Machine Vision:
Image Segmentation

Wu Wei

ww@hdu.edu.cn

Contents

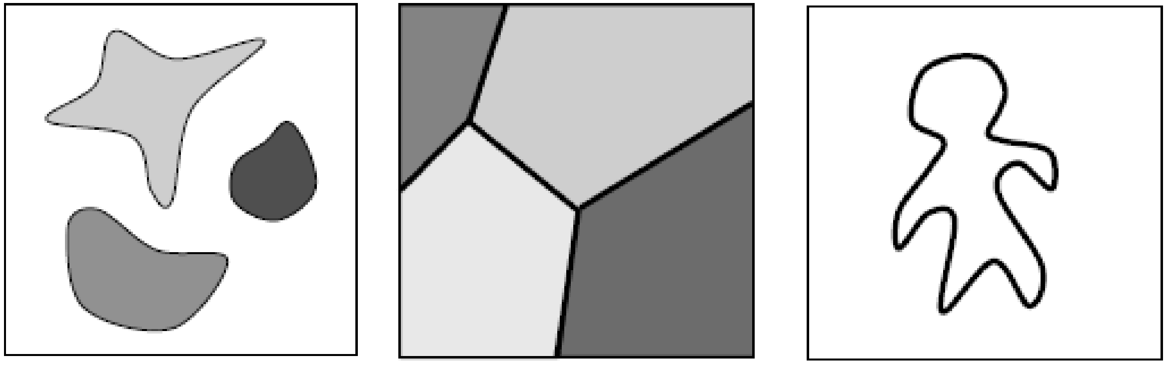
Segmentation

- Edge detection (Luminance: Discontinuity)
- Region detection (Luminance: Similarity- thresholding, region growing, region splitting / merging)

Segmentation

Image Segmentation= divide image into (continuous) regions or sets of pixels based on discontinuity or similarity of luminance.

1) Region Based
2) Edge Based
3) Boundary Based



Outlines

- Detection of Discontinuities
- Edge Linking and Boundary Detection
- Thresholding
- Region-Based Segmentation
- Color Segmentation

Outlines

- Detection of Discontinuities
- Edge Linking and Boundary Detection
- Thresholding
- Region-Based Segmentation
- Color Segmentation

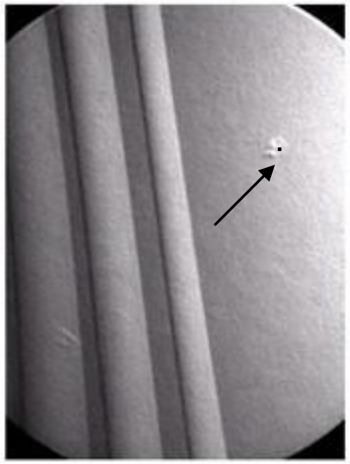
Detection of Discontinuities

The most common way to look for discontinuities is to run a mask through the image

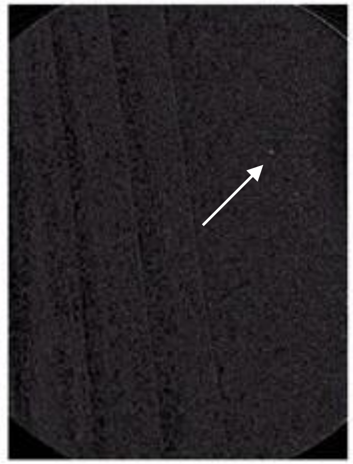
w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

$$R = \sum_{i=1}^{9} w_i z_i$$

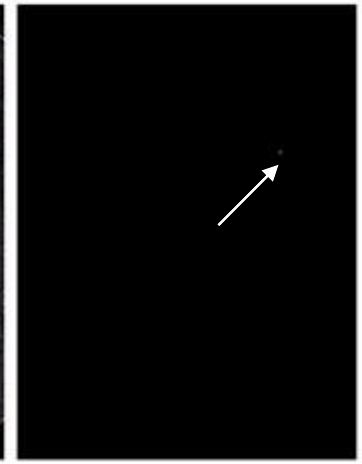
Point detection



X-ray image of a turbine blade with a porosity.



Result of point detection



Result of using threshold.

$$R = \sum_{i=1}^{9} w_i z_i$$

Point detection: $|R| \geq T$

Line Detection

−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

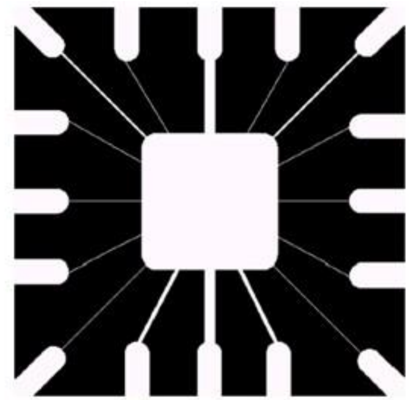
−45°

$$R = \sum_{i=1}^{9} w_i z_i$$

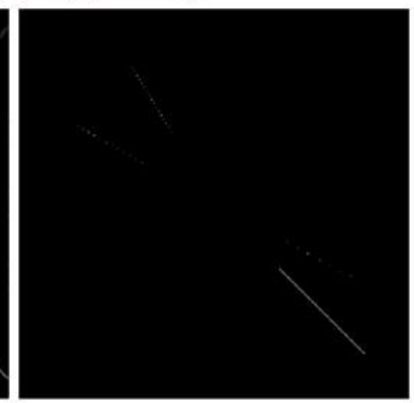
Point detection: $|R| \geq T$

How to remove the isolated point?

Binary wirebond mask.



Absolute value of result after processing with -45° line detector.



Result of thresholding image.

Edge detection –why important ?

Edge detection –why important ?

Information reduction

–replace image by a cartoon in which objects and surface markings are outlined

–these are the most informative parts of the image

Biological plausibility

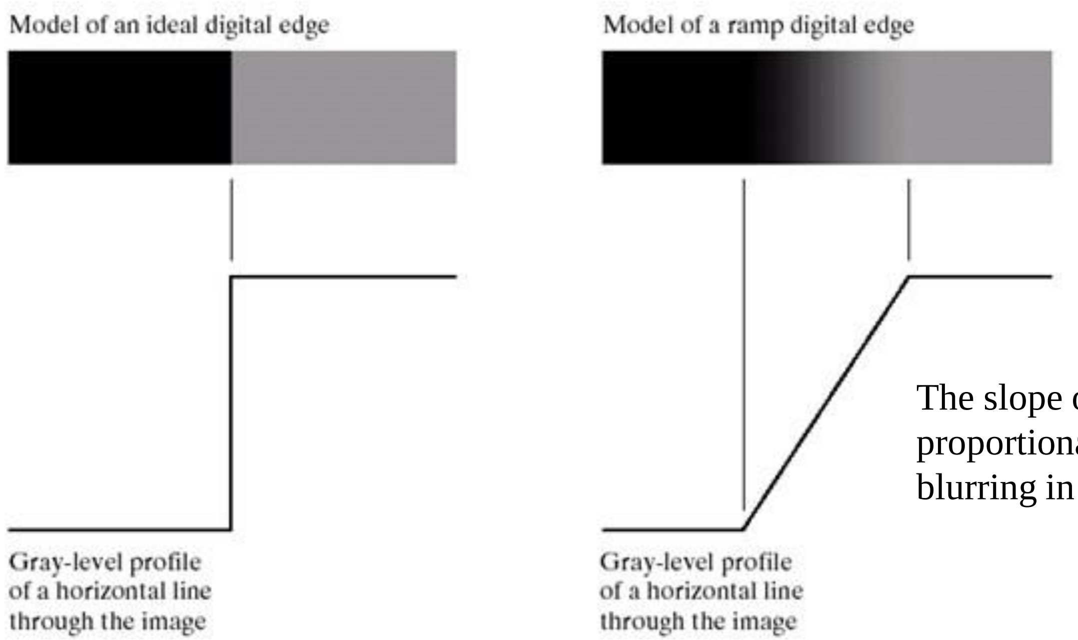
–initial stages of mammalian vision systems involve detection of edges and local features

Edge detection

De

–

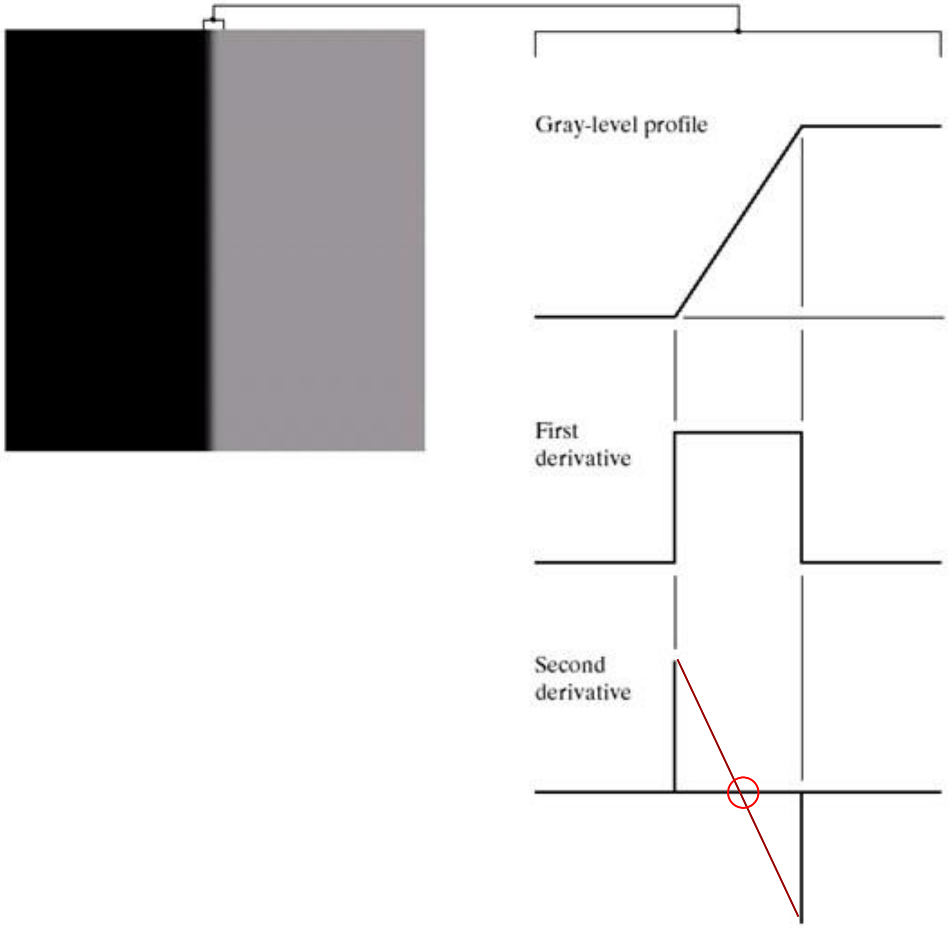
Edge Detection



The slope of the ramp is proportional to the degree of blurring in the edge.

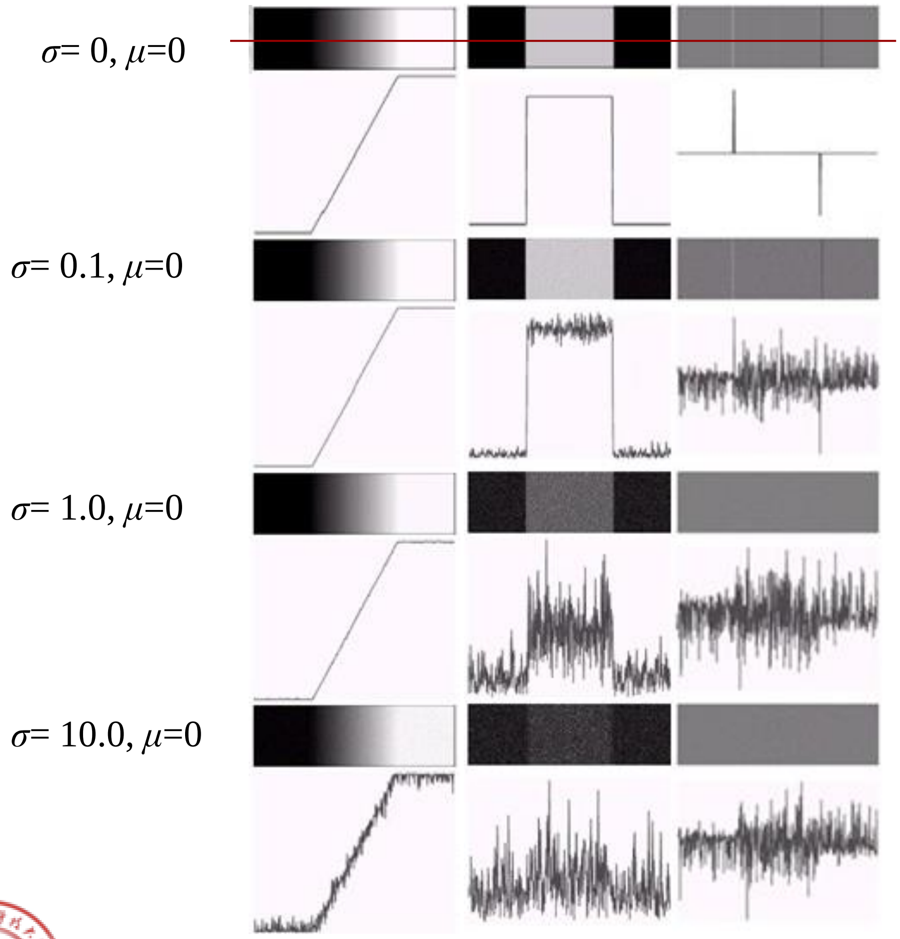
Optics, sampling, and other acquisition imperfections.

1st- and 2nd Derivatives



- The fist- and second-order digital derivatives are commonly used for the detection of edges in an image.

1st- and 2nd Derivatives in Noise



Fairly little noise can have a significant impact on the two key derivatives for edge detection in images.

Image smoothing should be a serious consideration prior to the use of derivatives in applications.

Gradient Operations

Review on the *gradient*:

$$\nabla \mathbf{f} = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

$$\nabla f = \mathrm{mag}(\nabla \mathbf{f}) = \left[G_x^2 + G_y^2\right]^{1/2} \qquad \nabla f \approx |G_x| + |G_y|$$

$$\alpha(x, y) = \tan^{-1}\left(\frac{G_y}{G_x}\right)$$

where the angle is measured with respect to the *x*-axis. The direction of an edge at (x, y) is *perpendicular* to the direction of the gradient vector at that point.

