

Histogram (Practical)

Digital Image Processing

Histogram

- Image Histogram
- Histogram Equalization
- Histogram Specification
- Other Histogram Modification techniques
 - Histogram Sliding
 - Histogram Stretching
 - Histogram Shrinking

Histogram

- The histogram of a monochrome image is a graphical representation of the frequency of occurrence of each grey level in the image.
- Each individual histogram entry can be expressed mathematically as

$$h(k) = n_k = \text{card}\{(x, y) | f(x, y) = k\}$$

where:

$k = 0, 1, 2, \dots, L - 1$ with L is the number of grey levels of the digitized, and

card denotes the cardinality of a set, that is, the number of elements in that set

Normalized Histogram

- A normalized histogram can be mathematically defined as

$$p(r_k) = \frac{n_k}{n}$$

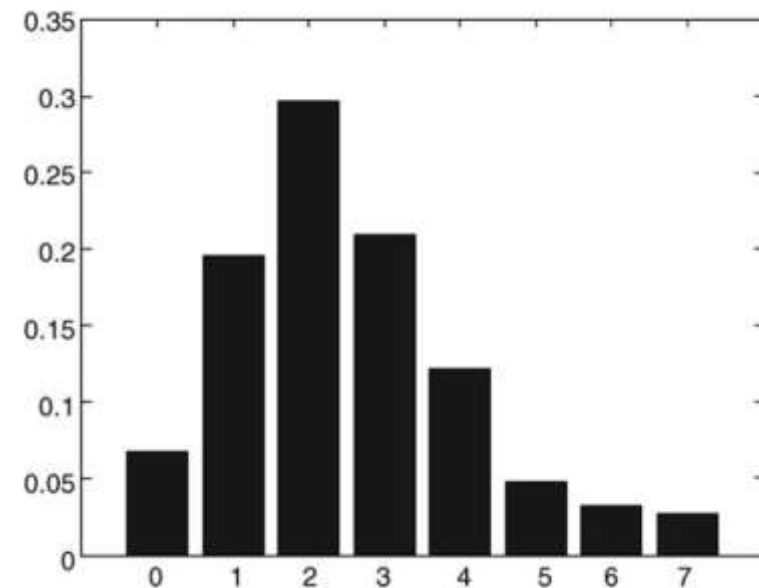
where :

n is the total number of pixels in the image and $p(r_k)$ is the probability (percentage) of the k th gray level (r_k).

Histogram (Matlab)

- *Matlab's build in function* untuk histogram citra adalah *imhist*.
- Fungsi alternatif menampilkan histogram adalah *bar*, *plot*, dan *stem*.

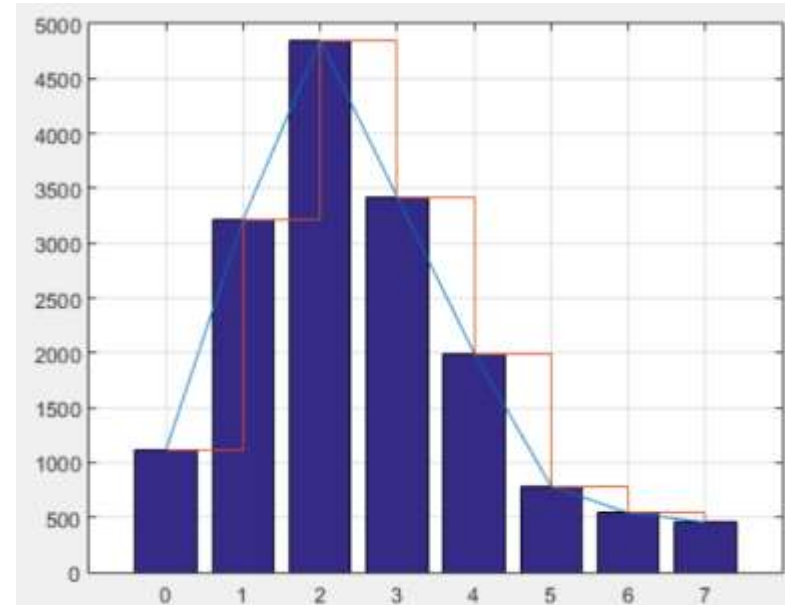
Gray Level (r_k)	n_k	$p(r_k)$
0	1120	0.068
1	3214	0.196
2	4850	0.296
3	3425	0.209
4	1995	0.122
5	784	0.048
6	541	0.033
7	455	0.028
Total	16,384	1.000



Histogram (Matlab)

Command Window

```
>> nk = [1120 3214 4850 3425 1995 784 541 455];  
>> rk = 0:1:7;  
>> bar(rk, nk); grid on;  
>> hold on; plot(rk, nk); grid on;  
>> hold on; stairs(rk, nk); grid on;  
fx >> |
```



Histogram Citra

- Menampilkan histogram citra dapat dilakukan untuk setiap derajat keabuan
- Akan tetapi, untuk citra dengan jumlah derajat keabuan lebih dari 8-bit akan menghasilkan larik (*array*) yang besar dan sulit dipakai
- Oleh karena itu, digunakan Teknik *binning*

Histogram Citra: *Binning*

- Teknik ini dapat direpresentasikan dalam notasi:

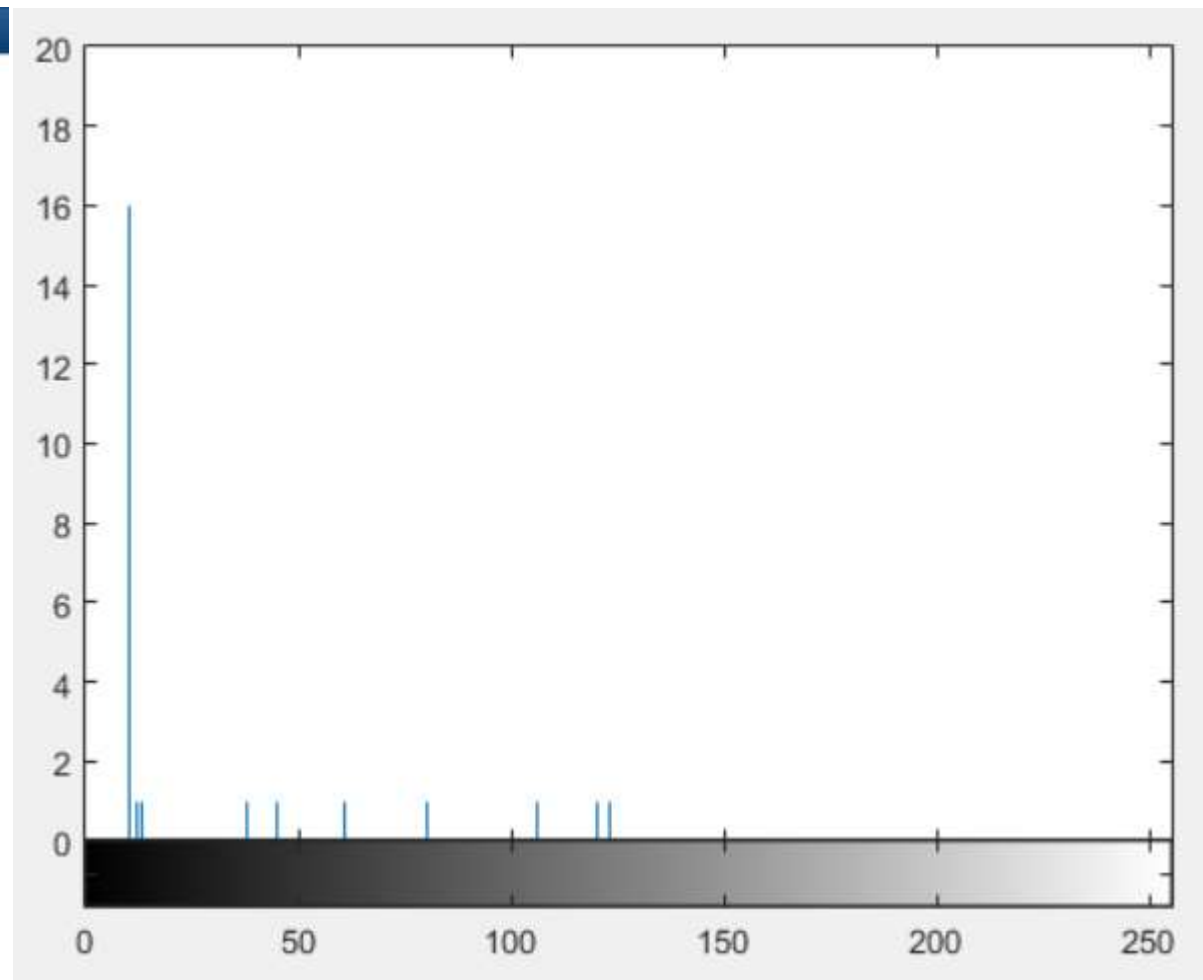
$$h(j) = \text{card}\{(x, y) | r_j \leq f(x, y) \leq r_{j+1}\} \text{ untuk } 0 \leq j < B$$

