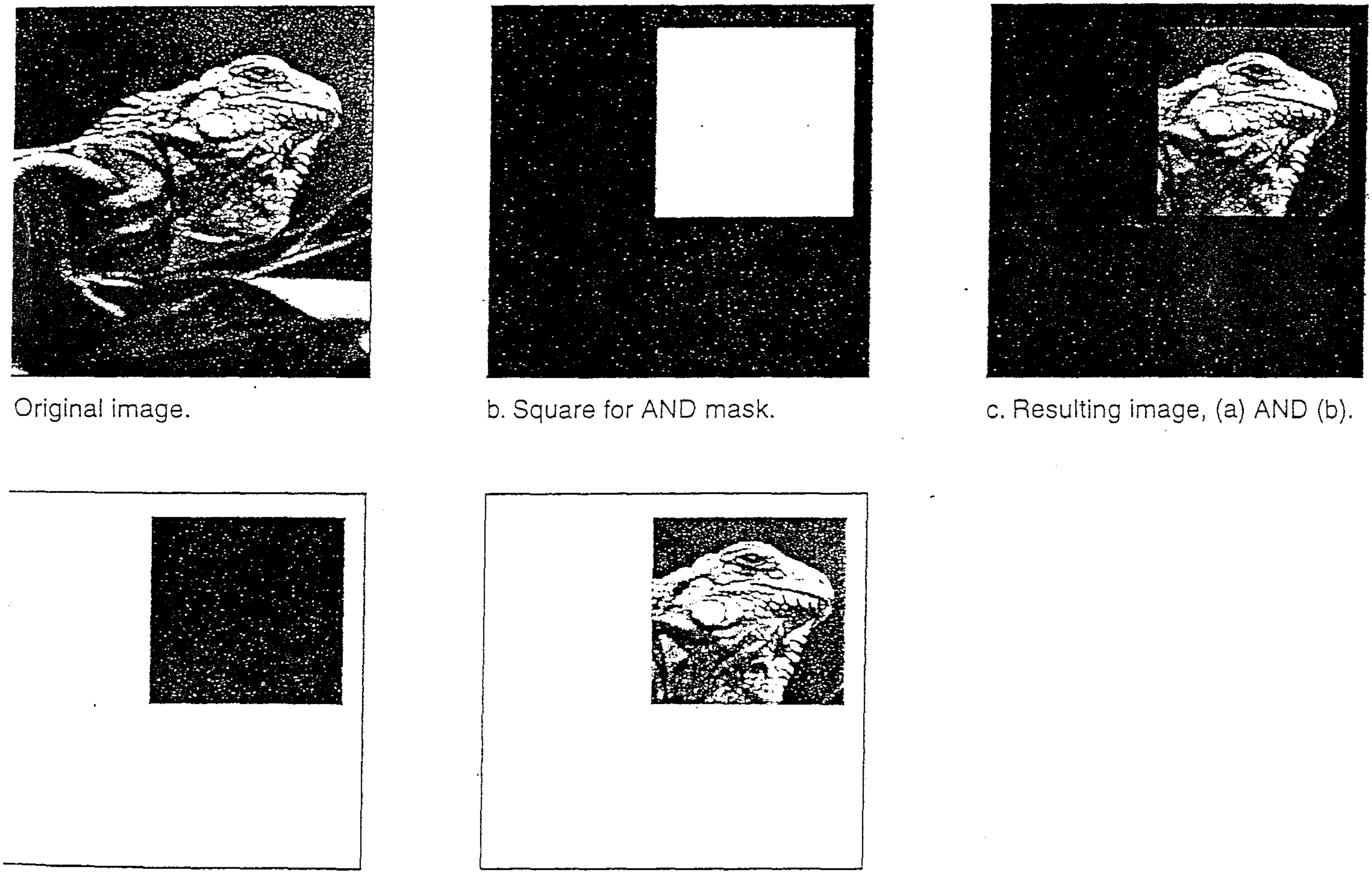
[illegible]

2.2.3 Spatial Filters

Spatial filtering is typically done for noise removal or to perform some type of image enhancement. These operators are called spatial filters to distinguish them from frequency domain filters, which are discussed in Section 2.5. The three types of filters discussed here include: 1) mean filters, 2) median filters, and 3) enhancement

igure 2.2-8 Image Masking



Original image.

b. Square for AND mask.

c. Resulting image, (a) AND (b).

Square for OR mask.

e. Resulting image, (a) OR (d).

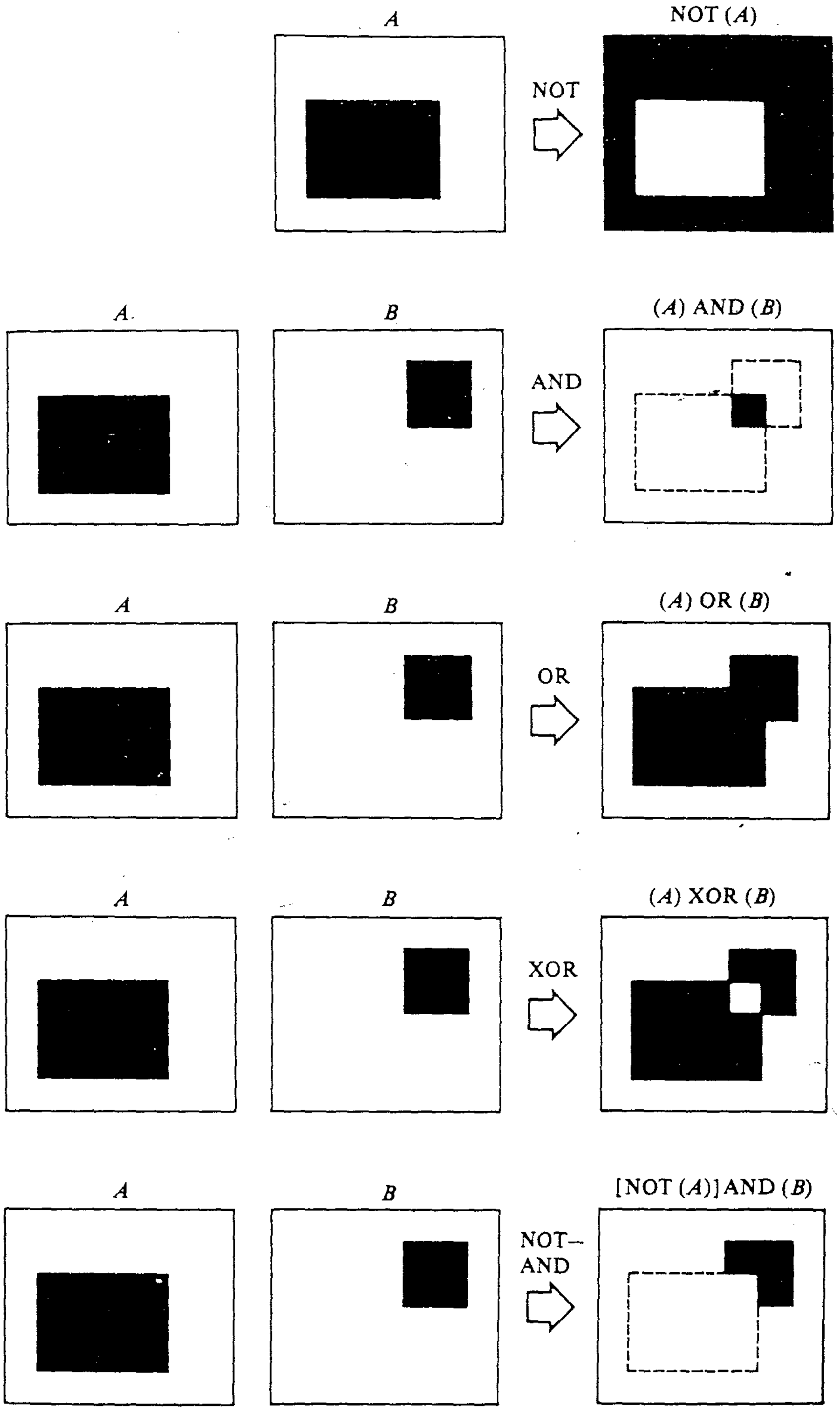
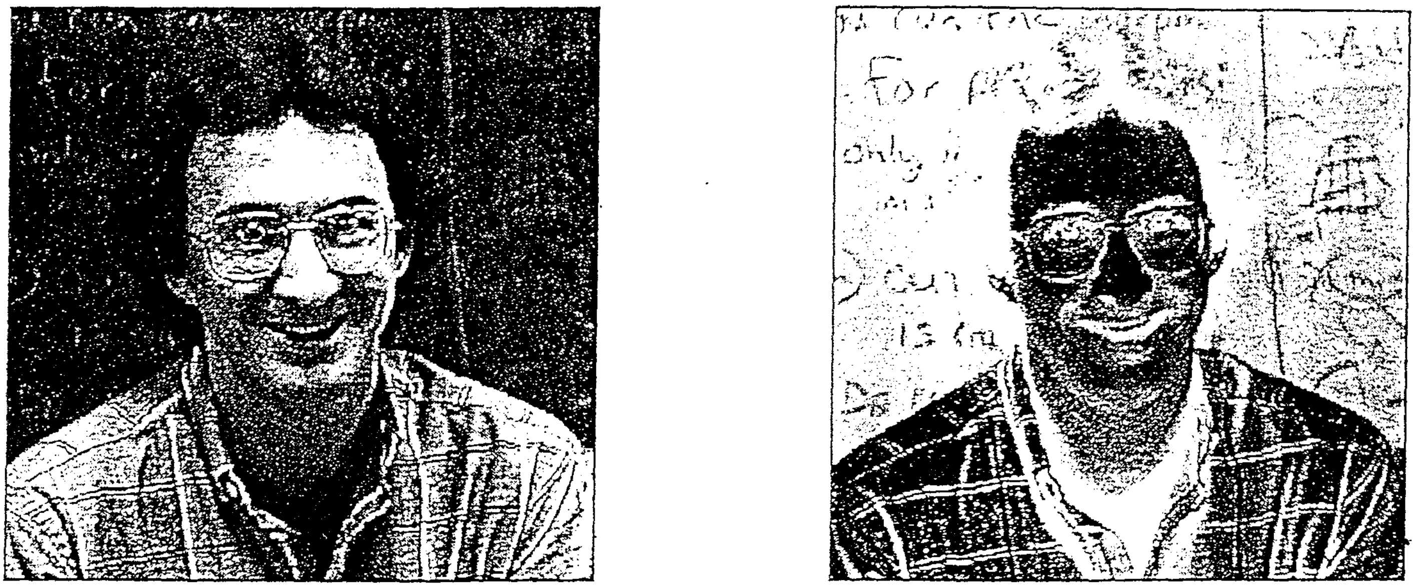


Figure 2.14 *Some examples of logic operations on binary images.*

2-9 Complement Image



a. Original image.

b. NOT operator applied to image.

9.1.6 Image subtraction

f moving
ir difference.

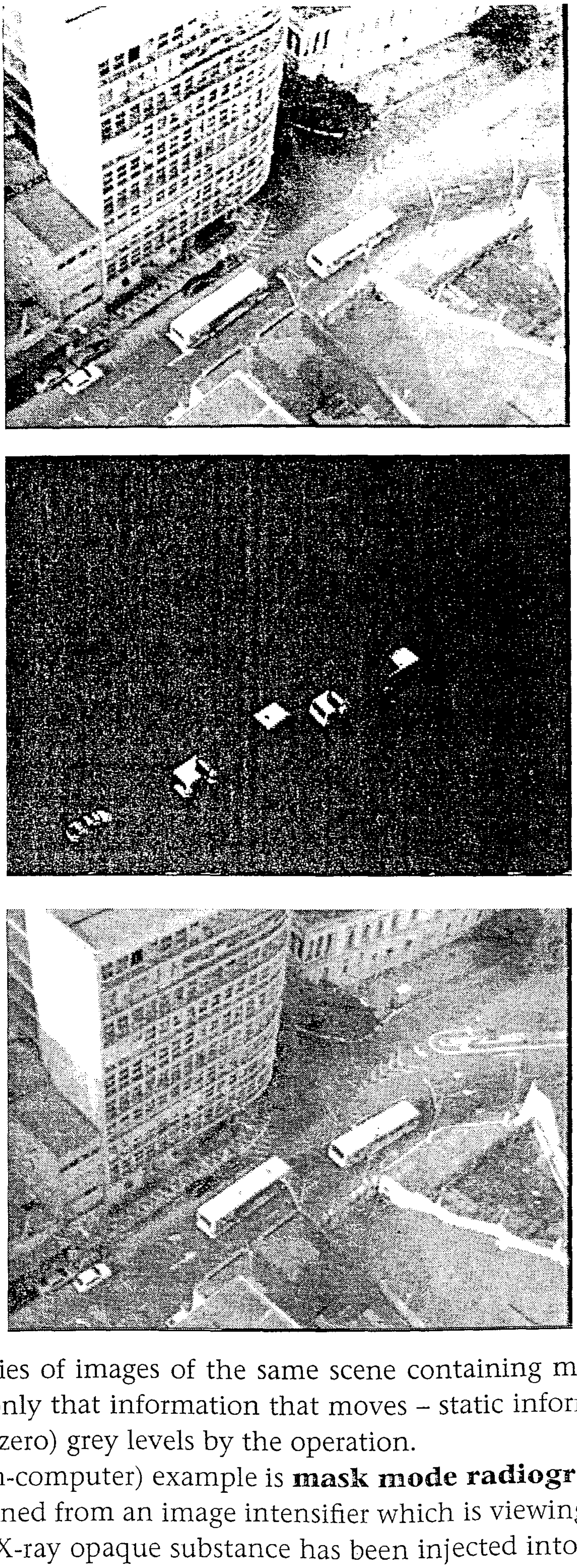


image and a series of images of the same scene containing motion, the subtrahends contain only that information that moves – static information is reduced to zero (or near zero) grey levels by the operation.

A classic (non-computer) example is **mask mode radiography**. Here X-ray images are obtained from an image intensifier which is viewing an area of blood vessels after an X-ray opaque substance has been injected into the bloodstream.

xample: Figure 4.17(a) shows an x-ray image of the top of a patient's head ior to injection of an iodine dye into the bloodstream. The camera yielding is image was positioned above the patient's head, looking down. As a reference point, the bright spot in the lower one-third of the image is the core of ie spinal column. Figure 4.17(b) shows the difference between the mask (Fig. 17a) and an image taken sometime after the dye was introduced into the loodstream. The bright arterial paths carrying the dye are unmistakably enanced in Fig. 4.17(b). These arteries appear quite bright because they are no ıbtracted out (that is, they are not part of the mask image). The overal ackground is much darker than that in Fig. 4.17(a) because differences beveen areas of little change yield low values, which in turn appear as dark

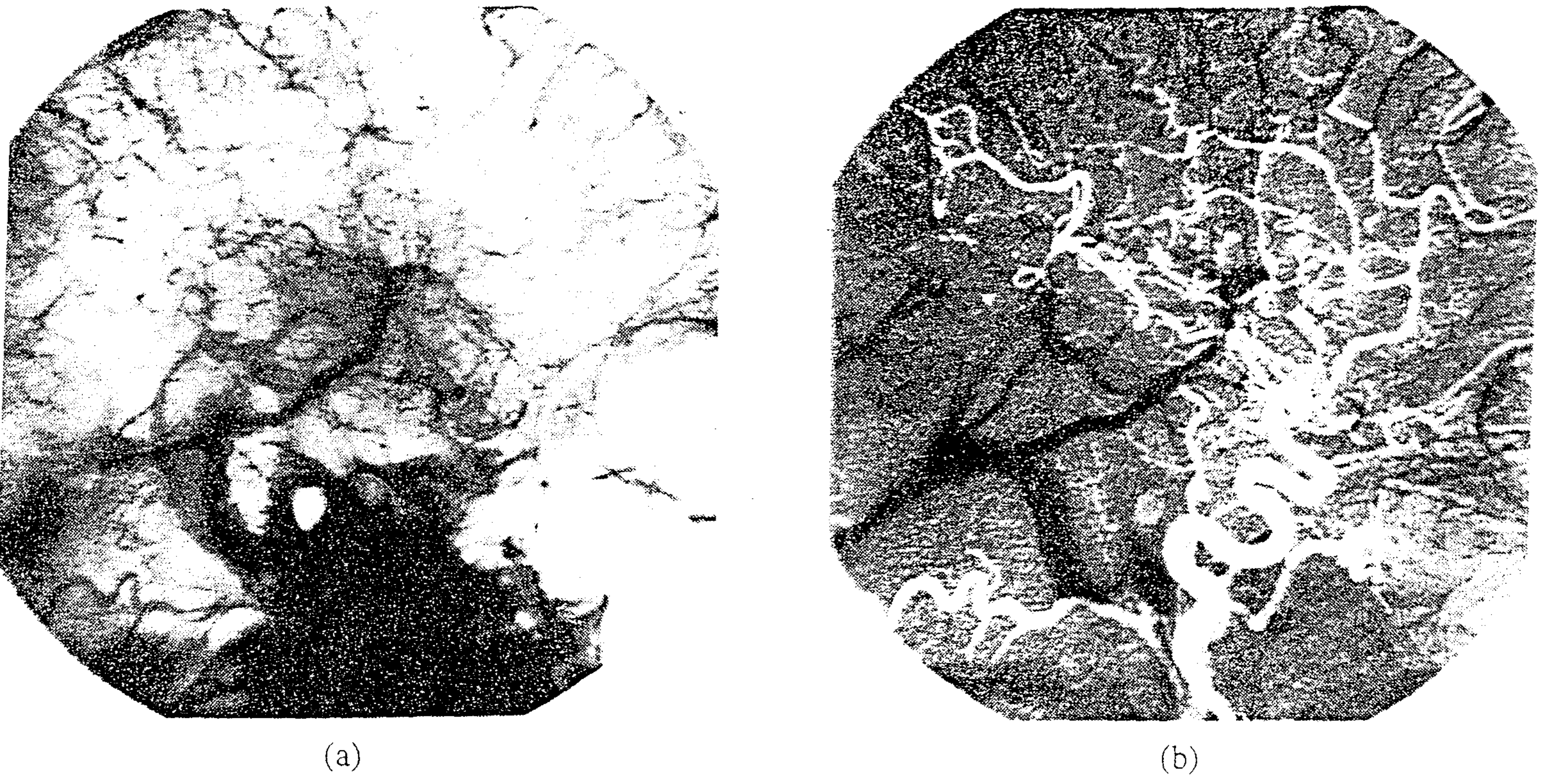


Figure 4.17 *Enhancement by image subtraction: (a) mask image; (b) image (after injectic of dye into the bloodstream) with mask subtracted out.*

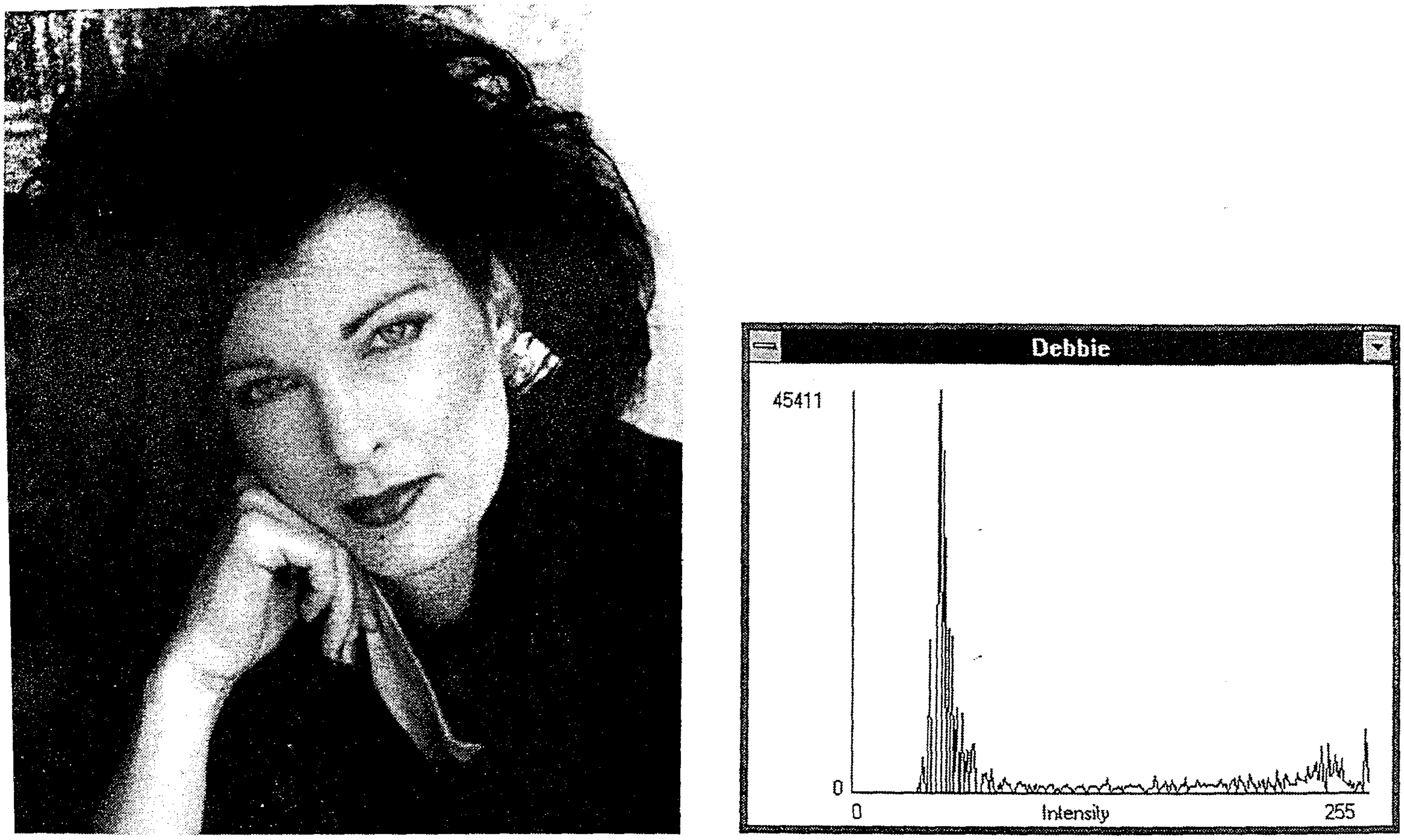


Figure 5–1 An image and its gray-level histogram

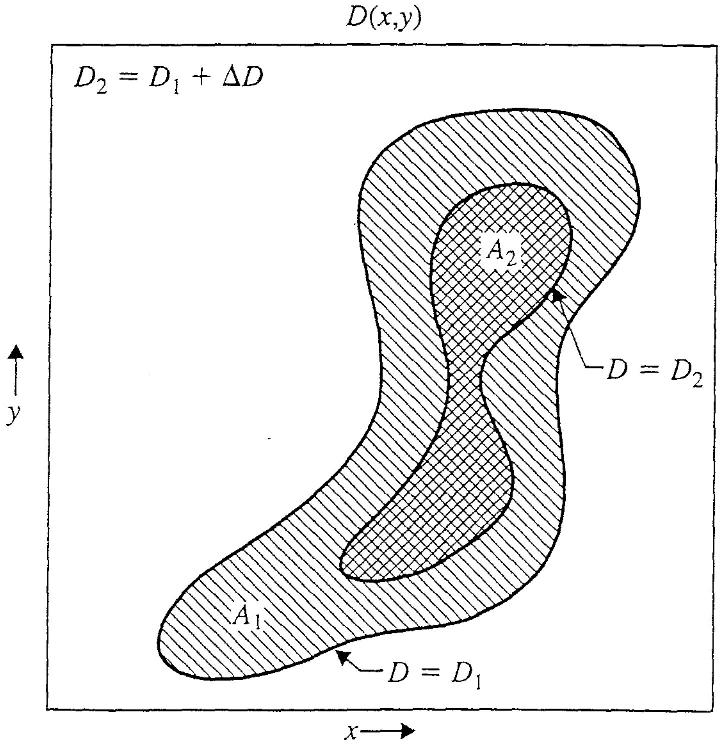


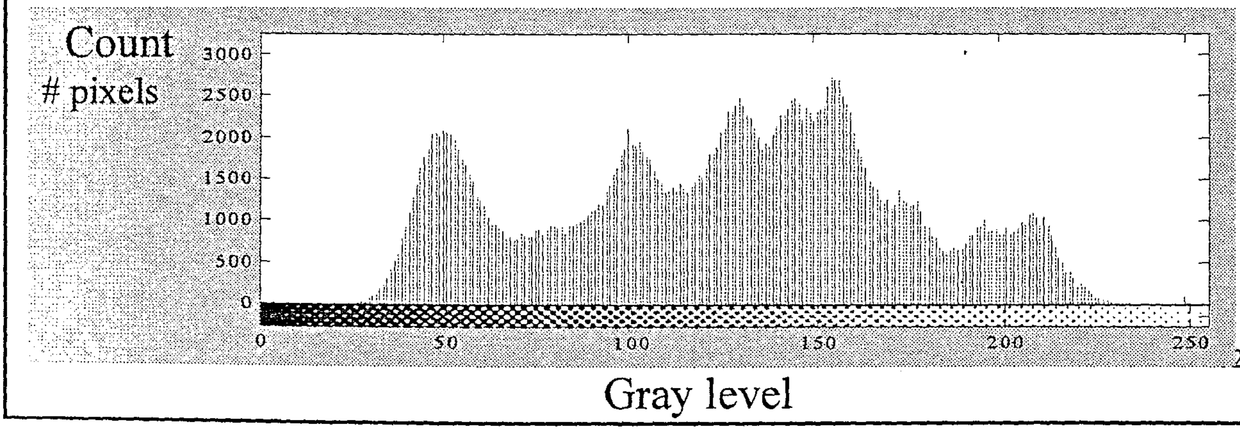
Figure 5–2 Contour lines in an image

Histogram (Normal)

x – image matrix (<u>intensity</u>)