Gradient Operations

A 3×3 region of an image (the z's are gray-level values) and various masks used to compute the gradient at point labeled z_5

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

−1	0	0	−1
0	1	1	0

Roberts

$$G_x = (z_9 - z_5)$$
$$G_y = (z_8 - z_6)$$

−1	−1	−1	−1	0	1
0	0	0	−1	0	1
1	1	1	−1	0	1

Prewitt

$$G_x = (z_7 + z_8 + z_9) - (z_1 + z_2 + z_3)$$
$$G_y = (z_3 + z_6 + z_9) - (z_1 + z_4 + z_7)$$

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

Sobel

$$G_x = (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)$$
$$G_y = (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)$$

Gradient Operations

Two additional Prewitt and Sobel masks for detecting discontinuities in the diagonal directions:

0	1	1
−1	0	1
−1	−1	0

−1	−1	0
−1	0	1
0	1	1

Prewitt

0	1	2
−1	0	1
−2	−1	0

−2	−1	0
−1	0	1
0	1	2

Sobel

Example of Gradient Operations

Original image

$|G_x|$, component of the gradient in the x-direction.

$|G_y|$, component of the gradient in the y-direction.

Gradient image, $|G_x|+|G_y|$

Sobel masks

Example of Gradient Operations

Original image smoothed with a 5×5 averaging filter

The averaging caused the response of all edges to be weaker.

Example of Gradient Operations

Diagonal edge detection.

0	1	2
−1	0	1
−2	−1	0

−2	−1	0
−1	0	1
0	1	2

The Laplacian

The Laplacian mask

0	−1	0
−1	4	−1
0	−1	0

−1	−1	−1
−1	8	−1
−1	−1	−1

$$\nabla^2 f = 4z_5 - (z_2 + z_4 + z_6 + z_8) \quad \nabla^2 f = 8z_5 - (z_1 + z_2 + z_3 + z_4 + z_6 + z_7 + z_8 + z_9)$$

Cons:

- The Laplacian is very sensitive to noise.
- The magnitude of the Laplacian produces double edges.
- The edge direction can’t be detected

Application

- Using its zero-crossing property for edge location.
- Deciding the pixel position (bright or dark)

The LoG

Consider a Gaussian function:

$$h(r) = -e^{-\frac{r^2}{2\sigma^2}}$$

where $r^2 = x^2 + y^2$, σ is the standard deviation. Convolving this function with an image blurs the image.

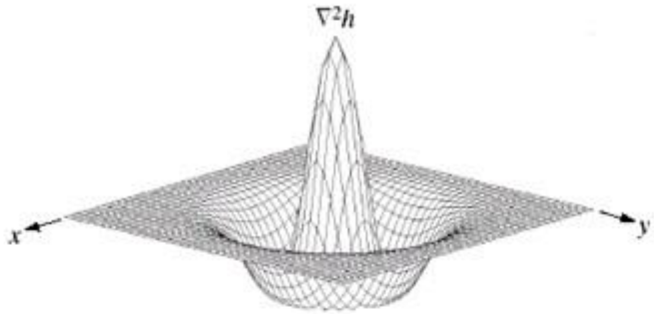
The *Laplacian of a Gaussian* (LoG) is:

$$\nabla^2 h(r) = -\left[\frac{r^2 - \sigma^2}{\sigma^4}\right] e^{-\frac{r^2}{2\sigma^2}}$$

Convolving an image with LoG is the same as convolving the image with the Gaussian smoothing function h first and then computing the Laplacian of the result.

The LoG

LoG



3-D plot

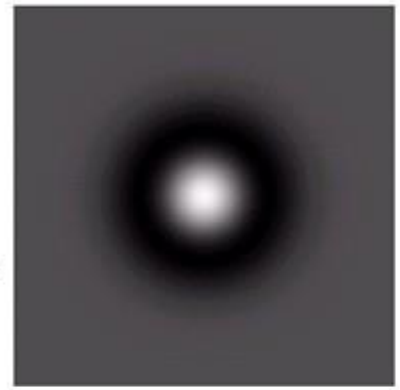
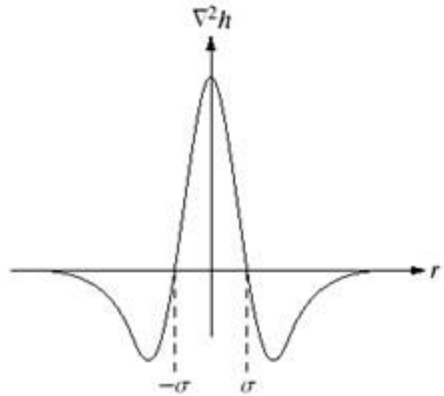


Image (black is negative, gray is the zero plane, and white is positive)



Cross section showing zero crossings

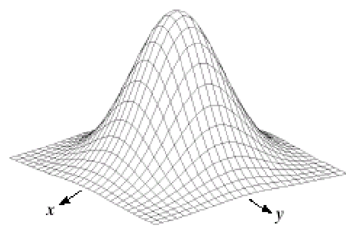
0	0	−1	0	0
0	−1	−2	−1	0
−1	−2	16	−2	−1
0	−1	−2	−1	0
0	0	−1	0	0

5×5 mask approximation to the shape of LoG

An Example of Zero-Crossing

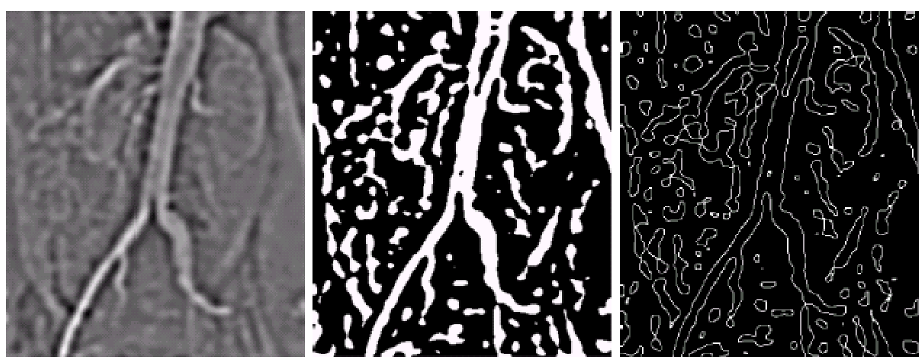
gradient image
(Sobel)

27*27



−1	−1	−1
−1	8	−1
−1	−1	−1

LoG image



zero-corssing

(spaghetti effect)

Outlines

- Detection of Discontinuities
- Edge Linking and Boundary Detection
- Thresholding
- Region-Based Segmentation
- Color Segmentation

Edge Linking and Boundary Detection

- The set of pixels seldom characterizes an edge completely because of noise and breaks in the edge.

- Edge detection algorithms typically are followed by linking procedures to assemble edge pixels into meaningful edges.

Local Processing

- Given an edge point (pixel) at (x, y), try to find in its neighborhood (say 3 x 3 or 5 x 5) which pixel should be connected to it by some pre-defined *similarity criteria*.

$$\left|\nabla f(x,y) - \nabla f(x_0,y_0)\right| \le E$$

$$\left|\alpha(x,y) - \alpha(x_0,y_0)\right| < A$$

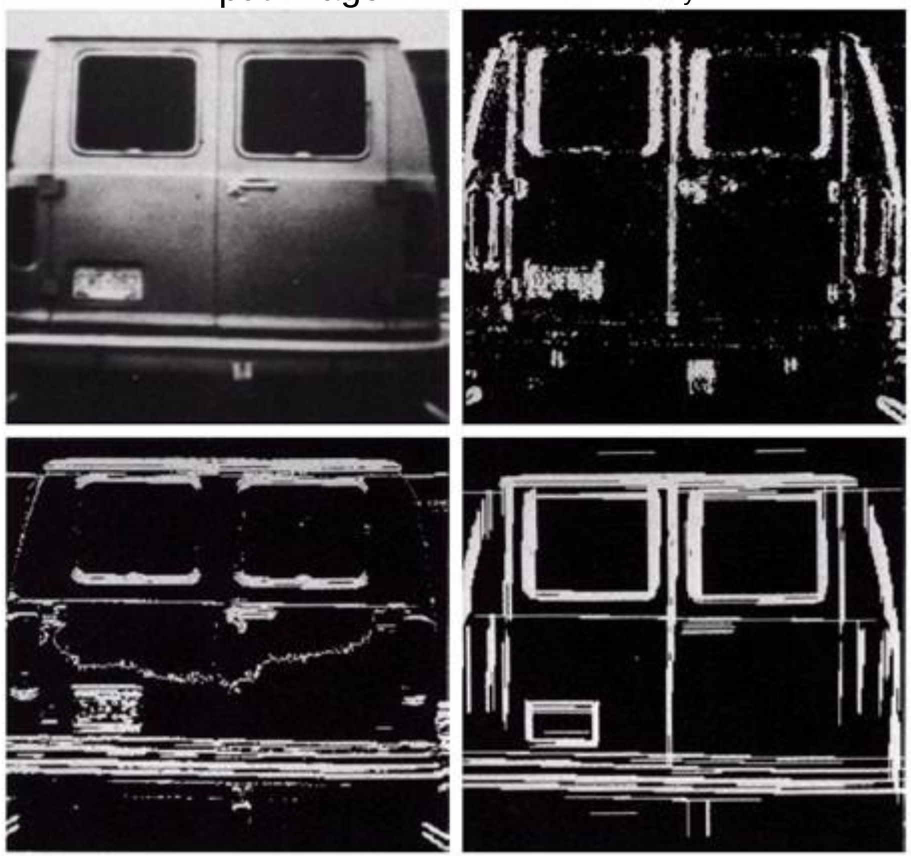
where E is a nonnegative gradient response threshold, and A is a non-negative angle threshold.

- A point in the predefined neighborhood of (x, y) is linked to the pixel at (x, y) if both magnitude and direction criteria are satisfied. This process is repeated at every location in the image.

Local Processing

Input image

G_y



G_x

Edge linking

Sobel masks

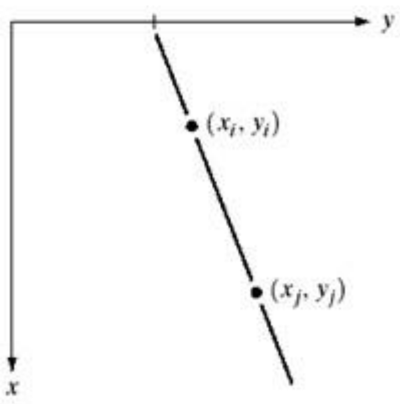
$$\nabla f(x,y) > 25$$

$$|\alpha(x,y) - \alpha(x_0,y_0)| < 15^\circ$$

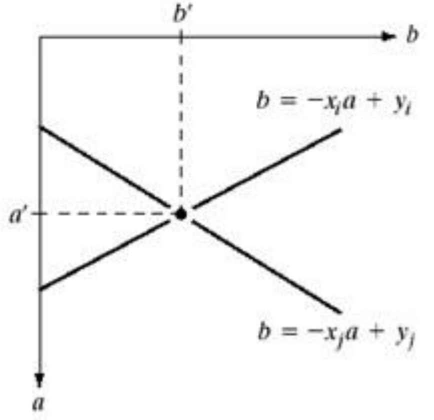
Hough Transform---- Line

- Given n points in an image, try to find subsets of these points that lie on straight lines.
- Consider a point (x_i, y_i), and a straight line through it in slop-intercept form:

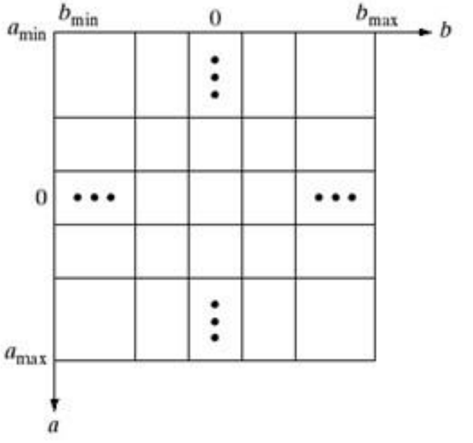
$$y_i = ax_i + b \;\Rightarrow\; b = -x_i a + y_i$$



xy plane



Parameter space

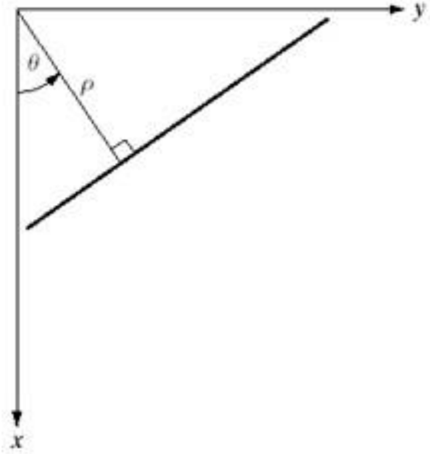


Subdivision of the parameter plane
for use in the Hough transform

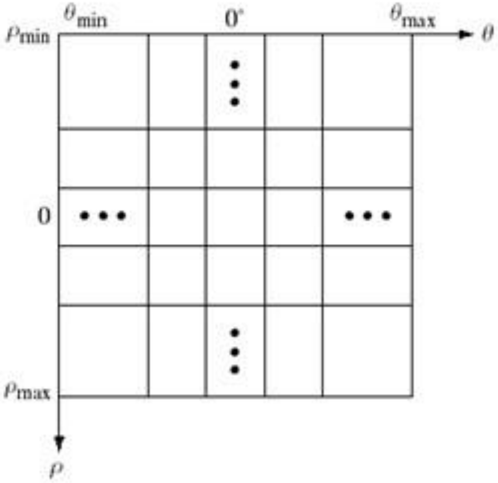
General Hough Transform

To avoid the slope (a) approaches infinity as the line approaches the vertical, use the normal representation of a line:

$$x\cos\theta + y\sin\theta = \rho$$



Normal representation of a line

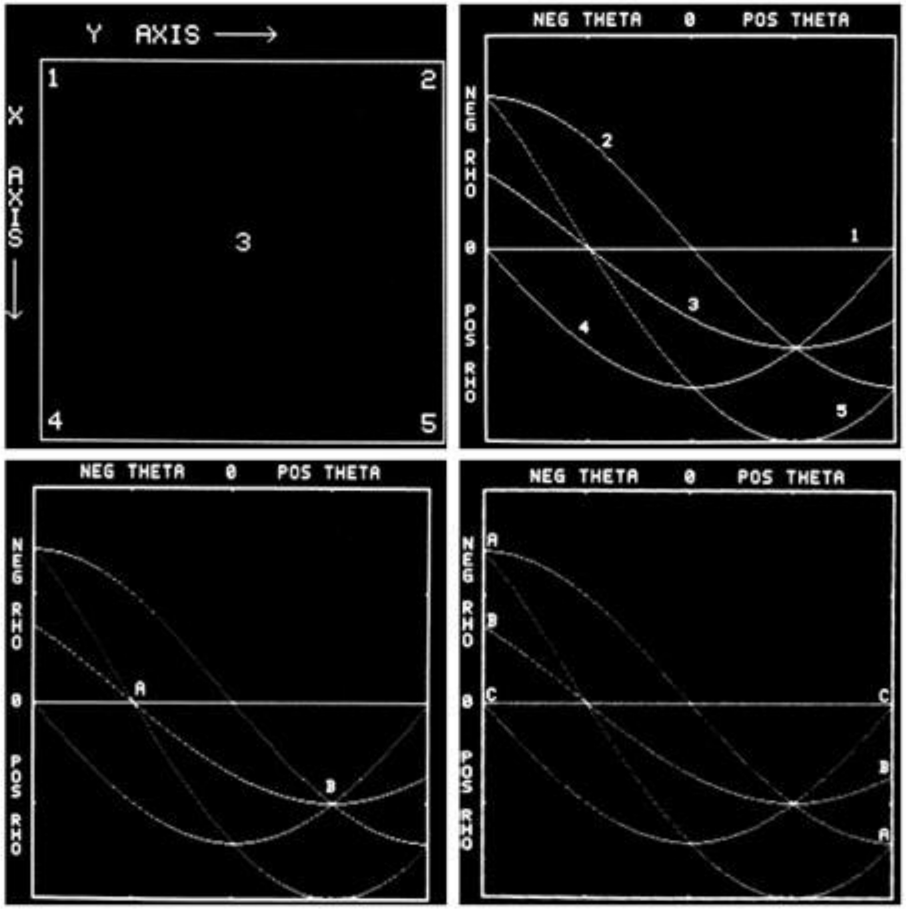


Subdivision of the $\rho\theta$–plane into cells

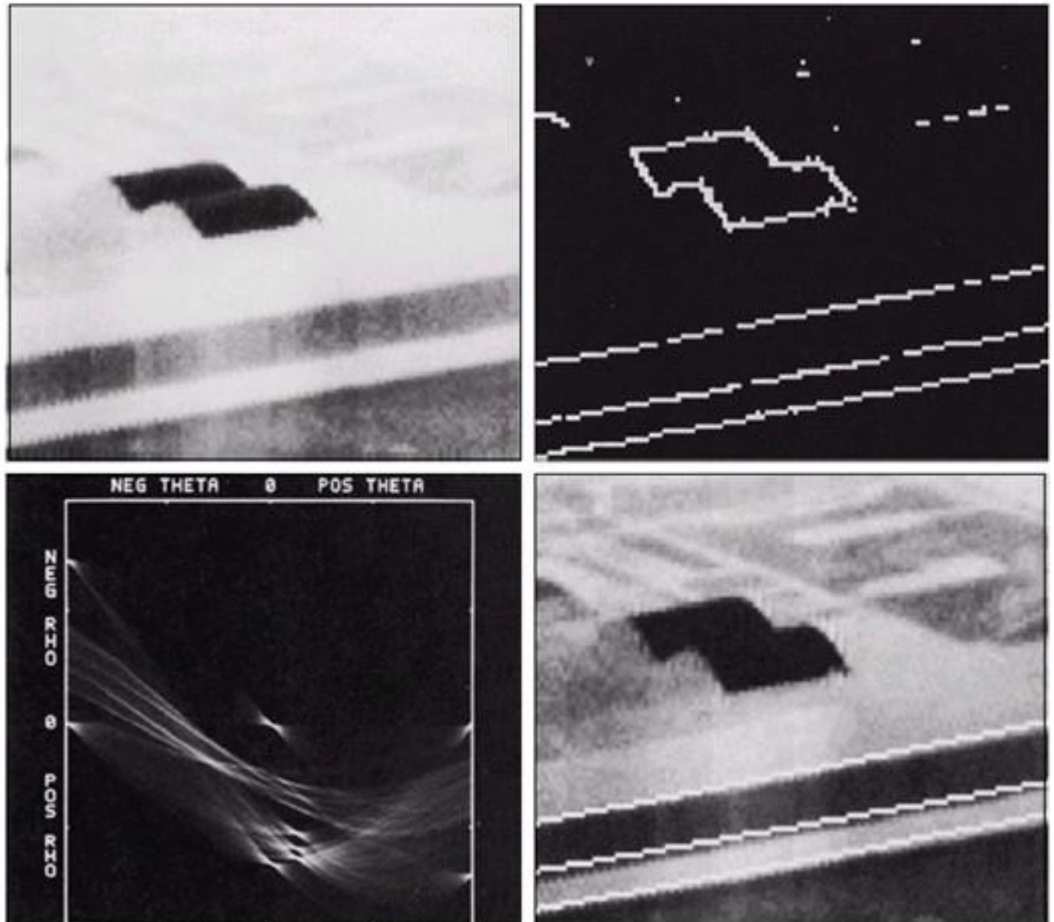
Instead of straight lines, the loci are sinusoidal curves in the $\rho\theta$ – plane.

Hough transform is applicable to any function of the form $g(\mathbf{v}, \mathbf{c})=0$, where $\mathbf{v}$ is a vector of coordinates and $\mathbf{c}$ is a vector of coefficients.

Illustration of the Hough Transform



An Example of the Hough Transform in Edge-Linking



Infrared image.

Thresholded gradient image.

Hough transform.

Linked pixels.

An Example of the Hough Transform

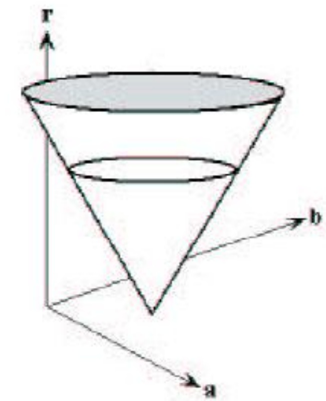
Input image

Edge detected by using Canny operator

Hough transform

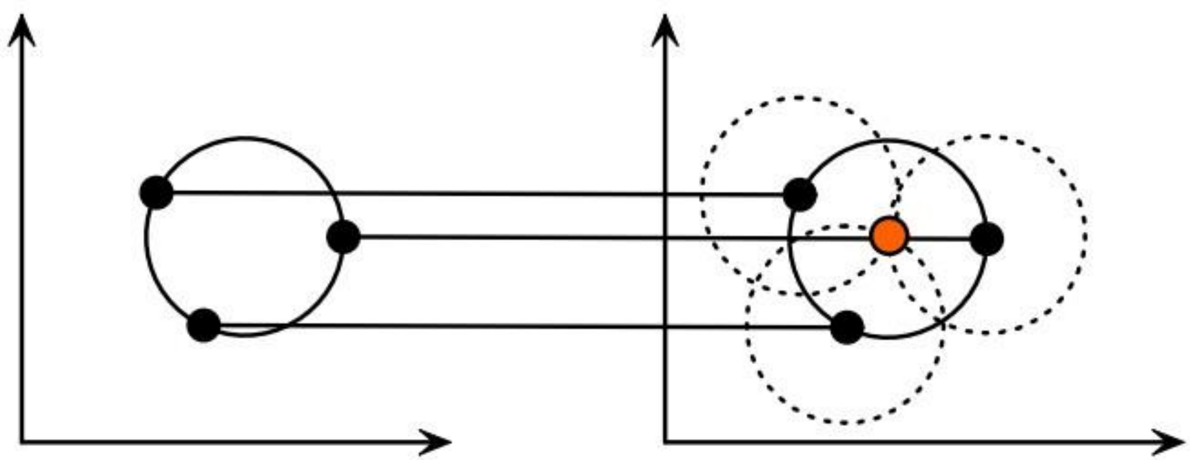
Hough Transform ---- circle

$$r^2 = (x-a)^2 + (y-b)^2 \longrightarrow \begin{aligned} x &= a + r\cos(\theta) \\ y &= b + r\sin(\theta) \end{aligned}$$

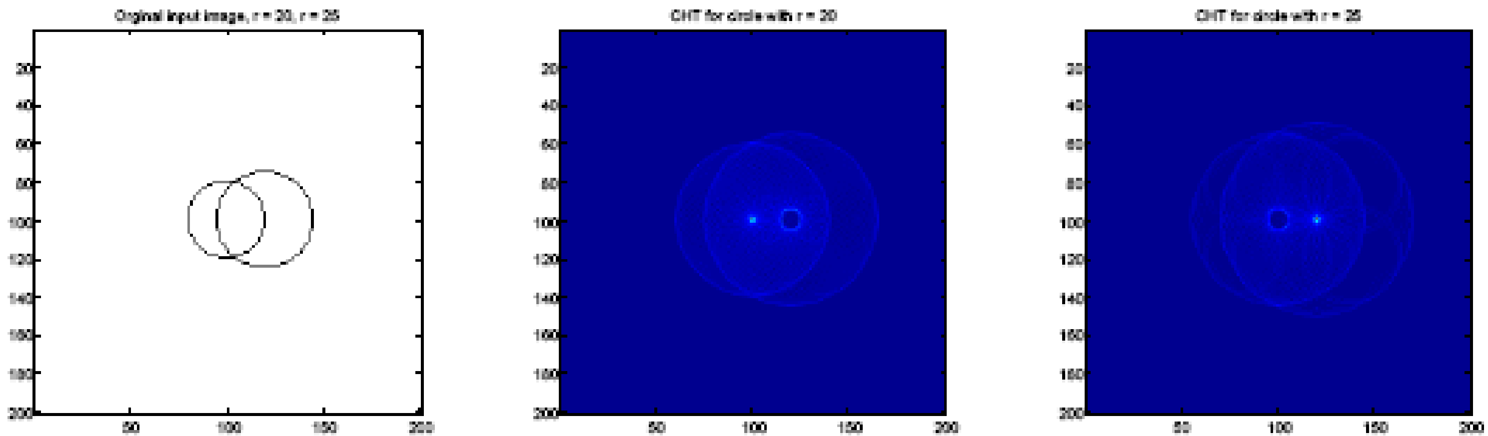


The parameter space

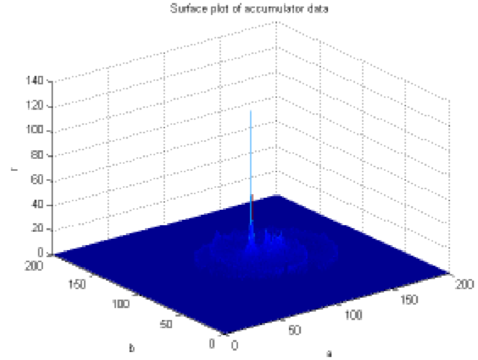
r = R



Hough Transform ---- circle



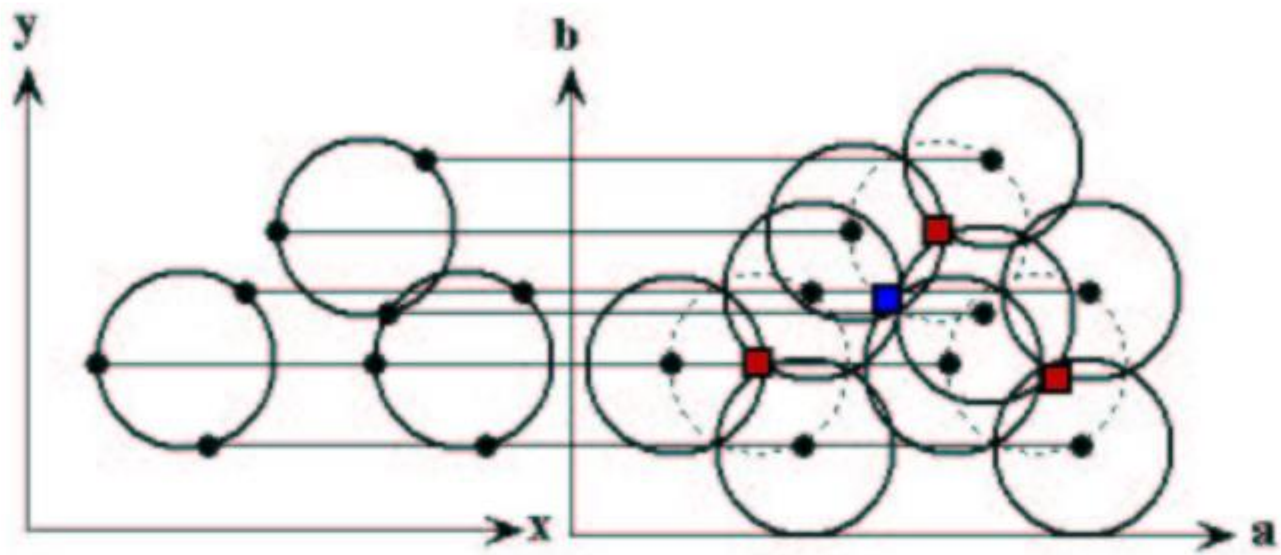
(a) The input image. (b) Hough transform where r = 20. (c) Hough transform where r = 25.