Dimana

- B adalah jumlah *bin(bucket)* dan $B \leq K$ dengan $0 \leq k \leq K$
- $r_j = j \frac{K}{B} = j \cdot k_b$ dengan k_b adalah Panjang setiap interval

Histogram Citra

```
Command Window  
  
>> A  
  
A =  
  
    10    10    13    12    10  
   120    10    10    10    10  
    10    10   123    10    10  
    80    61    45    10    10  
    10    38    10    10   106  
  
>> imhist(A); ylim([0 20]);
```

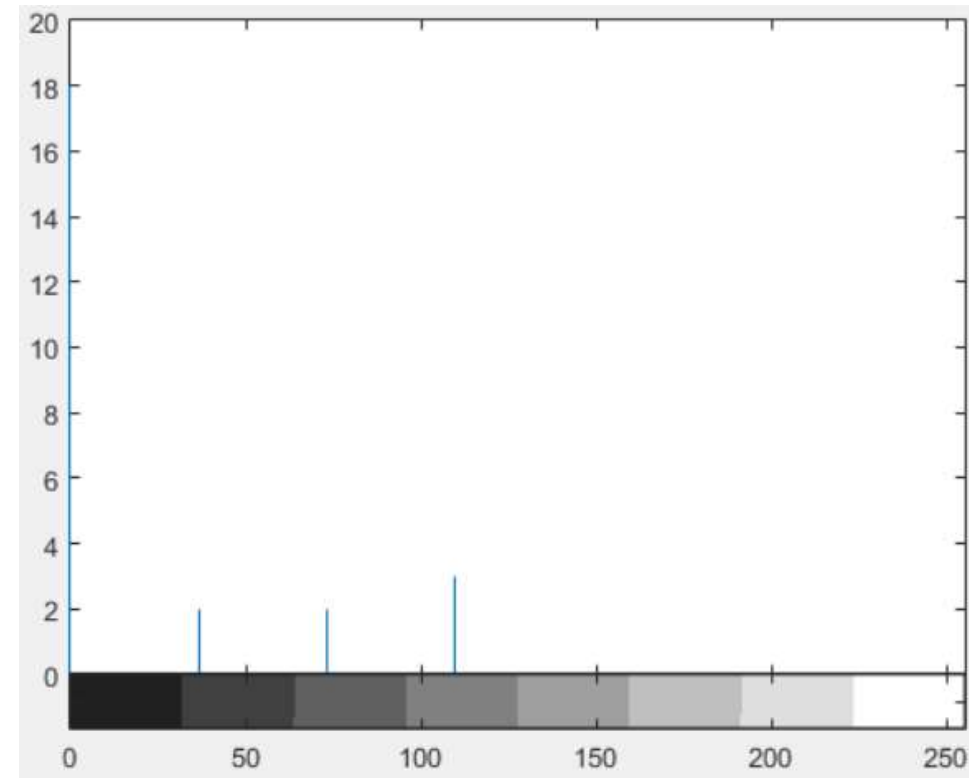


Histogram Citra: *Binning*

Command Window

```
A =  
  
    10    10    13    12    10  
   120    10    10    10    10  
    10    10   123    10    10  
    80    61    45    10    10  
    10    38    10    10   106  
  
>> B = 7;  
>> [counts, bin] = imhist(A, B+1);  
>> h = [bin, counts];
```

	1	2
1	0	18
2	36.4286	2
3	72.8571	2
4	109.2857	3
5	145.7143	0
6	182.1429	0
7	218.5714	0
8	255	0



Interpreting Histogram

Histograms provide an easy, practical, and straightforward way of evaluating image attributes, such as overall contrast and average brightness.



For a low contrast image, the histogram is clustered within a narrow range of gray levels (Figure 9.2a)

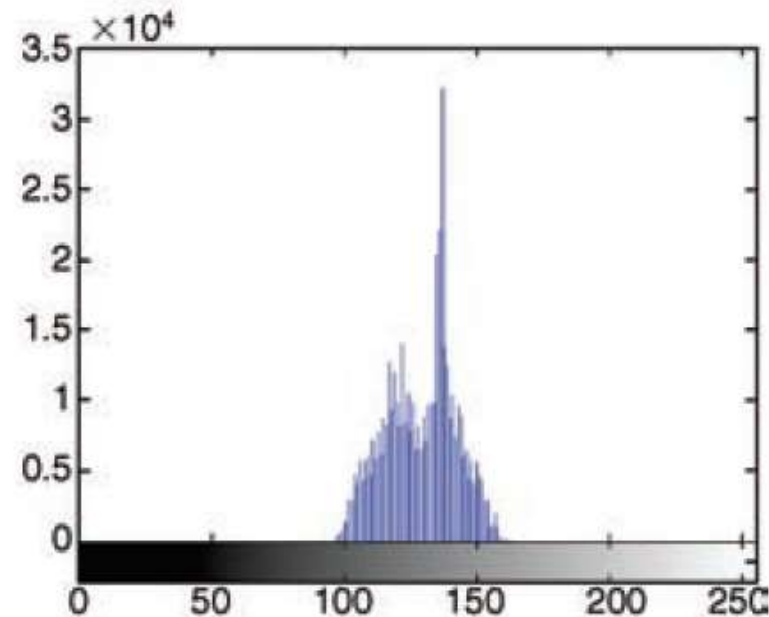


FIGURE 9.2 Examples of images and corresponding histograms. Original image in part (b): courtesy of MathWorks.

Interpreting Histogram

whereas a high contrast image usually exhibits a bimodal histogram with clear separation between the two predominant modes (Figure 9.2b)