`imhist(x)` **– display histogram**

```
[count, bin] = imhist(x);
```



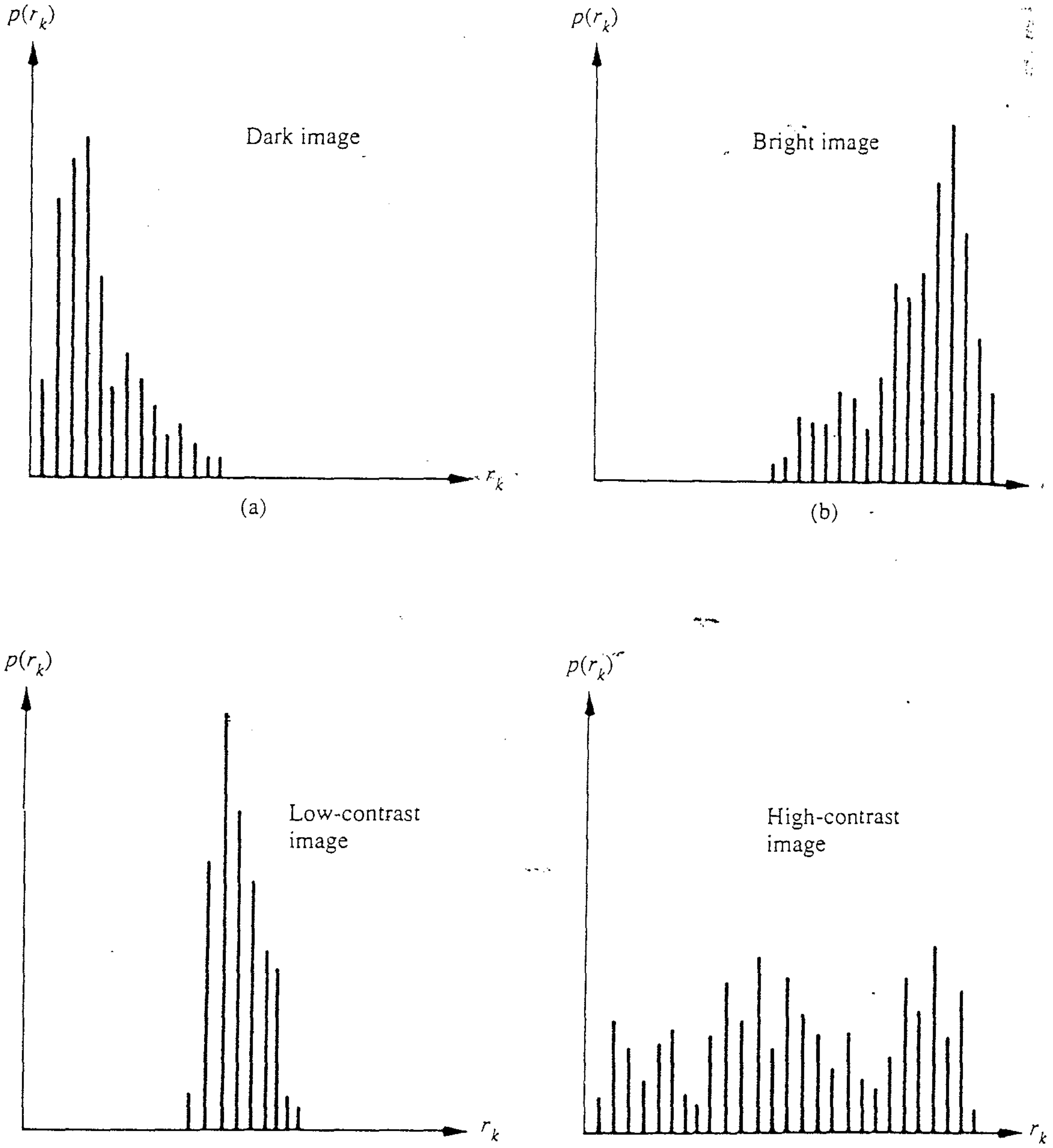
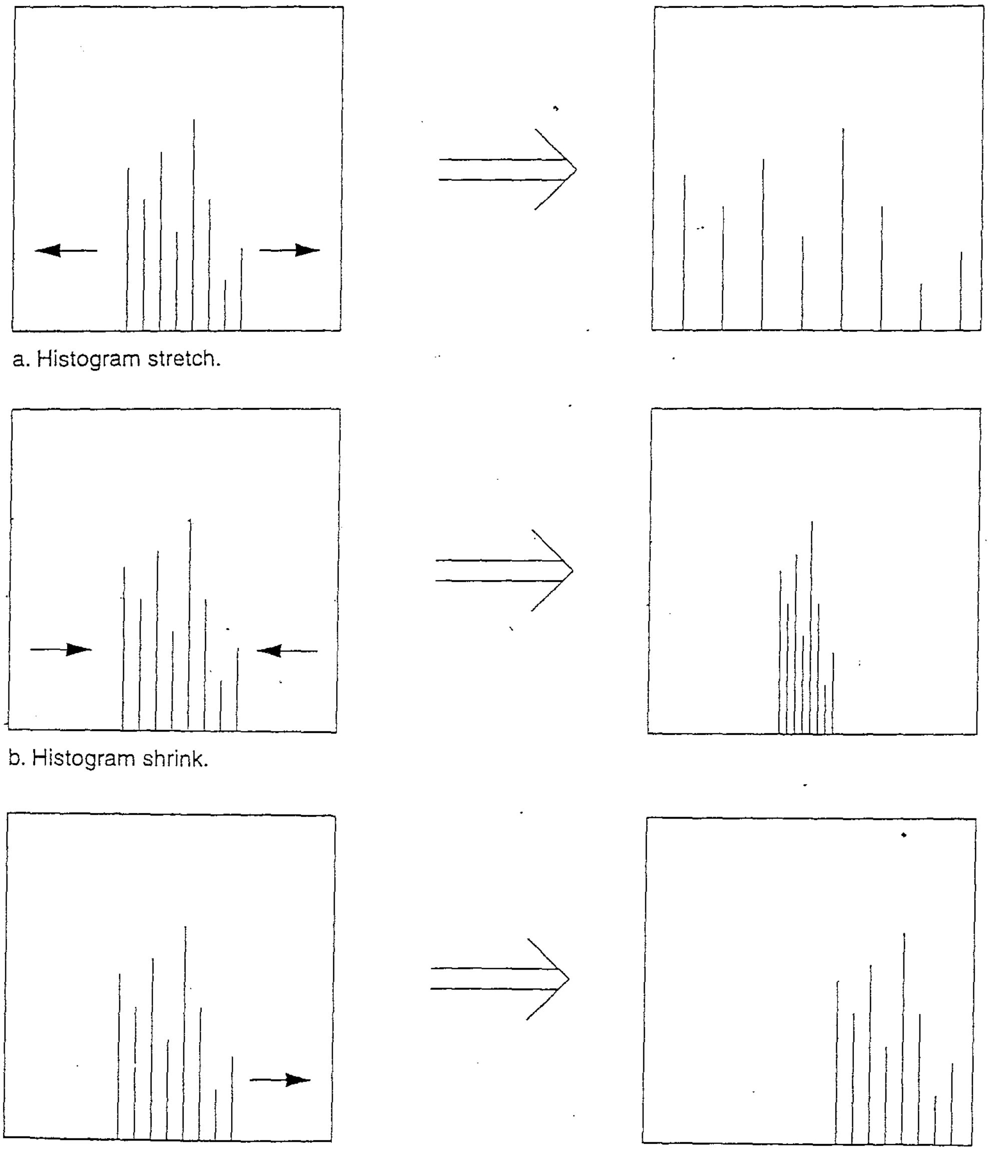


Figure 4.10 *Histograms corresponding to four basic image types.*

Figure 4.2-4 Histogram Modification



מתיחה

a. Histogram stretch.

מכווס

b. Histogram shrink.

הזזה

c. Histogram slide.

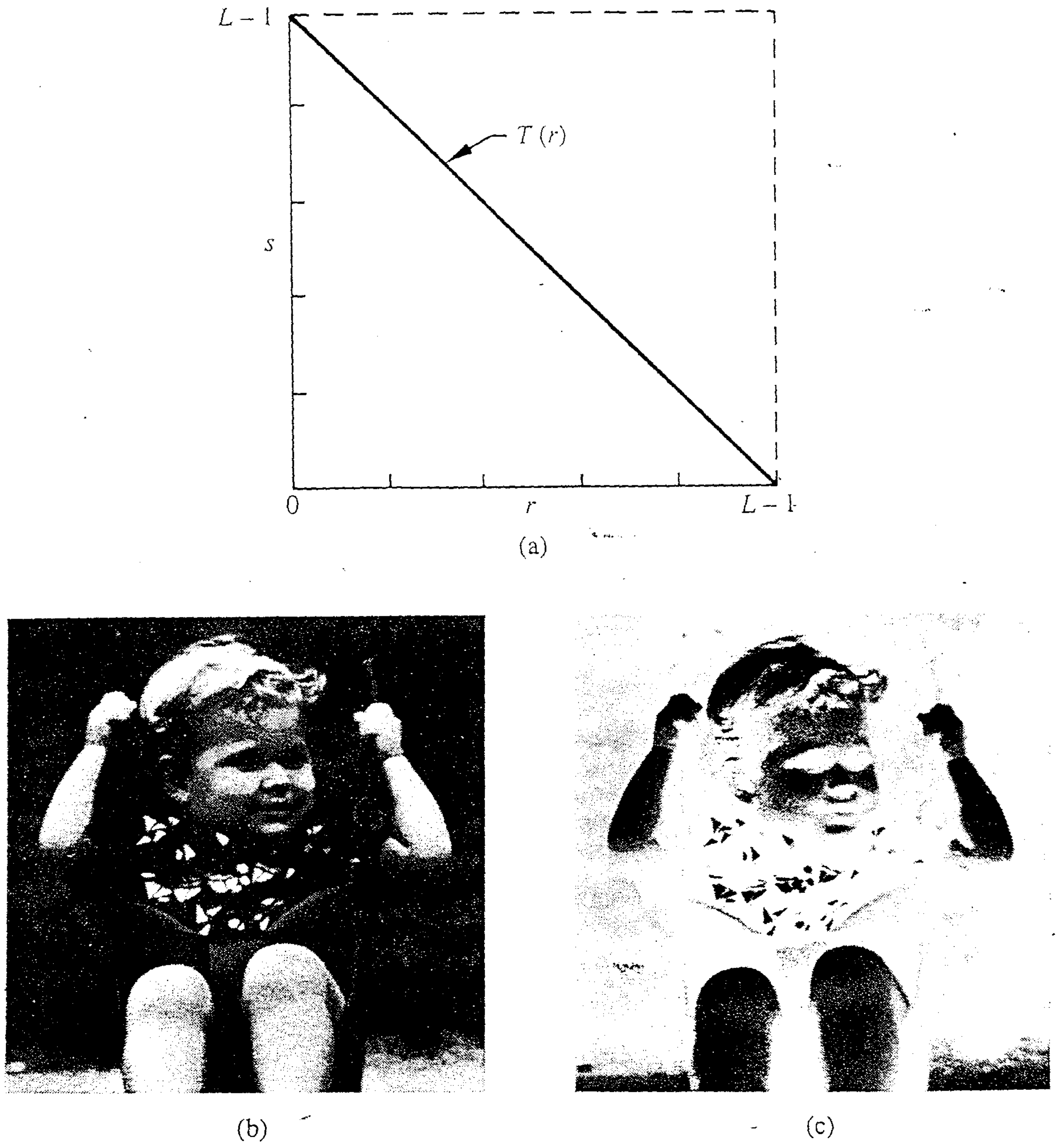


Figure 4.4 Obtaining the negative of an image: (a) gray-level transformation function; (b) an image; and (c) its negative. In (a), r and s denote the input and output gray levels, respectively.

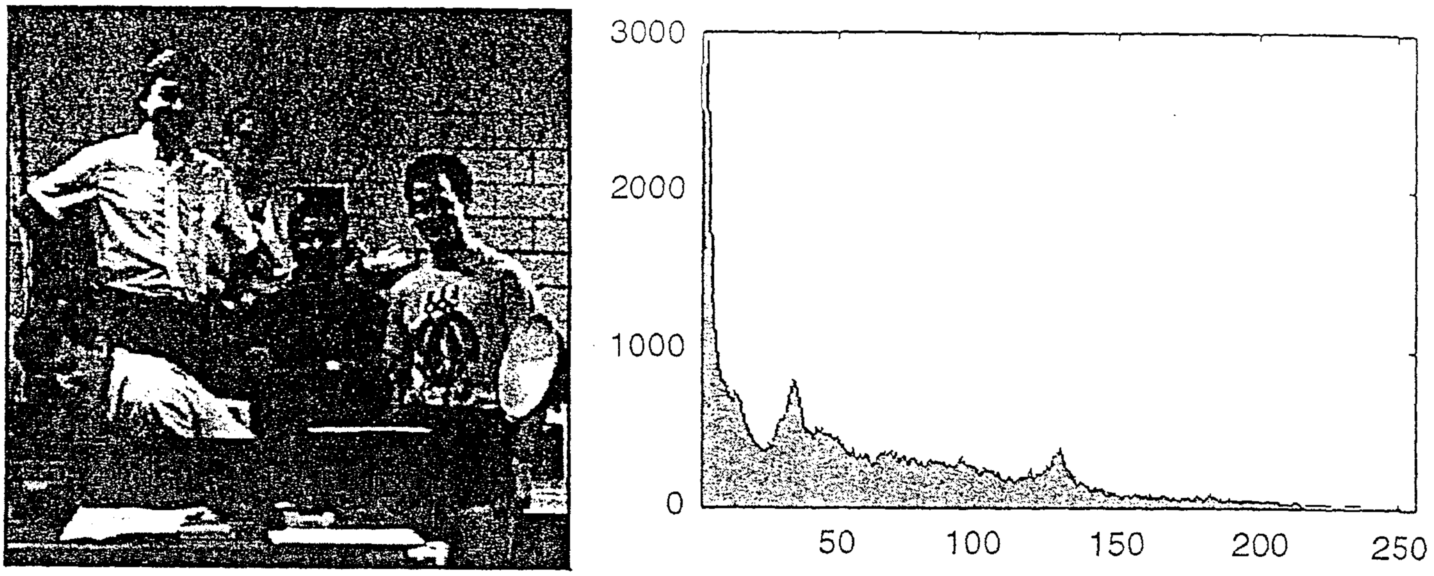


FIGURE 2 The digital image "students" (left) and its histogram (right). The gray levels of this image are skewed toward the left, and the image appears slightly underexposed.

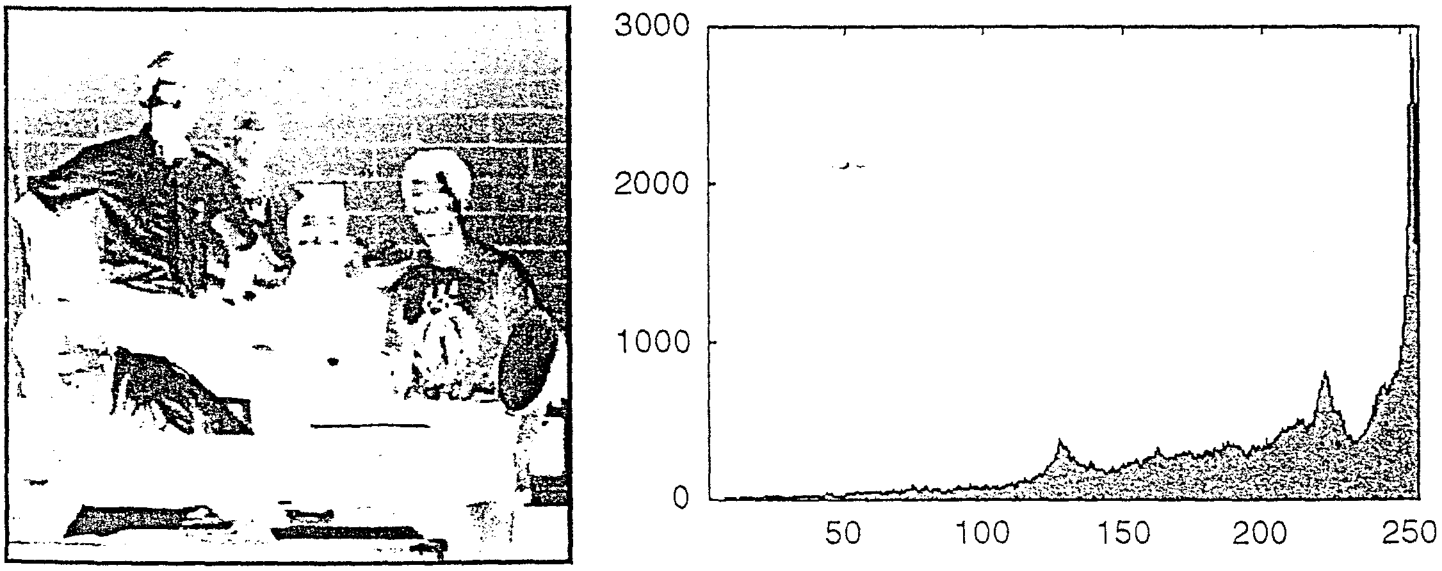
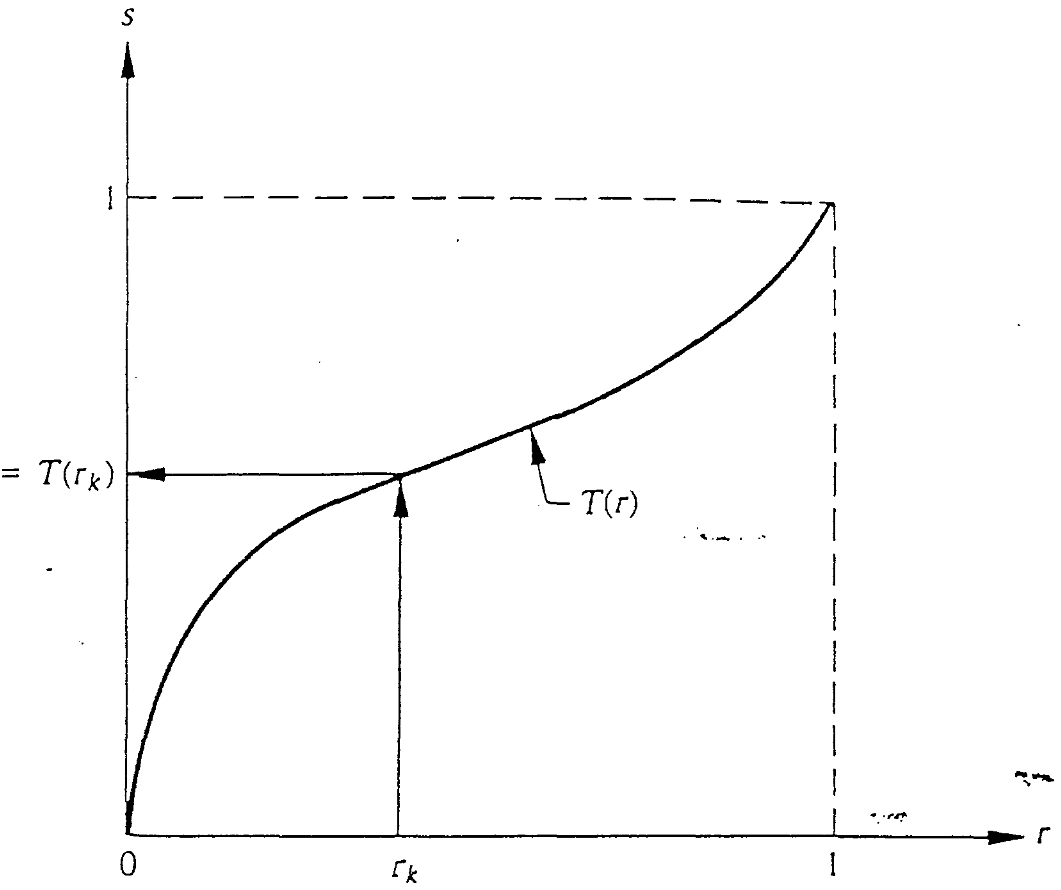


FIGURE 11 Example of an image negative with the resulting reversed histogram.



ay-level transformation function.

