$



−1
1

−1	1

−1	0		0	−1
0	1		1	0

Roberts

−1	−2	−1		−1	0	1
0	0	0		−2	0	2
1	2	1		−1	0	1

0	1	2		−2	−1	0
−1	0	1		−1	0	1
−2	−1	0		0	1	2

Sobel

Laplacian $\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$

Sobel H/V or 45/135 deg.

Gianni Ramponi
http://www.units.it/ramponi
Digital Image Processing

Image segmentation: discontinuities

a b
c d

FIGURE 10.16 (a) Original image of size 834 × 1114 pixels, with intensity values scaled to the range [0, 1]. (b) $|g_x|$, the component of the gradient in the x-direction, obtained using the Sobel mask in Fig. 10.14(f) to filter the image. (c) $|g_y|$, obtained using the mask in Fig. 10.14(g). (d) The gradient image, $|g_x| + |g_y|$.

Image segmentation: discontinuities

a b
c d

FIGURE 10.18 Same sequence as in Fig. 10.16, but with the original image smoothed using a 5×5 averaging filter prior to edge detection.

Image segmentation: discontinuities

Edges from diagonal masks

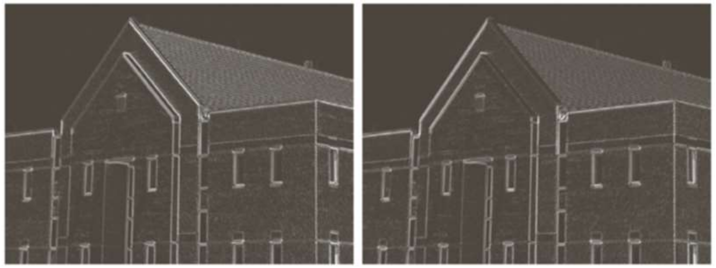
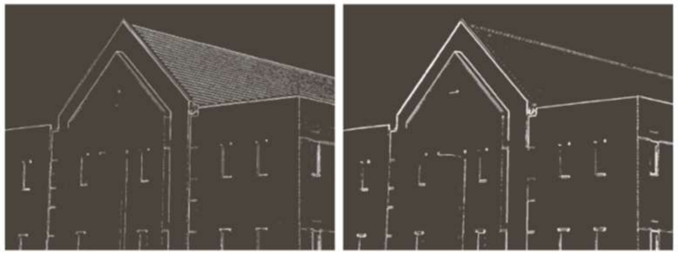


Image segmentation: discontinuities

Thresholded images



a b

FIGURE 10.20 (a) Thresholded version of the image in Fig. 10.16(d), with the threshold selected as 33% of the highest value in the image; this threshold was just high enough to eliminate most of the brick edges in the gradient image. (b) Thresholded version of the image in Fig. 10.18(d), obtained using a threshold equal to 33% of the highest value in that image.

Image segmentation: discontinuities

Edge detection based on **zero crossings of second-order derivative** (*Marr-Hildreth operator*)

Standard implementation of a Laplacian:

0	−1	0
−1	4	−1
0	−1	0

−1	−1	−1
−1	8	−1
−1	−1	−1

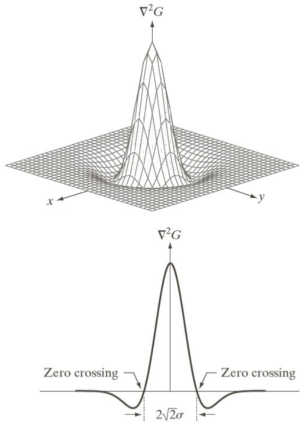
- Its magnitude produces double edges
- Unable to detect the edge direction
- Very sensitive to noise → use Laplacian of Gaussian instead:

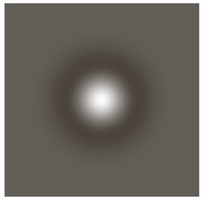
$$G(r) = \exp(-r^2/2\sigma^2)\,; \qquad r^2 = x^2 + y^2\,, \quad -K \le x, y \le K$$

$$\nabla G(r) = \left(\frac{-r}{\sigma^2}\right)\exp(-r^2/2\sigma^2)\,; \qquad \nabla^2 G(r) = -\left(\frac{r^2-\sigma^2}{\sigma^2}\right)\exp(-r^2/2\sigma^2)$$

Image segmentation: discontinuities

"Mexican hat"





0	0	−1	0	0
0	−1	−2	−1	0
−1	−2	16	−2	−1
0	−1	−2	−1	0
0	0	−1	0	0

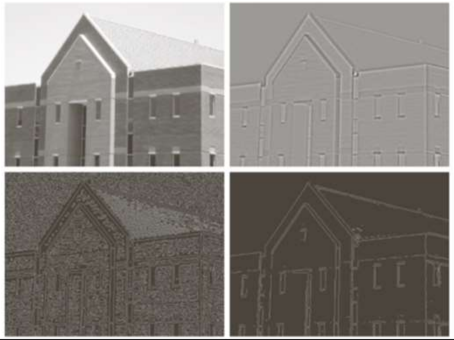
a b
c d

FIGURE 10.21 (a) Three-dimensional plot of the *negative* of the LoG. (b) Negative of the LoG displayed as an image. (c) Cross section of (a) showing zero crossings. (d) 5 × 5 mask approximation to the shape in (a).