Surface plot of ab-plane with r = 20

Hough Transform ---- circle

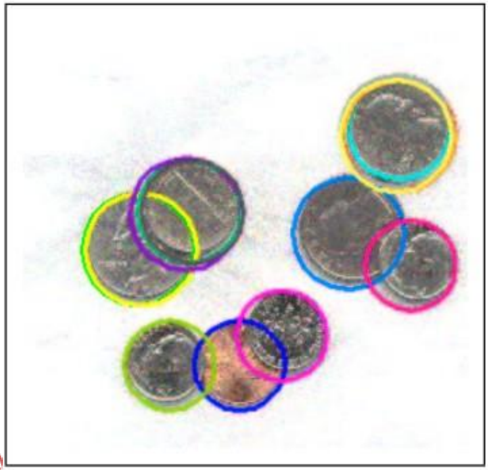
Multiple Circles with known R



Example of CHT

k	R	C_x	C_y	Count
1	58	327	479	394
2	64	301	190	392
3	73	90	190	379
4	73	569	436	369
5	73	504	151	331
6	73	504	155	310
7	54	164	383	303
8	73	569	433	287
9	54	620	230	269
10	54	388	322	268

Example of CHT



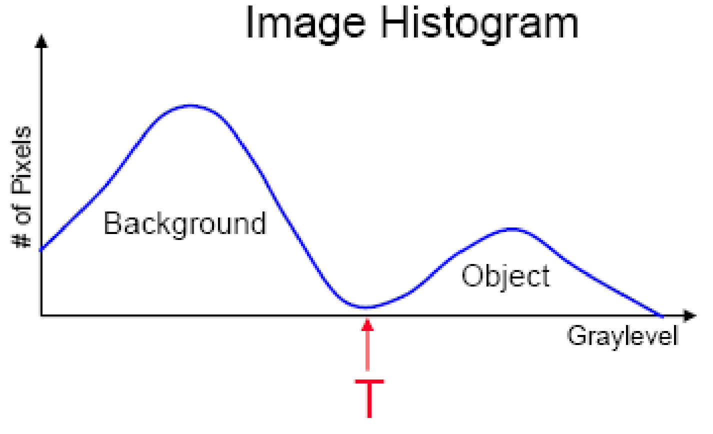
k	R	C_x	C_y	Count
1	73	489	409	266
2	73	149	258	212
3	59	281	106	196
4	67	489	416	192
5	59	335	145	187
6	73	153	258	184
7	73	489	406	183
8	59	189	106	174
9	67	215	307	171
10	73	423	279	169
11	73	211	304	167
12	59	503	237	165
13	67	219	307	163
14	73	485	413	159
15	73	485	406	158

Outlines

- Detection of Discontinuities
- Edge Linking and Boundary Detection
- Thresholding
- Region-Based Segmentation
- Color Segmentation

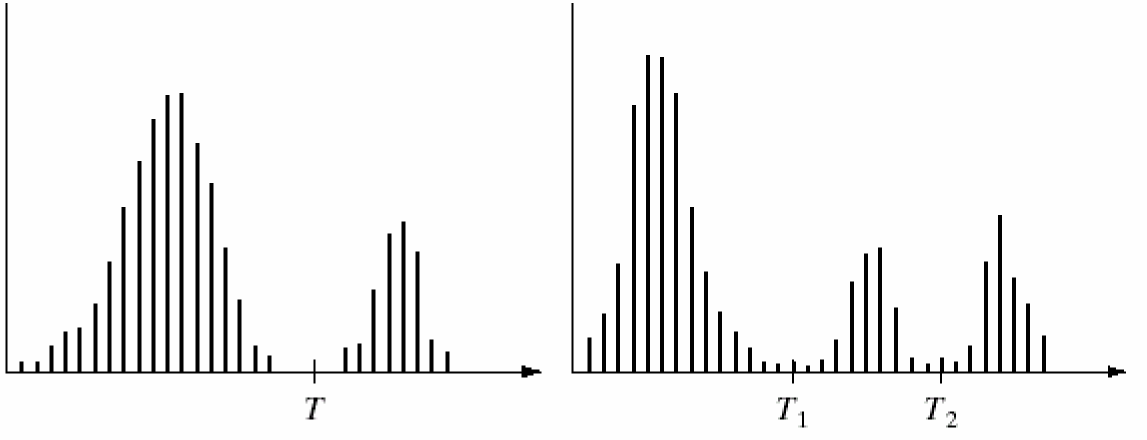
Thresholding

Global Thresholding = Choose threshold T that separates object from background.



Segmentation Using Thresholding

Image thresholding enjoys a central position in applications of image segmentation.



Thresholding may be reviewed as an operation that involves test against a function T of the form

$$T = T\left[x, y, p(x, y), f(x, y)\right]$$

where $p(x, y)$ denotes some local property of this point, average gray level of a neighborhood, e.g.

Thresholding

$$T = T\left[x, y, p(x,y), f(x,y)\right]$$

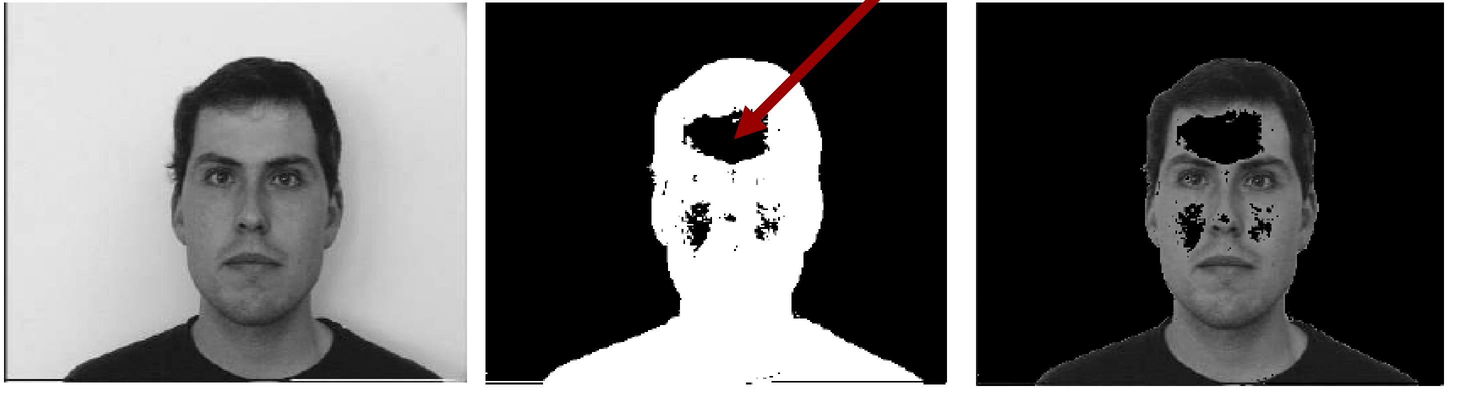
A thresholded image $g(x, y)$ is defined as

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

- When T depends only on $f(x, y)$, the threshold is called *global*.
- If T depends only on both $f(x, y)$ and $p(x, y)$, the threshold is called *local*.
- If T depends also on the spatial coordinates x and y, it is called *dynamic* or *adaptive*.

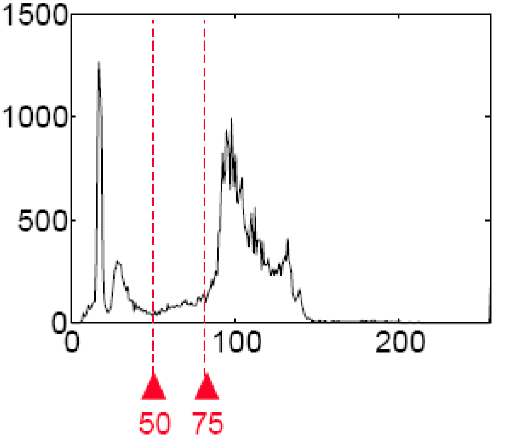
Thresholding

How can holes be filled?



Segmentation Using Thresholding

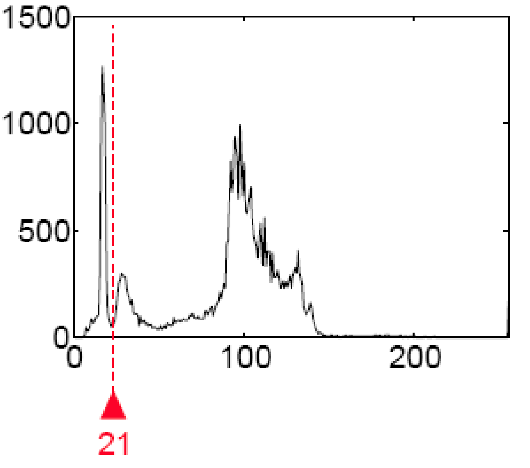
Original



Histogram

Segmentation Using Thresholding

Original

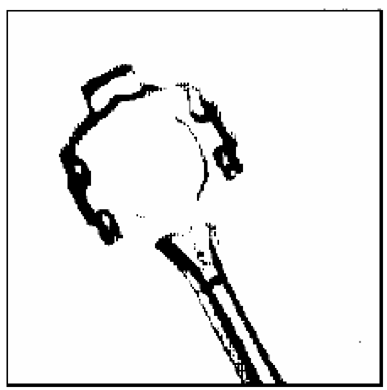


Histogram

Segmentation Using Thresholding

Original image

Thresholded image

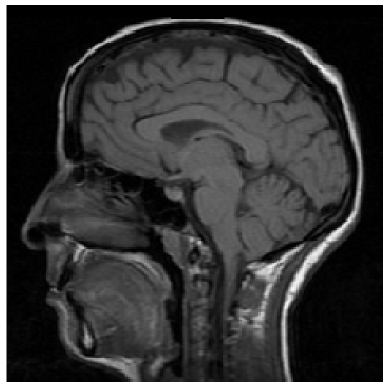


Threshold too low

Threshold too high

Segmentation Using Thresholding

Original image



T = 80

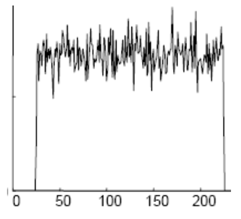
T = 71

T = 88

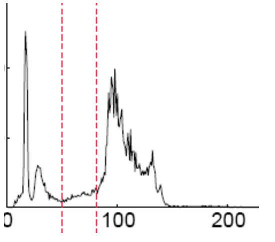
Segmentation Using Thresholding

Question

1. Noise in image

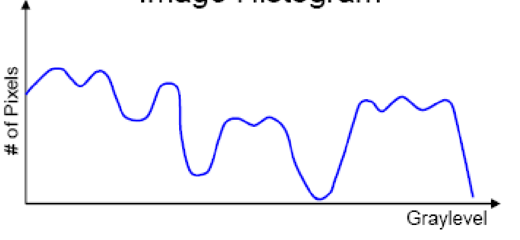


2. How to choose the best thresholding?



3. Many objects at different gray levels.

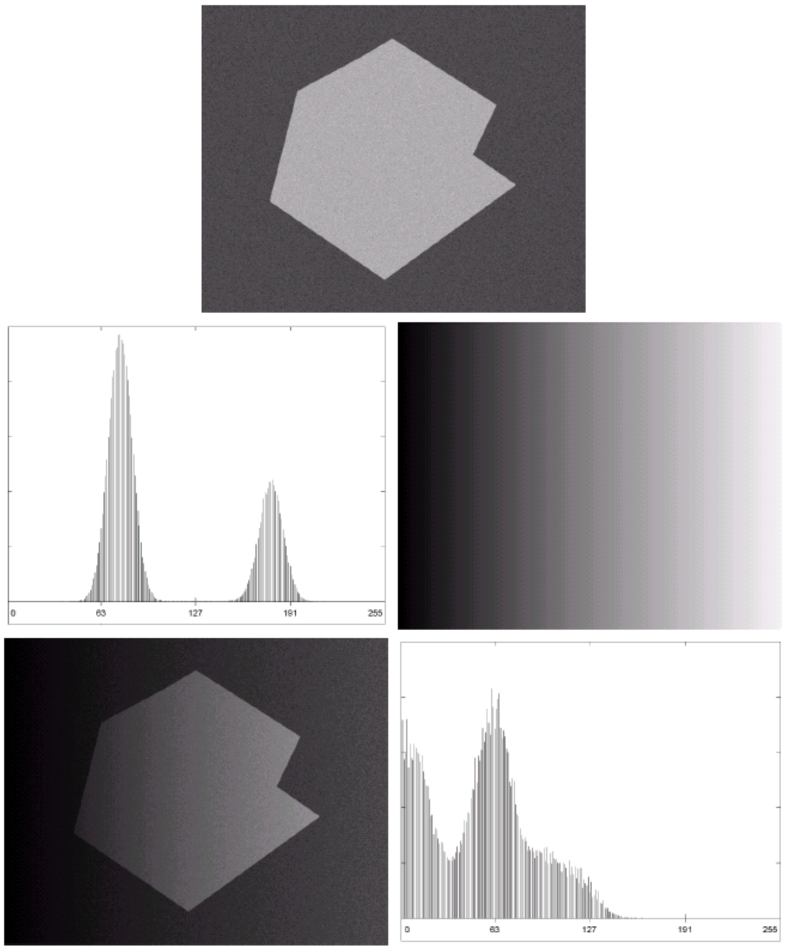
Image Histogram



4. Variations in back ground gray level.

Simple thresholding is not always possible.

The Role of Illumination



a
b c
d e

FIGURE 10.27 (a) Computer generated reflectance function. (b) Histogram of reflectance function. (c) Computer generated illumination function. (d) Product of (a) and (c). (e) Histogram of product image.

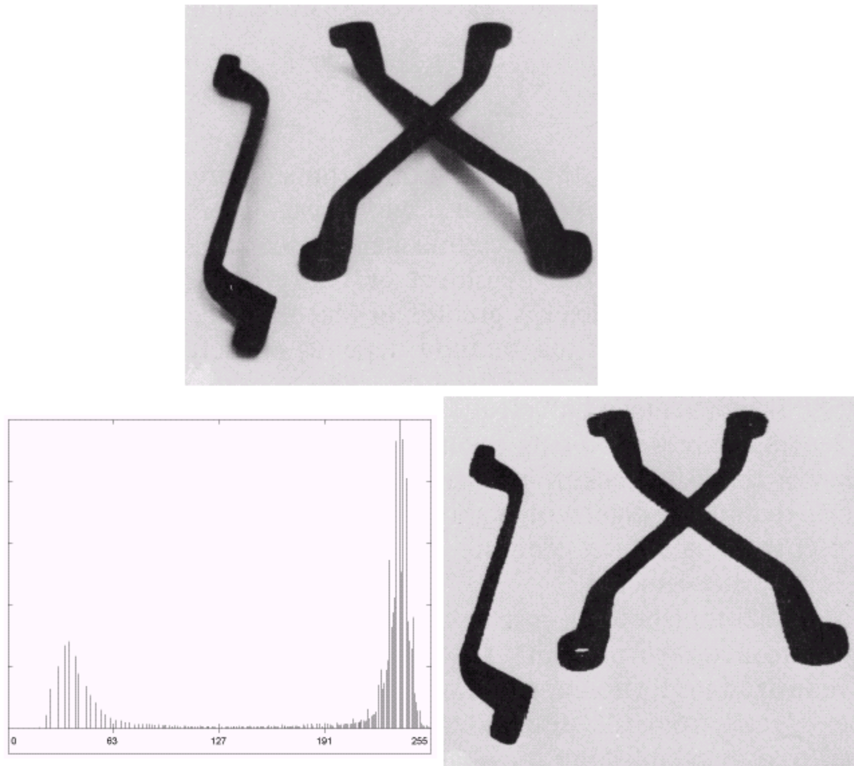
Nonuniformity compensation:

Project the illumination pattern onto a constant, white reflective surface to obtain a $g(x, y) = ki(x, y)$. Then:

$$h(x, y) = f(x, y) / g(x, y)$$

Basic Global Thresholding

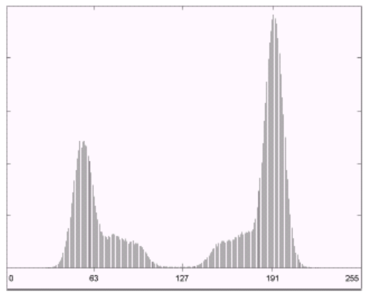
The threshold was specified by using heuristic approach, based on visual inspection.



a
b c

FIGURE 10.28 (a) Original image. (b) Image histogram. (c) Result of global thresholding with T midway between the maximum and minimum gray levels.