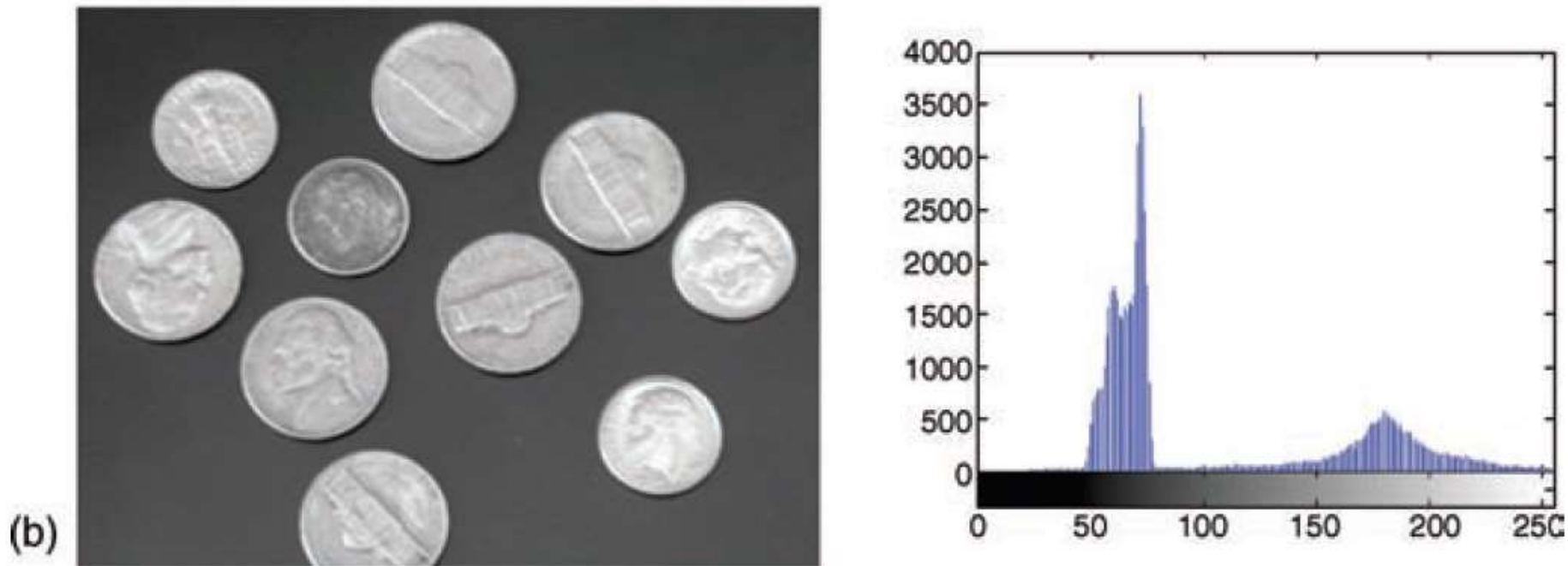


FIGURE 9.2 Examples of images and corresponding histograms. Original image in part (b): courtesy of MathWorks.

Interpreting Histogram

The histogram of a predominantly dark image contains a concentration of bars on the lower end of the gray-level range (Figure 9.2c)

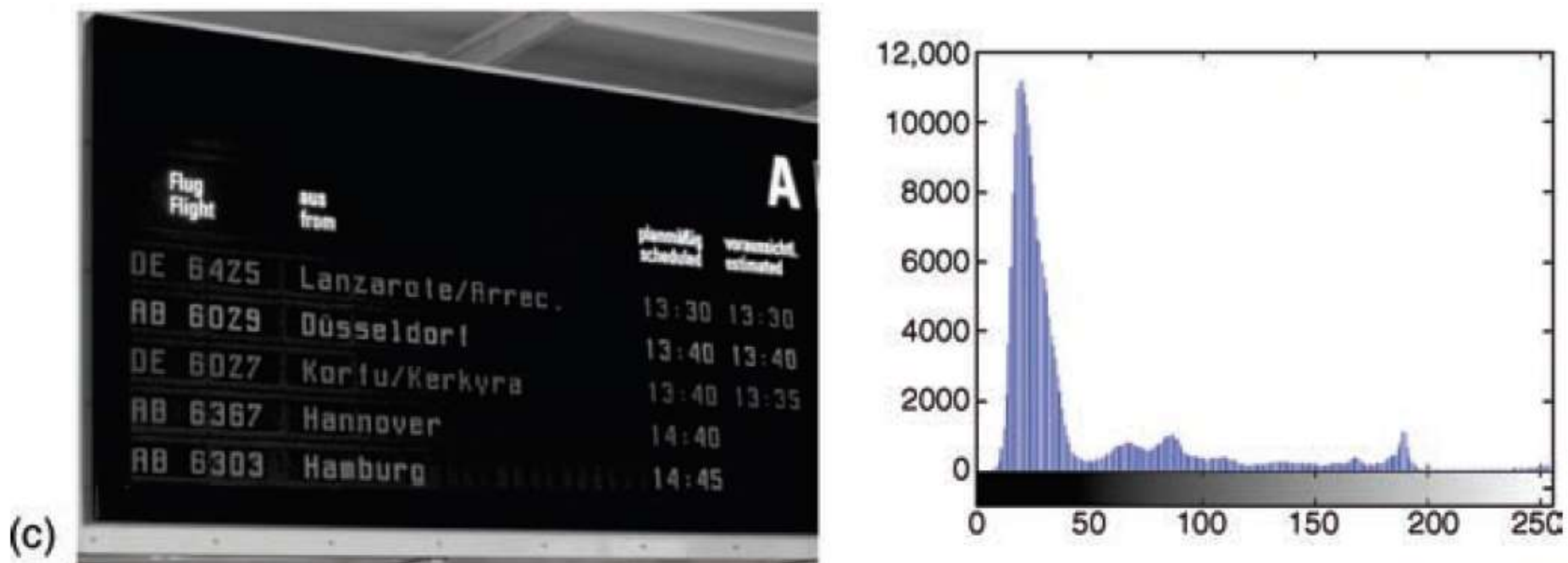


FIGURE 9.2 Examples of images and corresponding histograms. Original image in part (b): courtesy of MathWorks.

Interpreting Histogram

the histogram for a bright image is mostly concentrated on the opposite end (Figure 9.2d)

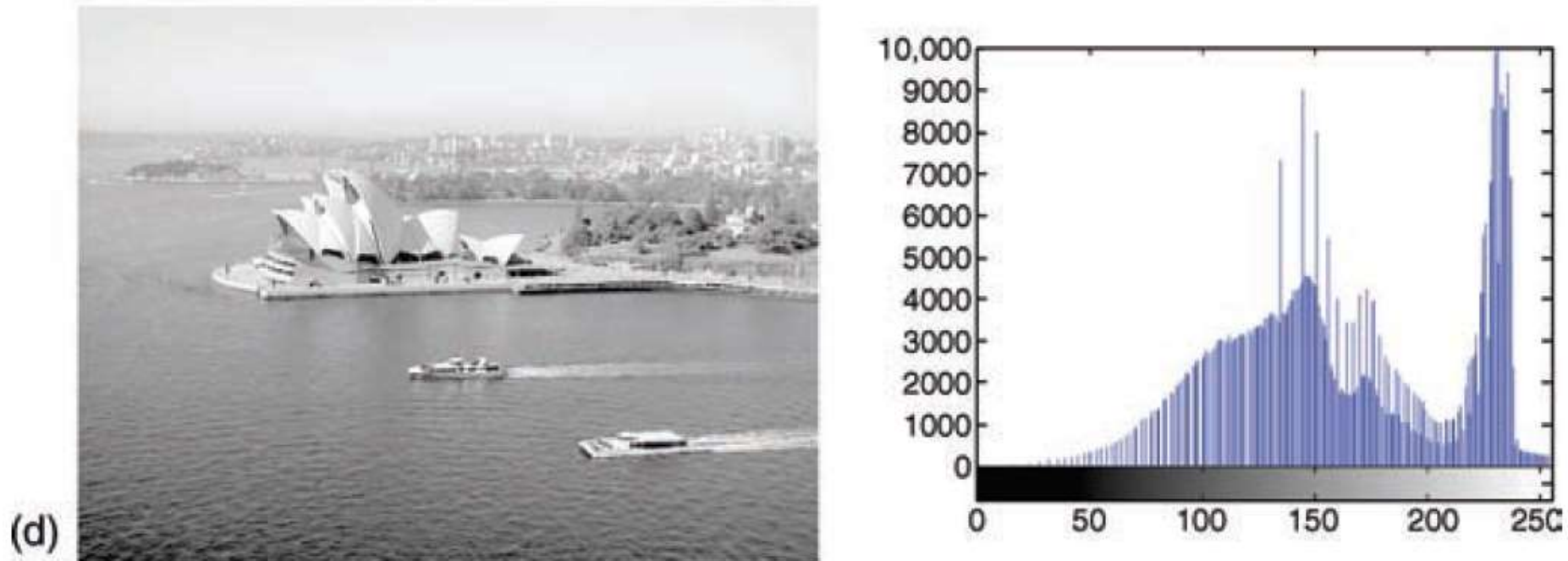


FIGURE 9.2 Examples of images and corresponding histograms. Original image in part (b): courtesy of MathWorks.