Image segmentation: discontinuities

To detect the zero crossings of the LoG image:

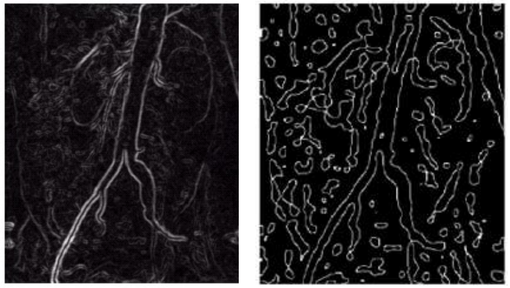
- center a 3x3 mask on each pixel $p(x,y)$ of the LoG image
- check all pairs (p_1,p_2) of opposite neighbors (l/r, u/d, 45, 135)
- p is edge if at least in one pair pixels have different sign
- to reduce "*noise*", consider only {p: abs(p_1-p_2)>thresh.}



a b
c d

FIGURE 10.22 (a) Original image of size 834 × 1114 pixels, with intensity values scaled to the range [0, 1]. (b) Results of Steps 1 and 2 of the Marr-Hildreth algorithm using $\sigma = 4$ and $n = 25$. (c) Zero crossings of (b) using a threshold of 0 (note the closed-loop edges). (d) Zero crossings found using a threshold equal to 4% of the maximum value of the image in (b). Note the thin edges.

Image segmentation: discontinuities



Comparison between Sobel (before thresholding) and zero crossings of LoG

Edges in LoG are thinner and tend to form loops; objects size is altered

Quality criteria

- small false alarm rate, small missed detection rate
- good localization
- one-pixel-wide edges

→ **Canny** edge detector

Image segmentation: discontinuities

Canny operator

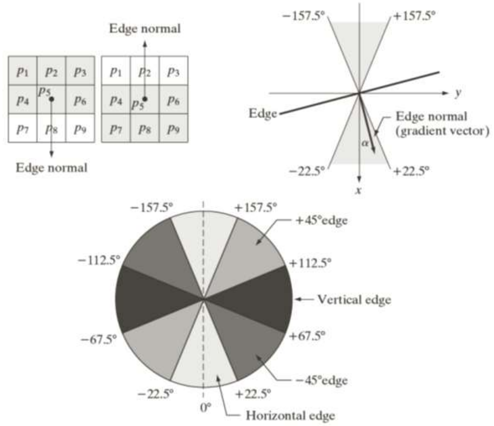
- A good approximation of an ideal detector for *1-D noisy step-edges* is the derivative of the Gaussian

$$\nabla G(x) = \left(\frac{-x}{\sigma^2}\right)\exp(-x^2/2\sigma^2)$$

- In 2D, it should be applied orthogonally to the edge
  → circularly symmetric lowpass Gaussian filter, followed by computation of the gradient
- $|G(x,y)|$ shows thick patterns
  → non-maxima suppression:
    1. determine the quantized direction d_k of the gradient
    2. if $|G(x,y)|$ < at least one of its neighbours along d_k then set it to zero
- Apply a threshold to reduce false alarms

Image segmentation: discontinuities

Canny operator: quantized edge directions



a b
c

FIGURE 10.24 (a) Two possible orientations of a horizontal edge (in gray) in a 3 × 3 neighborhood. (b) Range of values (in gray) of α, the direction angle of the *edge normal*, for a horizontal edge. (c) The angle ranges of the edge normals for the four types of edge directions in a 3 × 3 neighborhood. Each edge direction has two ranges, shown in corresponding shades of gray.

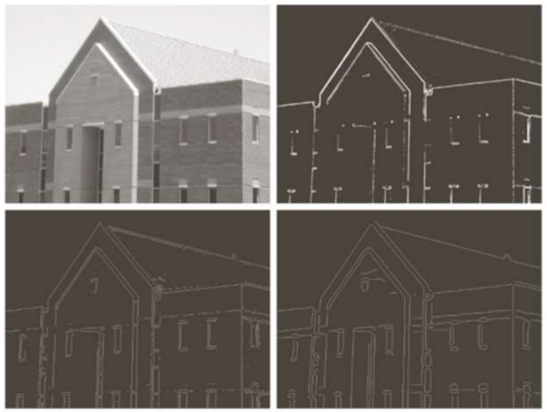
Image segmentation: discontinuities

Canny operator: hysteresis thresholding reduces both false alarms and missed detections

1. Set two thresholds: T_L and T_H , with $T_H \cong 3\ T_L$
2. Generate two binary images:
   $G_H = |G(x,y)| > T_H$ (strong edges)
   $G_L = |G(x,y)| > T_L$ (strong and weak edges)
3. Eliminate from G_L all strong edge pixels (pixels that are nonzero in G_H): $G_L = G_L - G_H$
4. Label all pixels in G_H as edge
5. Fill *edge gaps*:
   a. Visit each nonzero pixel p in G_H and mark as edge all pixels in G_L that are 8-connected to p
   b. Reset all unmarked pixels in G_L
   c. Final edge image = $G_H + G_L$

Image segmentation: discontinuities

$T_L = 0.04$, $T_H = 0.10$ (normalized pixel values)
$\sigma = 4$, mask size =25x25