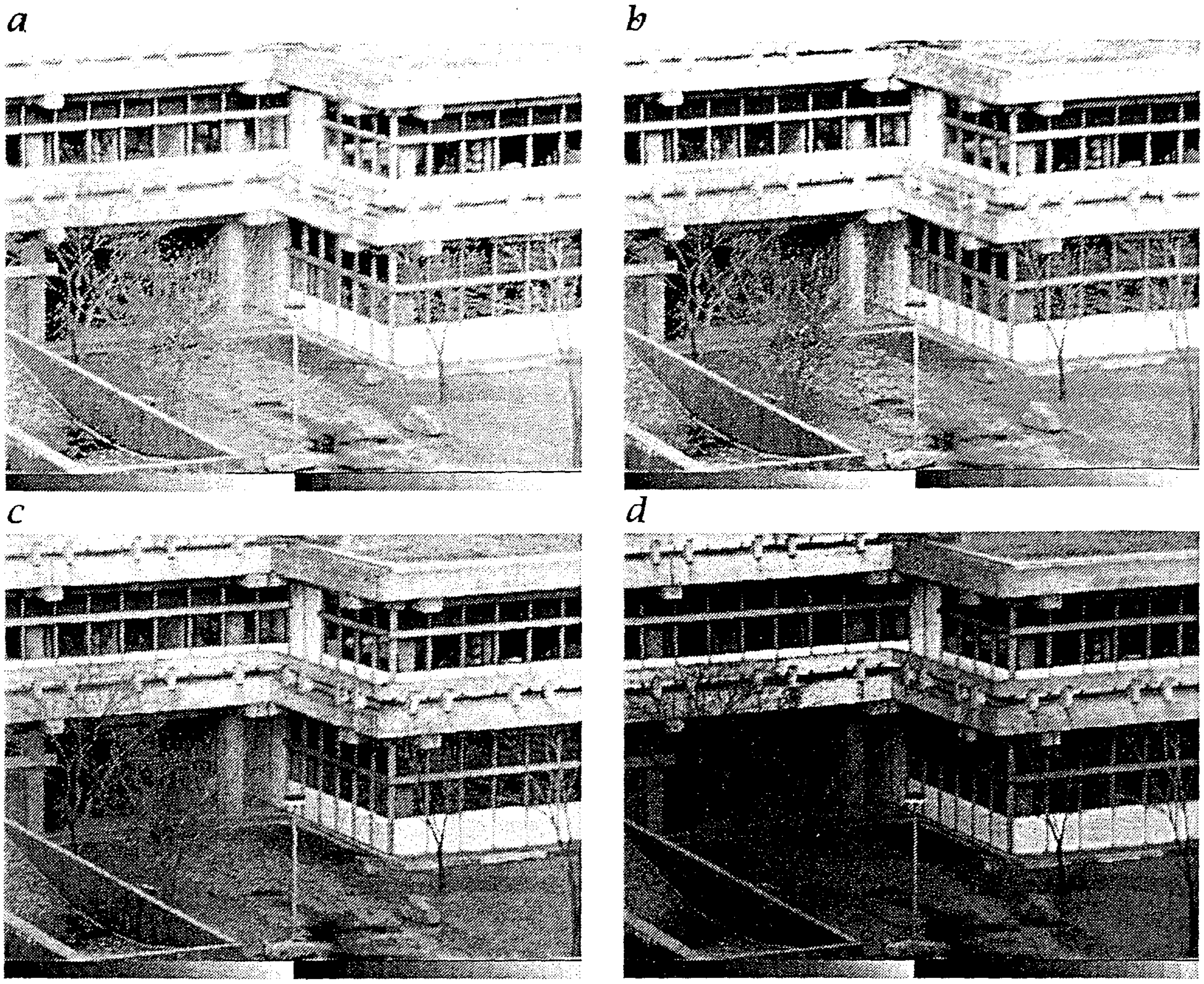


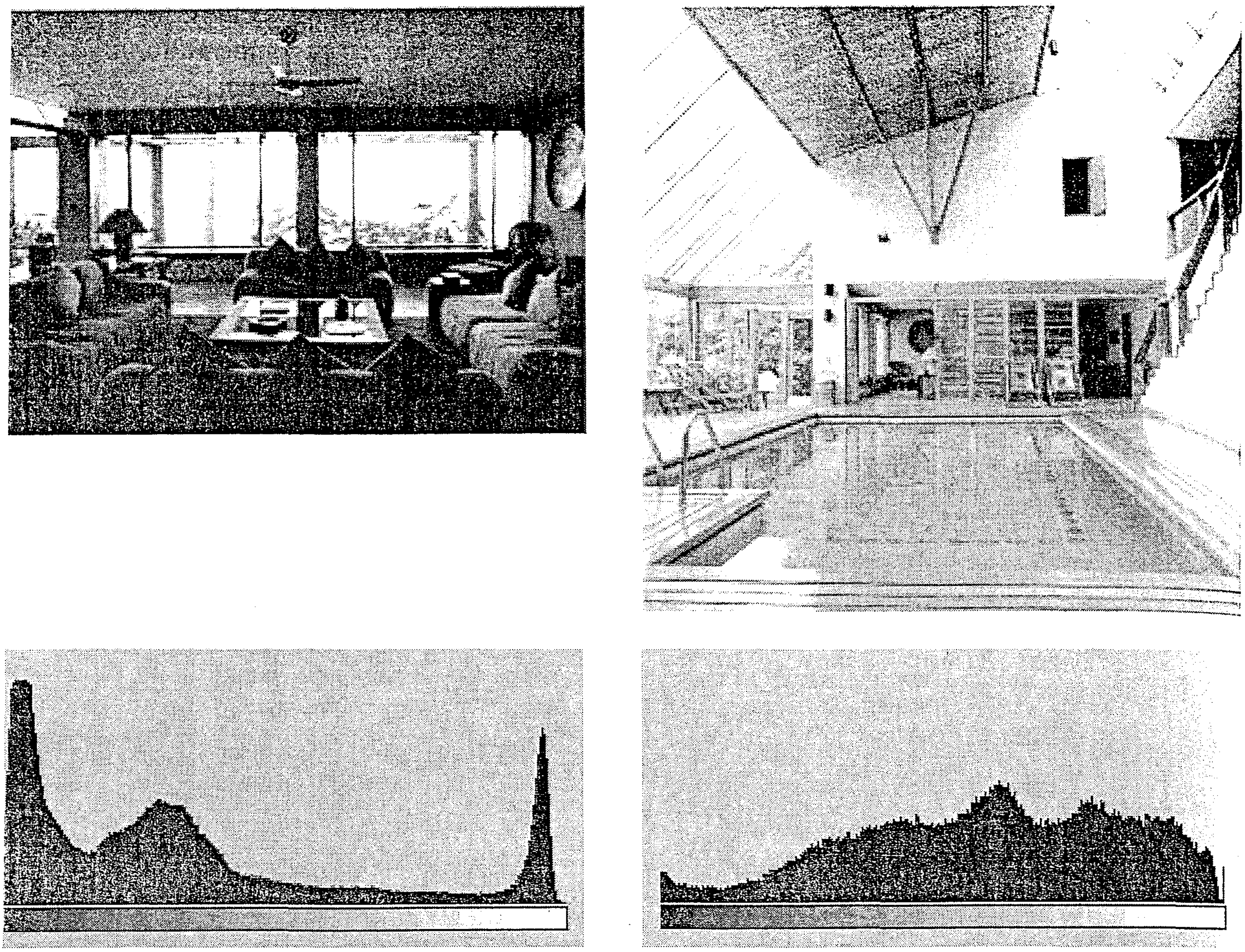
Figure 8.8: Presentation of an image with different gamma values: ***a*** *0.5,* ***b*** *0.7,* ***c*** *1.0, and* ***d*** *2.0 (exercise 8.3).*

Range Compression. In comparison to the human visual system, a digital image has a considerably smaller dynamical range. If a minimum resolution of 10% is demanded, the gray values must not be lower than 10. Therefore, the maximum dynamical range in an 8-bit image is only $255/10 \approx 25$. The low contrast range of digital images makes them appear of low quality when high-contrast scenes are encountered. Either the bright parts are bleached or no details can be recognized in the dark parts.

The dynamical range can be increased by a transform that was introduced in Sect. 2.2.6 as the *gamma transform.* This nonlinear homogeneous point operation has the form

$$q' = \frac{255}{255^{\gamma}} q^{\gamma}. \tag{8.10}$$

The factors in (8.10) are chosen such that a range of $[0, 255]$ is mapped onto itself. This transformation allows a larger dynamic range to be recognized at the cost of resolution in the bright parts of the image.



9.5
s with 'unsatisfactory'
rams.

ation based on the **shape** of the histogram. The first stage gives us global knov
edge about the nature of the image as far as its intensity levels a
concerned and enables a suitable transformation to be derived

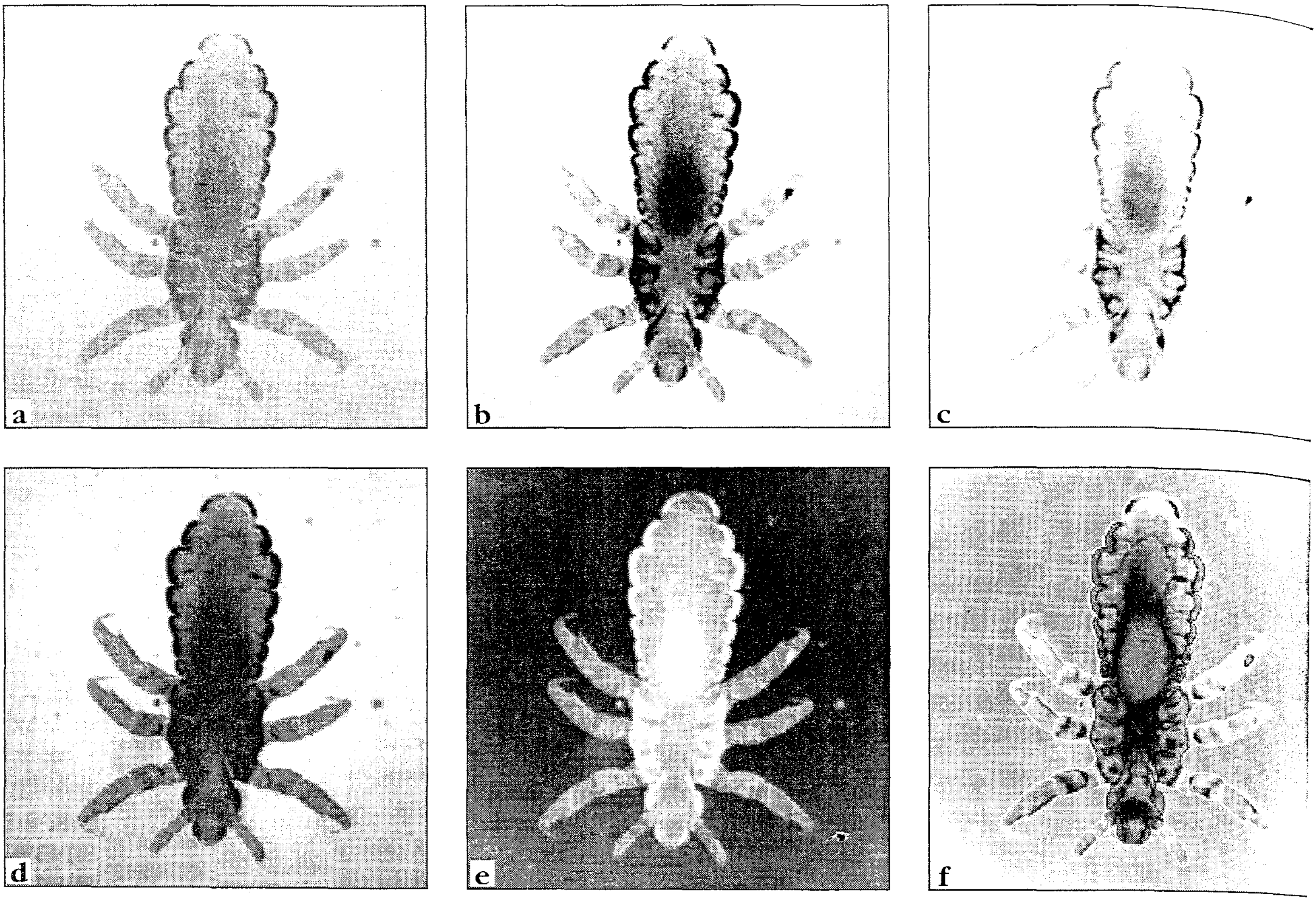


Figure 3. Manipulation of the grey scale transfer function:

a) *an original, moderately low-contrast transmission light microscope image (prepared slide of a head louse);*

b) *expanded linear transfer function adjusted to the minimum and maximum brightness values in the image;*

c) *positive gamma (log) function;*

d) *negative gamma (log) function;*

e) *negative linear transfer function;*

f) *nonlinear transfer function (high slope linear contrast over central portion of brightness range, with negative slope or solarization for dark and bright portions).*

Any of these functions may be used in addition to contr
expansion, which stretches the original scale to the full range
the display. Curves or tables of values for these transfer fu
tions are typically precalculated and stored, so that they can
loaded quickly to modify the display LUT. Many systems all
quite a few different tables to be kept on hand for use when
image requires it, just as a series of color LUTs may be availa
on disk for pseudo-color displays. **Figure 3** illustrates the use
several transfer functions to enhance the visibility of structure
an image.

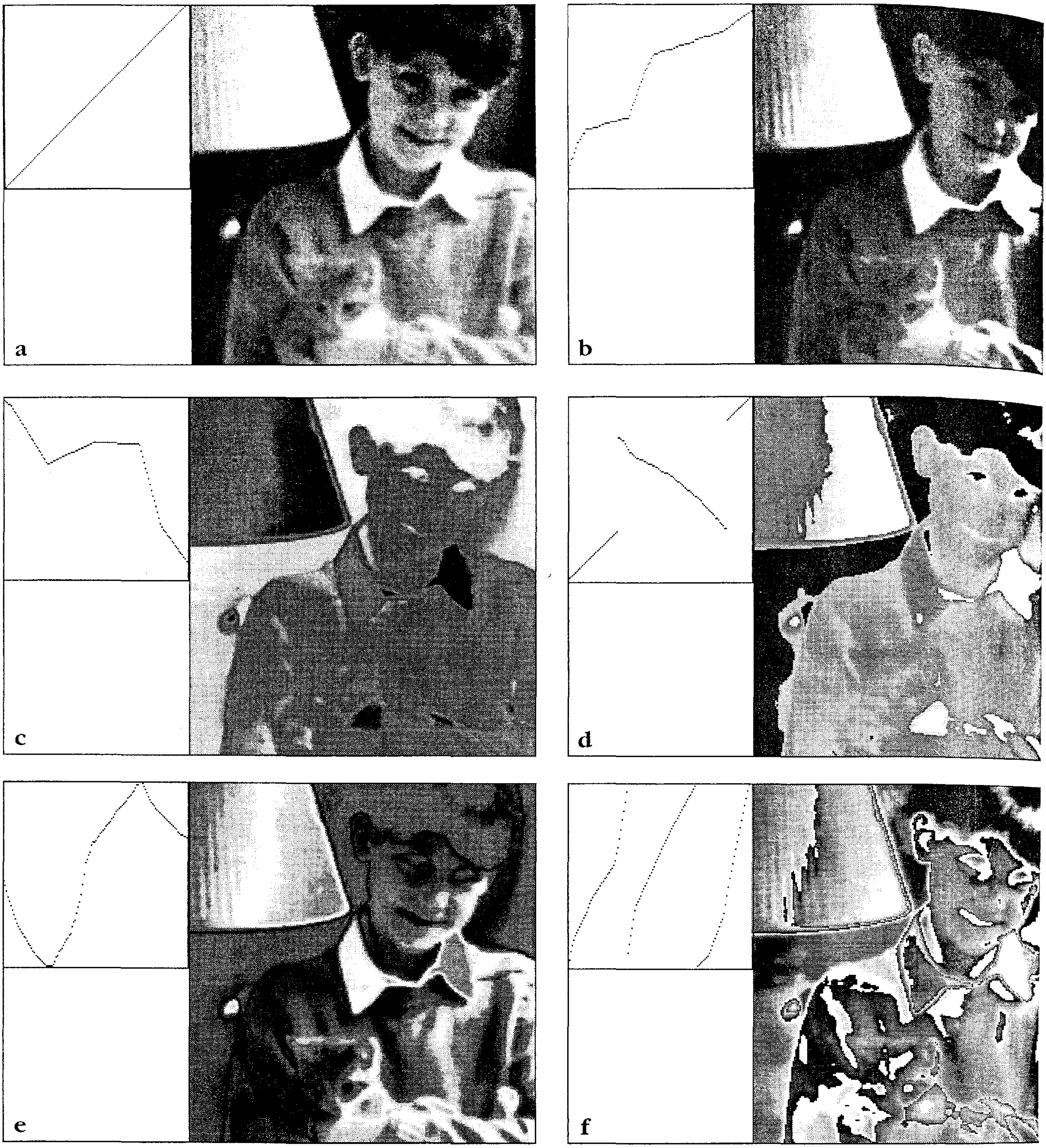


Figure 1. *An original image with a full range of brightness values and several examples of arbitrary ha drawn display transfer functions which expand or alter the contrast in various parts of the range. T plot with each image shows the stored pixel brightness values on the horizontal axis and the display brightness on the vertical axis.* ***Image a)*** *has a transfer function that is the identity function, so tha actual stored brightnesses are displayed.* ***Images b)*** *through* ***f)*** *illustrate various possibilities, incl reversal and increased or decreased slope over parts of the brightness range.*

smooth noise by kernel averaging or median filtering. Re

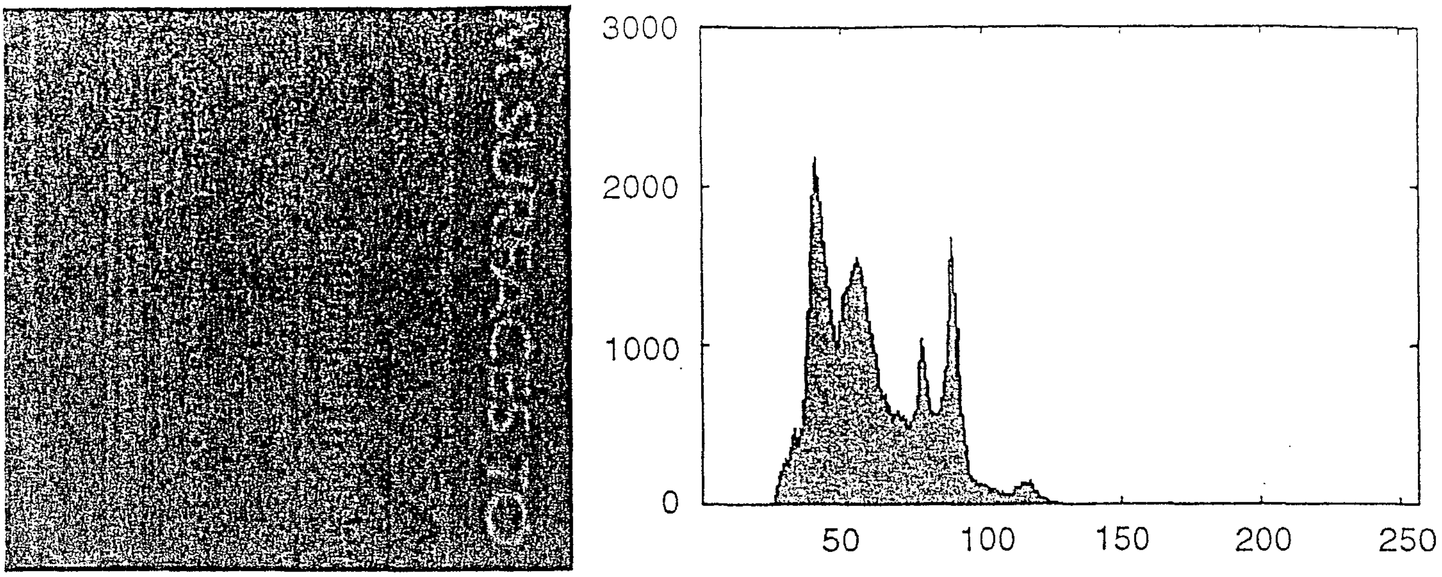


FIGURE 4 Digital image "books" (left) and its histogram (right). The image makes poor use of the available gray-scale range.

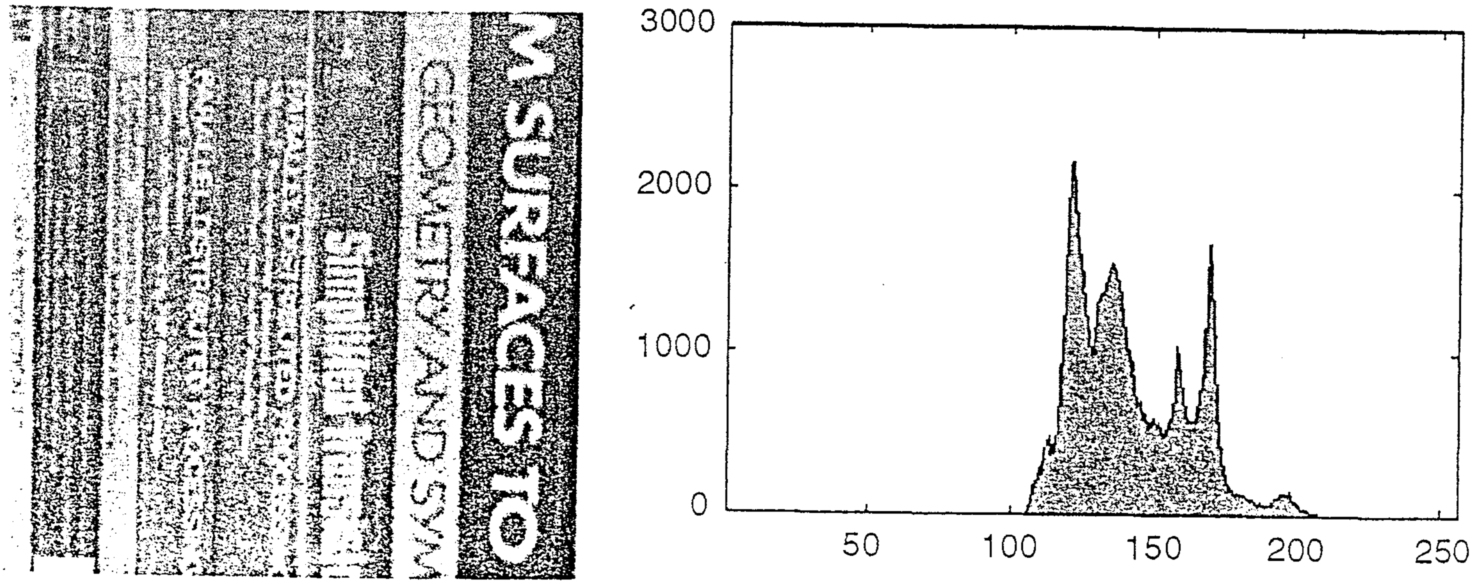
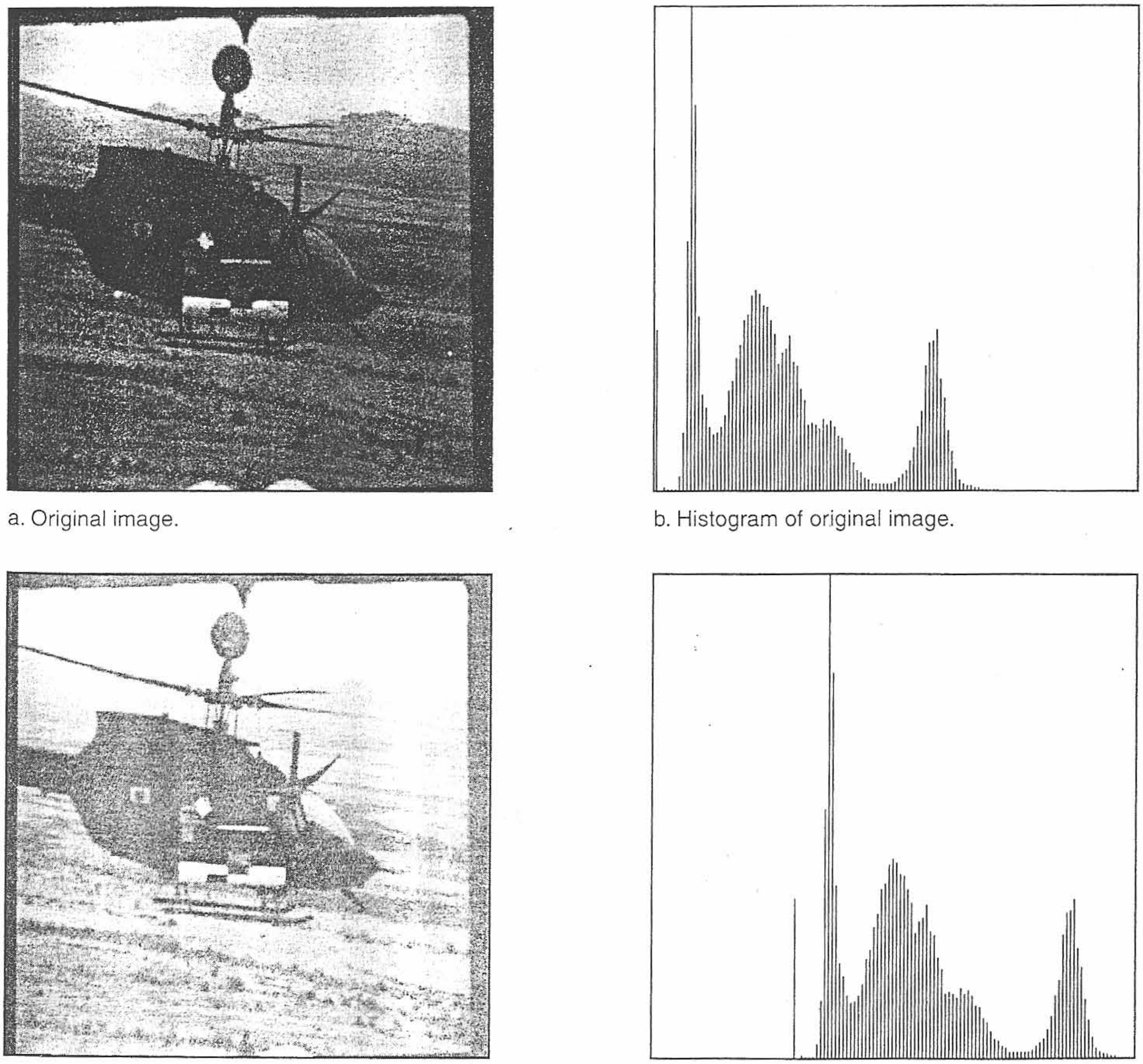


FIGURE 7 Left: Additive offset of the image of books in Fig. 4 by amount 80.

$$g(n) = f(n) + L$$

Figure 4.2-8 Histogram Sliding



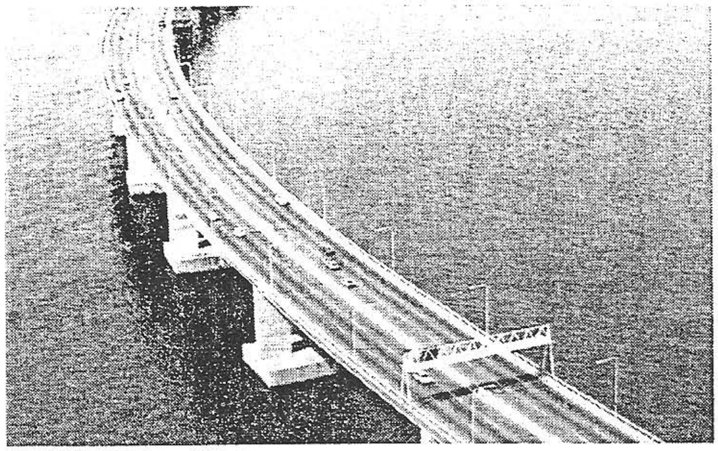
a. Original image.

b. Histogram of original image.