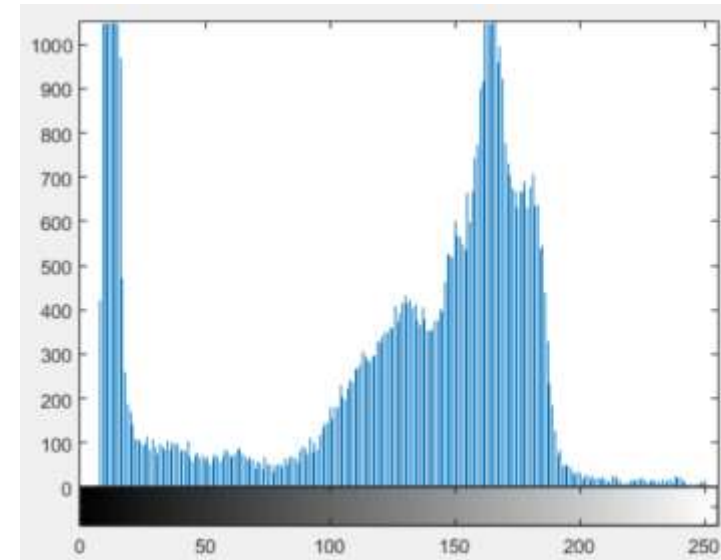
Interpreting Histogram

Command Window

```
>> minValue = min(min(img));  
>> maxValue = max(max(img));  
>> average = mean2(img);  
>> modeValue = mode(reshape(img, [1 size(img,1)*size(img,2)]));  
>> medianValue = median(reshape(img, [1 size(img,1)*size(img,2)]));  
>> stdDev = std(double(reshape(img, [1 size(img,1)*size(img,2)])));  
  
>> statImg = [minValue; maxValue; average; modeValue; medianValue; stdDev]
```

```
statImg =
```

```
7  
253  
119  
14  
144  
62
```



Interpreting Histogram

- It is important to note that even though a histogram carries significant qualitative and quantitative information about the corresponding image
 - e.g., minimum, average, and maximum gray-level values, dominance of bright or dark pixels, etc.
- Other qualitative conclusions can be reached only upon examining the image itself
 - e.g., overall quality of the image, presence or absence of noise, etc.
- Moreover, although a histogram provides the frequency distribution of gray levels in an image, it tells us nothing about the spatial distribution of the pixels whose gray levels are represented in the histogram.

Blank Space

Histogram Equalization

- Histogram equalization is a technique by which the gray-level distribution of an image is changed in such a way as to obtain a uniform (flat) resulting histogram, in which the percentage of pixels of every gray level is the same.
- To perform histogram equalization, it is necessary to use an auxiliary function, called the *transformation function*, $T(r)$. Such transformation function must satisfy two criteria
 - $T(r)$ must be a monotonically increasing function in the interval $0 \leq r \leq L - 1$.
 - $0 \leq T(r) \leq L - 1$ for $0 \leq r \leq L - 1$

Histogram Equalization

- The most usual transformation function is the *cumulative distribution function (cdf)* of the original probability mass function, given by:

$$s_k = T(r_k) = \sum_{j=0}^k \frac{n_j}{n} = \sum_{j=0}^k p(r_j)$$

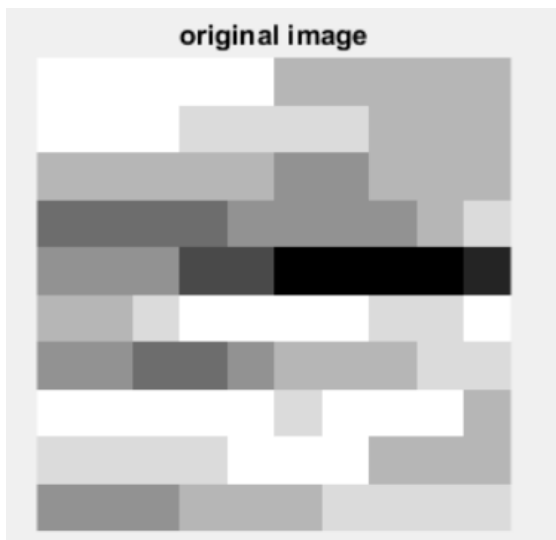
- where s_k is the new (mapped) gray level for all pixels whose original gray level used to be r_k .

In Matlab

- MATLAB's IPT has a built-in function to perform histogram equalization of a monochrome image: *histeq*.
- For the sake of histogram equalization, the syntax for *histeq* is usually $J = \text{histeq}(I, n)$, where n (whose default value is 64) is the number of desired gray levels in the output image.
- In addition to histogram equalization, this function can also be used to perform histogram matching

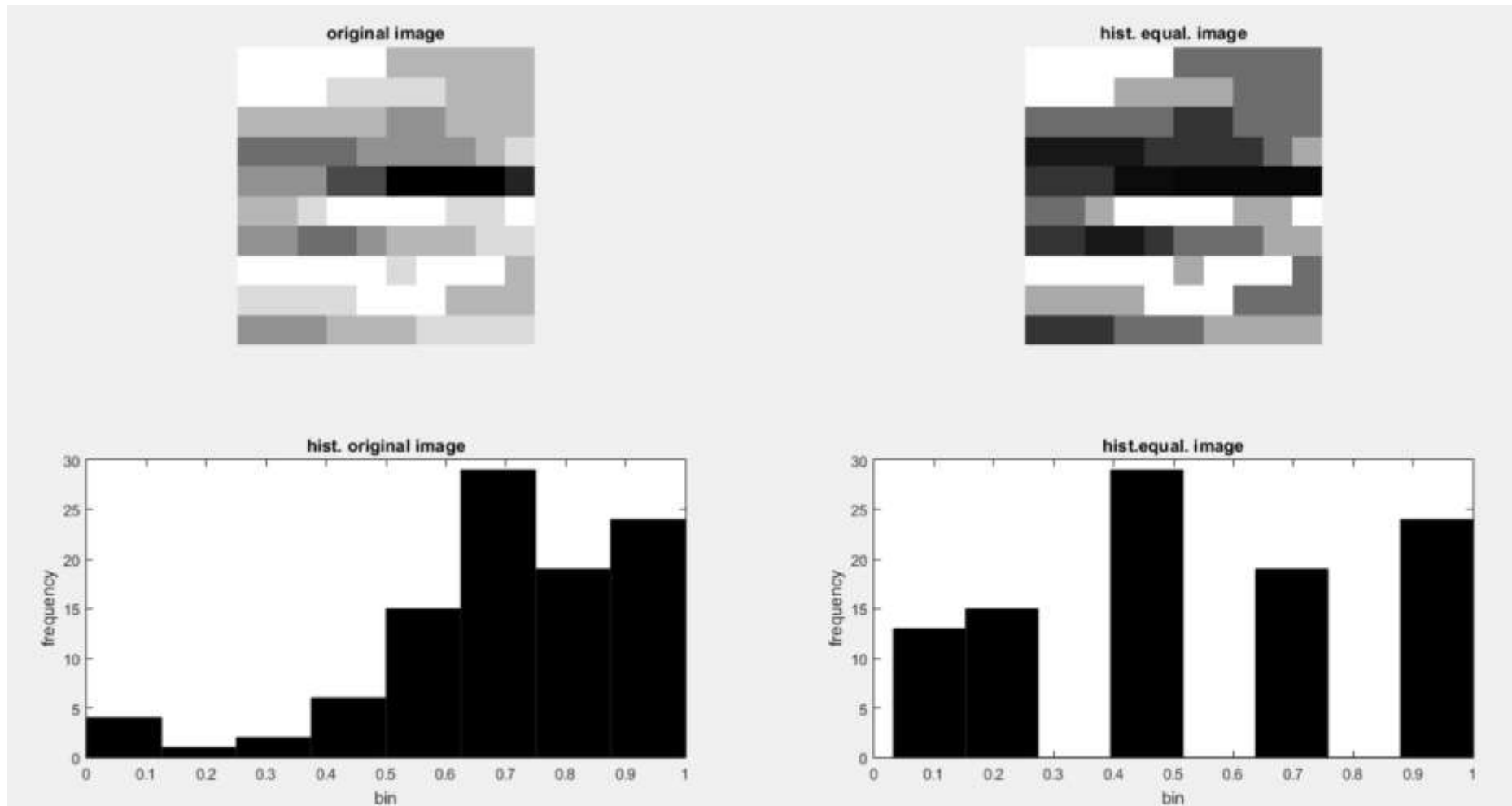
Equalisasi Histogram

7	7	7	7	7	5	5	5	5	5
7	7	7	6	6	6	6	5	5	5
5	5	5	5	5	4	4	5	5	5
3	3	3	3	4	4	4	4	5	6
4	4	4	2	2	0	0	0	0	1
5	5	6	7	7	7	7	6	6	7
4	4	3	3	4	5	5	5	6	6
7	7	7	7	7	6	7	7	7	5
6	6	6	6	7	7	7	5	5	5
4	4	4	5	5	5	6	6	6	6