The success of this method depends entirely on how well the histogram can be partitioned.



n=3，T=125.4

. estimate an initial T,

. segment the image using T,

$$\mu_1 = E\{g_1\}, \quad \mu_2 = E\{g_2\}$$

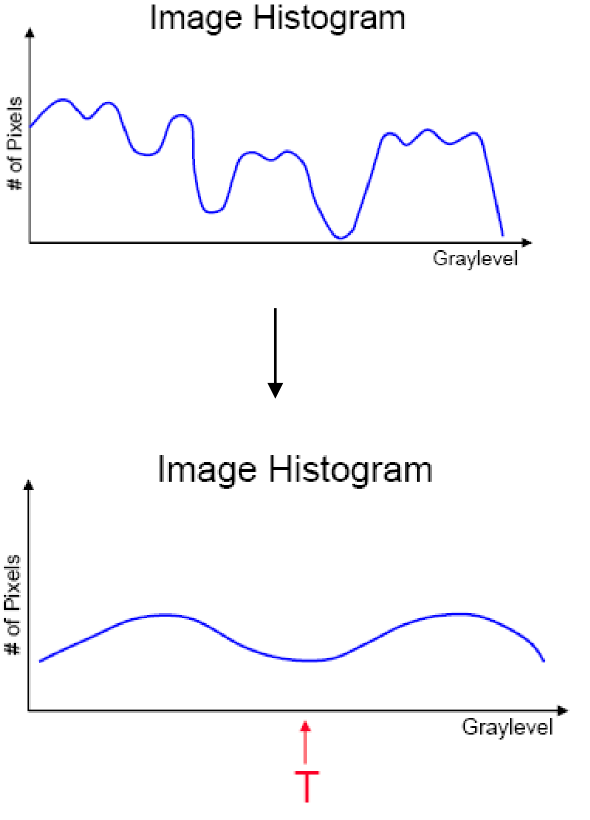
$$\text{new } T = \frac{1}{2}(\mu_1 + \mu_2)$$

. until the change of T is small enough.

Threshold Segmentation of Noisy Images

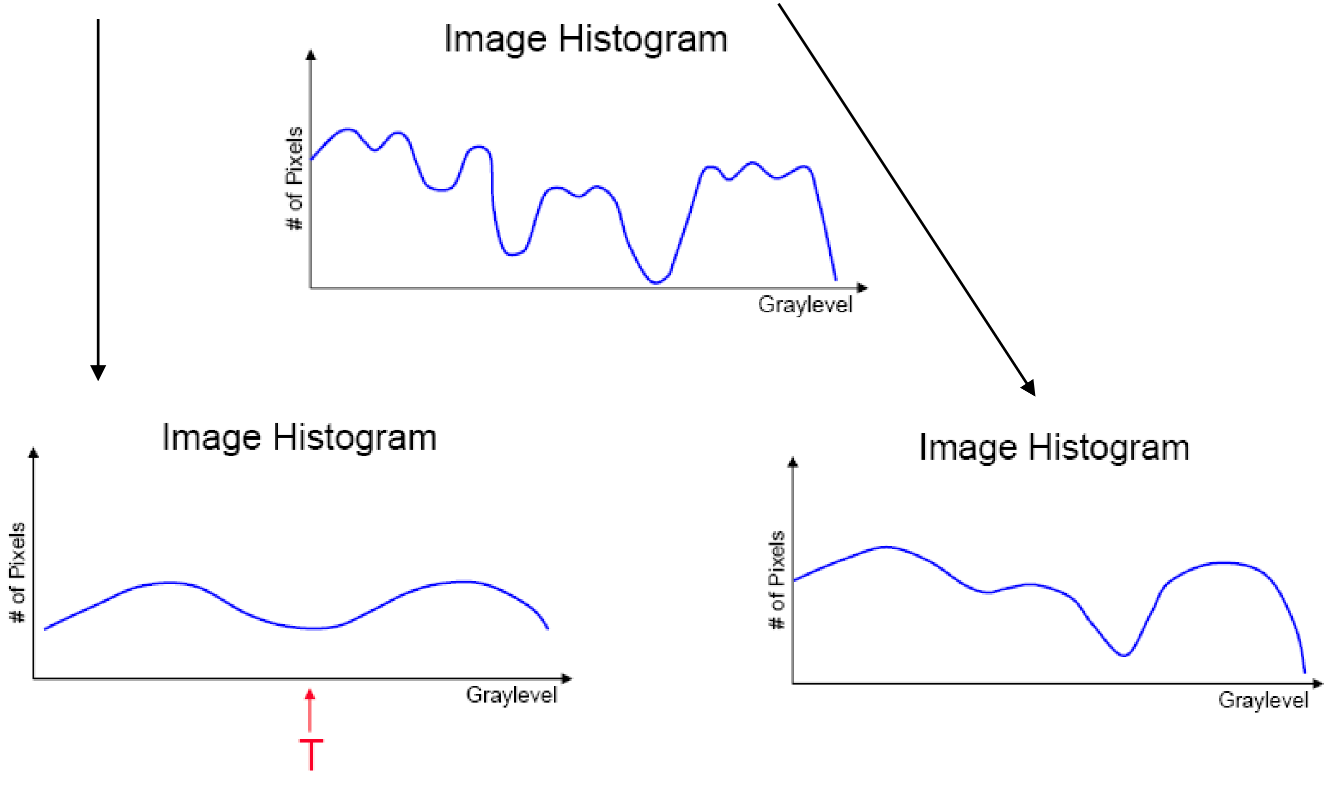
Noise inhibits localization of threshold.

Smooth image and obtain a histogram for which threshold is easily determined.

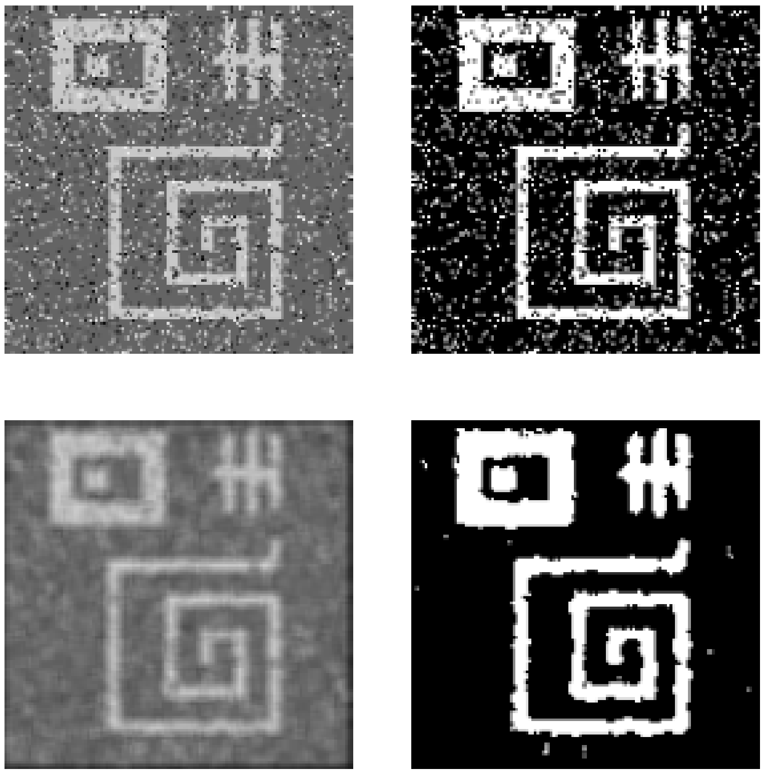


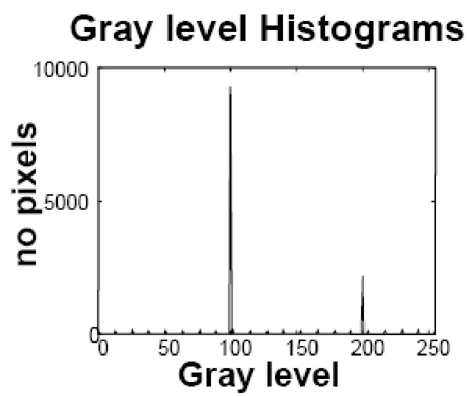
Threshold Segmentation of Noisy Images

Note: Smooth the image, not the histogram...

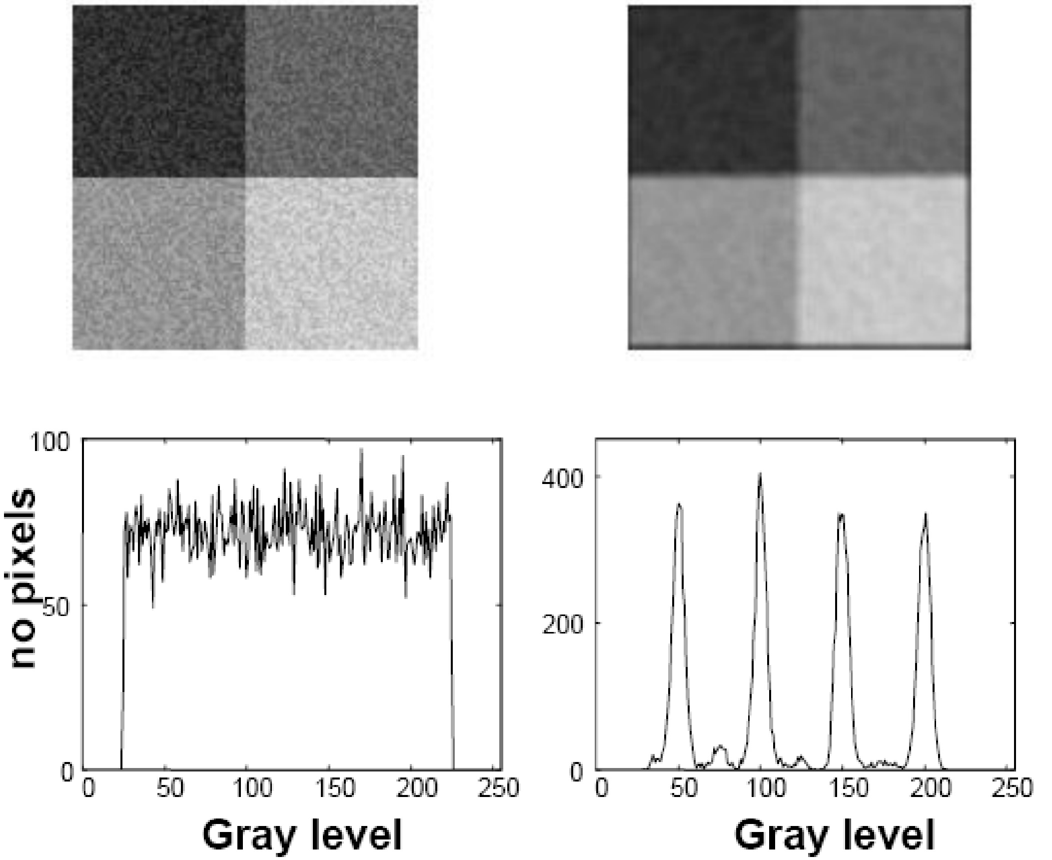


Threshold using Average



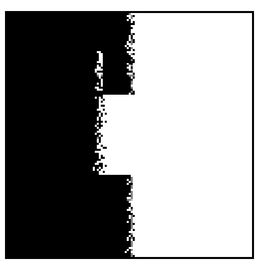


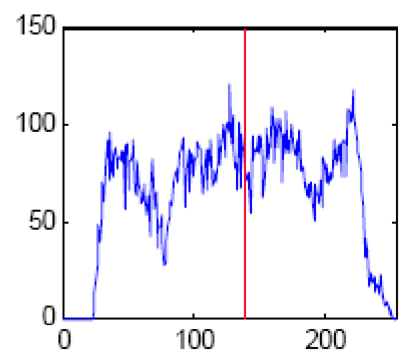
Threshold using Average



Local Thresholding

Single Global Threshold



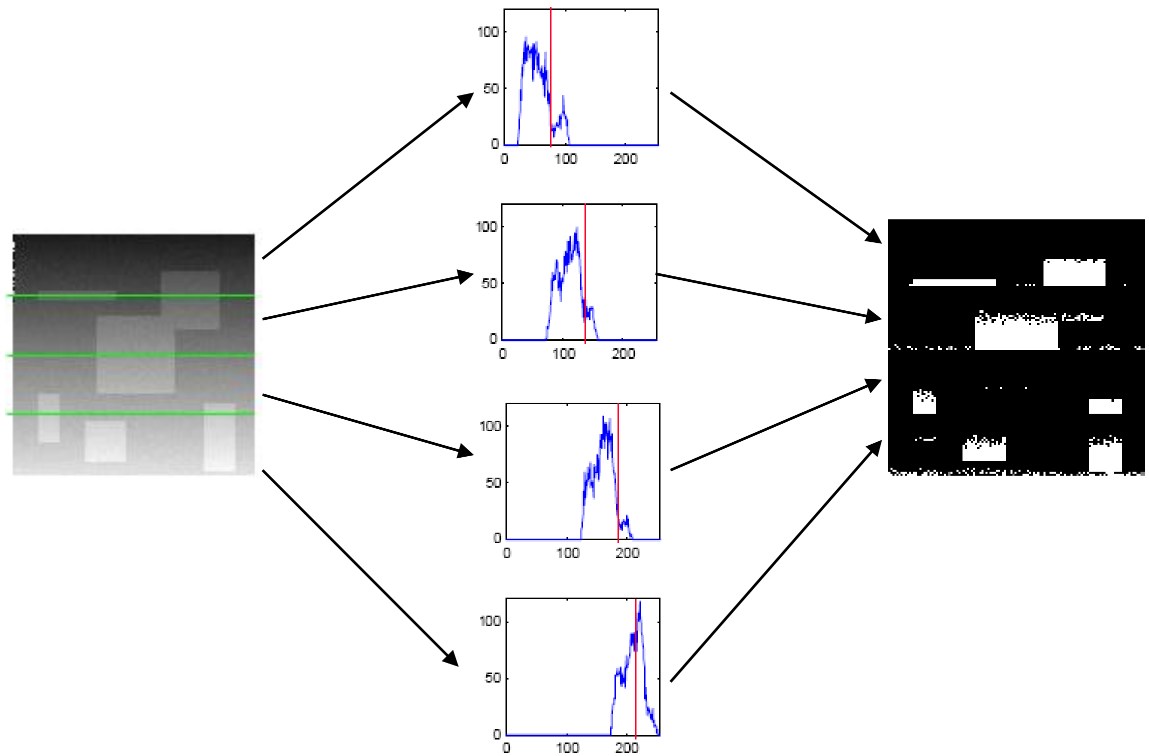


T = 128

Local Thresholding

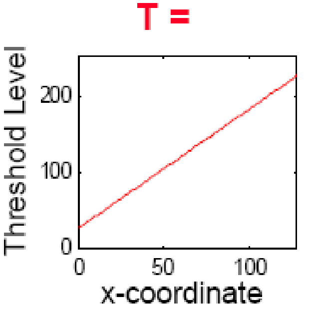
Divide image in to regions.