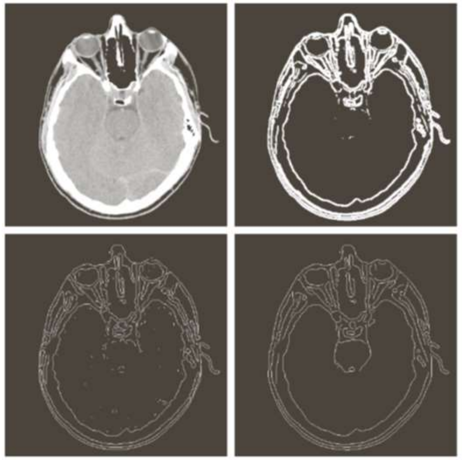


a b
c d

FIGURE 10.25 (a) Original image of size 834 × 1114 pixels, with intensity values scaled to the range [0, 1]. (b) Thresholded gradient of smoothed image. (c) Image obtained using the Marr-Hildreth algorithm. (d) Image obtained using the Canny algorithm. Note the significant improvement of the Canny image compared to the other two.

Image segmentation: discontinuities

$T_L = 0.05$,
$T_H = 0.15$
$\sigma = 2$,
mask size =13x13



a b
c d

FIGURE 10.26 (a) Original head CT image of size 512 × 512 pixels, with intensity values scaled to the range [0, 1]. (b) Thresholded gradient of smoothed image. (c) Image obtained using the Marr-Hildreth algorithm. (d) Image obtained using the Canny algorithm. (Original image courtesy of Dr. David R. Pickens, Vanderbilt University.)

Image segmentation: edge linking

- *local processing: image points with similar gradient*

Join the detected edge pixels according to their similarity (e.g., similar *amplitude* and *direction* of the gradient), and form a boundary

E.g.: looking for rectangles

- calculate image gradient $G(x,y)$
- scan $G(x,y)$ along rows and build binary edge image, setting pixels where $|G| > K\%\ |G|_{max}$ and $G_{angle} = \pm 90 \pm \delta$ deg.
- re-scan by rows and fill gaps shorter than L
- do the same by columns, $G_{angle} = 0 \pm \delta$, or $180 \pm \delta$ deg.
- add the two resulting images

Image segmentation: edge linking

K = 30
δ = 45 deg.
L = 25 px.

→ *Note detected license plate*

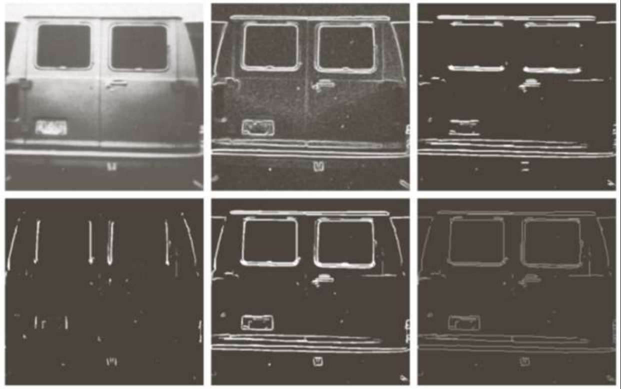


FIGURE 10.27 (a) A 534 × 566 image of the rear of a vehicle. (b) Gradient magnitude image. (c) Horizontally connected edge pixels. (d) Vertically connected edge pixels. (e) The logical OR of the two preceding images. (f) Final result obtained using morphological thinning. (Original image courtesy of Perceptics Corporation.)

Image segmentation: edge linking

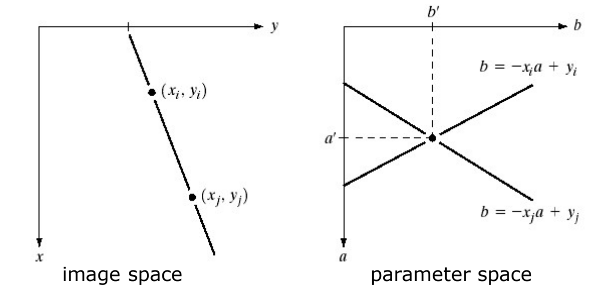
- *global processing: the Hough transform*

- A more efficient method to detect straight lines
- Can be generalized to curves

1. *Generic* line through a point in the image (x_i, y_i): $y_i = ax_i + b$
2. In the parameters space (a,b), (x_i, y_i) define a line $b = -x_i a + y_i$
3. Take a second point in the *image* along the *same* generic line; its representation in the parameters space is:
$$(x_j, y_j): \quad y_j = ax_j + b \qquad \Rightarrow \qquad b = -x_j a + y_j$$
4. Let (a', b') be the coordinates at which the two lines intercept in the parameters space
5. a', b' are the *slope* and *intercept* of the *specific* line through $(x_i, y_i), (x_j, y_j)$
6. All points located on such a line in the image plane have lines in the parameters space which intersect at (a', b') [indeed, the line in the image plane sets a well-defined pair of (slope,intersect) values]

Image segmentation: edge linking

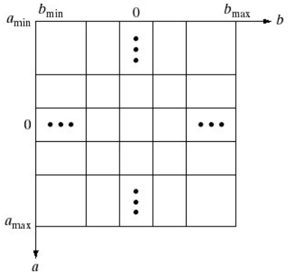
- *global processing: the Hough transform*



6. All points located on such a line in the image plane have lines in the parameters space which intersect at (a', b') [indeed, the line in the image plane sets a well-defined pair of (slope,intersect) values]