k	rk	nk	p(rk)	sig(p(rk))	T(rk)
0	0	4	0.04	0.04	0
1	0.142857	1	0.01	0.05	0
2	0.285714	2	0.02	0.07	0
3	0.428571	6	0.06	0.13	0
4	0.571429	15	0.15	0.28	1
5	0.714286	29	0.29	0.57	3
6	0.857143	19	0.19	0.76	5
7	1	24	0.24	1	7
Total		100	1		

Equalisasi Histogram



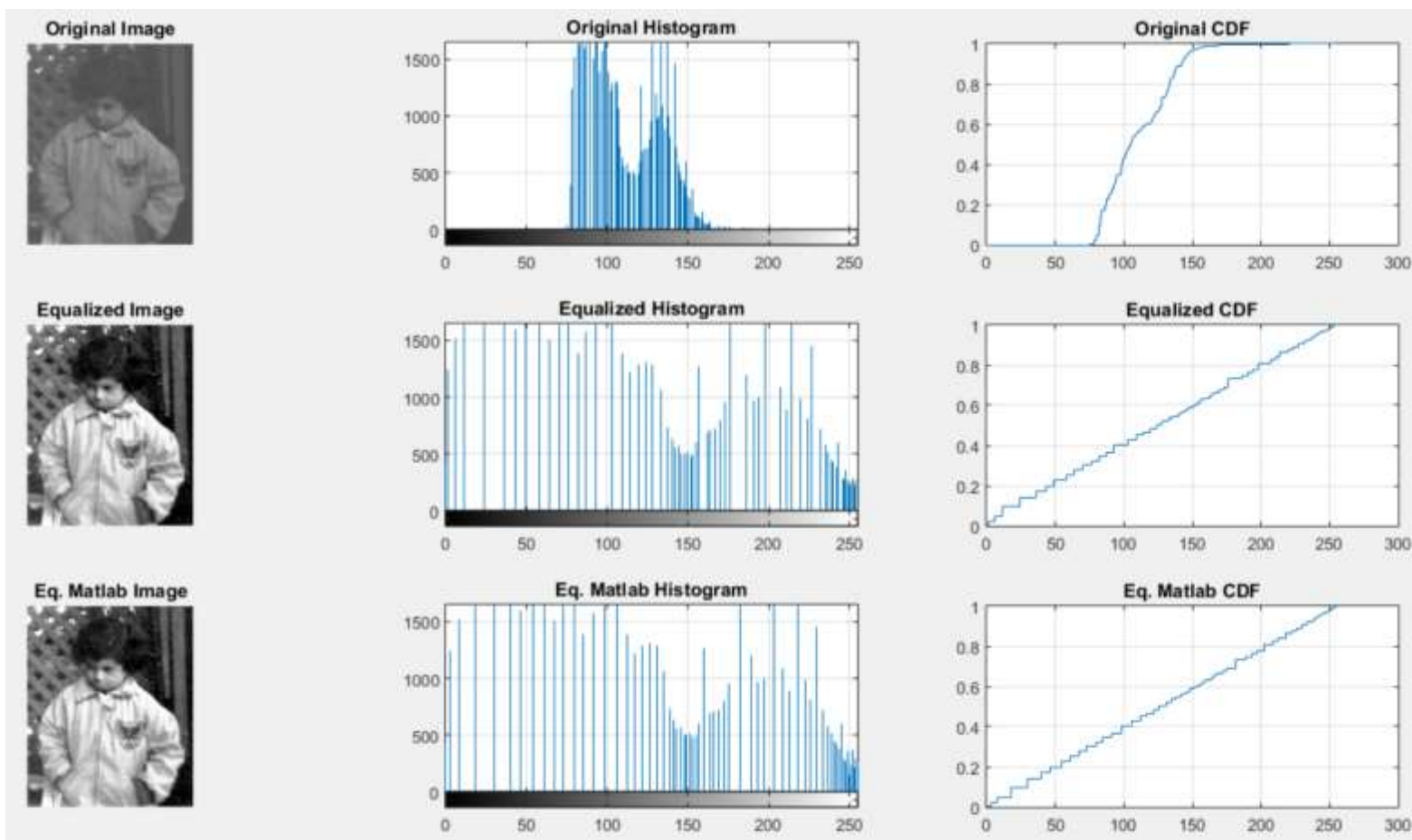
Equalisasi Histogram (Matlab Build in Function)

```
clear; clc;
img = [ 7 7 7 7 7 5 5 5 5 5; 7 7 7 6 6 6 6 5 5 5;
        5 5 5 5 5 4 4 5 5 5; 3 3 3 3 4 4 4 4 5 6;
        4 4 4 2 2 0 0 0 0 1; 5 5 6 7 7 7 7 6 6 7;
        4 4 3 3 4 5 5 5 6 6; 7 7 7 7 7 6 7 7 7 5;
        6 6 6 6 7 7 7 5 5 5; 4 4 4 5 5 5 6 6 6 6];

img = img/7;
imgeq = histeq(img);%Histogram Equalization
subplot(2, 2, 1); imshow(img); title('original image');
subplot(2, 2, 2); imshow(imgeq); title('hist. equal. image');
subplot(2, 2, 3); hist(reshape(img,[1 100]), 8);
title('hist. original image');
xlabel('bin'); ylabel('frequency');
subplot(2, 2, 4); hist(reshape(imgeq,[1 100]), 8);
title('hist.equal. image');
xlabel('bin'); ylabel('frequency');
```

Histogram Equalization

The Code vs Matlab Build in Function

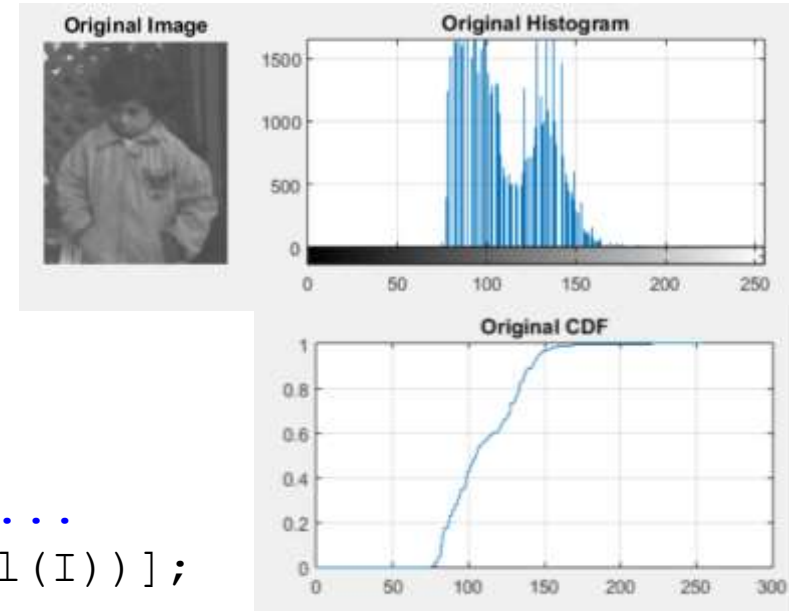


Histogram Equalization

```
clear; clc;
```

```
I = imread('pout.tif');  
[countI, binI] = imhist(I);  
c = [cumsum(countI), ...  
     cumsum(countI)./numel(I), ...  
     255*cumsum(countI)./numel(I), ...  
     floor(255*cumsum(countI)./numel(I))];
```

```
figure,  
subplot(3,3,1), imshow(I), title('Original Image');  
subplot(3,3,2), imhist(I), grid on, title('Original  
Histogram');  
subplot(3,3,3), stairs(0:1:255, c(:,2)), grid on,  
title('Original CDF');
```