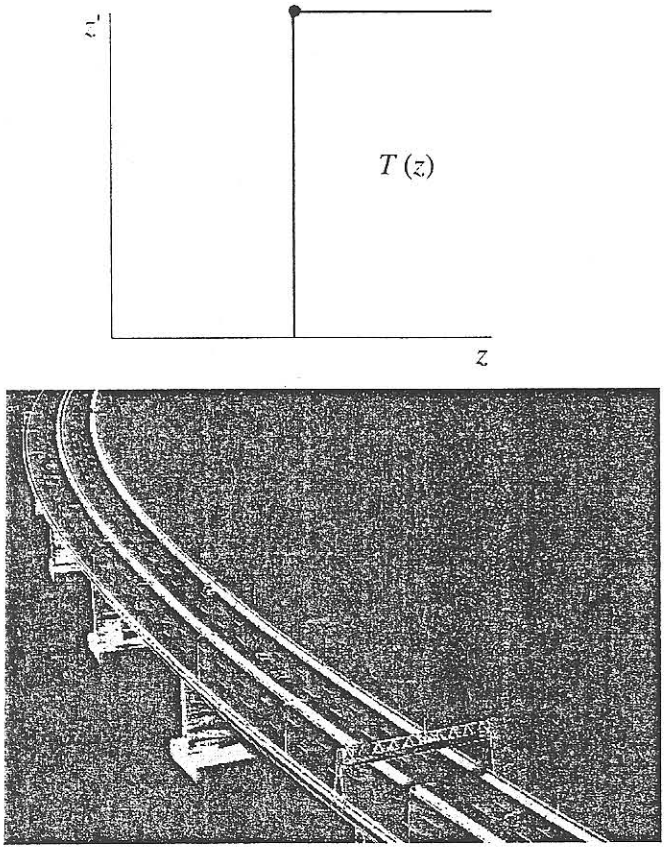
c. Image after positive-value histogram sliding.

d. Histogram of image after sliding.

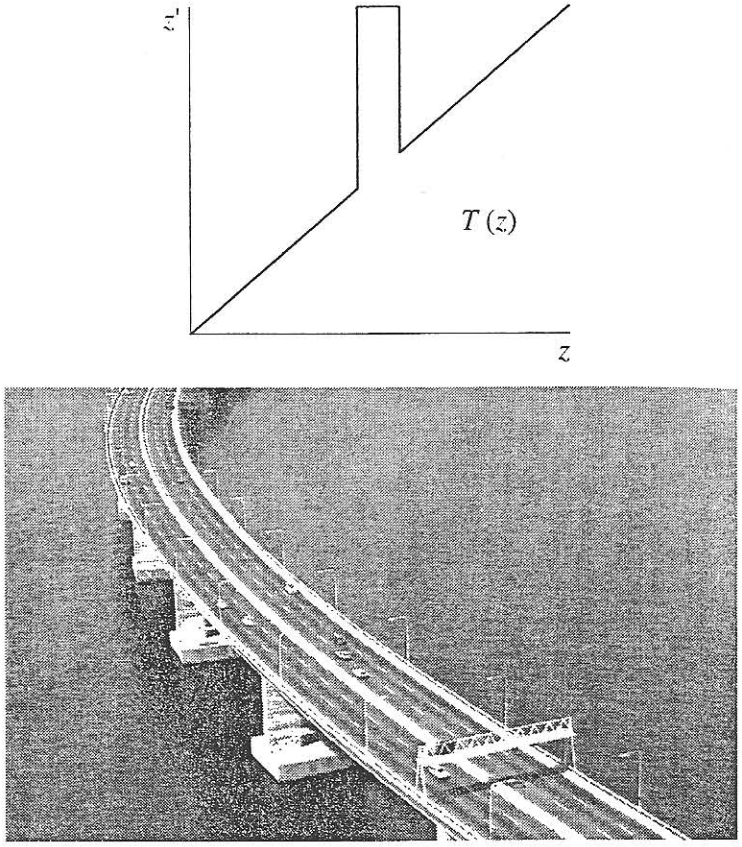
l 'slicing'.



Original image (Ponte Rio-Niteroi)



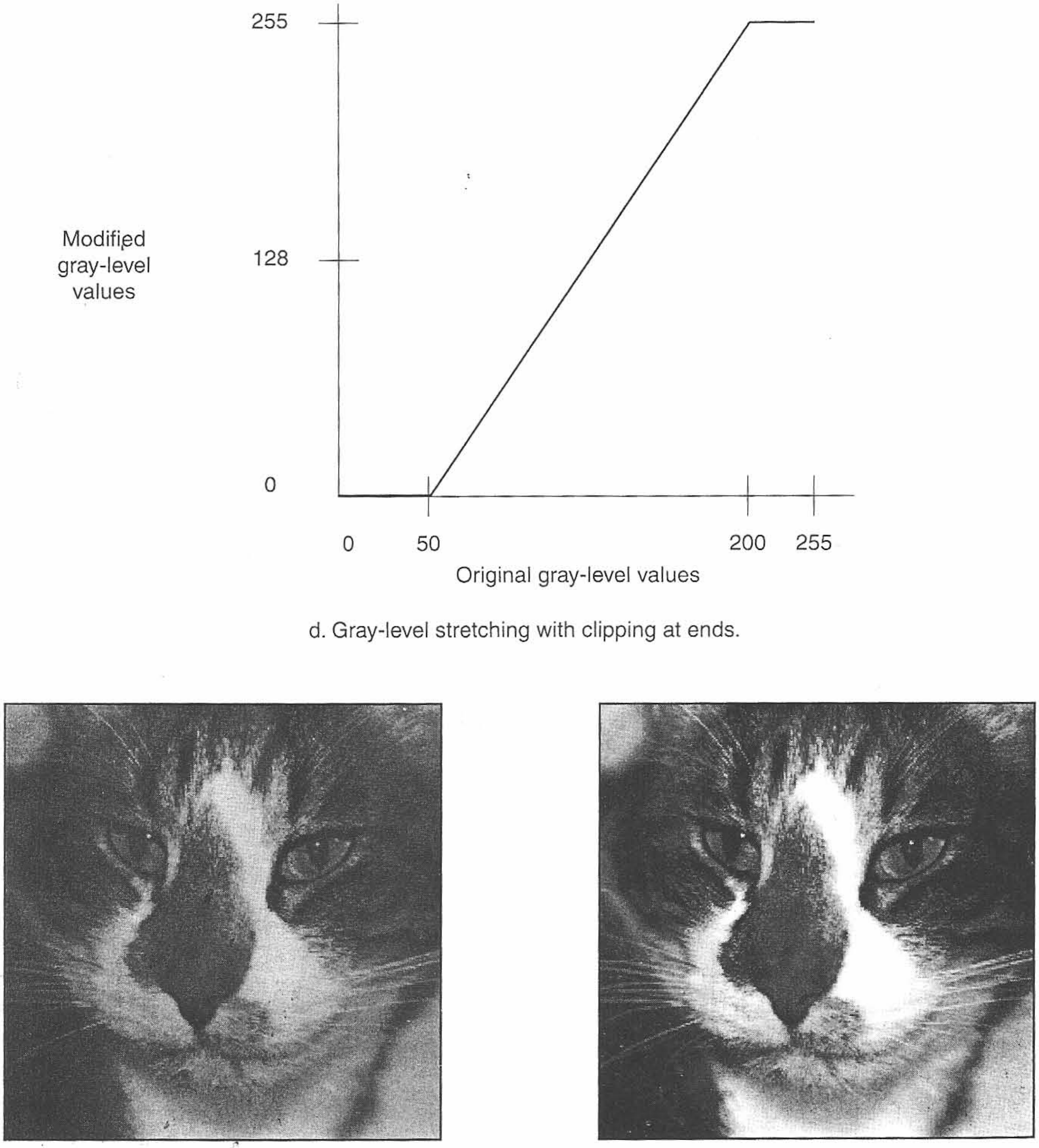
Thresholded



Highlighting an intensity
band of interest

and high ends. Figure 4.2-1d illustrates a linear function to stretch the gray levels between 50 and 200, while clipping any values below 50 to 0 and any values above 200 to 255. The original and modified images are shown in Figures 4.2-1e, f, where we see the resulting enhanced image.

Figure 4.2-1 (Continued)

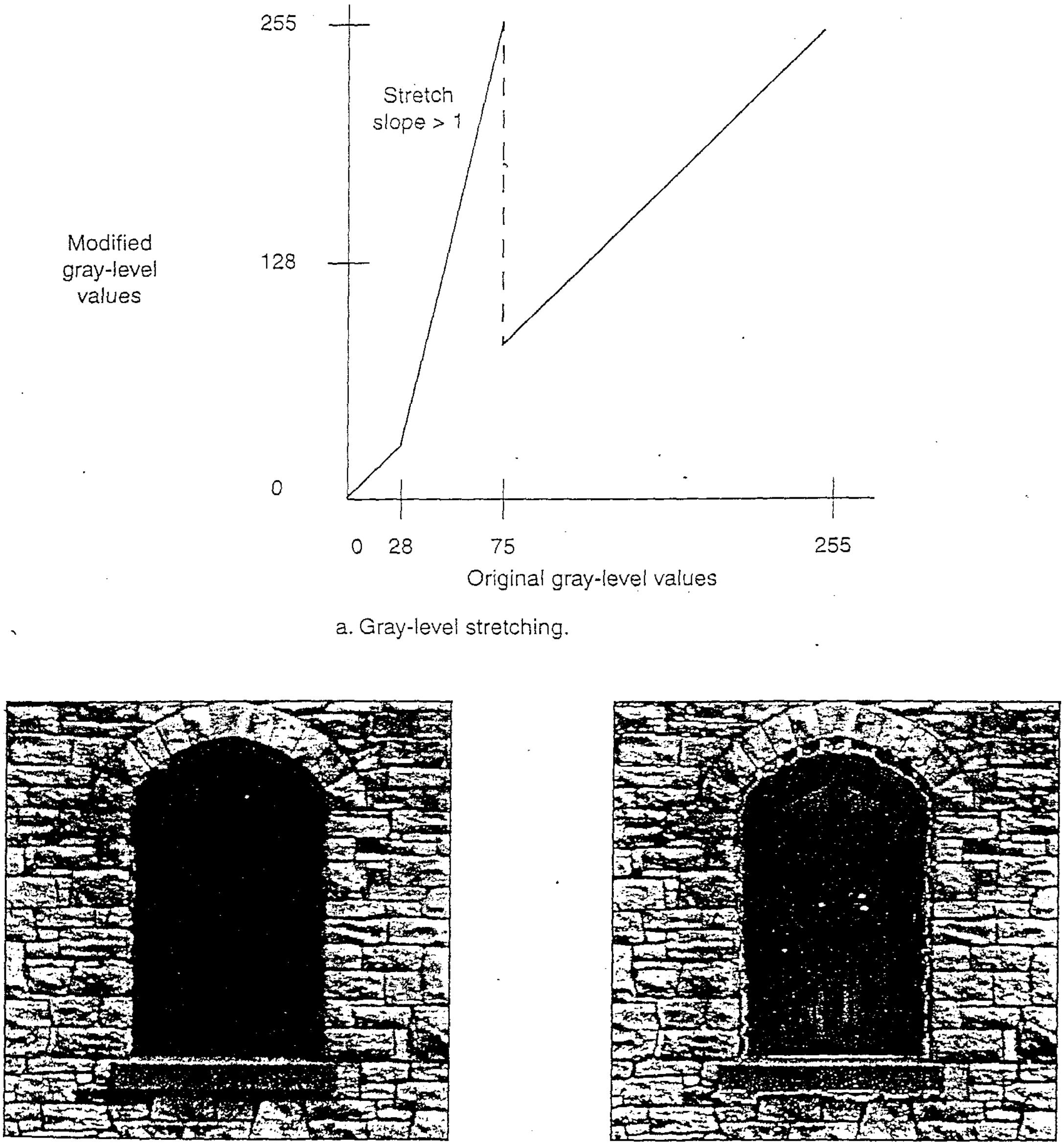


d. Gray-level stretching with clipping at ends.

e. Original image.

f. Image after modification.

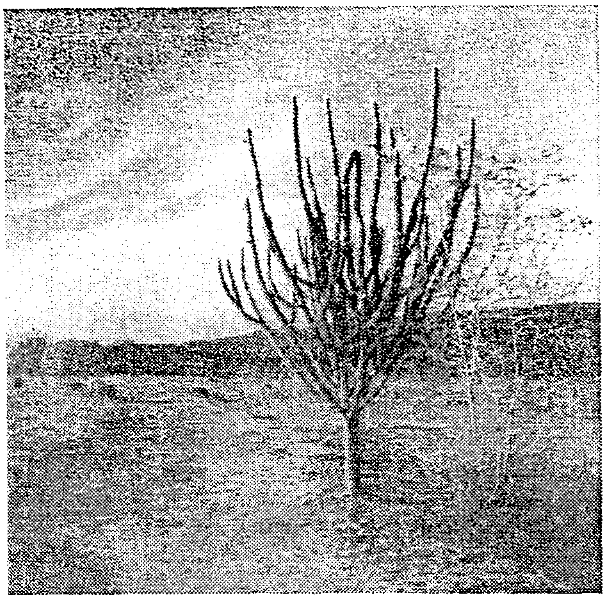
Figure 4.2-1 Gray-Scale Modification



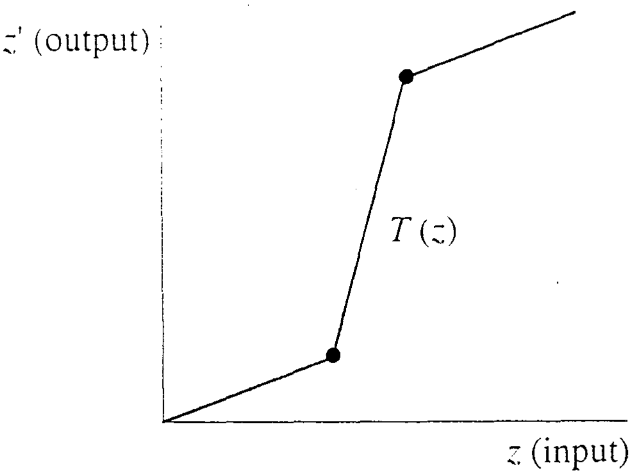
a. Gray-level stretching.

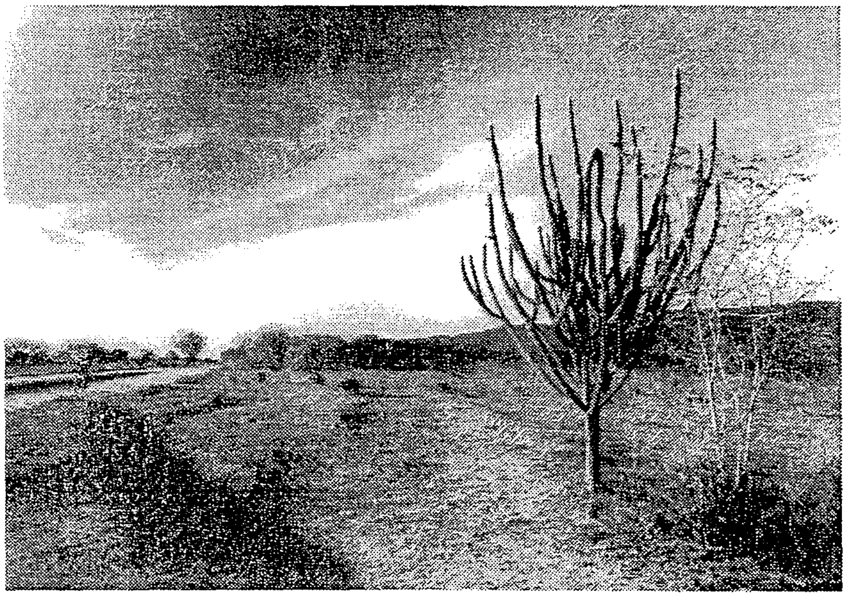
b. Original image.

c. Image after modification.



Original (N.E. Brazil)





Contrast stretched

breakpoints while viewing the processed image, or better still both images sid by side.

Intensity level slicing

This simple technique selects a particular contiguous range of grey levels to dis play. It is used in contexts where intensity variations of interest in an imag are too slight to be seen well by the viewer. Providing these variations occup

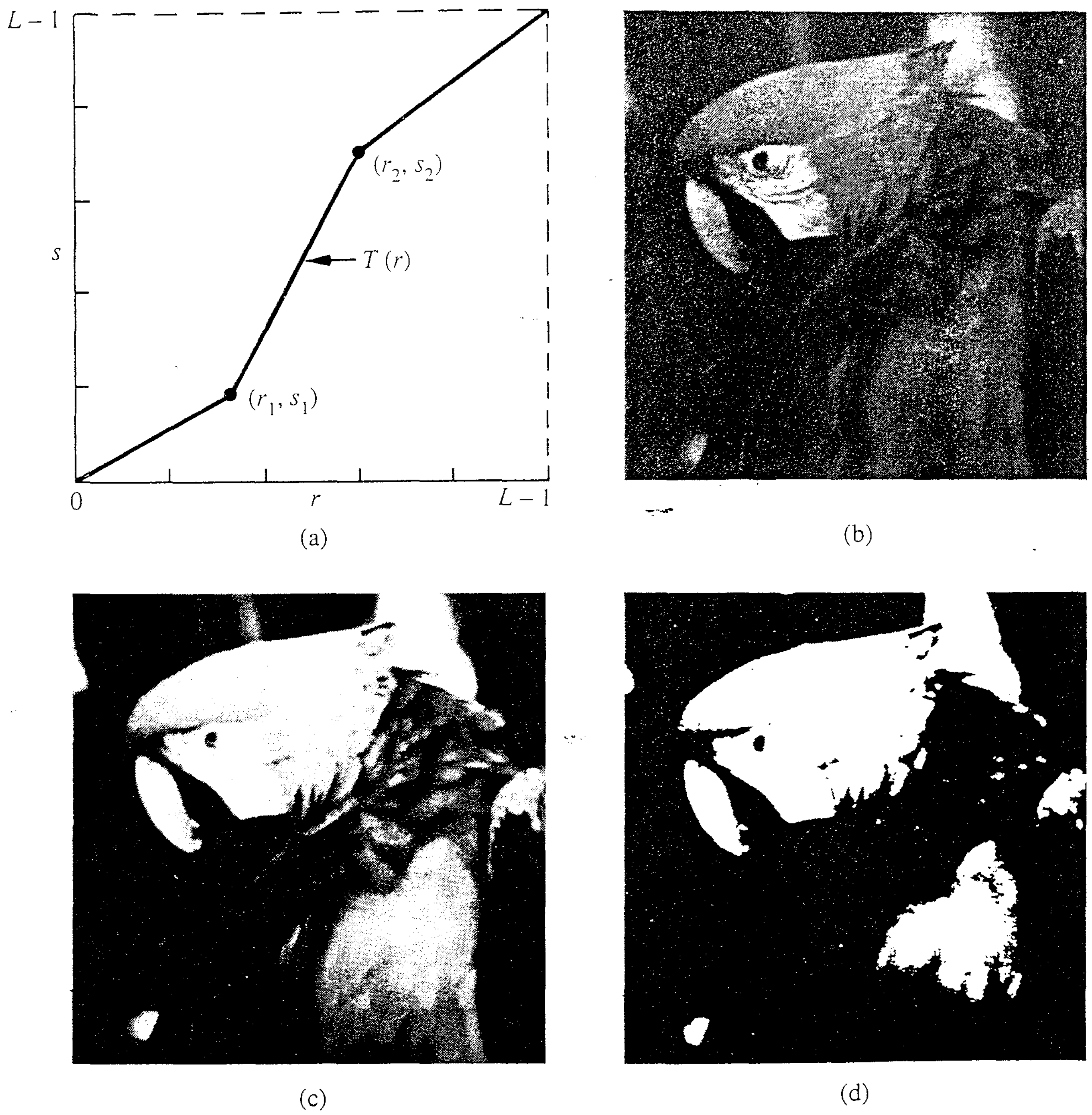


Figure 4.5 *Contrast stretching: (a) form of transformation function; (b) a low-contrast image; (c) result of contrast stretching; (d) result of thresholding.*

an image on film. One of the classic illustrations of this problem is in the display of the Fourier spectrum of an image, as discussed in Section 3.3. An effective way to compress the dynamic range of pixel values is to perform the following intensity transformation:

$$s = c \log(1 + |r|) \tag{4.2-1}$$

where c is a scaling constant, and the logarithm function performs the desired

The mapping function for a histogram stretch can be found by the followi equation:

$$\text{Stretch}(I(r, c)) = \left[\frac{I(r, c) - I(r, c)_{\text{MIN}}}{I(r, c)_{\text{MAX}} - I(r, c)_{\text{MIN}}}\right][\text{MAX} - \text{MIN}] + \text{MIN}$$

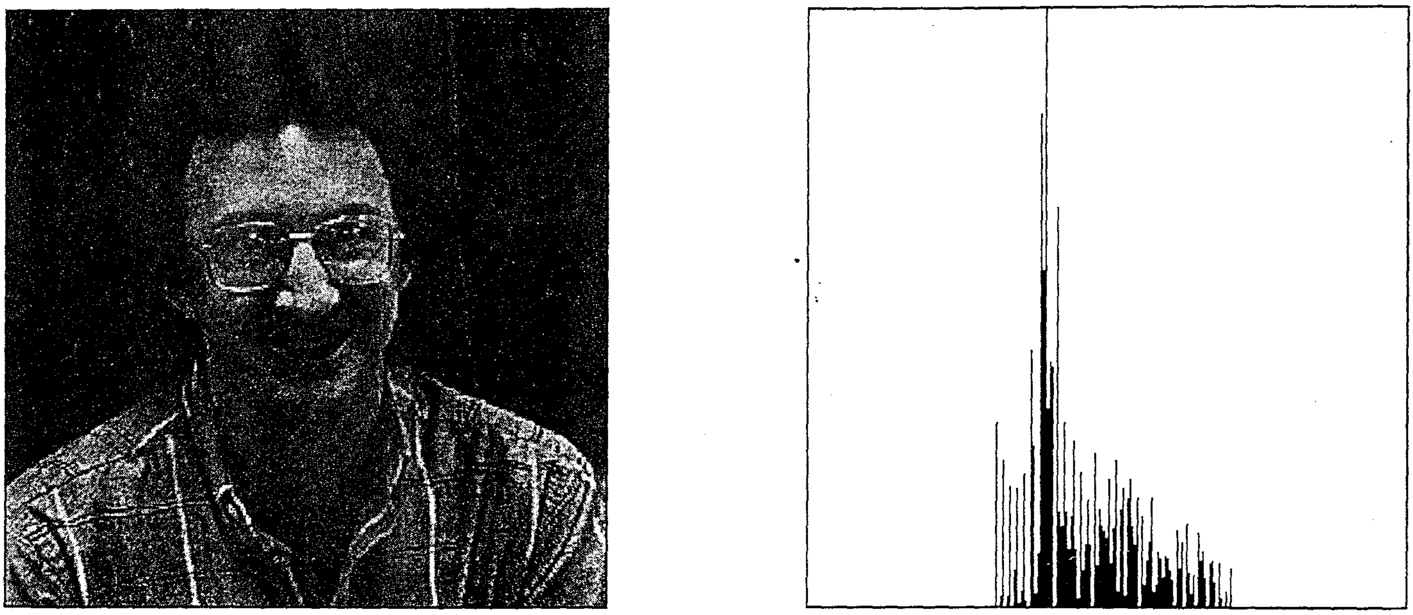
where

$I(r, c)_{\text{MAX}}$ is the largest gray-level value in the image $I(r, c)$

$I(r, c)_{\text{MIN}}$ is the smallest gray-level value in $I(r, c)$

MAX and MIN correspond to the maximum and minimum gray-level values pc sible (for an 8-bit image these are 0 and 255)