Image segmentation: edge linking

- *global processing: the Hough transform*




```
Subdivide the parameters space into a
    matrix A(a,b) of cells and reset it
For each edge point (x_i, y_i) in the image
    For each value of a
        solve  b = -x_i a + y_i
        increment A(a,b)
```

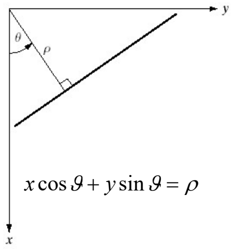
Each run of the inner loop plots a line in *A* that corresponds to an edge point

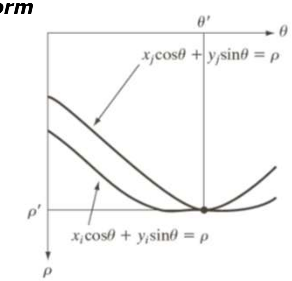
The value *Q* in *A(a,b)* indicates that *Q* edge points in the image lie on a line of slope *a* and intersect *b*
→ Bright points in *A* show the parameters of the main edges in the image.

Problem: vertical edges are difficult to represent since *a* tends to infinity
Solution: use the *normal form* (trigonometric form) of a line:

Image segmentation: edge linking

- *global processing: the Hough transform*



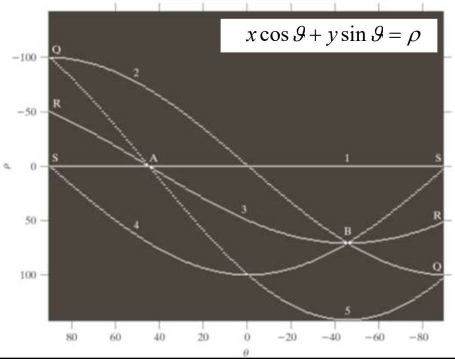


Points in the image space now define a sinusoid in the parameters space
Let (x_i, y_i), (x_j, y_j) define two sinusoids that intersect in (ρ', θ') → these are the parameters of a line through (x_i, y_i), (x_j, y_j) in the image

```
Define a matrix A(ρ,θ) and reset it.
    For each edge point in the image
        For each value of θ; find ρ; increment A(ρ,θ)
```

→ Bright points in *A* show the parameters of the main edges in the image

Image of size 101x101; origin in pt.1;
pt.2: 0 + 100 sinθ = ρ
pt.4: 100 cosθ + 0 = ρ
pt.3: 50 cosθ + 50 sinθ = ρ



Points 1,…5 in (*x,y*) are mapped to sinusoids with amplitude ρ

1,3,5 are aligned; sinusoids intersect at (0,45) (A)

2,3,4 are aligned; sinusoids intersect at (70.7, -45) (B)

ρ reflects specularly at θ=90, θ=-90 (Q,R,S): edges both vertical

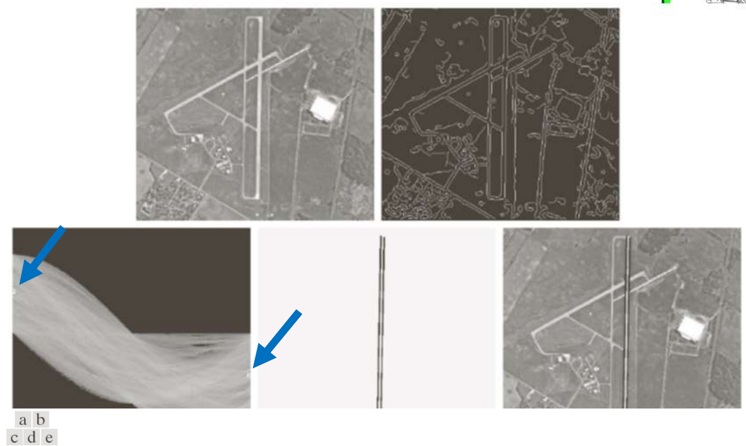


FIGURE 10.34 (a) A 502 × 564 aerial image of an airport. (b) Edge image obtained using Canny's algorithm. (c) Hough parameter space (the boxes highlight the points associated with long vertical lines). (d) Lines in the image plane corresponding to the points highlighted by the boxes). (e) Lines superimposed on the original image.

Image segmentation: edge linking

- *global processing: the Hough transform*

- **Note 1:**

Length of a segment is determined looking back at the positions of the edge points (first, last, aligned clusters) that contribute to $A(\rho,\theta)$

- **Note 2:**

The HT can be used in principle for any edge shape, represented by a function of the type $g(\boldsymbol{v},\boldsymbol{coef})=0$, where ***v*** is a vector of coordinates and ***coef*** is a vector of coefficients.

→ E.g.: looking for circular objects: (*x*-*a*)^2 + (*y*-*b*)^2 = *c*^2

Three parameters (*a,b,c*), 3-D parameter space, cube-like cells, accumulator takes the form *A*(*i,j,k*).