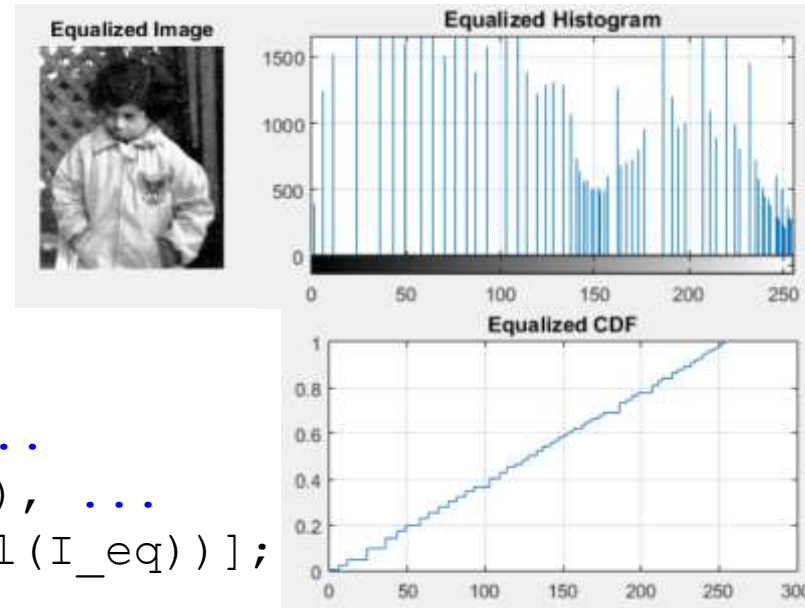


Histogram Equalization

```
map = c(:,4);  
I_eq = map(I+1);  
I_eq = uint8(I_eq);
```

```
[countE, binE] = imhist(I_eq);  
cE = [cumsum(countE), ...  
      cumsum(countE)./numel(I_eq), ...  
      255*cumsum(countE)./numel(I_eq), ...  
      floor(255*cumsum(countE)./numel(I_eq))];
```

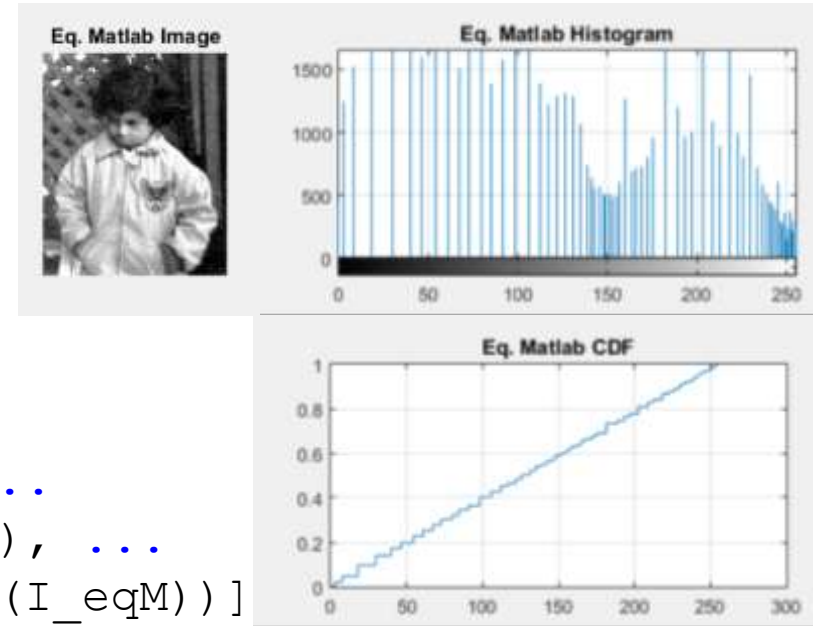
```
subplot(3,3,4), imshow(I_eq), title('Equalized Image');  
subplot(3,3,5), imhist(I_eq), grid on, title('Equalized  
Histogram');  
subplot(3,3,6), stairs(0:1:255, cE(:,2)), grid on,  
title('Equalized CDF');
```



Histogram Equalization

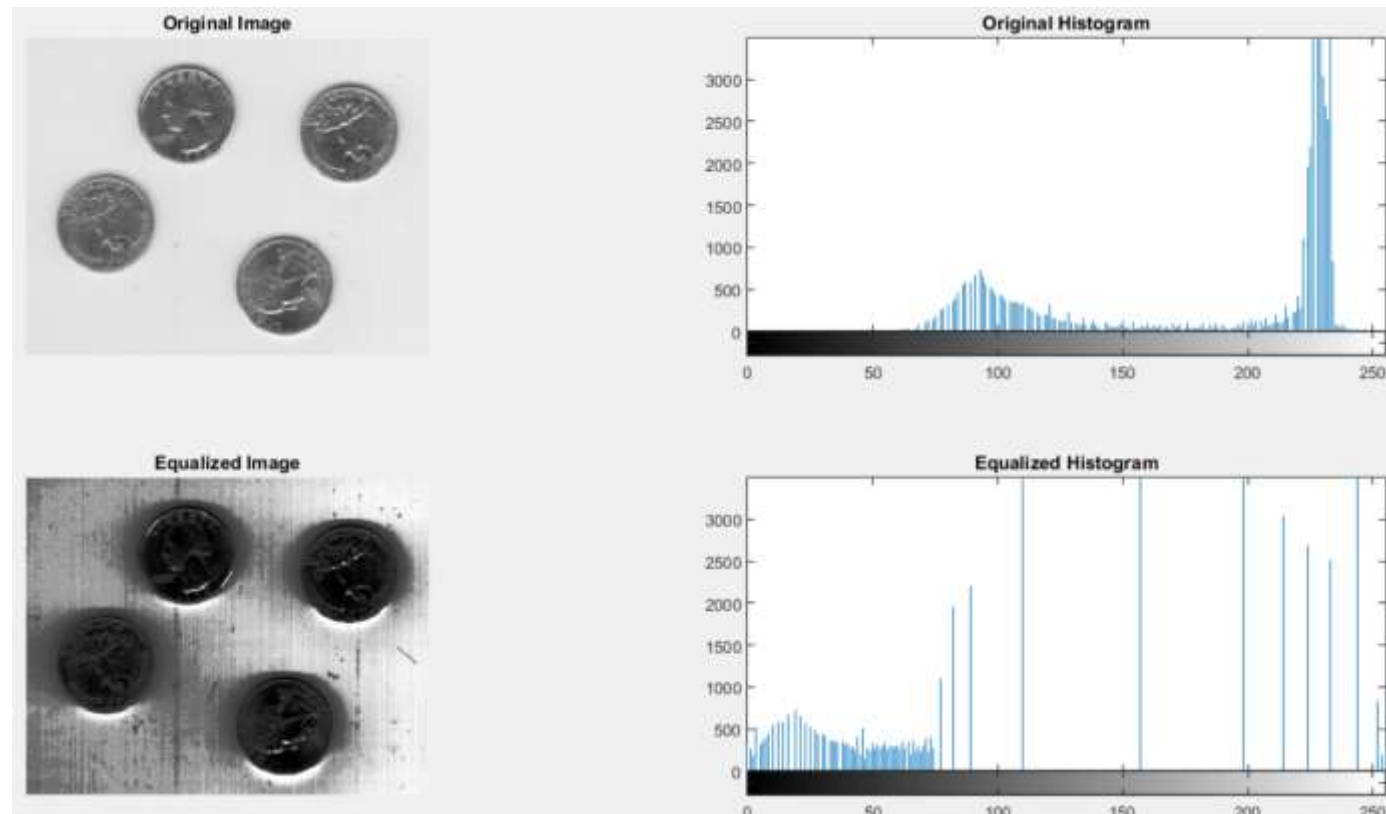
```
[I_eqM, T] = histeq(I, 256);  
[countEM, binEM] = imhist(I_eqM);  
cEM = [cumsum(countEM), ...  
       cumsum(countEM) ./ numel(I_eqM), ...  
       255*cumsum(countEM) ./ numel(I_eqM), ...  
       floor(255*cumsum(countEM) ./ numel(I_eqM))] ]
```

```
subplot(3,3,7), imshow(I_eqM), title('Eq. Matlab Image');  
subplot(3,3,8), imhist(I_eqM), grid on, title('Eq. Matlab  
Histogram');  
subplot(3,3,9), stairs(0:1:255, cEM(:,2)), grid on,  
title('Eq. Matlab CDF');
```