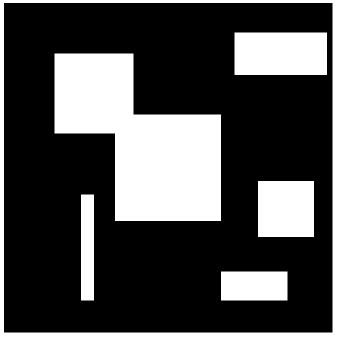
Perform thresholding independently in each region.



Adaptive Thresholding

Every pixel in image is thresholded according to the histogram of the pixel neighborhood.

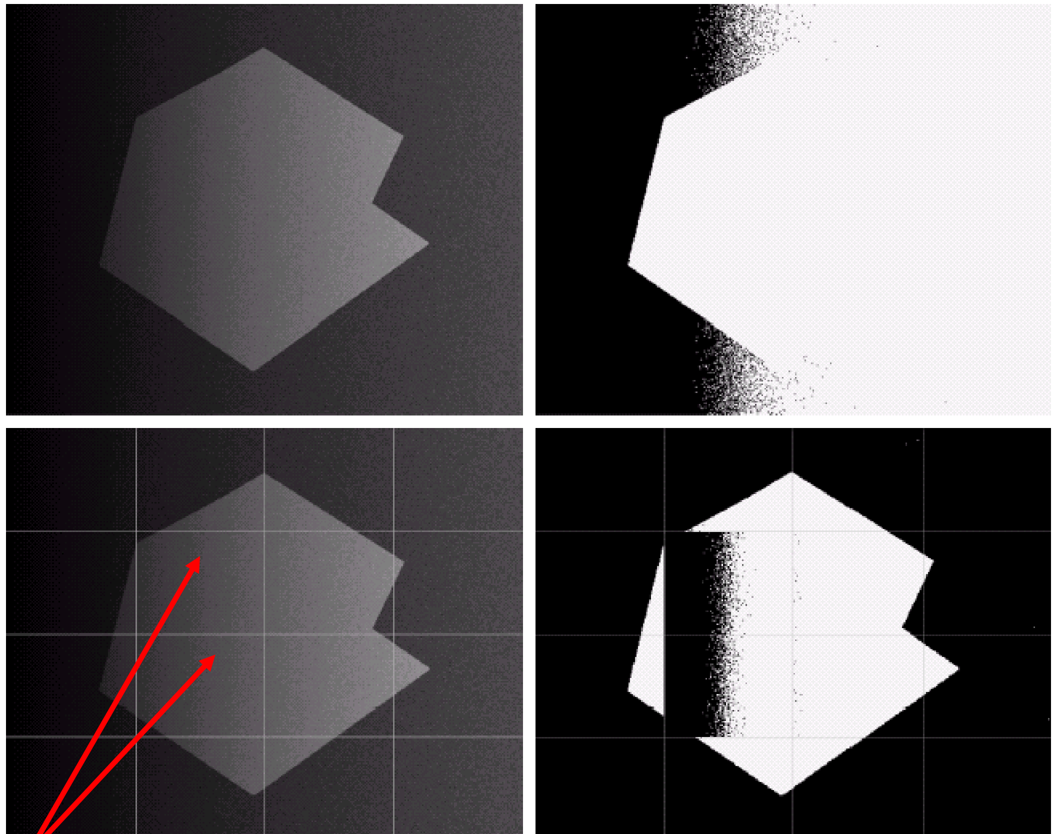




Adaptive Thresholding

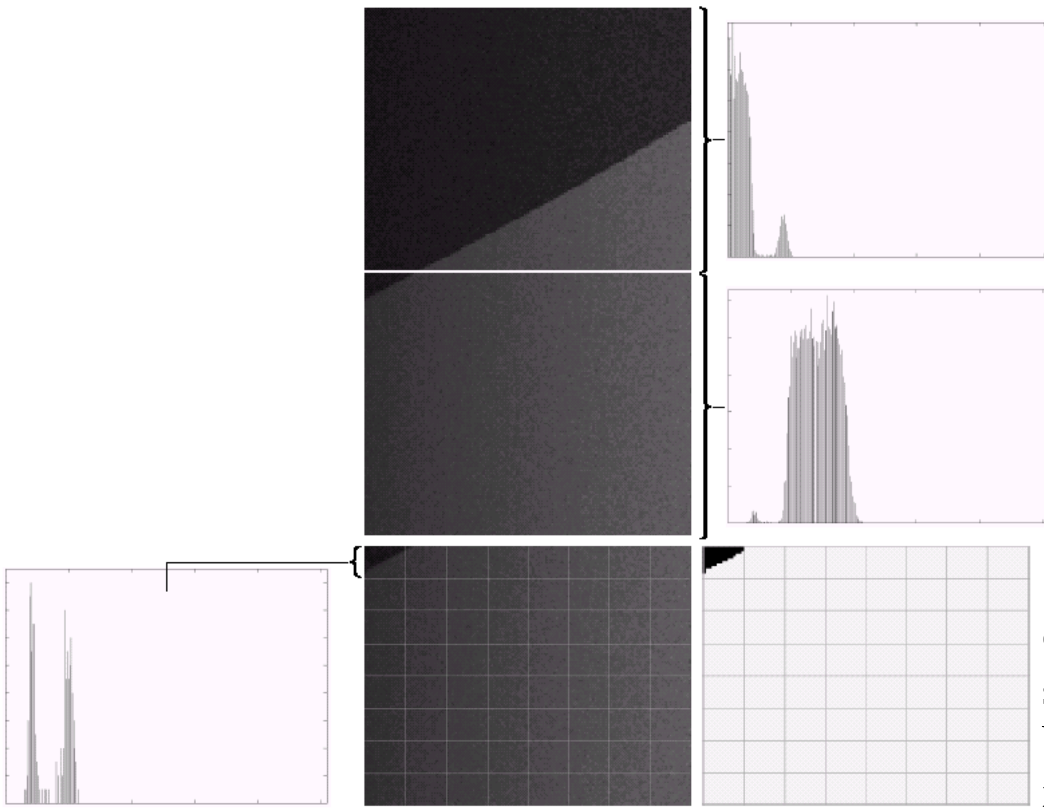
a	b
c	d

FIGURE 10.30 (a) Original image. (b) Result of global thresholding. (c) Image subdivided into individual subimages. (d) Result of adaptive thresholding.



almost unimodal histogram

Adaptive Thresholding



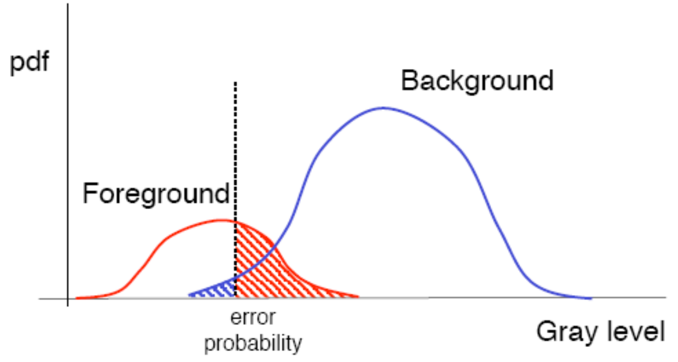
$\sigma^2 > 100$ $\sigma^2 < 75$

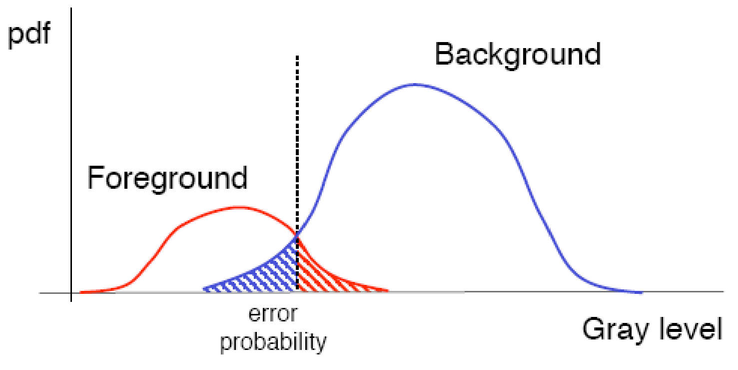
Considerable more accurate segmentation can be achieved by subdividing the entire image into subimages of this small size

Otsu Algorithm

- Otsu's method is named after Nobuyuki Otsu(大津展之).
- it is used to automatically perform histogram shape-based image thresholding, or, the reduction of a graylevel image to a binary image.
- The algorithm assumes that the image contains two classes of pixels or bi-modal histogram (e.g. foreground and background) then calculates the optimum threshold separating those two classes so that their combined spread (intra-class variance) is minimal.
- The extension of the original method to multi-level thresholding is referred to as the Multi Otsu method.