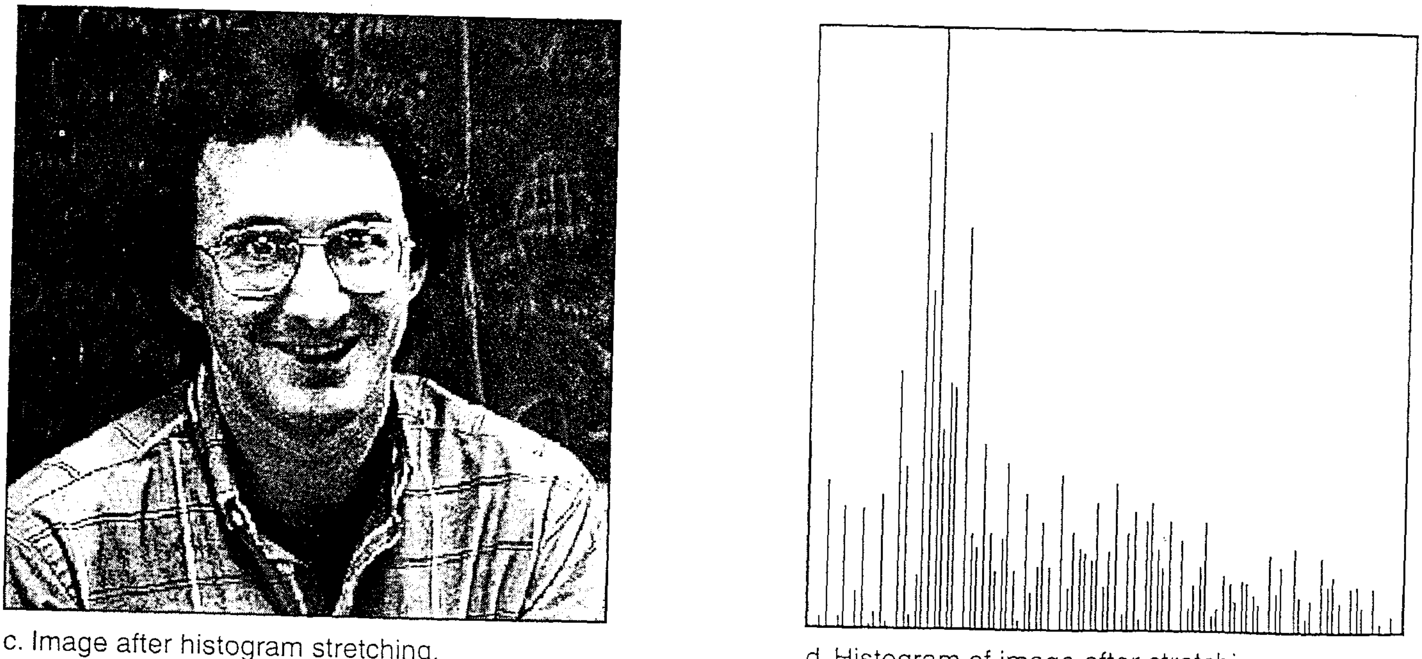
Figure 4.2-5 Histogram Stretching



a. Low-contrast image.

b. Histogram of low-contrast image.

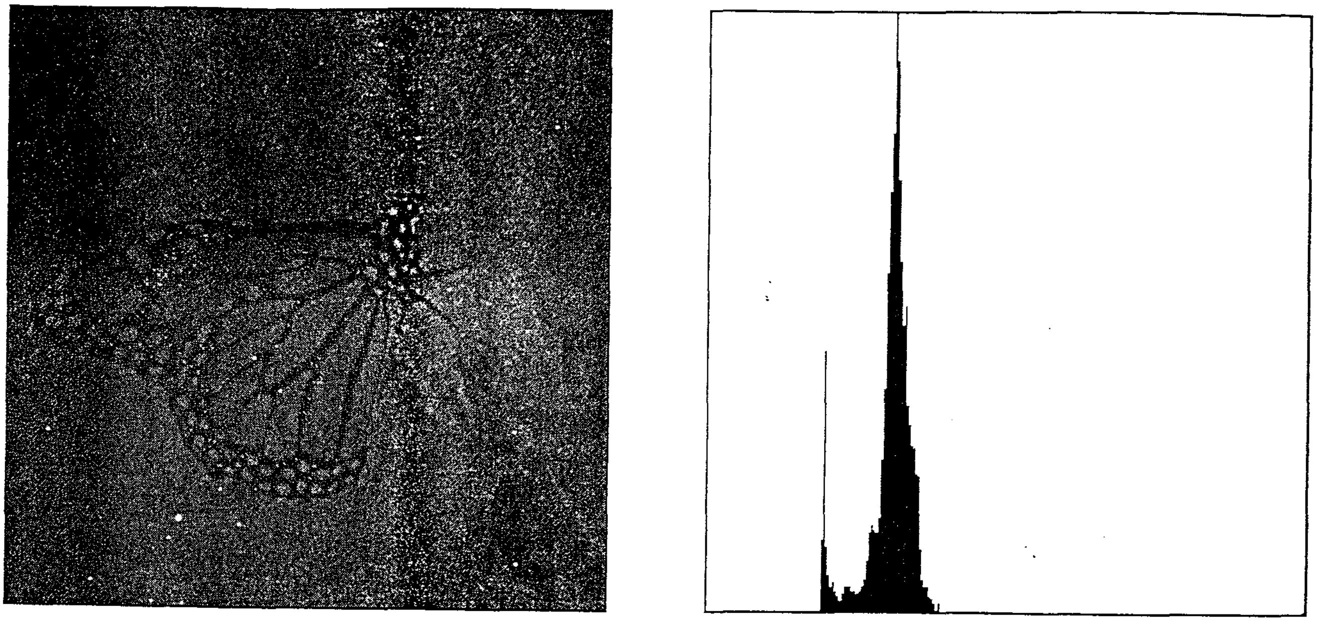
Figure 4.2-5 (Continued)



c. Image after histogram stretching.

d. Histogram of image after stretching.

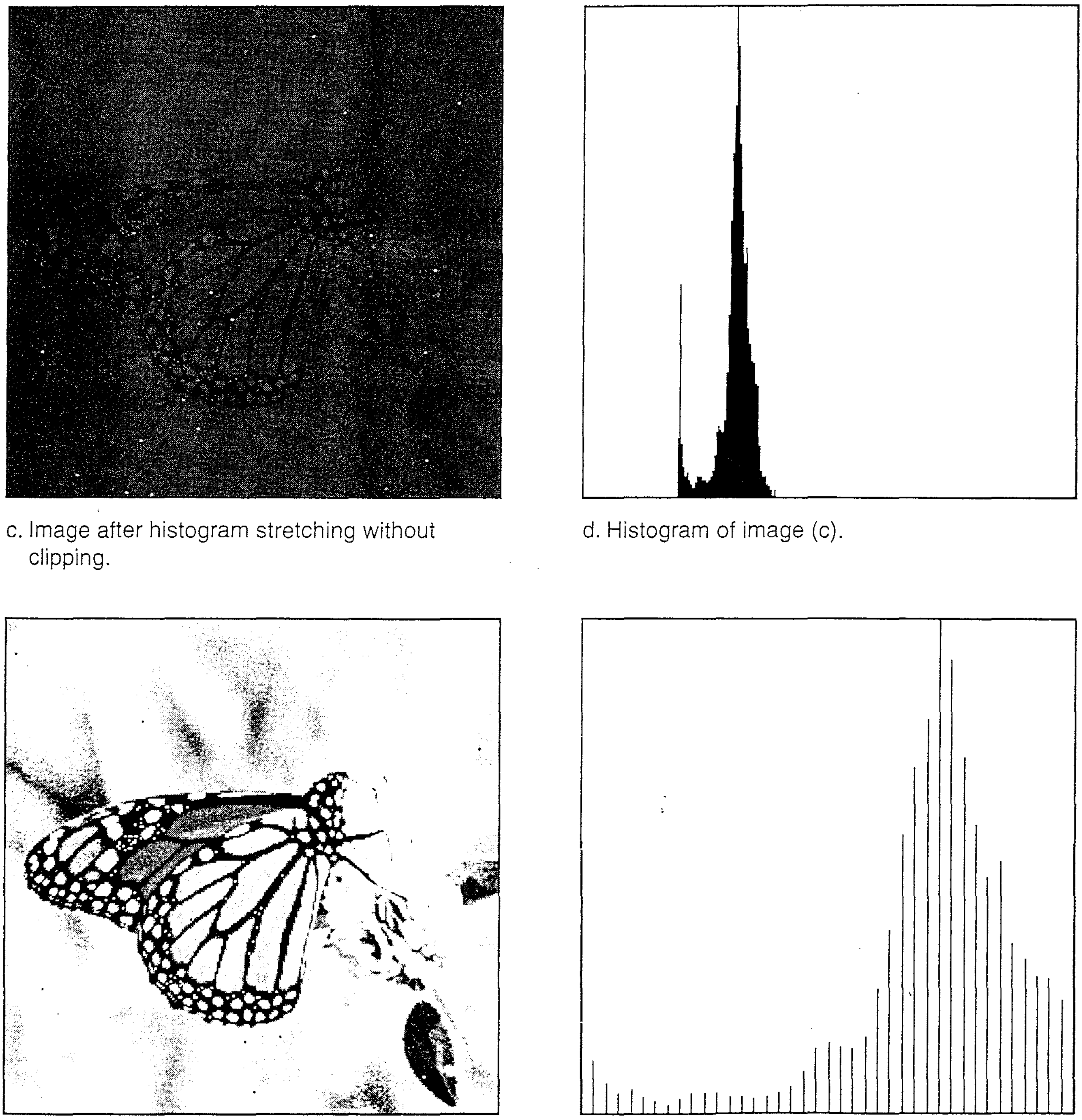
Figure 4.2-6 Histogram Stretching with Clipping



a. Original image.

b. Histogram of original image.

Figure 4.2-6 (Continued)



c. Image after histogram stretching without clipping.

d. Histogram of image (c).

e. Image after histogram stretching with clipping

f. Histogram of image (e).

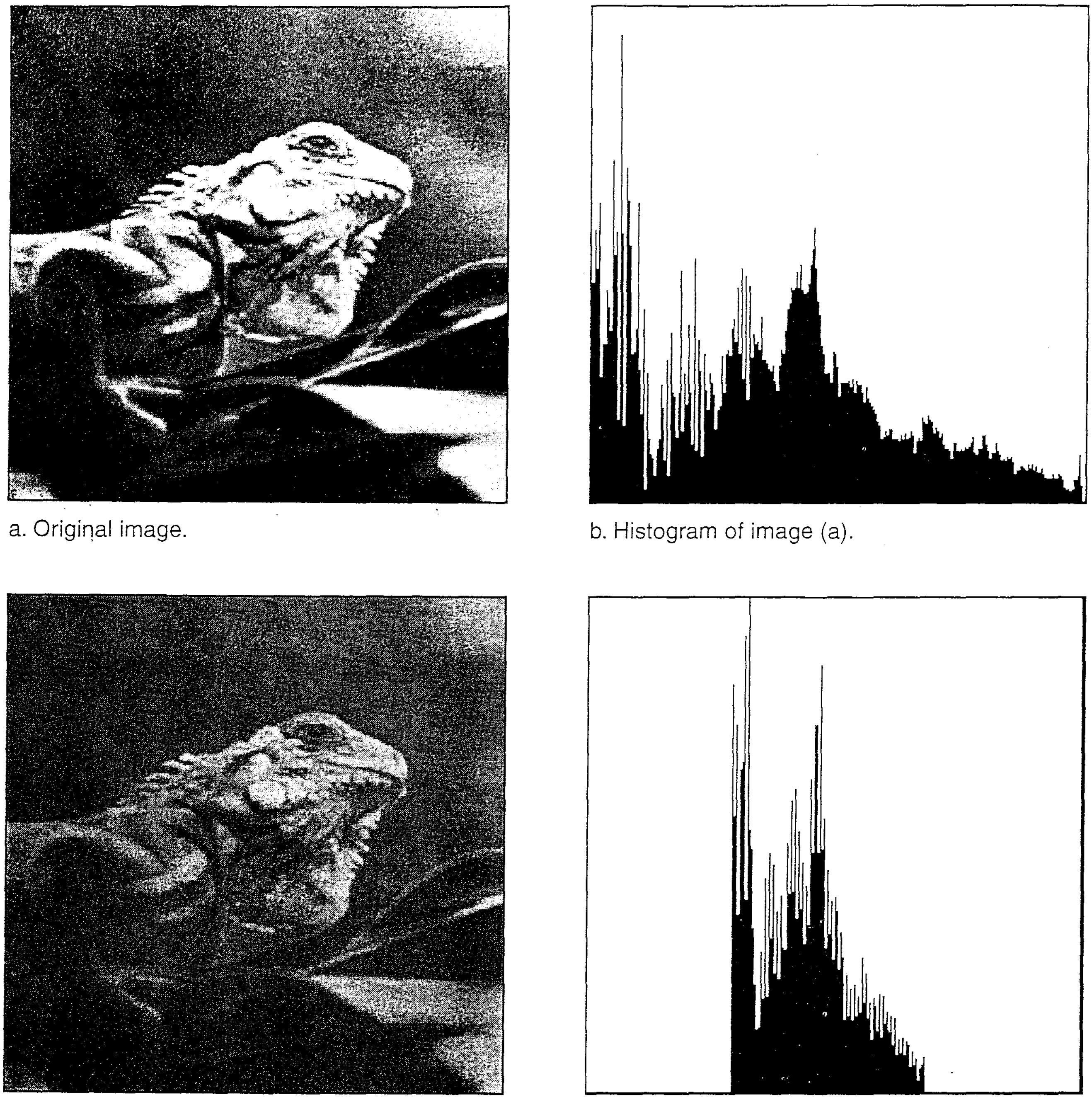
$$\text{Shrink}(I(r,c)) = \left[\frac{\text{Shrink}_{\text{MAX}} - \text{Shrink}_{\text{MIN}}}{I(r,c)_{\text{MAX}} - I(r,c)_{\text{MIN}}}\right]\left[I(r,c) - I(r,c)_{\text{MIN}}\right] + \text{Shrink}_{\text{MIN}}$$

where

$I(r, c)_{\text{MAX}}$ is the largest gray-level value in the image $I(r, c)$

$I(r, c)_{\text{MIN}}$ is the smallest gray-level value in $I(r, c)$

gure 4.2-7 Histogram Shrinking



a. Original image.

b. Histogram of image (a).

c. Image after shrinking the histogram to the range [75,175].

d. Histogram of image (c).

Histogram Equalization

A=input B=output

gray level change

$$D_B = f(D_A) \qquad D_A = f^{-1}(D_B)$$

Assume f is monotonic and $0 \le f \le 1$ when $0 \le D \le 1$

Conserving "gray levels" we have

$$\int_{D_B}^{D_B+\Delta D_B} H_B(D)dD = \int_{D_A}^{D_A+\Delta D_A} H_A(D)dD$$

So $H_B(D_B)\Delta D_B = H_A(D_A)\Delta D_A$

and

$$H_B(D_B) = \frac{H_A(D_A)}{\Delta D_B \big/ \Delta D_A} \rightarrow \frac{H_A(D_A)}{\frac{dD_B}{dD_A}} = \frac{H_A(D_A)}{\frac{df(D_A)}{dD_A}}$$

or

$$H_B\left(f(D_A)\right) = \frac{H_A(D_B))}{f'(D_B)}$$

Dropping the subscript we get

$$H_B(f(D)) = \frac{H_A(D)}{f'(D)} = \text{ constant} \qquad f' = \frac{df}{dD}$$

If we indeed we have a constant then we get

$$f'(D) = \frac{D_m}{A_0} H(D) \qquad A_0 = \text{area of image} \quad \text{D}_m = \text{ maximum gray level}$$

integrate

$$f(D) = \frac{D_m}{A_0} \int_0^D H(t)dt = D_m \bullet \text{CDF}$$

Sometimes use logarithmic transformation as being closer to what the eye sees

Discrete Equalization

Let $h(x_i) = \#$ pixels with gray level x_i

Normalize

$$p(x_i) = \frac{h(x_i)}{\sum_{l=0}^{L-1} h(x_i)} \qquad i = 0 \ldots L-1 \quad L = \#\text{gray levels}$$

Define discrete CDF

$$v_i = \sum_{l=0}^{i} p(x_i)$$

$$v'_i = INT\left[\left(\frac{v_i - v_{\min}}{L - v_{\min}}\right)(L-1) + \frac{1}{2}\right]$$

v' is approximately uniformly distributed

Note: Discretization will destroy exact equalization

Randomization

- **randomly move gray levels between pixels to give equalization**
- **add random small numbers to given gray levels**

Local histogram modification

improvement use mean and variance

$$g(x,y) = A\left[f(x,y) - m(x,y)\right] + m(x,y)$$

$A =$ chosen constant

$m(x,y) =$ mean inside windown centered at (x,y)

σ(x,y) = variance inside the window

choose $A = \frac{kM}{\sigma(x,y)}$ k constant and M the global average ıpproximation of the continuous pixel brightness transformation from equation (4.7) is

$$q = \mathcal{T}(p) = \frac{q_k - q_0}{N^2} \sum_{i=p_0}^{p} H(i) \; + q_0 \tag{4.8}$$

Formally, the algorithm to perform equalization is as follows.

Algorithm 4.1: Histogram equalization

1. For an $N \times M$ image of G gray-levels (often 256), create an array H of length G initialized with 0 values.

2. Form the image histogram: Scan every pixel and increment the relevant member of H—if pixel p has intensity g_p, perform

$$H[g_p] = H[g_p] + 1$$

3. Form the cumulative image histogram H_c:

$$\begin{aligned} H_c[0] &= H[0] \\ H_c[p] &= H_c[p-1] + H[p] \qquad p = 1, 2, \ldots, G-1 \end{aligned}$$

4. Set

$$T[p] = \text{round}\left(\frac{G-1}{NM} H_c[p]\right)$$

(This step obviously lends itself to more efficient implementation by constructing a look-up table of the multiples of NM, and making comparisons with the values in H_c, which are monotonic increasing).

5. Rescan the image and write an output image with gray-levels g_q, setting

$$g_q = T[g_p]$$

(This presentation assumes that the intensity range of source and destination images is [0,G-1]—the adjustment if this is not the case is trivial.)

These results can be demonstrated on an image of a lung. An input image and its equalization are shown in Figure 4.3; their respective histograms are shown in Figure 4.4.

The **logarithmic** gray-scale transformation function is another frequently used technique. It simulates the logarithmic sensitivity of the human eye to the light intensity.

Also belonging to the group of pixel brightness transformations are **adaptive neighborhood histogram modification** and **adaptive neighborhood contrast enhancement**. These methods are discussed later in Section 4.3.9 in the context of other adaptive neighborhood pre-processing approaches.

Pseudo-color is yet another kind of gray-scale transform. The individual brightnesses in the input monochromatic image are coded to some color. Since the human eye is much more sensitive to change in color than to change in brightness, much more detail can be perceived in pseudo-colored images.

find the running sum of the histogram values, 2) normalize the values from step 1 by dividing by the total number of pixels, 3) multiply the values from step 2 by the maximum gray level value and round, and 4) map the gray-level values to the results from step 3 using a one-to-one correspondence. An example will help to clarify this process:

EXAMPLE 4 – 1

We have an image with 3 bits/pixel, so the possible range of values is 0 to 7. We have an image with the following histogram:

Gray-Level Value	*Number of Pixels (Histogram values)*
0	10
1	8
2	9
3	2
4	14
5	1
6	5
7	2

STEP 1: Create a running sum of the histogram values. This means that the first value is 10, the second is 10 + 8 = 18, next is 10 + 8 + 9 = 27, and so on. Here we get 10, 18, 27, 29, 43, 44, 49, 51.

STEP 2: Normalize by dividing by the total number of pixels. The total number of pixels is 10 + 8 + 9 + 2 + 14 + 1 + 5 + 0 = 51 (note this is the last number from step 1), so we get: 10/51, 18/51, 27/51, 29/51, 43/51, 44/51, 49/51, 51/51.

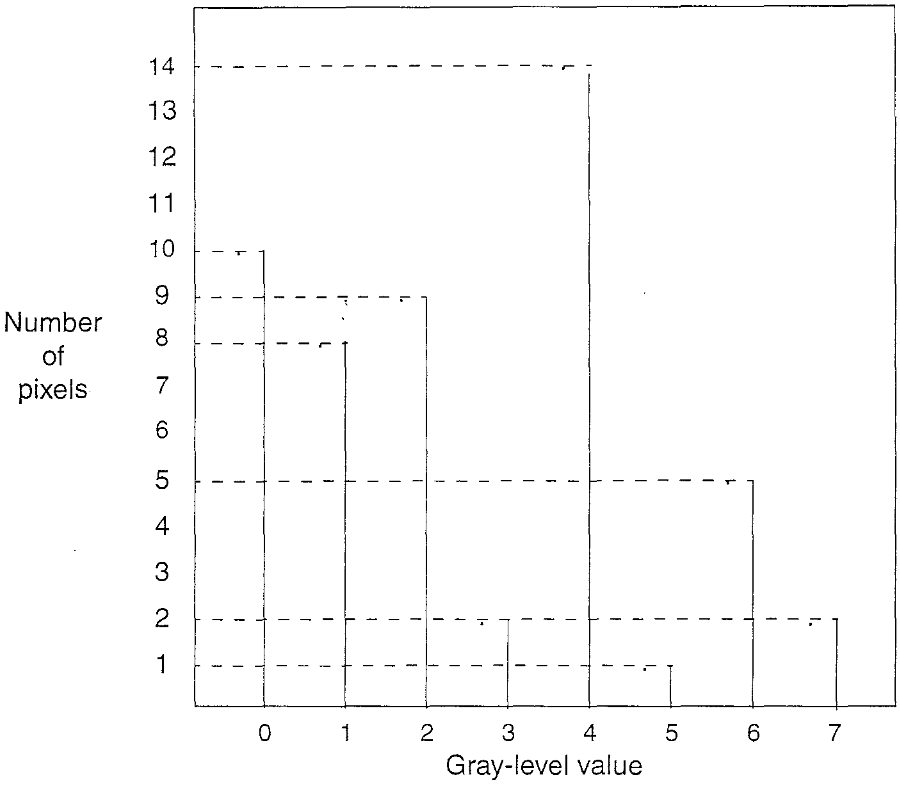
STEP 3: Multiply these values by the maximum gray-level values, in this case 7, and then round the result to the closest integer. After this is done we obtain 1, 2, 4, 4, 6, 6, 7, 7.