Histogram Equalization

- Histogram equalization does not always perform well.



Blank Space

Histogram (Direct) Specification

Given an image (and its original histogram) and the desired resulting histogram, the direct histogram specification consists of the following:

- Equalizing the original image's histogram using the cdf as a transformation function:

$$s_k = T(r_k) = \sum_{j=0}^k \frac{n_j}{n} = \sum_{j=0}^k p(r_j)$$

- Equalizing the desired probability mass function (in other words, equalizing the desired histogram):

$$v_k = G(z_k) = \sum_{j=0}^k p(z_j)$$

- Applying the inverse transformation function

$$z = G^{-1}(s)$$

TABLE 9.1 Example of a Histogram

Gray Level (r_k)	n_k	$p(r_k)$
0	1120	0.068
1	3214	0.196
2	4850	0.296
3	3425	0.209
4	1995	0.122
5	784	0.048
6	541	0.033
7	455	0.028
Total	16,384	1.000

TABLE 9.3 Desired Histogram

Gray Level (s_k)	n_k	$p(s_k)$
0	0	0.0
1	0	0.0
2	0	0.0
3	1638	0.1
4	3277	0.2
5	6554	0.4
6	3277	0.2
7	1638	0.1
Total	16,384	1.0

TABLE 9.2 Equalized Histogram: Values

Gray Level (s_k)	n_k	$p(s_k)$
0	1120	0.068
1	0	0.000
2	3214	0.196
3	0	0.000
4	4850	0.296
5	3425	0.209
6	1995	0.122
7	1780	0.109
Total	16,384	1.000

TABLE 9.4 Direct Histogram Specification: Summary

k	$p(r_k)$	$s_k (\times 1/7)$	Maps to	v_k	$p(z_k)$
0	0.068	0	z_2	0.00	0.000
1	0.196	2	z_4	0.00	0.000
2	0.296	4	z_5	0.00	0.000
3	0.209	5	z_5	0.10	0.100
4	0.122	6	z_6	0.30	0.200
5	0.048	7	z_7	0.70	0.400
6	0.033	7	z_7	0.90	0.200
7	0.028	7	z_7	1.00	0.100

Histogram Specification

The equalized histogram has already been calculated before and its results are shown in Table 9.2.

The next step consists in obtaining the cdf of the desired probability mass function. Using equation (9.10), we find the following values:

$$v_0 = 0, v_1 = 0, v_2 = 0, v_3 = 0.1, v_4 = 0.3, v_5 = 0.7, v_6 = 0.9, \text{ and } v_7 = 1.$$

The last step—and the most difficult to understand when studying this technique for the first time—is obtaining the inverse function. Since we are dealing with discrete values, the inverse function can be obtained simply by searching, for each value of s_k , the closest value of v_k . For instance, for $s_1 = 2/7 \simeq 0.286$, the closest value of v_k is $v_4 = G(z_4) = 0.3$. In inverse function notation, $G^{-1}(0.3) = z_4$. Therefore, pixels that were shifted to gray level s_1 after the original histogram equalization should be mapped to gray level z_4 . In other words, the 3214 pixels whose original gray level was $1/7$ and that were remapped to gray level $s_1 = 2/7$ during the equalization step should now be shifted again to gray level $z_4 = 4/7$. Similarly, for the remaining values of s_k , the following mappings should be performed:

TABLE 9.2 Equalized Histogram: Values

Gray Level (s_k)	n_k	$p(s_k)$
0	1120	0.068
1	0	0.000
2	3214	0.196
3	0	0.000
4	4850	0.296
5	3425	0.209
6	1995	0.122
7	1780	0.109
Total	16,384	1.000

$$s_0 = 0 \rightarrow z_2$$

$$s_1 = 2/7 \simeq 0.286 \rightarrow z_4$$

$$s_2 = 4/7 \simeq 0.571 \rightarrow z_5$$

$$s_3 = 5/7 \simeq 0.714 \rightarrow z_5$$

$$s_4 = 6/7 \simeq 0.857 \rightarrow z_6$$

$$s_5 = 1 \rightarrow z_7$$

$$s_6 = 1 \rightarrow z_7$$

$$s_7 = 1 \rightarrow z_7$$

TABLE 9.3 Desired Histogram

Gray Level (s_k)	n_k	$p(s_k)$
0	0	0.0
1	0	0.0
2	0	0.0
3	1638	0.1
4	3277	0.2
5	6554	0.4
6	3277	0.2
7	1638	0.1
Total	16,384	1.0

TABLE 9.4 Direct Histogram Specification: Summary

k	$p(r_k)$	$s_k (\times 1/7)$	Maps to	v_k	$p(z_k)$
0	0.068	0	z_2	0.00	0.000
1	0.196	2	z_4	0.00	0.000
2	0.296	4	z_5	0.00	0.000
3	0.209	5	z_5	0.10	0.100
4	0.122	6	z_6	0.30	0.200
5	0.048	7	z_7	0.70	0.400
6	0.033	7	z_7	0.90	0.200
7	0.028	7	z_7	1.00	0.100

In Matlab

- The `histeq` function introduced previously can also be used for histogram matching. In this case, the syntax for `histeq` usually changes to $J = \text{histeq}(I, h)$, where h (a 1D array of integers) represents the specified histogram.