Otsu's Threshold





$$\omega_0 = \frac{N_0}{M * N}$$

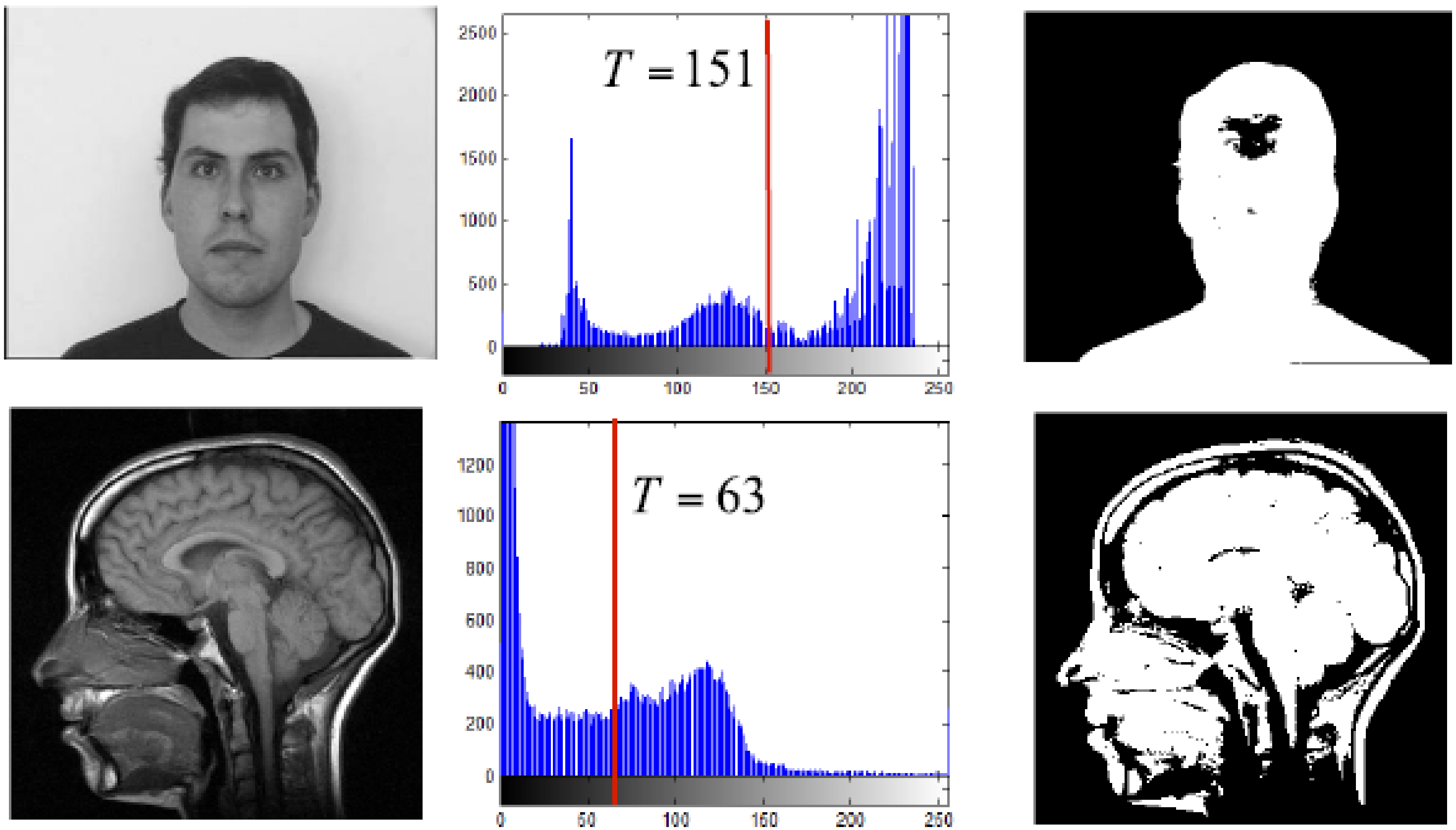
$$\omega_1 = \frac{N_1}{M * N}$$

$$N_0 + N_1 = M * N$$
$$\omega_0 + \omega_1 = 1$$

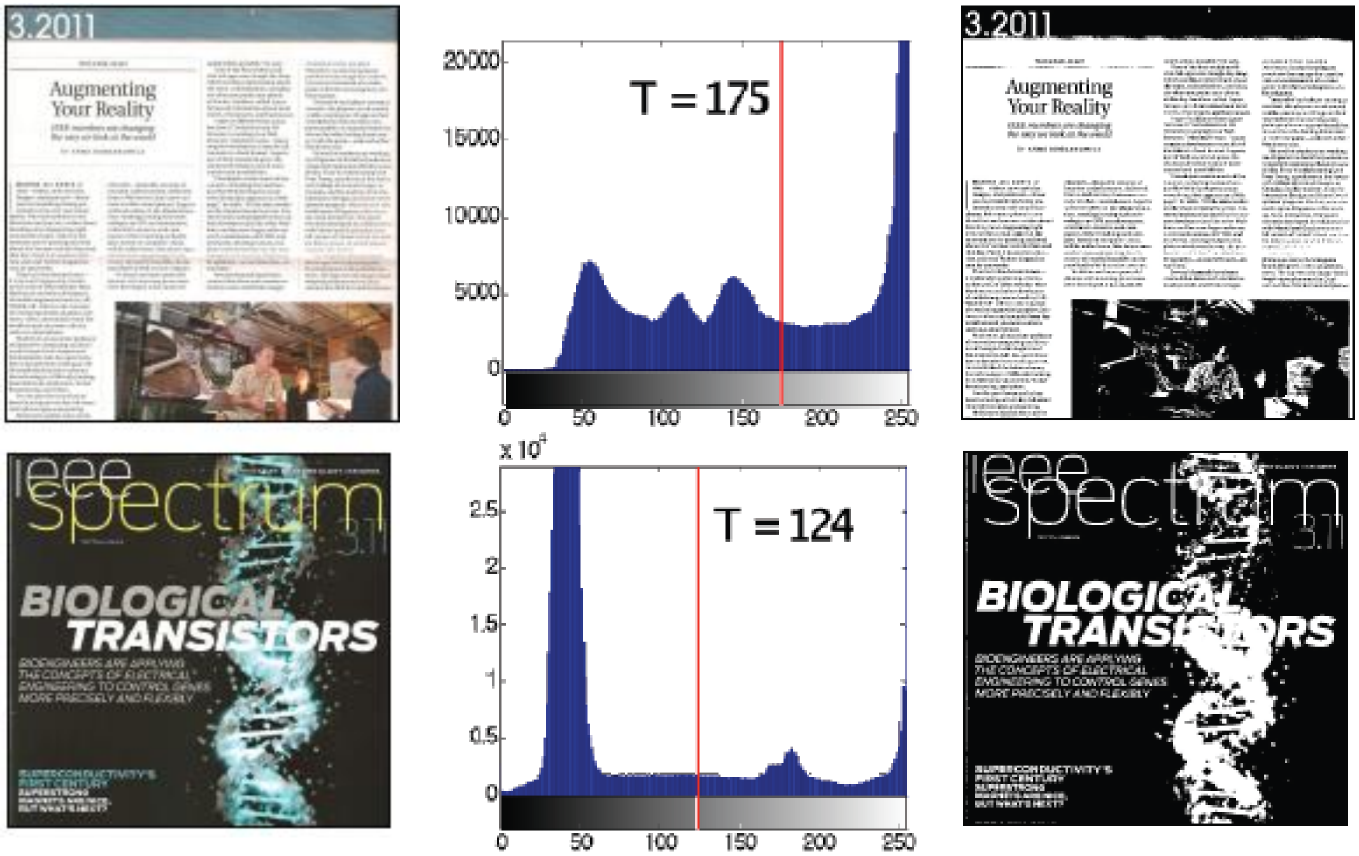
$$\begin{cases} \mu = \omega_0 * \mu_0 + \omega_1 * \mu_1 \\ g = \omega_0(\mu_0 - \mu)^2 + \omega_1(\mu_1 - \mu)^2 \end{cases}$$

$$\longrightarrow \quad g = \omega_0 \omega_1 (\mu_0 - \mu_1)^2$$

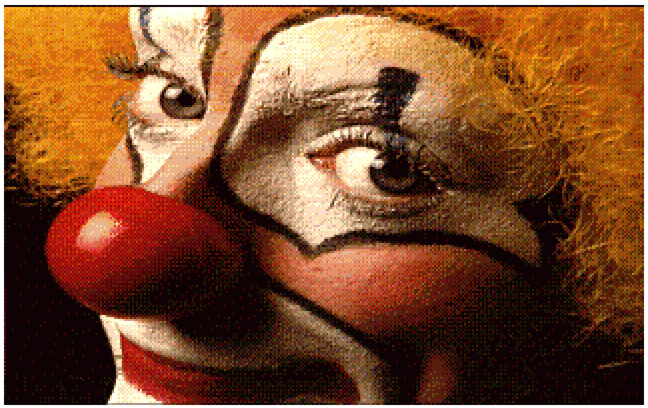
Otsu's Segmentation



Otsu’s Segmentation



Multi Otsu's Segmentation



n = 2

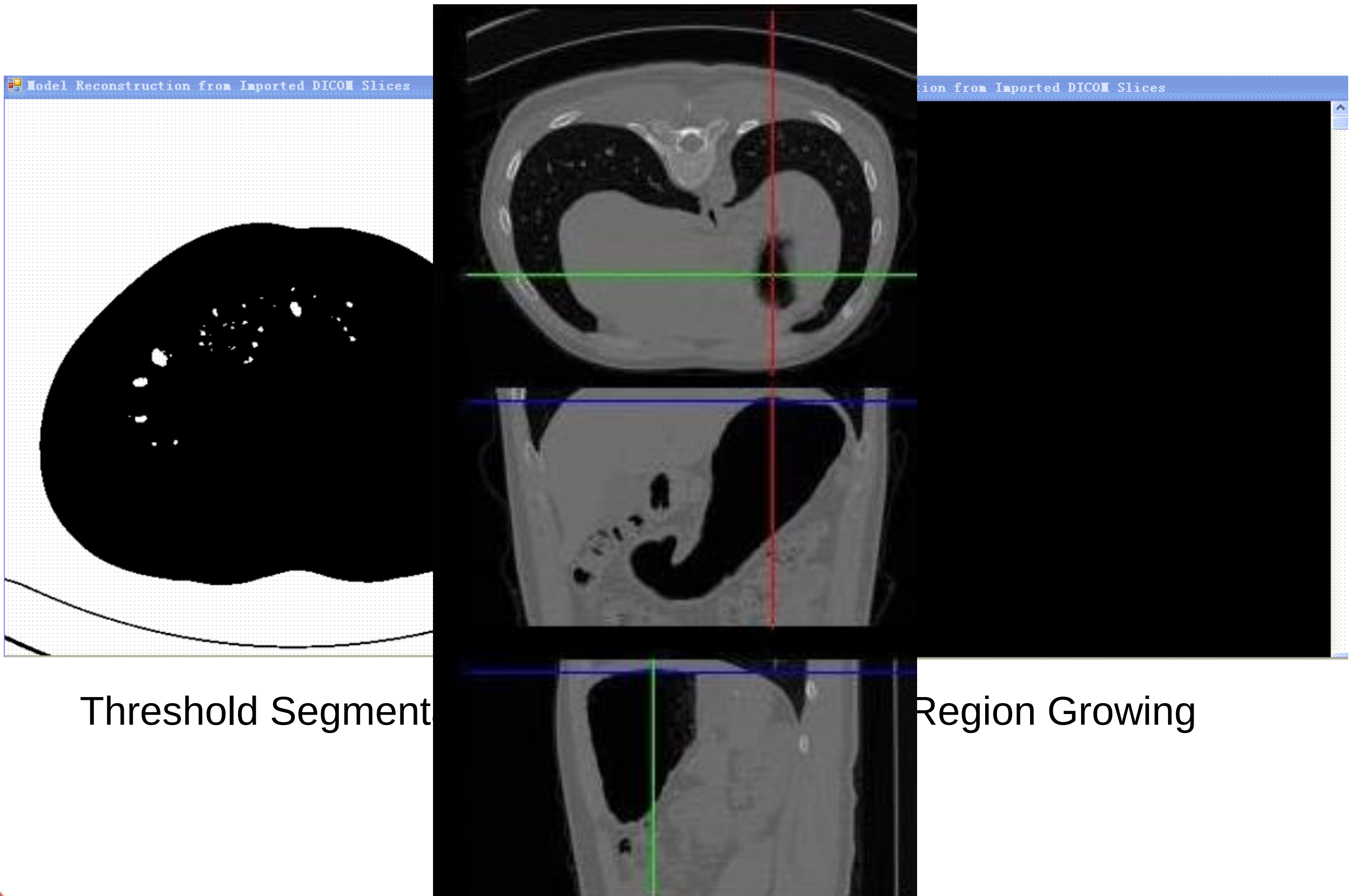
n = 3

n = 4

Outlines

- Detection of Discontinuities
- Edge Linking and Boundary Detection
- Thresholding
- Region-Based Segmentation
- Color Segmentation

Region Growing



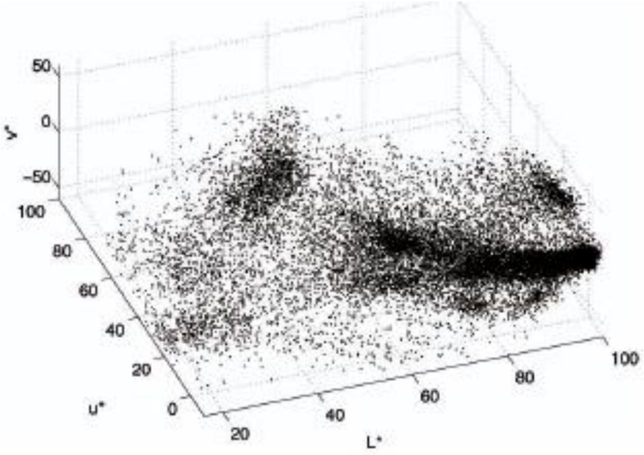
Threshold Segment

Region Growing

Clustering

Example of a feature space

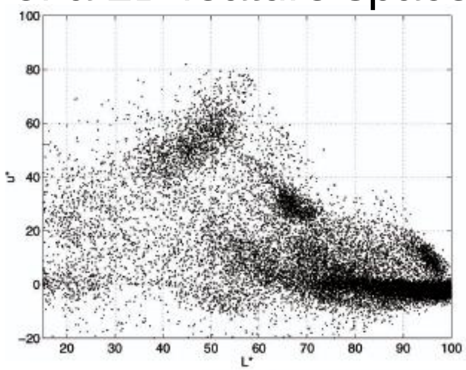
A 400*276 color image

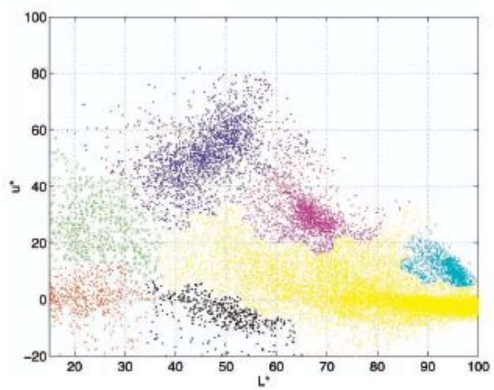


Corresponding L*u*v* color space with 110,400 data points

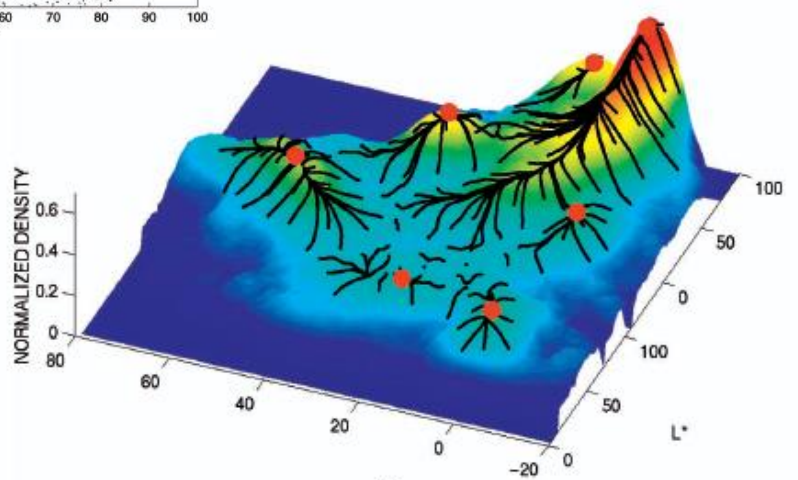
Clustering-Mean shift

Example of a 2D feature space analysis





Decomposition obtained by running 159 mean shift procedures with different initalizations



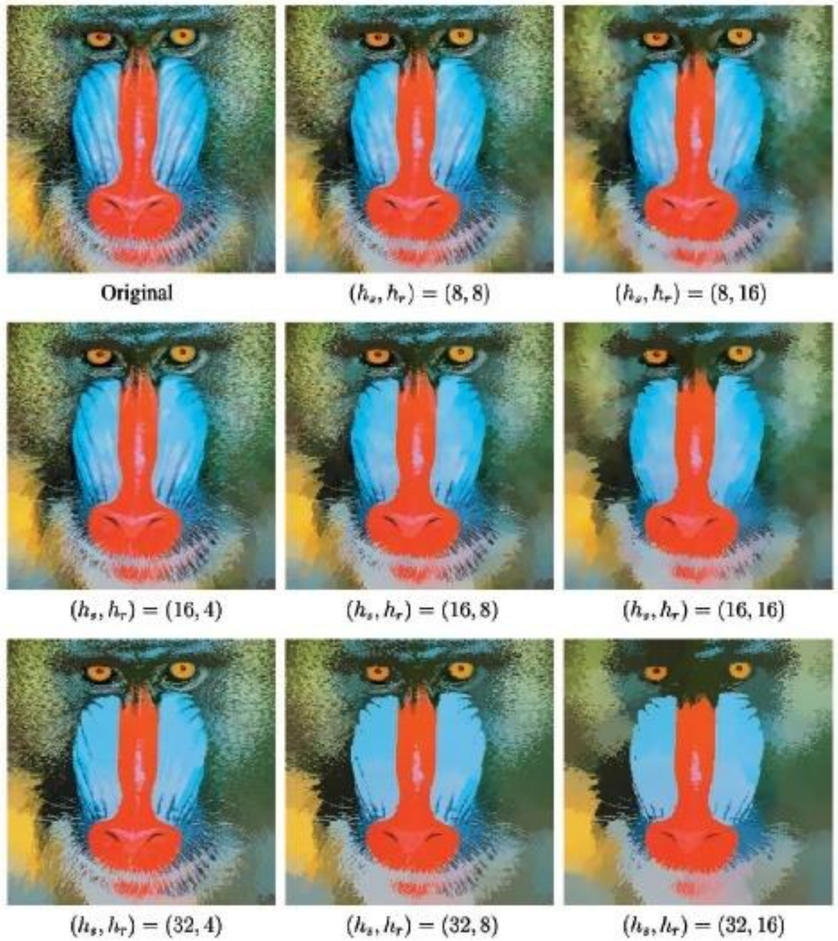
Trajectories of the mean shift procedures drawn over the Epanechnikov density estimated computed for the data set.