STEP 4: Map the original values to the results from step 3 by a one-to-one correspondence. This is done as follows:

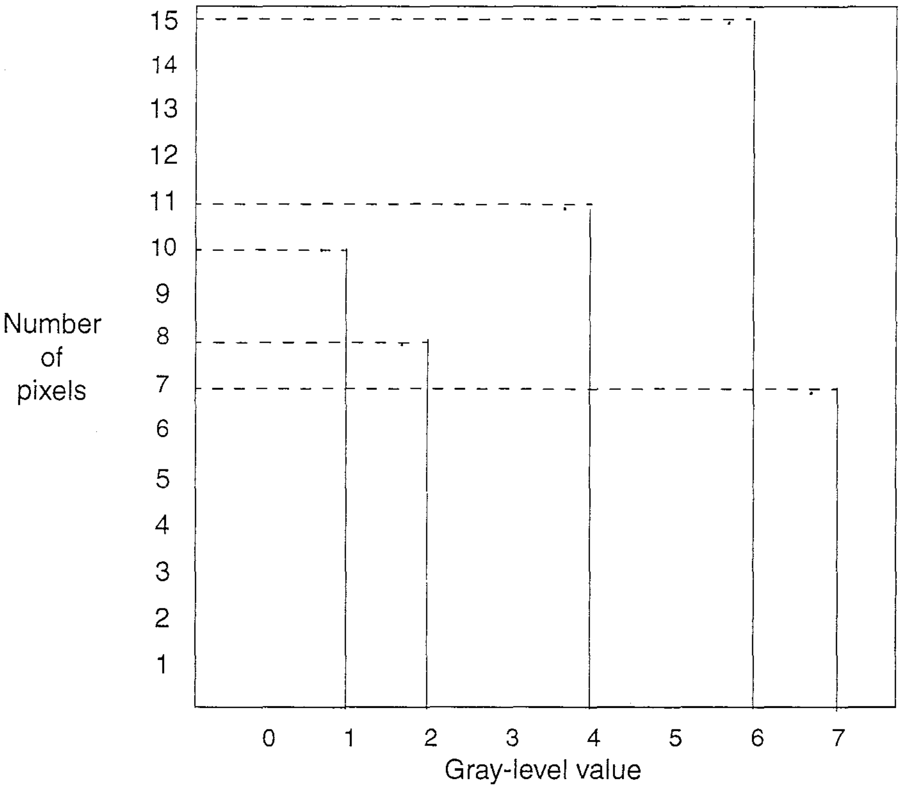
Original Gray-Level Value	*Histogram Equalized Values*
0	1
1	2
2	4
3	4
4	6
5	6
6	7
7	7

All pixels in the original image with gray level 0 are set to 1, values of 1 are set to 2, 2 set to 4, 3 set to 4, and so on. In Figure 4.2-9 we see the original histogram and the resulting histogram-equalized histogram. Although the result is not flat, it is closer to being flat than the original histogram.

Figure 4.2-9 Histogram Equalization



a. Original histogram.



b. After histogram equalization.

Histogram Matching

Key idea: Equalize input and output and solve a discrete inverse problem

Algorithm:

1. get a histogram of the input
2. $s_k = T(r_k) = \sum_{j=0}^{k} p_r(r_j) = \sum_{j=0}^{k} \frac{n_j}{n} \quad k = 0,1,\ldots,L-1$
3. Given $p_z(z)$ define G

   $v_k = g(z_k) = \sum_{j=0}^{k} p_z(z_j) = s_k \quad k = 0,1,\ldots,L-1$
4. Given s_k with $G\left(z_k\right) = s_k$

   discretely:

   start $z = 0$ increase z until $G(z) \geq s_k \quad k = 1$

   continue for each k but start search at z_{k-1}

   precompute #2 and #4

5. Given any gray level r_k in the input

$$r_k \to s_k \quad (2)$$
$$s_k \to z_k \quad (4) \quad \text{precomputed}$$

#3

O	*H*	*S*	*M*
0	1	0	1
1	2	1	2
2	4	2	3
3	4	4	3
4	6	7	4
5	6	7	4
6	7	7	4
7	7	7	4

rows
4,5,6,7

O = original
H = equalization
S = specified
M = final

The M column for this table is obtained by mapping the value in H to the closest value in S and then using the corresponding row in O for the entry in M. For example, the first entry in H is 1. We find the closest value in S, which is 1. This 1 from S appears in row 1, so we write a 1 for that entry in M. Another example, the third entry in H is 4. We find the closest value in S, which is 4. This 4 from S appears in row 3, so we write a 3 for that entry in M. If we consider the fifth entry in H, we see that 6 must map to 7 (the closest value), but the 7 appears on rows 4, 5, 6, 7. Which one do we select? It depends on what we want; picking the largest value will provide maximum contrast, but picking the smallest (closest) value will produce a more gradually changing image. Typically, the smallest is chosen because we can always perform a histogram stretch or equalization on the output image, if we desire to maximize contrast.

In practice, the desired histogram is often specified by a continuous (possibly nonlinear) function, for example, a sine or a log function. To obtain the numbers for the specified histogram, the function is sampled, and the values are normalized to 1 and then multiplied by the total number of pixels in the image. Using a hyperbolic function for the specified histogram is called *histogram hyperbolization* and is based on a mathematical model of the response of the cones in the human visual system. Histogram hyperbolization is an attempt to make the perceived brightness levels equally likely—sort of a "histogram equalization" based on a model of the visual system.

EXAMPLE 4-2

STEP 1: For this we will use the data from the previous example, where the histogram-equalization mapping table is given by:

Original Gray-Level Value—O	*Histogram Equalized Values—H*
0	1
1	2
2	4
3	4
4	6
5	6
6	7
7	7

STEP 2: Specify the desired histogram:

Gray-Level Value	*Number of Pixels in Desired Histogram*	sum
0	1	
1	5	6
2	10	16
3	15	31
4	20	51
5	0	
6	0	
7	0	

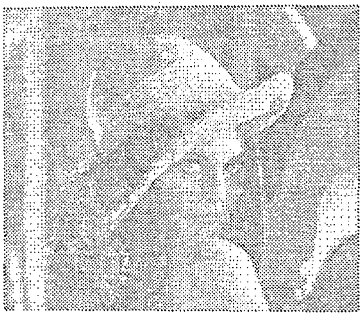
STEP 3: Find the histogram equalization mapping table for the desired histogram:

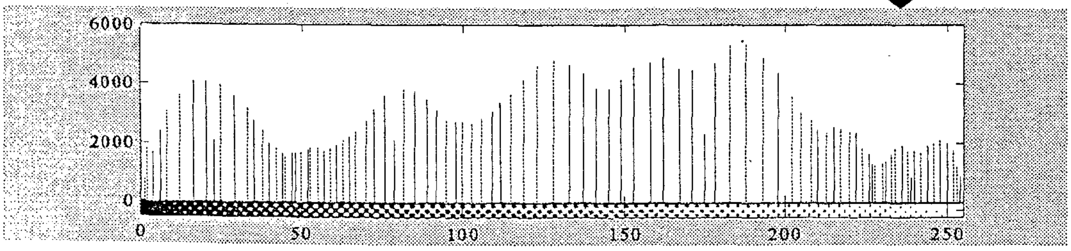
Gray-Level Value	*Histogram Equalized Values—S*
0	round(1/51)*7 = 0
1	round(6/51)*7 = 1
2	round(16/51)*7 = 2
3	round(31/51)*7 = 4
4	round(51/51)*7 = 7
5	round(51/51)*7 = 7
6	round(51/51)*7 = 7
7	round(51/51)*7 = 7

STEP 4: Map the original values to values from step 3 by using the table from step 1. This is done by setting up a table created by combining the tables from steps 1 and 3. We will denote O for the original gray levels, H for the histogram-equalized levels, S for the specified and histogram-equalized values, and M for our final mapping, which will provide the desired histogram. The combined table consists of O and H from the step 1 table, S from the second column of the step 3 table, and M, which provides the resulting gray-level values and will be generated in this step. Here is the resulting table:

Histogram equalization

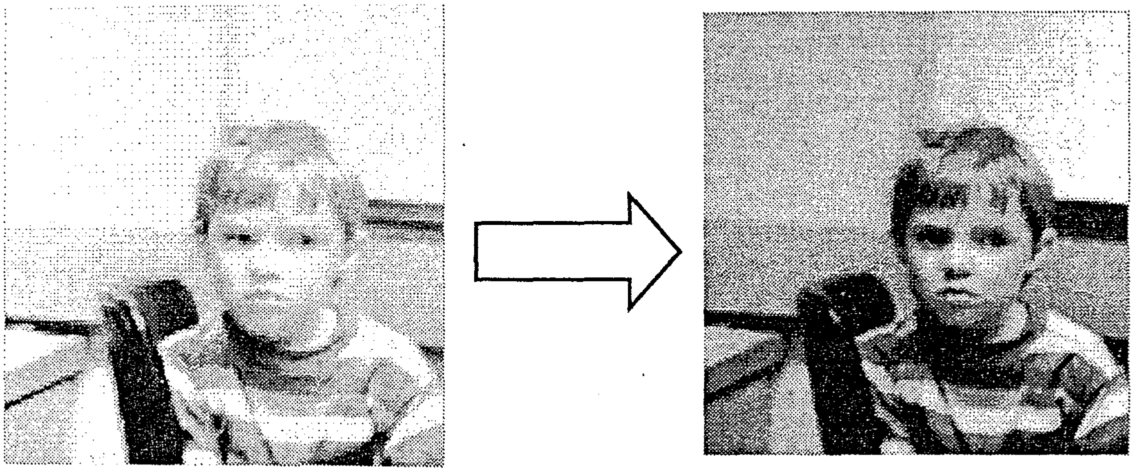
```
y = histeq(x,256);
```

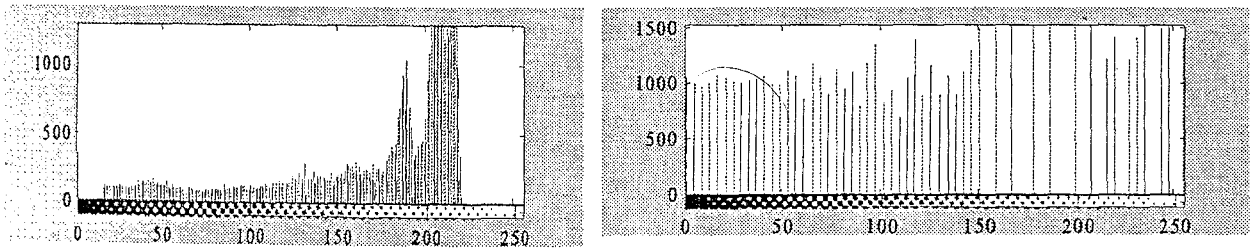




Histogram equalization (2)

Adam

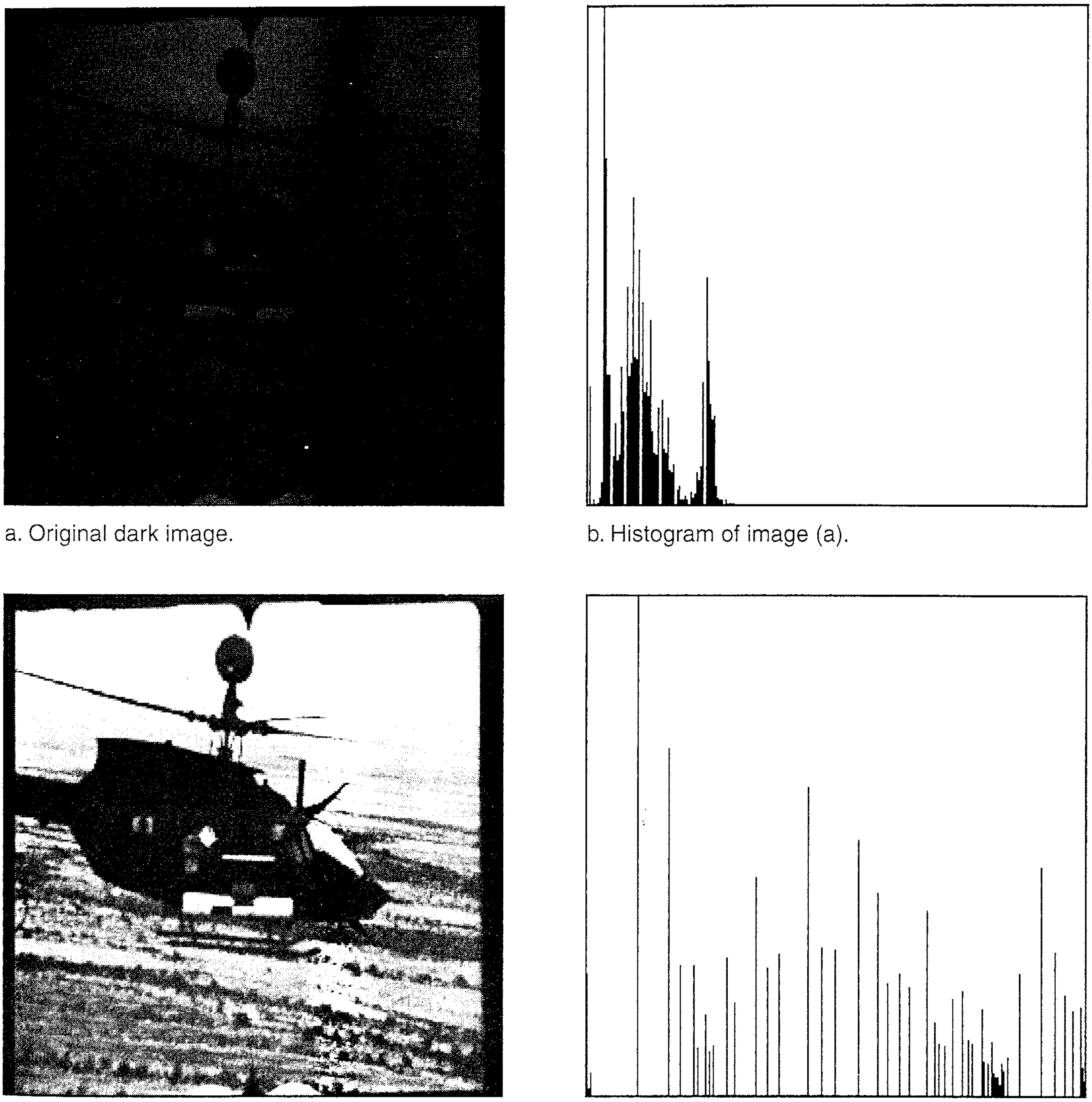




Histogram equalization of a digital image will not typically provide a histogra that is perfectly flat, but it will make it as flat as possible. For the equalized histogra to be completely flat, the pixels at a given gray level might need to be redistribut across more than one gray level. This could be done but would greatly complicate tl process, as some redistribution criteria would need to be defined. In most cases tl visual gains achieved by doing this would be negligible and could in some cases l negative. In practice, it is not done.

Figure 4.2-10 shows the result of histogram equalizing two images with very po contrast. In Figures 4.2-10a–d we see the results of applying histogram equalization a dark image, and in Figures 4.2-10e–h we see results from a bright image. The resul of this process are often very dramatic.

Figure 4.2-10 Histogram Equalization Example



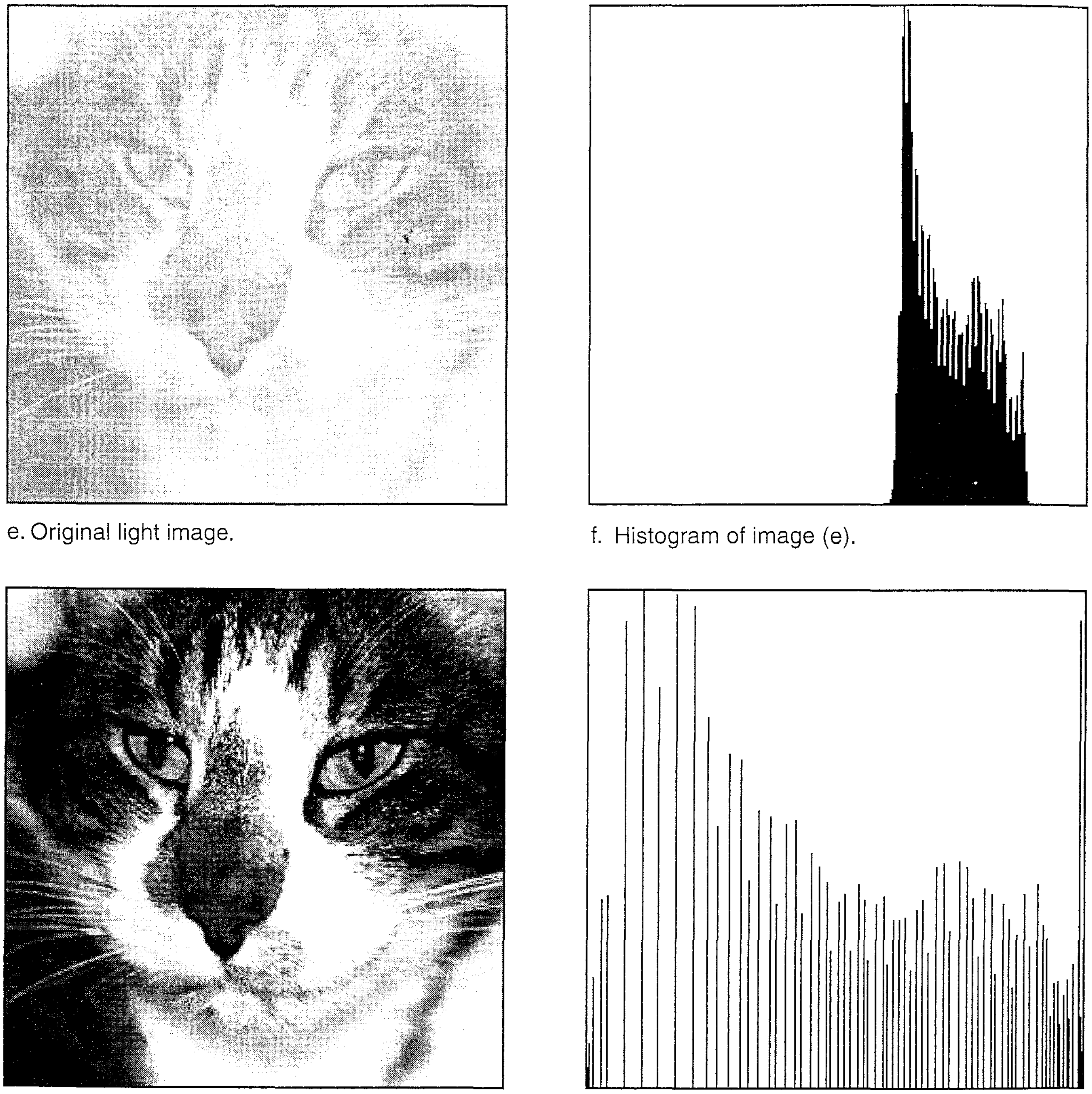
a. Original dark image.

b. Histogram of image (a).

c. Dark image after histogram equalization.

d. Histogram of image (c).

Figure 4.2-10 (Continued)



e. Original light image.

f. Histogram of image (e).

g. Light image after histogram equalization.

h. Histogram of image (g).

Histogram equalization may not always provide the desired effect because goal is fixed—to distribute the gray-level values as evenly as possible. To allow 1 interactive histogram manipulation, the ability to specify the histogram is necessa *Histogram specification* is the process of defining a histogram and modifying the his gram of the original image to match the histogram as specified. This process can implemented by: 1) finding the mapping table to histogram-equalize the image, specifying the desired histogram, 3) finding the mapping table to histogram-equali the values of the desired histogram, and 4) mapping the original values to the valu from step 3, by using the table from step 1. This process is best illustrated by examp

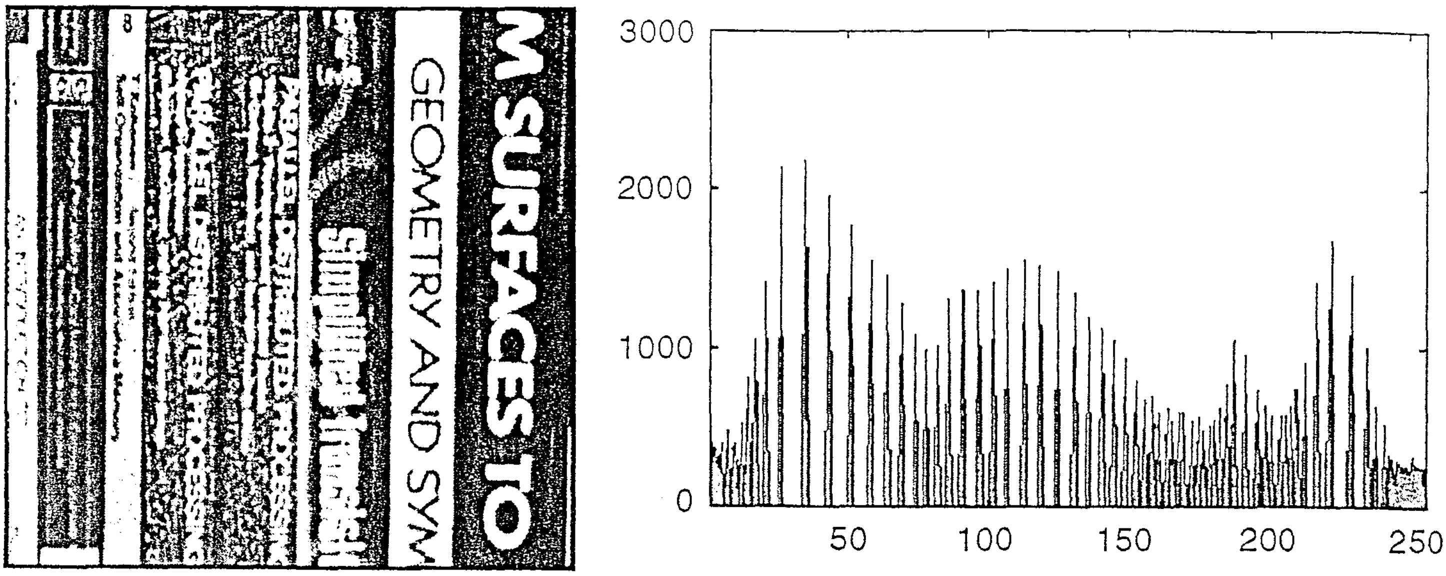


FIGURE 15 Histogram equalization applied to the image of books.

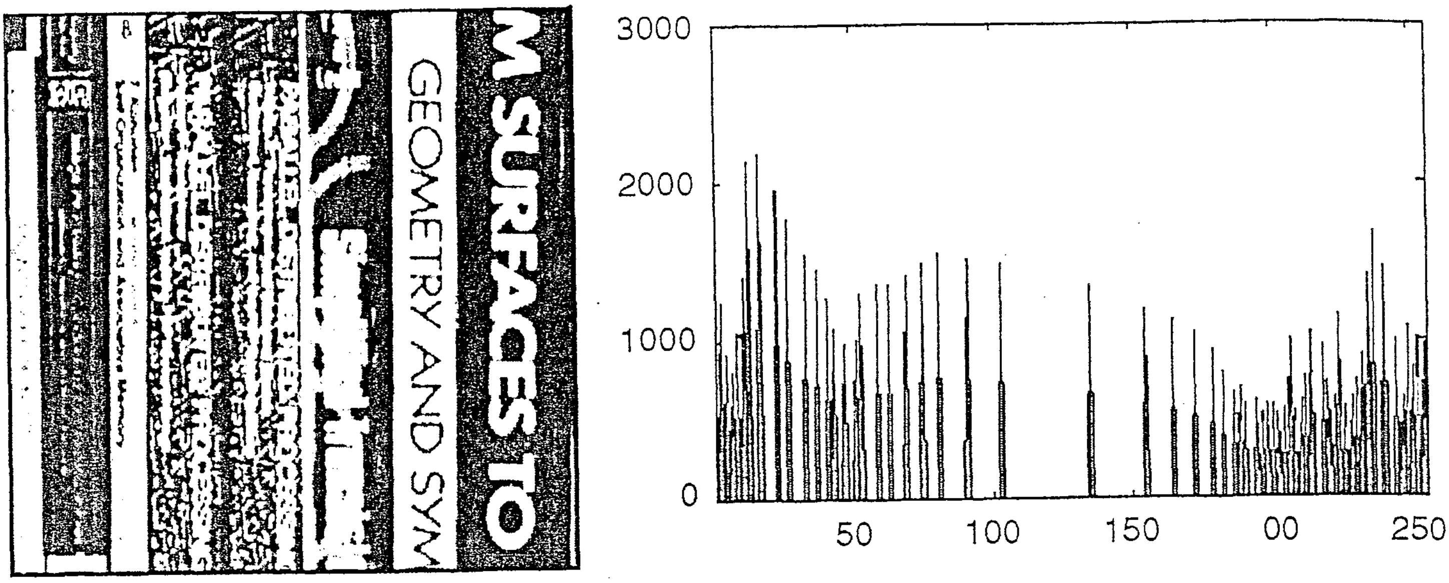


FIGURE 16 Histogram of the image of books shaped to match a "V".