Procedure:

```
1. Increment a and b
2. Solve for c
3. Update the accumulator associated with (a,b,c)
```

it.mathworks.com/help/images/hough-transform.html

Gianni Ramponi
http://www.units.it/ramponi
Digital Image Processing

Image segmentation: snakes

- Active (elastic) contour models: ***Snakes***

slide: z10_snakes 1 e 2

Matlab: z10_snakes_Matlab.zip

Gianni Ramponi
http://www.units.it/ramponi
Digital Image Processing

Image segmentation: similarities

Thresholding

$$g(x,y)=\begin{cases}1 & \text{if } f(x,y)>T\\ 0 & \text{if } f(x,y)\le T\end{cases}$$

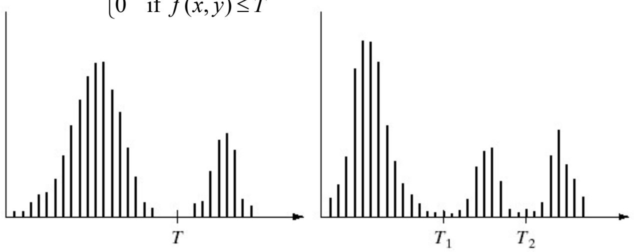


FIGURE 10.26 (a) Gray-level histograms that can be partitioned by (a) a single threshold, and (b) multiple thresholds.

Gianni Ramponi
http://www.units.it/ramponi
Digital Image Processing

Image segmentation: similarities

Thresholding (global) noise

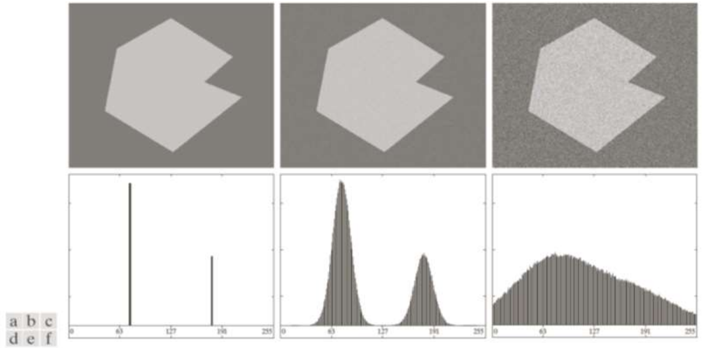


FIGURE 10.36 (a) Noiseless 8-bit image. (b) Image with additive Gaussian noise of mean 0 and standard deviation of 10 intensity levels. (c) Image with additive Gaussian noise of mean 0 and standard deviation of 50 intensity levels. (d)–(f) Corresponding histograms.

(edge-preserving noise smoothing preprocessing may be useful)

Image segmentation: similarities

Thresholding (global) illumination x reflectance

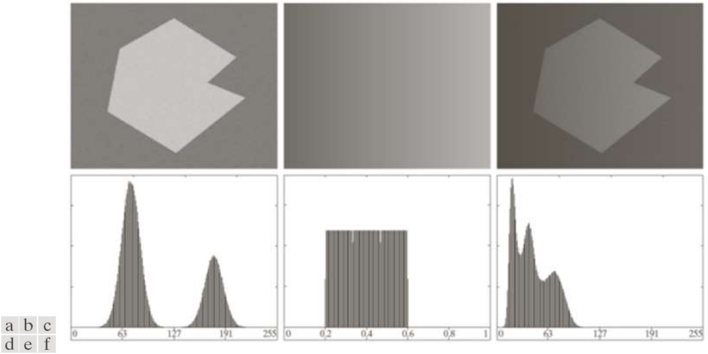
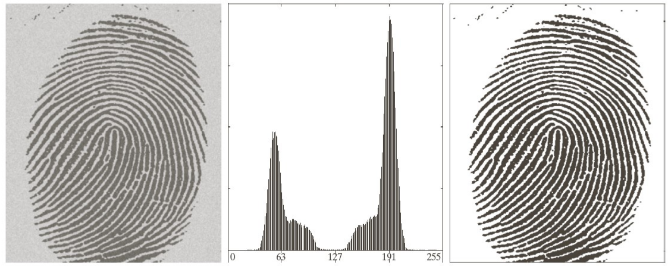


FIGURE 10.37 (a) Noisy image. (b) Intensity ramp in the range [0.2, 0.6]. (c) Product of (a) and (b). (d)–(f) Corresponding histograms.

(Morphological or Retinex preprocessing may be useful)

Image segmentation: similarities

Thresholding (global)



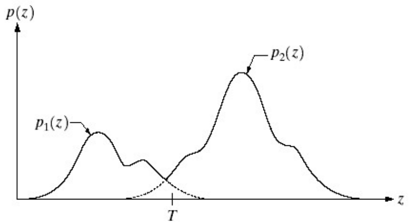
A simple algorithm: Select a first value for *T=T0*; threshold the image; evaluate the average gray-levels *Ga, Gb* of the two groups; set *T1=(Ga+Gb)/2*; repeat until $|T(k)-T(k-1)| < \varepsilon$

This approach is acceptable only for visibly bimodal histograms

Image segmentation: similarities

Thresholding (global, optimal)

FIGURE 10.32 Gray-level probability density functions of two regions in an image.



More formally: A bimodal histogram can indicate the presence of two objects in the image, i.e. it can be the weighted sum of two unimodal densities (one for light, one for dark areas): $p(z) = P1\ p1(z) + P2\ p2(z)$

The parameters (probabilities *P1* and *P2,* with *P1+P2=1*) are proportional to the areas of the picture of each brightness.

If a mathematical expression for the densities *p1(z), p2(z)* is known or assumed, determining an optimal (e.g. MMSE) threshold is possible.

Image segmentation: similarities

Thresholding (global, optimal)

Consider a simple case (Ming Jiang 5.1.2).