Clustering-Mean shift

original

Mean shift filtered $(h_s, h_r) = (8, 4)$

Clustering-Mean shift

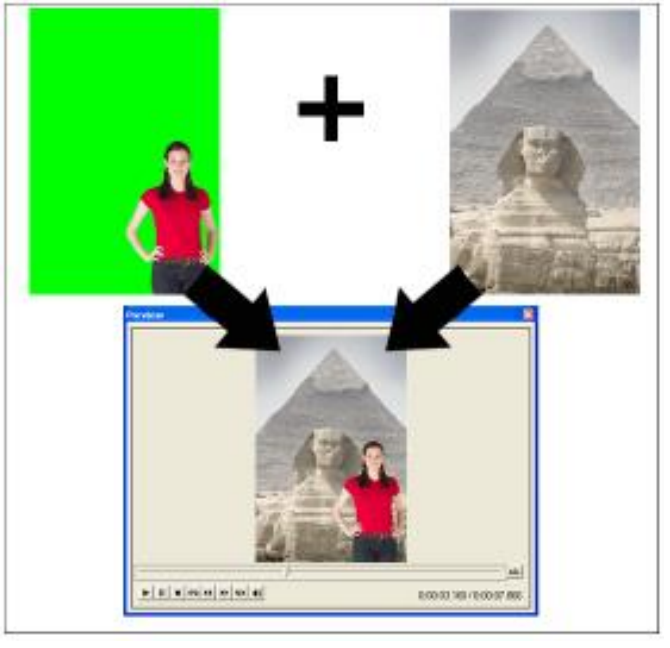


Outlines

- Detection of Discontinuities
- Edge Linking and Boundary Detection
- Thresholding
- Region-Based Segmentation
- Color Segmentation

Color Image Segmentation

- Color is more powerful for pixel-wise segmentation: 3-d vs. 1-d space
- Take picture in front of a blue screen (or green, or orange)



Color Image Segmentation

Original image

Skin color detector

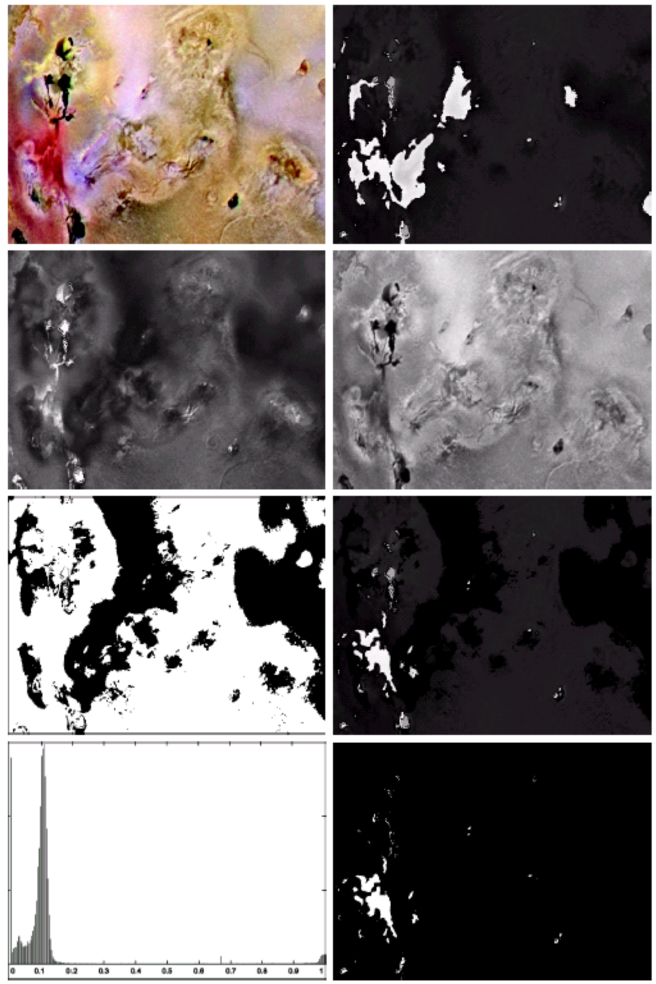
Color Segmentation – A Brief Introduction

- If we wish to segment an image based on color, it often implies to separate “a color” from others. In this case, unlike segmentation in gray-scale images, the intensity is of less important because it carries no color information.

- It is natural to think first of the HSI space because color is conveniently represented in the hue image.

- Typically, saturation is used as a masking image in order to isolate further regions of interest in the hue image.

Segmentation in HSI Color Space



Hue

Saturation

Intensity

binary mask by thresholding the saturation image 10%*max(S)

product of hue and the binary mask

histogram of the product image

segmentation result of reddish area by thresholding the product image (0.9)

Segmentation in RGB Vector Space

- Although working in HSI space is more intuitive, segmentation is one area in which better results generally are obtained by using RGB color vectors.
- The problems are how to define the interested color(s) and the measure of similarity.
- Given a set of sample color points representative of the colors of interest, we obtain an estimate of the "average" color denoted by RGB vector **a** that wish to segment.

Measure of Similarity

- The simplest measure is the Euclidean distance.
- Let **z** denote an arbitrary point in RGB space, the Euclidean distance between **z** and **a** is given by

$$
\begin{aligned}
D(\mathbf{z},\mathbf{a}) &= \|\mathbf{z}-\mathbf{a}\| = \left[(\mathbf{z}-\mathbf{a})^T(\mathbf{z}-\mathbf{a})\right]^{\frac{1}{2}} \\
&= \left[(z_R-a_R)^2+(z_G-a_G)^2+(z_B-a_B)^2\right]^{\frac{1}{2}}
\end{aligned}
\qquad (6.7\text{-}1)
$$

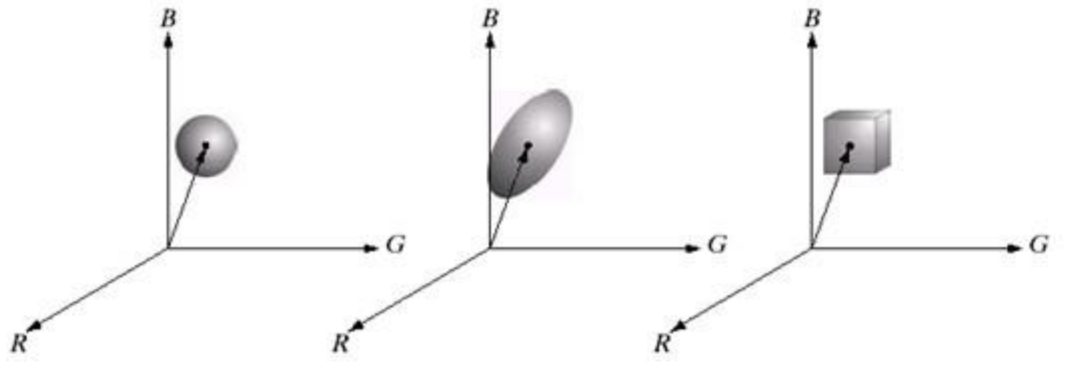
if D is less then a specified threshold D_0, we say **z** is similar to **a**. The locus of points such that D(**z**, **a**) ≦ D_0 is a solid sphere of radius D_0.

- A useful generalization of the Euclidean distance is of the form

$$D(\mathbf{z},\mathbf{a}) = \left[(\mathbf{z}-\mathbf{a})^T\mathbf{C}^{-1}(\mathbf{z}-\mathbf{a})\right]^{\frac{1}{2}}$$

where **C** is the covariance matrix of the samples representative of the color to be segmented. The locus of points such that D(**z**, **a**) ≦ D_0 describes a solid 3-D elliptical body whose principal axes are oriented in the direction of maximum data spread.

Measure of Similarity

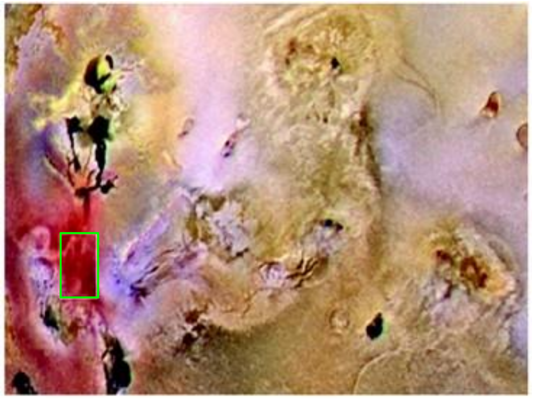


Three approaches for enclosing data regions for RGB vector segmentation

- To simplify the computation, use a bounding box instead of the spherical or elliptical enclosure.
- The box is centered on **a**, and its dimensions along each of the color axes is chosen proportional to the standard deviation of the samples along each of the axis.

An Example in RGB Vector Space

Segmentation in RGB space



Original image with colors of interest shown enclosed by a rectangle

Result of segmentation in RGB vector space

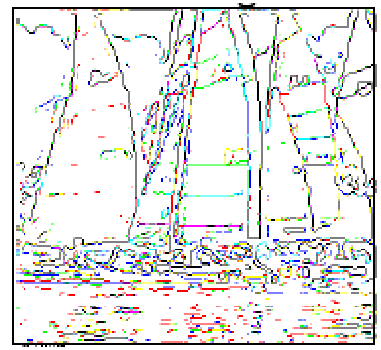
$L = 1.25\sigma$

Result of segmentation in HSI vector space

Color Edge Detection

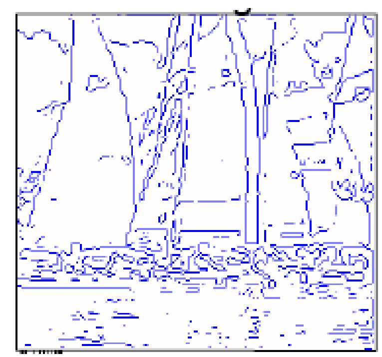
Original image

All-edges



R-edges

G-edges



B-edges

Color Edge Detection

Original image

All-edges

R-edges

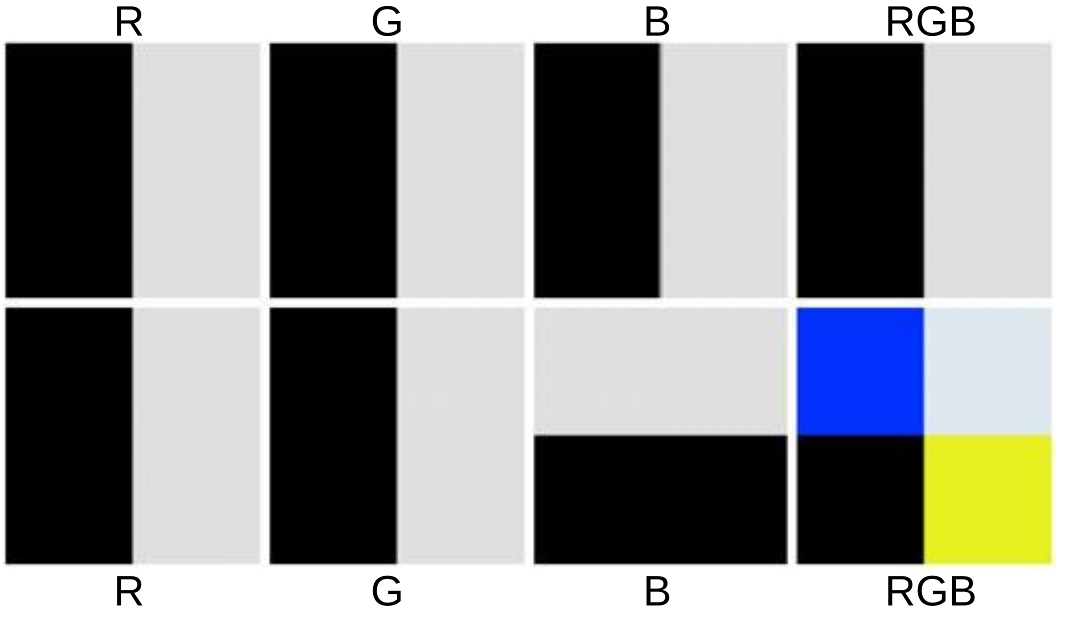
G-edges

B-edges

Color Edge Detection

Example:

In both cases (a) and (b), the standard approach would give the same gradient magnitude at the center of the image However in (b) we would expect a lower magnitude as only two edges are in the same direction.



Color Edge Detection

- The goal is to find a vector pointing in the direction of maximum rate of change of

$$\mathbf{c}(\mathrm{x,y})=[R(x,y)\ G(x,y),B(x,y)]^{\mathrm{T}}$$

  (this is the definition of the gradient).

- Let r, g and b be unit vectors along the R, G and B axes and define:

$$\mathbf{u}=\frac{\partial R}{\partial x}\mathbf{r}+\frac{\partial G}{\partial x}\mathbf{g}+\frac{\partial B}{\partial x}\mathbf{b} \qquad \mathbf{v}=\frac{\partial R}{\partial y}\mathbf{r}+\frac{\partial G}{\partial y}\mathbf{g}+\frac{\partial B}{\partial y}\mathbf{b}$$

Color Edge Detection

Let also:

$$g_{xx} = \mathbf{u} \bullet \mathbf{u} = \mathbf{u}^T \bullet \mathbf{u} = \left|\frac{\partial R}{\partial x}\right|^2 + \left|\frac{\partial G}{\partial x}\right|^2 + \left|\frac{\partial B}{\partial x}\right|^2$$

$$g_{yy} = \mathbf{v} \bullet \mathbf{v} = \mathbf{v}^T \bullet \mathbf{v} = \left|\frac{\partial R}{\partial y}\right|^2 + \left|\frac{\partial G}{\partial y}\right|^2 + \left|\frac{\partial B}{\partial y}\right|^2$$

$$g_{xy} = \mathbf{u} \bullet \mathbf{v} = \mathbf{u}^T \bullet \mathbf{v} = \frac{\partial R}{\partial x}\frac{\partial R}{\partial y} + \frac{\partial G}{\partial x}\frac{\partial G}{\partial y} + \frac{\partial B}{\partial x}\frac{\partial B}{\partial y}$$

Color Edge Detection

- The direction of maximum rate of change of **c**(x,y)=[$R(x,y)$ $G(x,y)$,$B(x,y)$]^T is [Di Zenzo 86]:

$$\theta(x,y) = \frac{1}{2}\tan^{-1}\left(\frac{2g_{xy}}{g_{xx}-g_{yy}}\right)$$

- and the value of that rate of change is:

$$F_\theta(x,y) = \left\{\frac{1}{2}\left[(g_{xx}+g_{yy}) + (g_{xx}-g_{yy})\cos\left(2\theta(x,y)\right) + 2g_{xy}\sin\left(2\theta(x,y)\right)\right]\right\}^{\frac{1}{2}}$$

Color Edge Detection

RGB image

Gradient computed in RGB color vector space

Gradient computed on a per-image basis and then added

Difference between two gradient image

The end