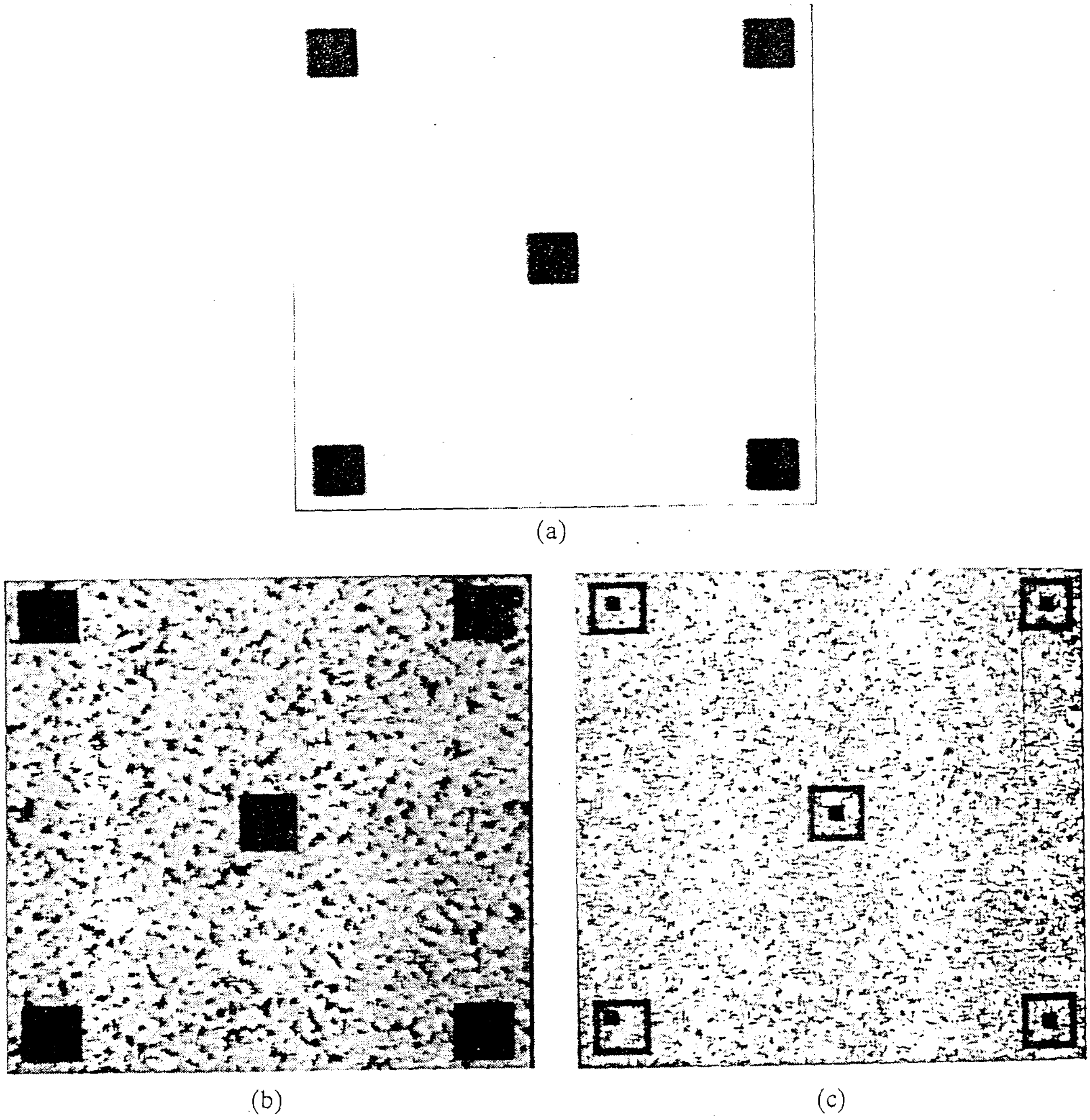
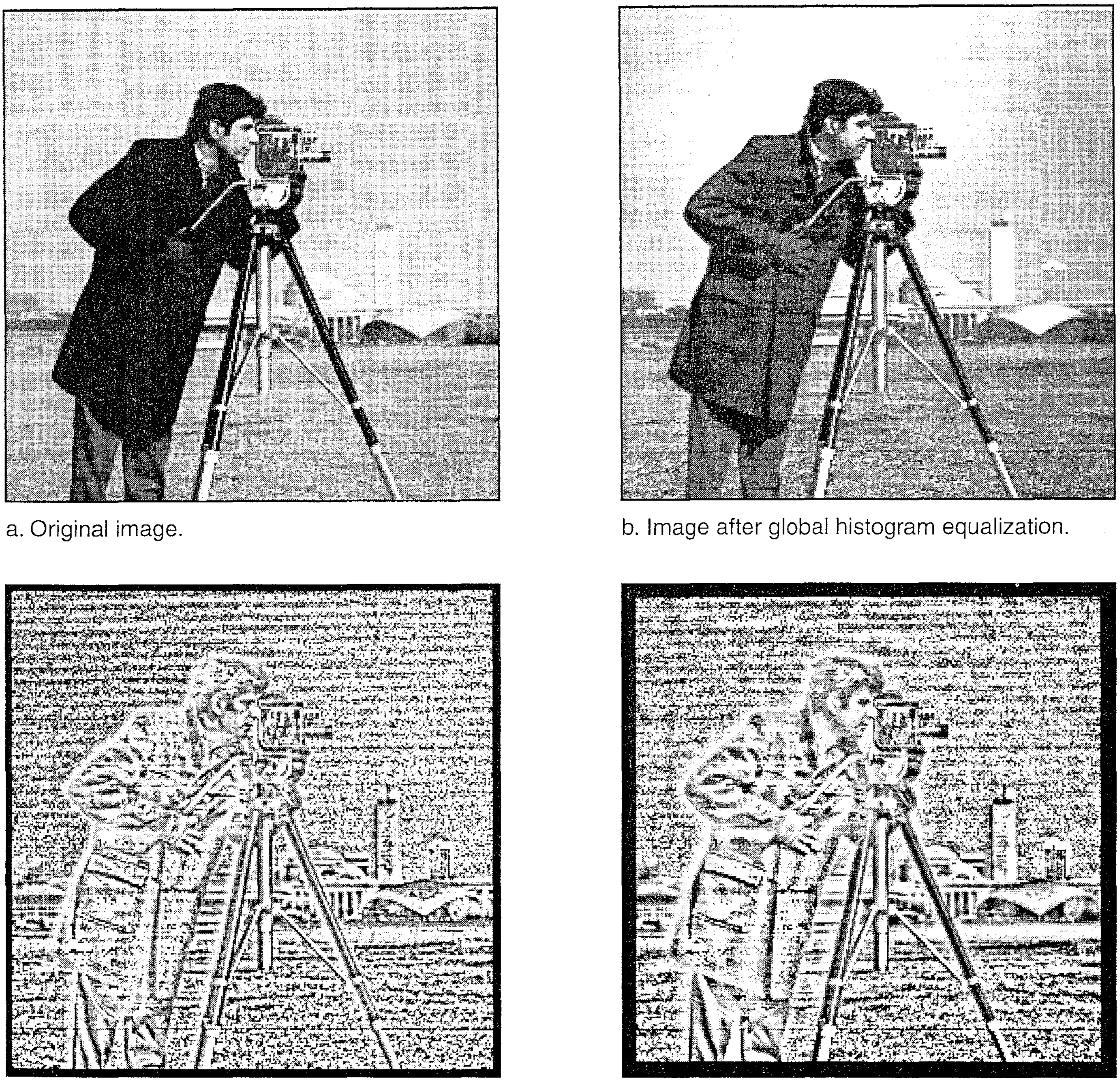


Figure 4.15 *(a) Original image; (b) result of global histogram equalization; (c) result of local histogram equalization using a 7 × 7 neighborhood about each pixel. (From Fu, Gonzalez, and Lee [1987].)*

Figure 4.2-11 Local Histogram Equalization



a. Original image.

b. Image after global histogram equalization.

c. Image after local histogram equalization, window = 7 × 7.

d. Image after local histogram equalization, window = 15 × 15.

modification techniques, which use only global parameters and result in fixed gr level transformations. The image is processed using the sliding window concept (s Figure. 3.3-1), the local image statistics are found by considering only the current w dow (subimage), and the global parameters are found by considering the entire ima It is defined as follows:

$$\text{ACE} = k_1 \left[\frac{m_{I(r,c)}}{\sigma_l(r,c)} \right] \left[I(r,c) - m_l(r,c) \right] + k_2 \; m_l(r,c)$$

along sudden changes in intensity, or "edges," which may appear jagged following nearest neighbor interpolation.

7.2 Bilinear Interpolation

Bilinear interpolation produces a smoother interpolation than does the nearest-neighbor approach. Given four neighboring image coordinates $f(n_{10}, n_{20})$, $f(n_{11}, n_{21})$, $f(n_{12}, n_{22})$, and $f(n_{13}, n_{23})$ — these can be the four nearest neighbors of $f[\mathbf{a}(\mathbf{n})]$ — then the geometrically transformed image $g(n_1, n_2)$ is computed as

$$g(n_1, n_2) = A_0 + A_1 n_1 + A_2 n_2 + A_3 n_1 n_2, \tag{47}$$

which is a bilinear function in the coordinates (n_1, n_2). The bilinear weights A_0, A_1, A_2, and A_3 are found by solving

$$\begin{bmatrix} A_0 \\ A_1 \\ A_2 \\ A_3 \end{bmatrix} = \begin{bmatrix} 1 & n_{10} & n_{20} & n_{10}n_{20} \\ 1 & n_{11} & n_{21} & n_{11}n_{21} \\ 1 & n_{12} & n_{22} & n_{12}n_{22} \\ 1 & n_{13} & n_{23} & n_{13}n_{23} \end{bmatrix}^{-1} \begin{bmatrix} f(n_{10}, n_{20}) \\ f(n_{11}, n_{21}) \\ f(n_{12}, n_{22}) \\ f(n_{13}, n_{23}) \end{bmatrix}. \tag{48}$$

Thus, $g(n_1, n_2)$ is defined to be a linear combination of the gray levels of its four nearest neighbors. The linear combination defined by Eq. (48) is in fact the value assigned to $g(n_1, n_2)$ when the best (least-squares) planar fit is made to these four neighbors. This process of optimal averaging produces a visually smoother result.

Regardless of the interpolation approach that is used, it is possible that the mapping coordinates $a_1(n_1, n_2)$, $a_2(n_1, n_2)$ do not fall within the pixel ranges

$$0 \leq a_1(n_1, n_2) \leq N - 1$$

and/or (49)

$$0 \leq a_2(n_1, n_2) \leq M - 1,$$

in which case it is not possible to define the geometrically transformed image at these coordinates. Usually a nominal value is assigned, such as $g(\mathbf{n}) = 0$, at these locations.

7.3 Image Translation

The most basic geometric transformation is the *image translation*, where

$$a_1(n_1, n_2) = n_1 - b_1, \qquad a_2(n_1, n_2) = n_2 - b_2, \tag{50}$$

where (b_1, b_2) are integer constants. In this case $g(n_1, n_2) = f(n_1 - b_1, n_2 - b_2)$, which is a simple shift or translation of g by an amount b_1 in the vertical (row) direction and an amount

it is also used in algorithms, such as image convolu
ter 2.3), where images are shifted relative to a refe
integer shifts can be defined in either direction, the
no need for the interpolation step.

7.4 Image Rotation

Rotation of the image g by an angle θ relative to th
(n_1) axis is accomplished by the following transfor

$$a_1(n_1, n_2) = n_1 \cos\theta - n_2 \sin\theta,$$

$$a_2(n_1, n_2) = n_1 \sin\theta + n_2 \cos\theta.$$

The simplest cases are: $\theta = 90°$, where $[a_1(n_1, n_2), a$
$(-n_2, n_1)$; $\theta = 180°$, where $[a_1(n_1, n_2), a_2(n_1, n_2)] =$
and $\theta = -90°$, where $[a_1(n_1, n_2), a_2(n_1, n_2)] = (n_2,$
the rotation point is not defined here as the center
the arguments of Eq. (51) may fall outside of the im
This may be ameliorated by applying an image trans
before or after the rotation to obtain coordinate
nominal range.

7.5 Image Zoom

The *image zoom* either magnifies or minifies the
according to the mapping functions

$$a_1(n_1, n_2) = n_1/c, \qquad a_2(n_1, n_2) = n_2$$

where $c \geq 1$ and $d \geq 1$ to achieve magnification,
$d < 1$ to achieve minification. If applied to the
then the image size is also changed by a factor c
vertical (horizontal) direction. If only a small p
age is to be zoomed, then a translation may be
corner of that region, the zoom applied, and t
cropped.

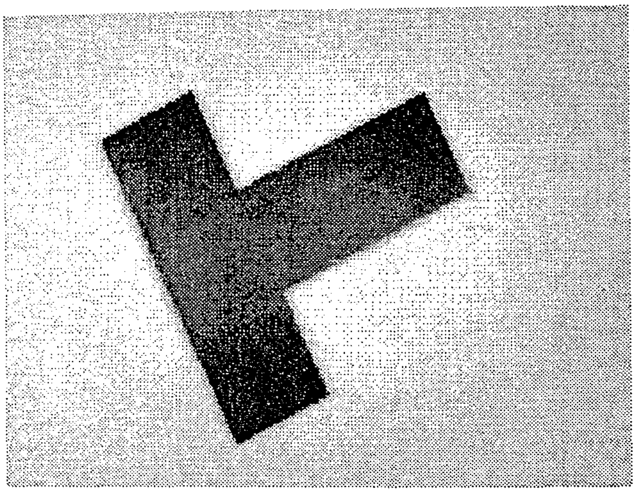
The image zoom is a good example of a geome
for which the type of interpolation is important,
high magnifications. With nearest neighbor inter
values in the zoomed image may be assigned the s
resulting in a severe "blotching" or "blocking" effe
interpolation usually supplies a much more viab

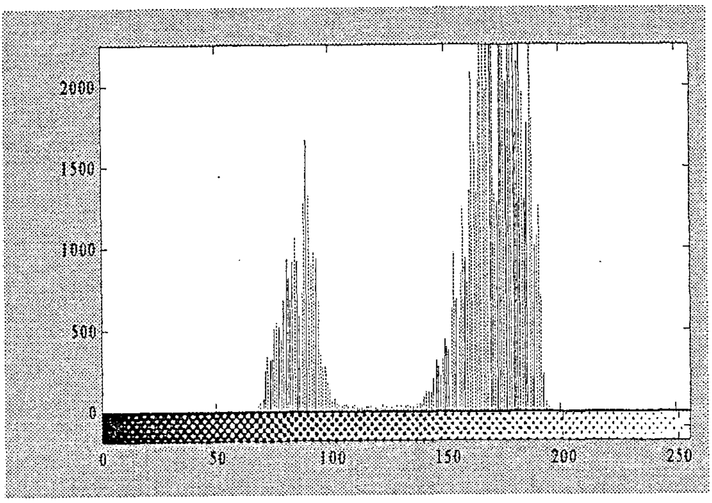
Figure 19 depicts a 4× zoom operation appli
in Fig. 13 (logarithmically transformed "studen
was first zoomed, creating a much larger ima
many pixels). The image was then translated to a
(selected, e.g., by a mouse), and then it was cropp
256 pixels around this point. Both nearest-neigh
interpolation were applied for the purpose of co
provide a nice close-up of the original, making
more identifiable. However, the bilinear result is
and it does not contain the blocking artifacts

Thresholding

80

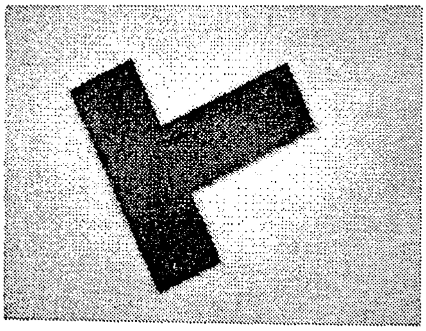
separate objects from background



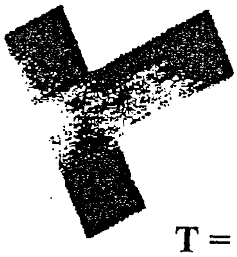


Thresholding (2)

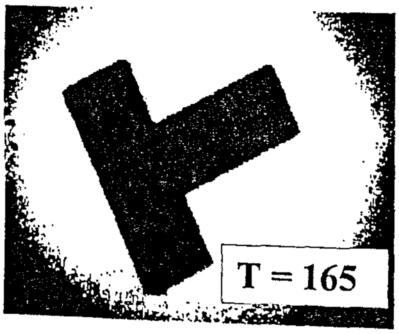
`t`



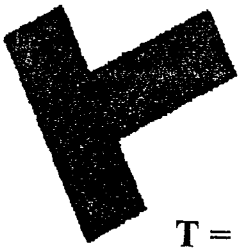
`imagesc(t > T)`



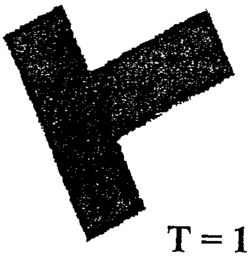
T = 90



T = 165



T = 120

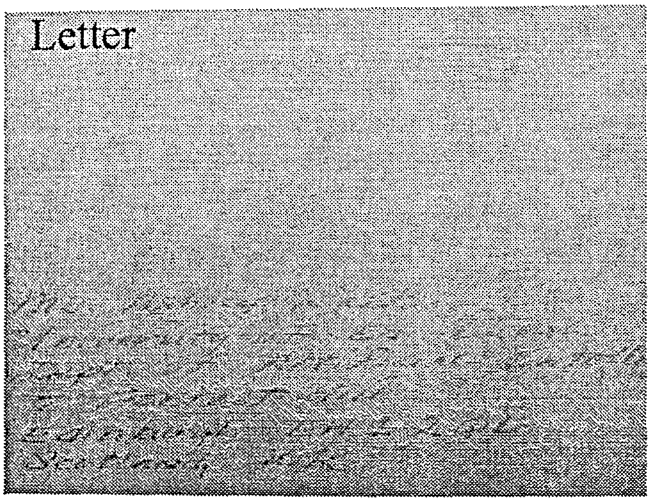


T = 100

Background separation

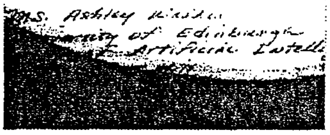
Problem: background is not uniform

Letter

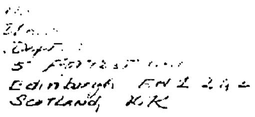


imagesc(Letter > T)

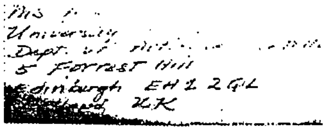
T = 100



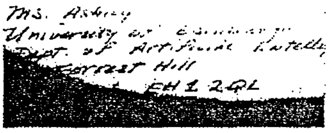
T = 70



T = 80

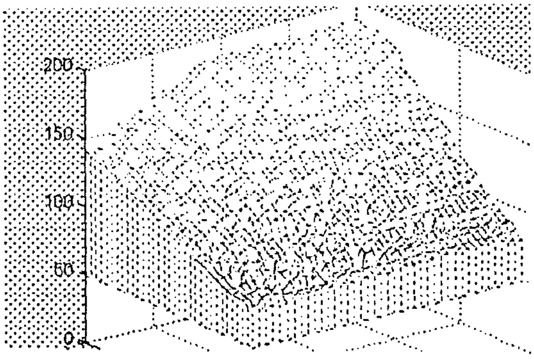


T = 90

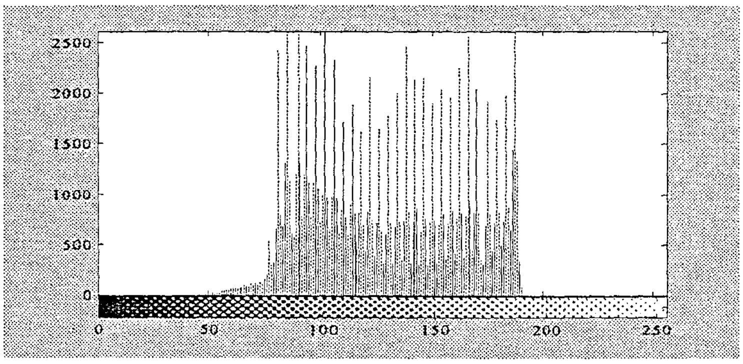


Background separation (2)

3D map of "Letter"

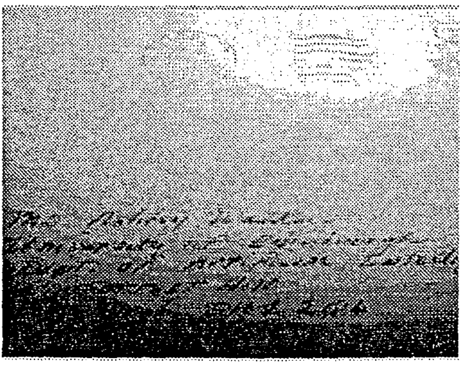


Histogram of "Letter"



Meshz(Letter)
Colormap copper

Histogram equalization does not
Help here!



size of block

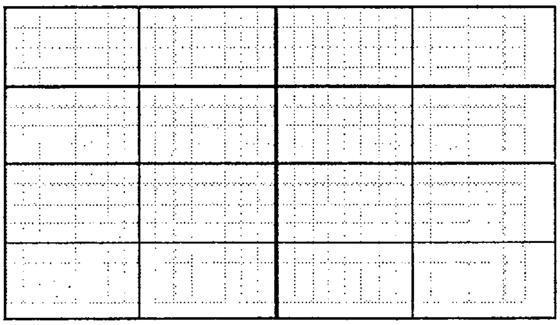
Background approximation

function

```
bg = blkproc(x,[10 10],'mean2(x)*ones(size(x))');
```

bg

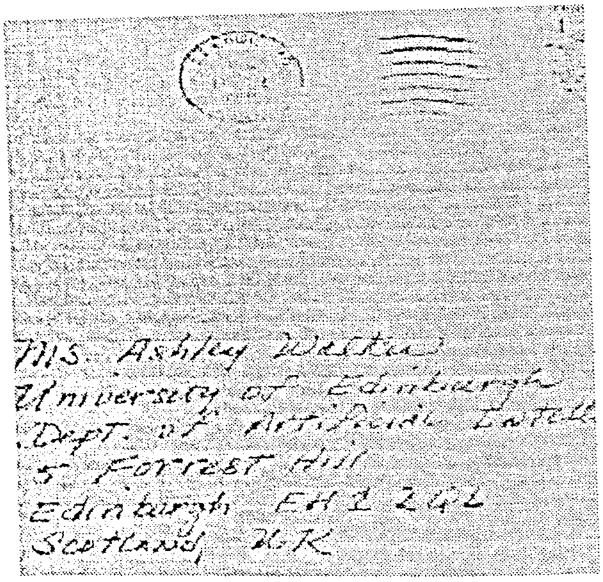
Block processing



Better approximations may be used (e.g., splines)