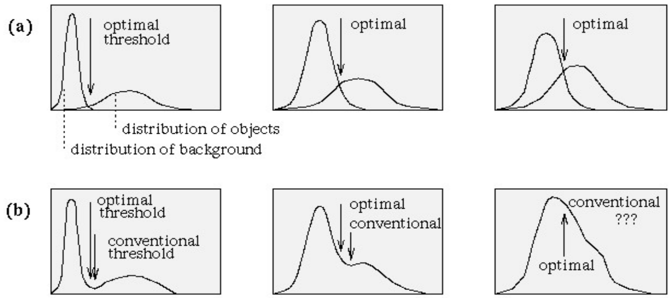
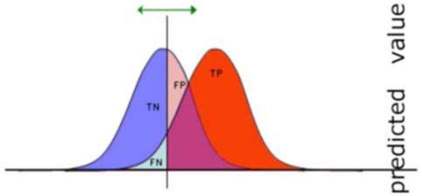


Figure 5.4 *Grey level histograms approximated by two normal distributions; the threshold is set to give minimum probability of segmentation error: (a) Probability distributions of background and objects, (b) corresponding histograms and optimal threshold.*

Image segmentation: similarities

Thresholding (global, optimal)



		actual value pos	actual value neg	
predicted value	pos	TP	FP	P'
predicted value	neg	FN	TN	N'
		1	1	tot
	tot	P	N	

Above threshold — TP: # true positive; FP: # false positive

Below threshold — TN: # true negative; FN: # false negative

Gianni Ramponi
http://www.units.it/ramponi
Digital Image Processing

Image segmentation: similarities

Thresholding (global, optimal)

Assume the image consists of the objects and background, where the objects occupy *P1* of the pixels (*P1+P2=1*). Assume both objects and background are subject to a Normal distribution; by the total probability rule, the image has the following density function:

$$p(z)=\frac{P_1}{\sigma_1\sqrt{2\pi}}\exp(-\frac{(z-\mu_1)^2}{2\sigma_1^2})+\frac{P_2}{\sigma_2\sqrt{2\pi}}\exp(-\frac{(z-\mu_2)^2}{2\sigma_2^2})$$

Let *T* be the threshold. The mis-segmentation takes place in two cases:

* Background pixels mis-classified into object pixels (FP): the error probability (or the number of errors) is *E1*

* Object pixels mis-classified into background pixels (FN): the error probability (or the number of errors) is *E2*

$$E1(T)=\int_T^{\infty} p1(z)dz\,; \qquad E2(T)=\int_{-\infty}^{T} p2(z)dz$$

The total mis-segmentation error is *E(T) = P1 E1(T) + P2 E2(T)*

Gianni Ramponi
http://www.units.it/ramponi
Digital Image Processing

Image segmentation: similarities

Thresholding (global, optimal)

The total mis-segmentation error is *E(T) = P1 E1(T) + P2 E2(T)*

The optimal threshold is *T** = arg min{*E(T)*}. Differentiating *E(T)*:

P1 E1'(T) + P2 E2'(T) = 0

Substituting the formula for the Gaussian into the above equation:

$$\frac{P_1}{\sigma_1\sqrt{2\pi}}\exp(-\frac{(T-\mu_1)^2}{2\sigma_1^2})=-\frac{P_2}{\sigma_2\sqrt{2\pi}}\exp(-\frac{(T-\mu_2)^2}{2\sigma_2^2})$$

$$\frac{(T-\mu_1)^2}{2\sigma_1^2}-\frac{(T-\mu_2)^2}{2\sigma_2^2}=\log\frac{P_1\sigma_2}{P_2\sigma_1}$$

Two specific examples:

- If the s.d. are the same: $T^*=\frac{\sigma^2}{\mu_1-\mu_2}\log\frac{P_1}{P_2}+\frac{\mu_1+\mu_2}{2}$
- If the s.d. are the same and *P1=P2=1/2*: $T^*=\frac{\mu_1+\mu_2}{2}$

Gianni Ramponi
http://www.units.it/ramponi
Digital Image Processing

Image segmentation: similarities

Thresholding (Otsu) *(no hypotheses on the distribution)*

For any (sub)set of gray levels (*K1…K2*), define CDF, mean, variance:

$$\omega=\sum_{i=K1}^{K2}P_i=(\sum_{i=K1}^{K2}n_i)/N\,; \qquad \mu=\sum_{i=K1}^{K2}i\,P_i\,; \qquad \sigma^2=\sum_{i=K1}^{K2}(i-\mu)^2P_i\,;$$

All are functions of (*K1*,*K2*), omitted

Let class 1 be formed by all pixels whose gray level is <= a threshold *T*; class 2 by pixels > *T*;

The between-class variance is the variation of the mean values for each class from the *global* (*G*) intensity mean of all pixels:

$$\mu_G=\omega_1\mu_1+\omega_2\mu_2$$
$$\sigma_B^2=\omega_1(\mu_1-\mu_G)^2+\omega_2(\mu_2-\mu_G)^2=\omega_1\omega_2(\mu_1-\mu_2)^2$$

Image segmentation: similarities

Thresholding (Otsu)

Otsu: All possible thresholds are evaluated, and the one (T^*) that maximizes the between-class variance is chosen.