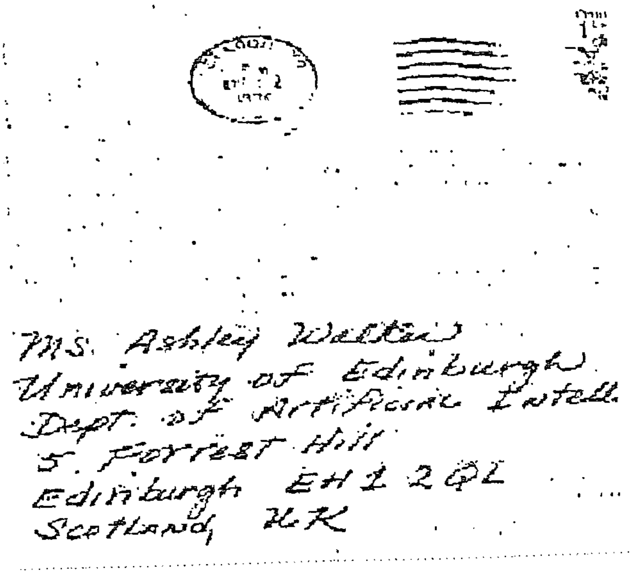
Background subtraction

Letter - bg



MS. Ashley Walker
University of Edinburgh
Dept. of Artificial Intell
5 Forrest Hill
Edinburgh EH1 2QL
Scotland, UK

Thresholding



MS. Ashley Walker
University of Edinburgh
Dept. of Artificial Intell.
5 Forrest Hill
Edinburgh EH1 2QL
Scotland, UK

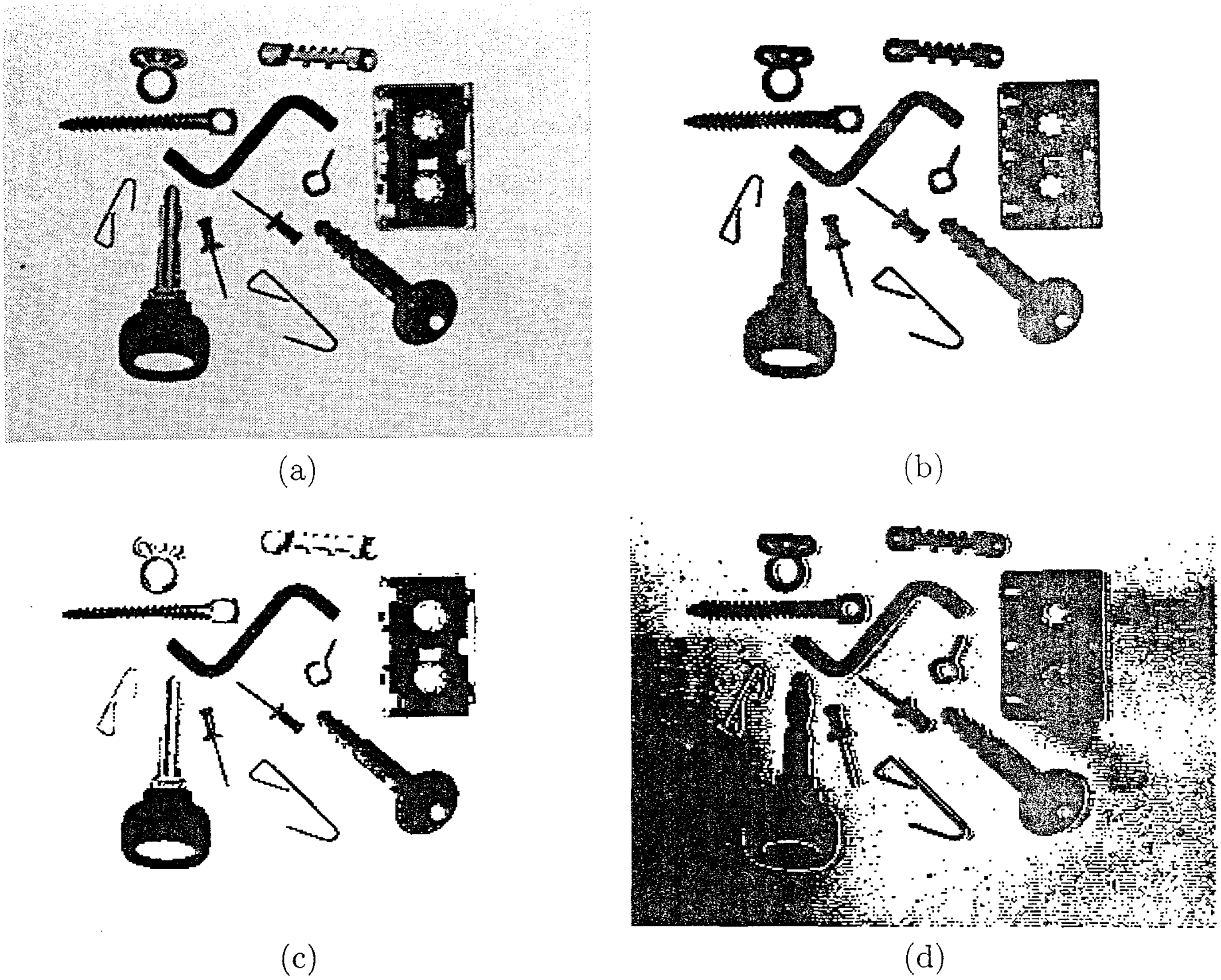


Figure 5.1: *Image thresholding: (a) original image; (b) threshold segmentation; (c) threshold too low; (d) threshold too high.*

thresholding), in which the threshold value varies over the image as a function of local image characteristics, can produce the solution in these cases.

A global threshold is determined from the whole image f:

$$T = T(f) \tag{5.3}$$

On the other hand, local thresholds are position dependent:

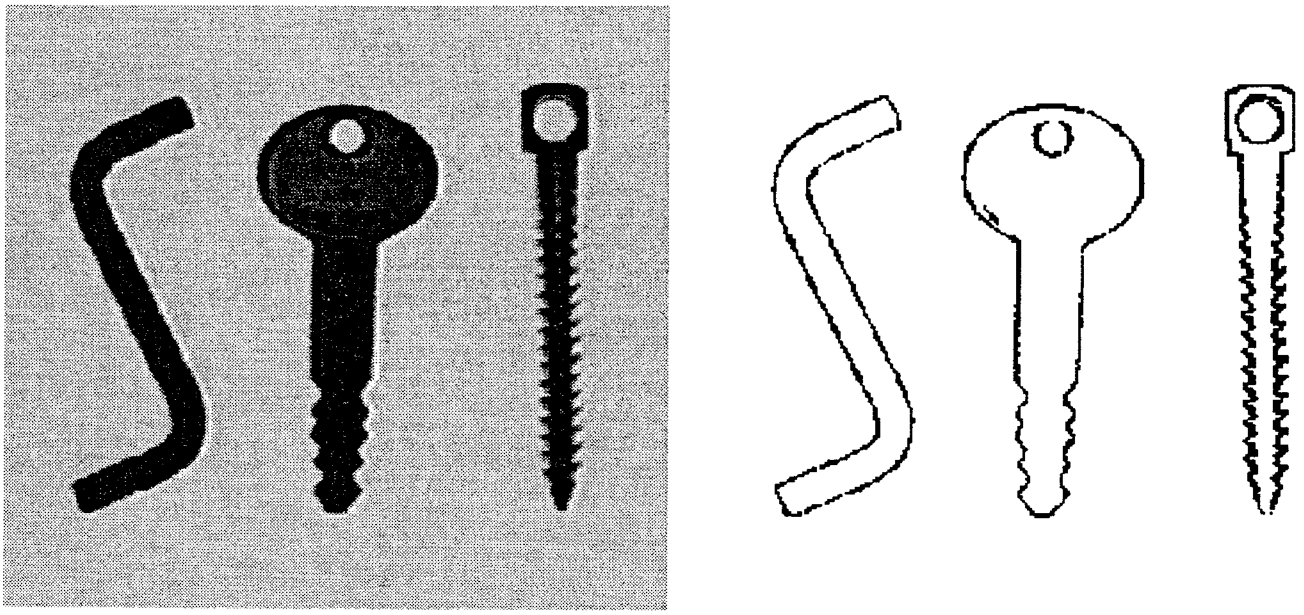
$$T = T(f, f_c) \tag{5.4}$$

where f_c is that image part in which the threshold is determined. One option is to divide the image f into subimages f_c and determine a threshold independently in each subimage; then if a threshold cannot be determined in some subimage, it can be interpolated from thresholds determined in neighboring subimages. Each subimage is then processed with respect to its local threshold.

Basic thresholding as defined by equation (5.2) has many modifications. One possibility is to segment an image into regions of pixels with gray-levels from a set D and into background otherwise (band thresholding):

$$g(i,j) \begin{array}{ll} = 1 & \text{for } f(i,j) \in D \\ = 0 & \text{otherwise} \end{array} \tag{5.5}$$

This thresholding can be useful, for instance, in microscopic blood cell segmentations, where a particular gray-level interval represents cytoplasma, the background is lighter, and the cell kernel darker. This thresholding definition can serve as a border detector as well; assuming dark objects on a light background, some gray-levels between those of objects and background can be found only in the object borders. If the gray-level set D is chosen to contain just these object-border gray-levels, and if thresholding according to equation (5.5) is used, object borders result as shown in Figure 5.2. Isolines of gray can be found using this appropriate gray-level set D.



(a) (b)

Figure 5.2: *Image thresholding modification: (a) original image; (b) border detection using band-thresholding.*

There are many modifications that use multiple thresholds, after which the resulting image is no longer binary, but rather an image consisting of a very limited set of gray-levels:

$$g(i,j) \begin{array}{ll} = 1 & \text{for } f(i,j) \in D_1 \\ = 2 & \text{for } f(i,j) \in D_2 \\ = 3 & \text{for } f(i,j) \in D_3 \\ \dots & \\ = n & \text{for } f(i,j) \in D_n \\ = 0 & \text{otherwise} \end{array} \tag{5.6}$$

where each D_i is a specified subset of gray-levels.

Another special choice of gray-level subsets D_i defines **semi-thresholding**, which is sometimes used to make human-assisted analysis easier:

$$g(i,j) \begin{array}{ll} = f(i,j) & \text{for } f(i,j) \geq T \\ = 0 & \text{for } f(i,j) < T \end{array} \tag{5.7}$$

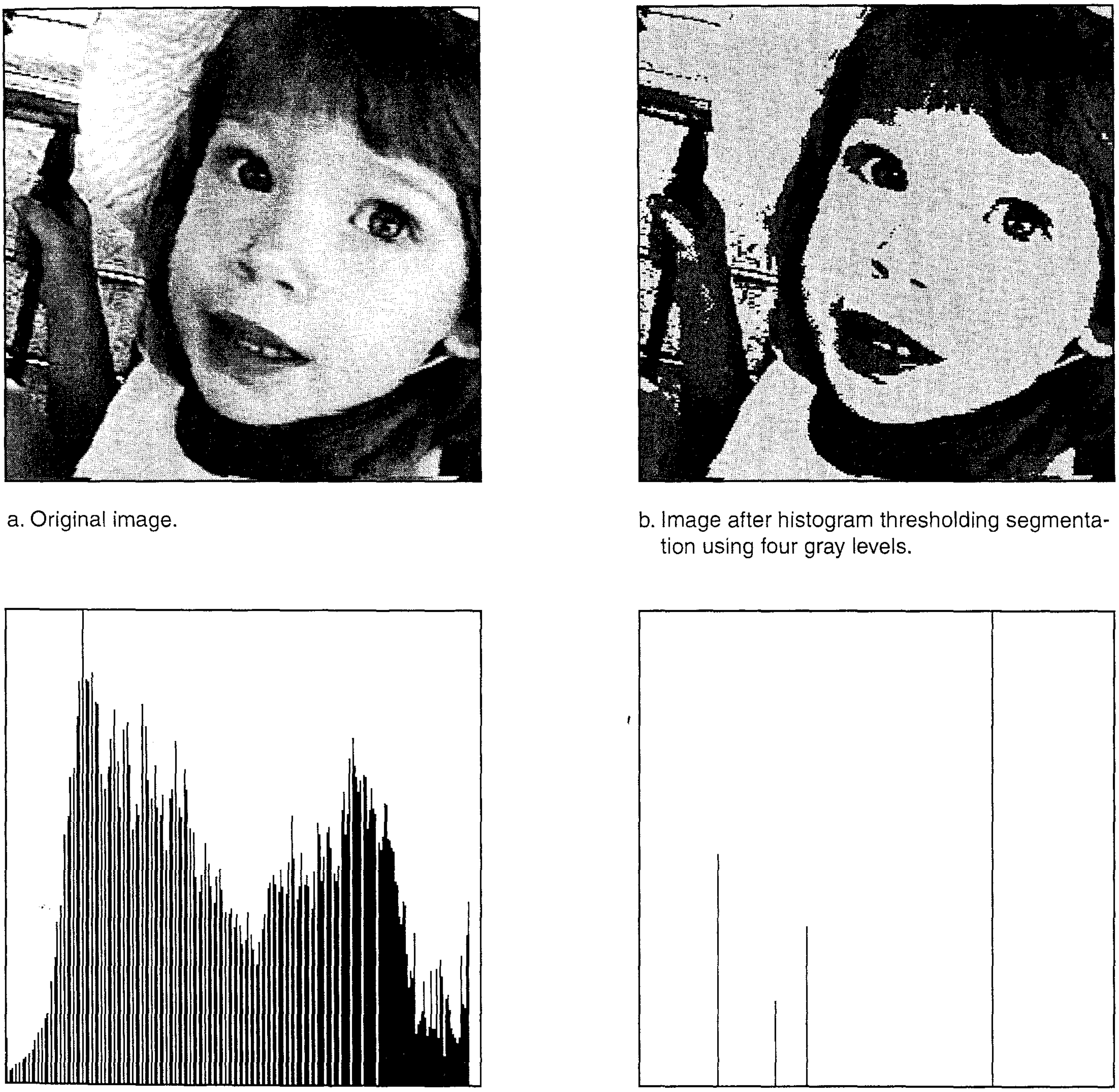


Figure 2.4-4 Histogram Thresholding Segmentation

a. Original image.

b. Image after histogram thresholding segmentation using four gray levels.

c. Histogram of image (a).

d. Histogram of image (b).

that as the white point is approached in the color space, a greater number of hues w be observable in a fixed area by the human visual system than on the perimeter of tl color triangle. This observation is application specific because it only applies to colo from white (in the center of the triangle) to the green and red vertices. Skin tumor cc ors typically range from white out to the red vertex. The SCT/Center segmentatic algorithm is outlined as follows:

1. Convert the (R,G,B) triple into spherical coordinates (L, angle A, angle B).

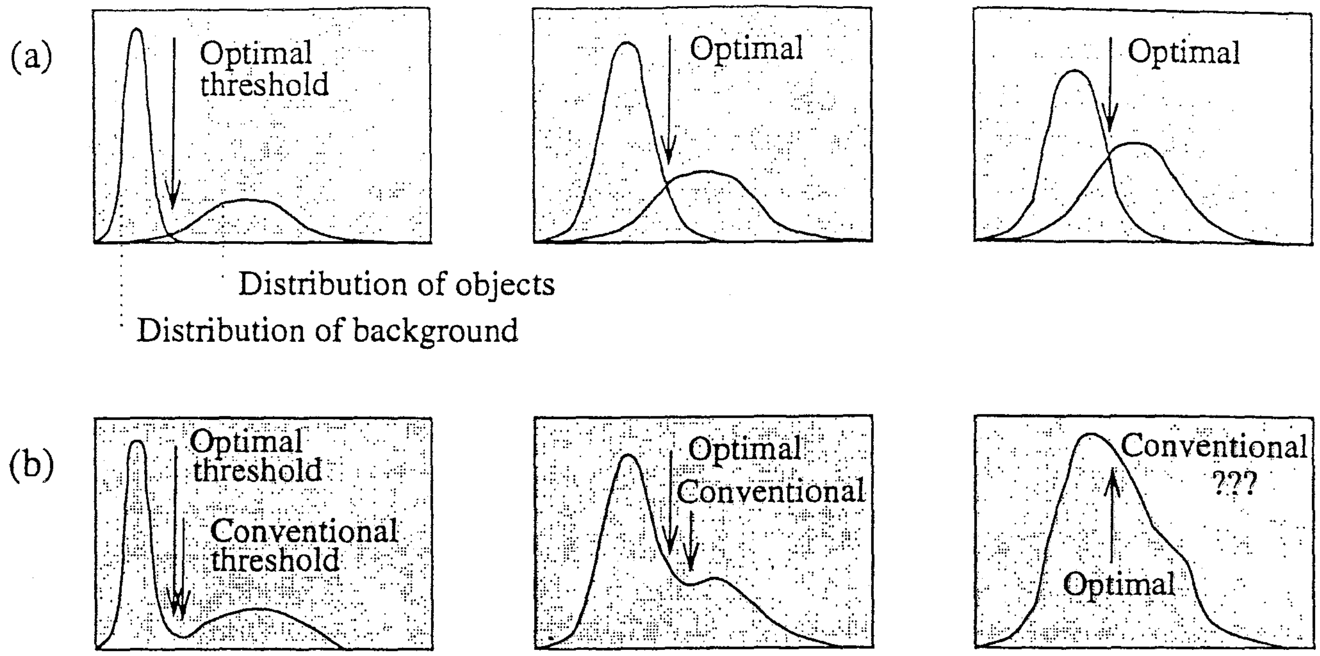


Figure 5.4: *Gray-level histograms approximated by two normal distributions— the threshold is set to give minimum probability of segmentation error: (a) probability distributions of background and objects; (b) corresponding histograms and optimal threshold.*

The following algorithm represents a simpler version that shows a rationale for this approach [Ridler and Calvard 78] and works well even if the image histogram is not bi-modal. This method assumes that regions of two main gray-levels are present in the image, thresholding of printed text being an example. The algorithm is iterative, four to ten iterations usually being sufficient.

Algorithm 5.2: Iterative (optimal) threshold selection

1. Assuming no knowledge about the exact location of objects, consider as a first approximation that the four corners of the image contain background pixels only and the remainder contains object pixels.

2. At step t, compute μ_B^t and μ_O^t as the mean background and object gray-level, respectively, where segmentation into background and objects at step t is defined by the

$$\mu_B^t = \frac{\sum_{(i,j)\in\text{background}} f(i,j)}{\#\text{background\_pixels}} \qquad \mu_O^t = \frac{\sum_{(i,j)\in\text{objects}} f(i,j)}{\#\text{object\_pixels}}$$

(5.

3. Set

$$T^{(t+1)} = \frac{\mu_B^t + \mu_O^t}{2}$$

(5.

$T^{(t+1)}$ now provides an updated background—object distinction.

4. If $T^{(t+1)} = T^{(t)}$, halt; otherwise return to step 2.

optimal thresholds can be determined for each voxel and used for segmentation. In [Fra et al. 95, Santago and Gage 93], the partial volume effect was also considered (in brain M images, the finite-size voxels can consist of a combination of, e.g., gray and white matt and a volume percentage corresponding to WM, GM, and CSF was calculated for each vox Figure 5.6 gives an example of such brain segmentation. The brighter the voxel location individual segmented images, the higher the volume percentage of the GM, WM, or CSF that particular voxel. In each voxel, the sum of partial volume percentages is 100%.

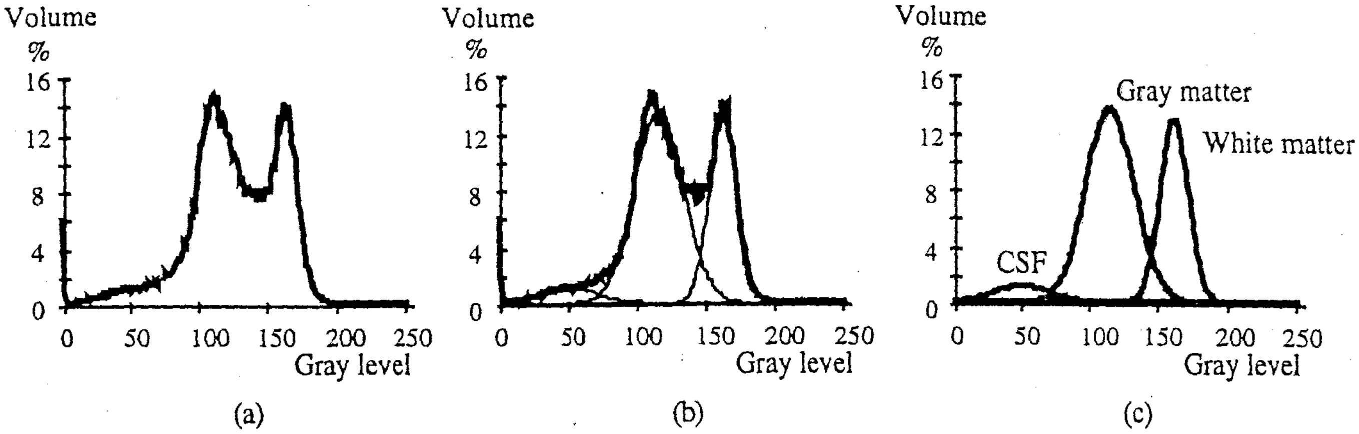


Figure 5.5: *Segmentation of 3D T1-weighted MR brain image data using optimal thresl olding: (a) local gray-level histogram; (b) fitted Gaussian distributions, global 3D image fi (c) Gaussian distributions corresponding to WM, GM, and CSF. Courtesy R.J. Frank, T. Grabowski, Human Neuroanatomy and Neuroimaging Laboratory, Department of Neurolog The University of Iowa.*

5.1.3 Multi-spectral thresholding

Many practical segmentation problems need more information than is contained in one spectral band. Color images are a natural example, in which information is coded in three spectra bands, for example, red, green, and blue; multi-spectral remote sensing images or meteorological satellite images may have even more spectral bands. One segmentation approacl determines thresholds independently in each spectral band and combines them into a singl segmented image.

Algorithm 5.3: Recursive multi-spectral thresholding

1. Initialize the whole image as a single region.

2. Compute a smoothed histogram (see Section 2.3.2) for each spectral band. Find the most significant peak in each histogram and determine two thresholds as local minima on either side of this maximum. Segment each region in each spectral band into sub-regions according to these thresholds. Each segmentation in each spectral band is projected into a multi-spectral segmentation—see Figure 5.7. Regions for the next processing steps are those in the multi-spectral image.

3. Repeat step 2 for each region of the image until each region's histogram contains only one significant peak.

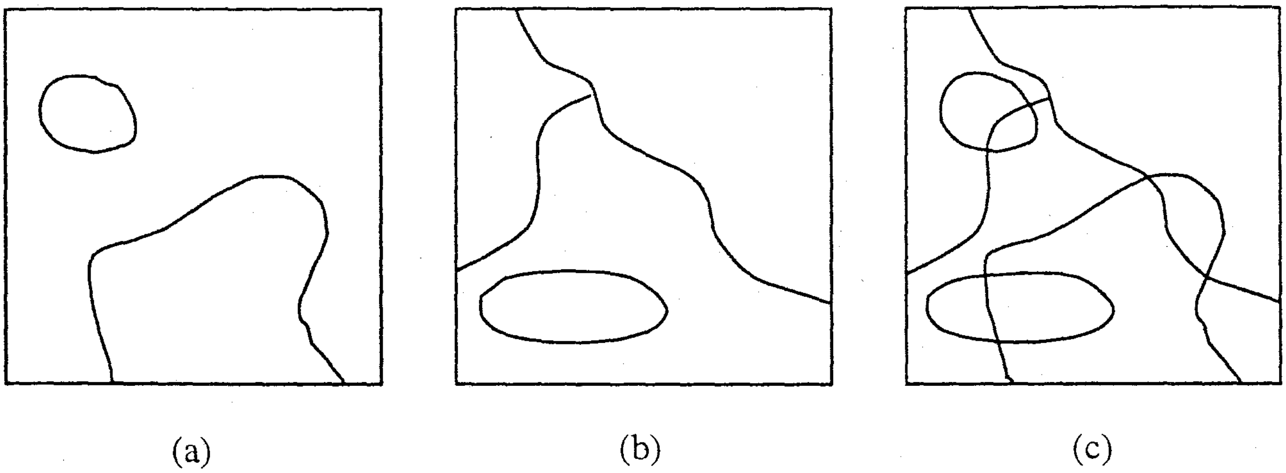


Figure 5.7: *Recursive multi-spectral thresholding: (a) band 1 thresholding; (b) band 2 thresholding; (c) multi-spectral segmentation.*