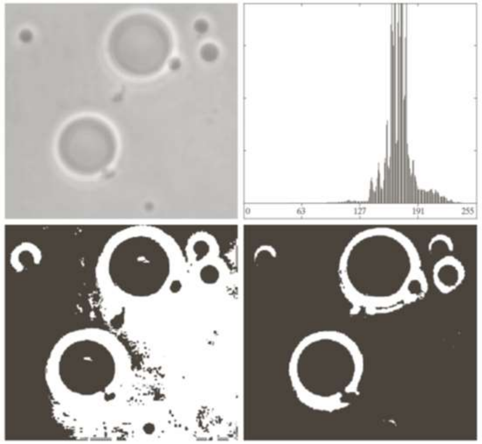
- BTW, this is equivalent to finding the minimum *intra-class* variance
- Quality of result is given by the *normalized between-class variance*, called separability, measured at T^*

$$\eta(T) = \sigma_B^2(T) / \sigma_G^2 \qquad 0 \le \eta(T) \le 1$$

  - ✓ 0 is attainable only by images with a single uniform gray level
  - ✓ 1 is attainable only by 2-valued images with gray levels 0 and L-1
- Otsu's method can be extended to multiple thresholds

Image segmentation: similarities

Thresholding (Otsu)



a b
c d

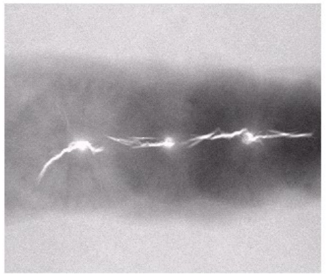
FIGURE 10.39 (a) Original image. (b) Histogram (high peaks were clipped to highlight details in the lower values). (c) Segmentation result using the basic global algorithm from Section 10.3.2. (d) Result obtained using Otsu's method.

T = 169

T^* = 181
H = 0.467

Image segmentation: similarities

Region growing



== *Defective weld*

1. Select *seed* regions (e.g. values in the upmost 1% of the distribution)

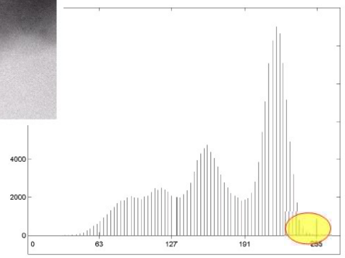


Image segmentation: similarities

Region growing

2. Detect all *connected components*, then *erode* them to one pixel
3. Define a predicate for the pixels of the image. E.g.:
   «the luminance difference wrt the average of the original seed area is below a threshold; and the pixel is 8-connected to at least one pixel in the region»
4. Sequentially *append* to the eroded seeds all the pixels that satisfy the predicate

Image segmentation: similarities

Region splitting and merging

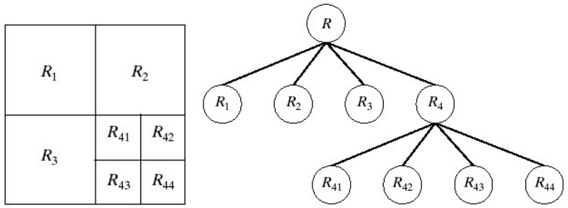


FIGURE 10.42 (a) Partitioned image. (b) Corresponding quadtree.

Define a predicate *P* [e.g., using *descriptors* (Ch. dip11)], and subdivide the image in regions for which *P* is satisfied. More precisely:

1. **Split** into four quadrants any region *Ri* for which *P(Ri)=false*. Stop when a given min. size is reached (e.g. 1x1) → a quadtree is created
2. **Merge** any *adjacent* regions *Ri, Rj* for which *P(Ri U Rj)=true.* Stop when no further merging is possible

Image segmentation: similarities

Watersheds

- The image is treated as if it were a topographic map: gray level = height
- *Watershed lines* divide *catchment basins*
- Flooding is applied (a hole is punched in each local minimum, and water enters from below)
- Dams are built to prevent merging between basins

a b
c d

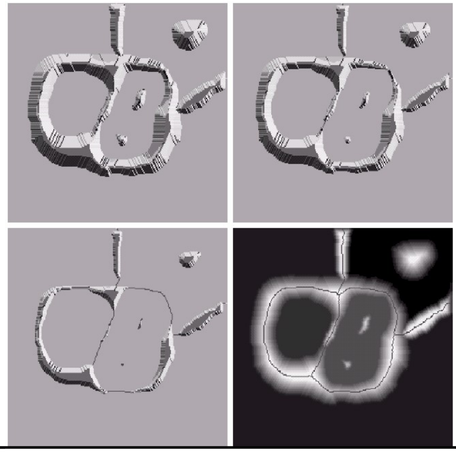
FIGURE 10.44 (a) Original image. (b) Topographic view. (c)–(d) Two stages of flooding.

Image segmentation: similarities

Watersheds

- The final dams are the desired segmentation result (fig. h)
- Watershed lines form a connected path → *continuous boundaries*

Watershed segmentation is often applied to the *gradient* of the image



e f
g h

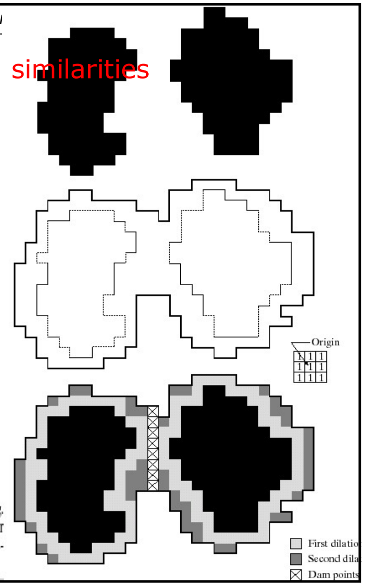
FIGURE 10.44 *(Continued)* (e) Result of further flooding. (f) Beginning of merging of water from two catchment basins (a short dam was built between them). (g) Longer dams. (h) Final watershed (segmentation) lines. (Courtesy of Dr. S. Beucher, CMM/Ecole des Mines de Paris.)

Image segmentation: similarities

Watersheds

A dam is built on a binary image (thresholded flooded image) using morphological dilation, with two rules:
- Dilation is constrained to the connected region formed by the merging of the two basins
- Dilation is not performed on a point if this would cause the two regions to merge

This operation is repeated for each gray level



a
b c
d

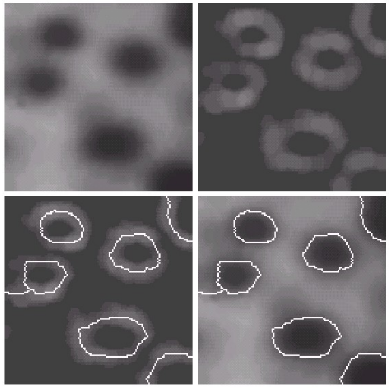
FIGURE 10.45 (a) Two partially flooded catchment basins at stage $n-1$ of flooding. (b) Flooding at stage n, showing that water has spilled between basins (for clarity, water is shown in white rather than black). (c) Structuring element used for dilation. (d) Result of dilation and dam construction.

Image segmentation: similarities

Watersheds

a b
c d

FIGURE 10.46
(a) Image of blobs. (b) Image gradient. (c) Watershed lines. (d) Watershed lines superimposed on original image. (Courtesy of Dr. S. Beucher, CMM/Ecole des Mines de Paris.)



Video segmentation

[Refers to both *spatial frame* segm. and *temporal shot* segm.]

MOTION is a useful cue for segmentation, even for humans.

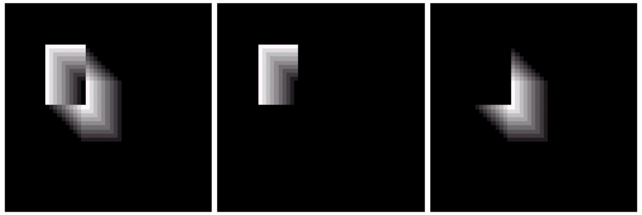
- **The trivial way**: compare two successive frames, pixel by pixel, and search pixels for *significant* changes; **difference image** takes value 1 in positions where $|\ frame_n(x,y) - frame_{n-k}(x,y)\ | > T \quad (k>=1)$

This is sensitive to noise, spatial misregistration (camera motion or shake), variations of illumination

- **Accumulative Difference Image (ADI)**: each pixel is a *counter*, incremented every time a significant difference is found between that location in a frame of the sequence and the same location in a *reference frame*. The reference frame can be the first one of the sequence.
  - Absolute ADI: increment if $|\ ref(x,y) - frame_n(x,y)\ | > T$
  - Positive ADI: increment if $ref(x,y) - frame_n(x,y) > T$
  - Negative ADI: increment if $ref(x,y) - frame_n(x,y) < -T$

Video frame segmentation

E.g.: BRIGHT rectangular object moving right/downwards on a DARK background:



A-ADI P-ADI N-ADI

- Nonzero area in P-ADI = object area (if sequence is long enough)
- P-ADI shows the object location in the reference frame
- P-ADI stops increasing when object is wholly displaced wrt ref. frame
- Direction and speed can be obtained by A-ADI and N-ADI

Video frame segmentation

Determination of the reference image is not trivial

Example: build a static reference image using ADIs
- when the white car has moved completely out of its position in the ref. frame, copy the corresponding background in the present frame into the ref. frame.
- repeat for all moving objects.

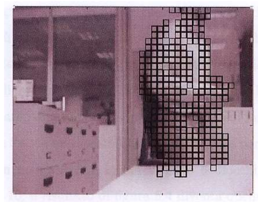
FIGURE 10.50 Building a static reference image. (a) and (b) Two frames in a sequence. (c) Eastbound automobile subtracted from (a) and the background restored from the corresponding area in (b). (Jain and Jain.)

Video frame segmentation

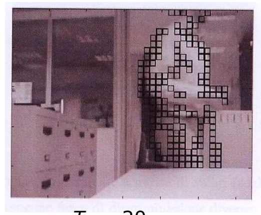
Example: intrusion detection on a bus in a garage:
- acquire a frame once per second and compare to the previous one
- divide each frame into 8x8 *blocks* and 32x32 *macroblocks* (MBs)
- compute the SAD (sum of absolute differences) for each block in the same position in the two frames
- calculate N_1: for each MB, number of blocks with SAD > T_1
- calculate N_2: number of MBs with $N_1 > T_2$
- if $N_2 > T_3$ → alarm

Gianni Ramponi
http://www.units.it/ramponi
Digital Image Processing

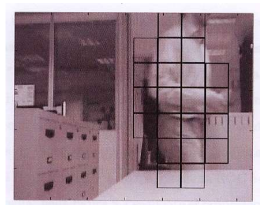
Video frame segmentation



$T_1 = 10$



$T_1 = 20$



MBs with at least 6 blocks
"with movement" ($T_2 = 6$)