Histogram Spesification

```
clear; clc;

img1 = imread('office_1.jpg'); img1 = rgb2gray(img1);
img2 = imread('office_3.jpg'); img2 = rgb2gray(img2);

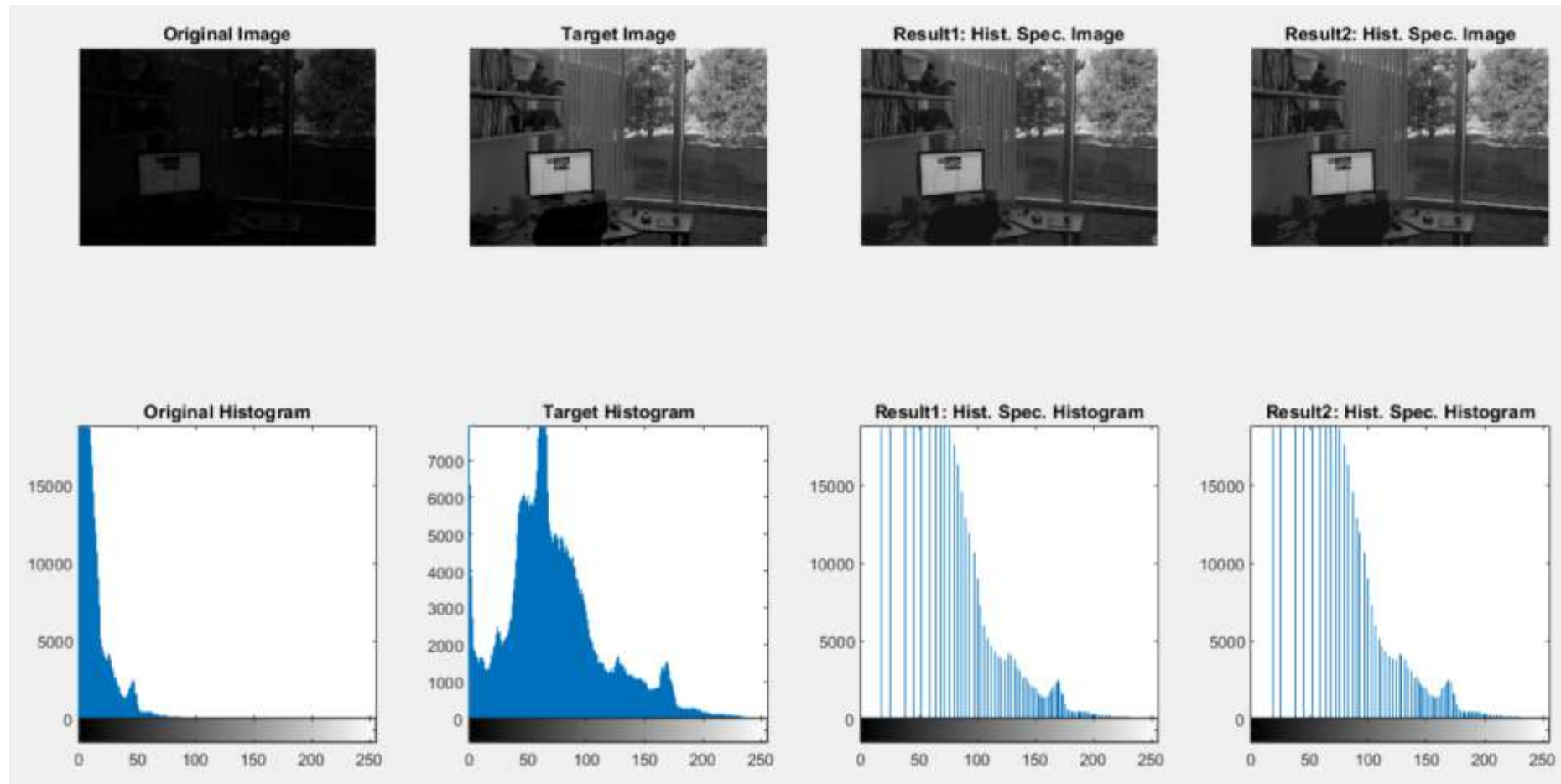
eqsImg1    = histeq(img1, 256);
normHist2  = imhist(img2)./numel(img2);

%histogram spesification
eqsImg1_1 = histeq(img1, normHist2);
eqsImg1_2 = imhistmatch(img1,img2, 256);

subplot(2,4,1); imshow(img1); title('Original Image');
subplot(2,4,2); imshow(img2); title('Target Image');
subplot(2,4,3); imshow(eqsImg1_1);title('Result1: Hist. Spec. Image');
subplot(2,4,4); imshow(eqsImg1_2);title('Result2: Hist. Spec. Image');

subplot(2,4,5); imhist(img1); title('Original Histogram');
subplot(2,4,6); imhist(img2); title('Target Histogram');
subplot(2,4,7); imhist(eqsImg1_1);title('Result1: Hist. Spec. Histogram');
subplot(2,4,8); imhist(eqsImg1_2);title('Result2: Hist. Spec. Histogram');
```

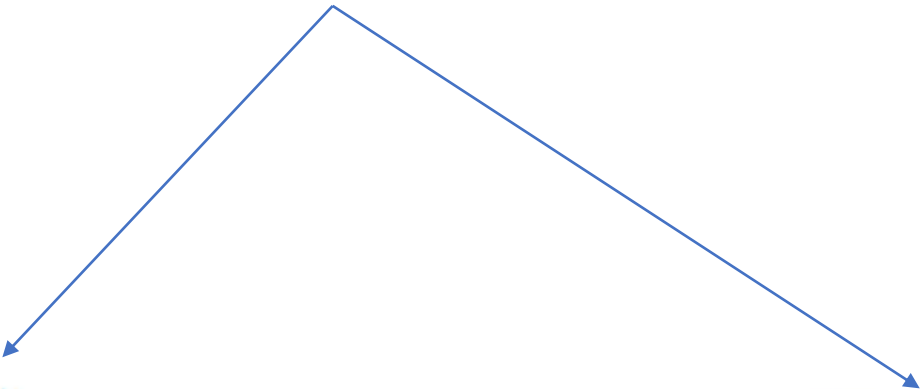
Histogram Spesification



Blank Space

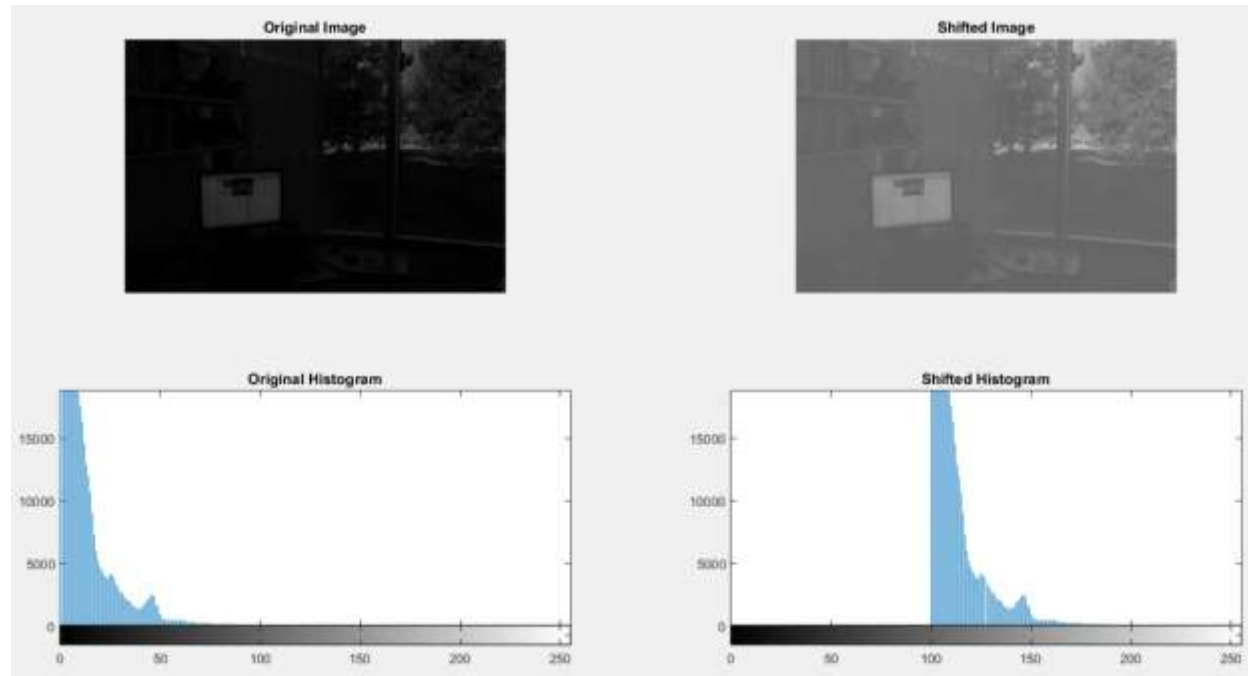
Histogram Modification

$$I_N = (I - \text{Min}) \frac{\text{newMax} - \text{newMin}}{\text{Max} - \text{Min}} + \text{newMin}$$


$$s = \left[\frac{s_{\max} - s_{\min}}{r_{\max} - r_{\min}} \right] (r - r_{\min}) + s_{\min} \quad s = \frac{r - r_{\min}}{r_{\max} - r_{\min}} \times (L - 1)$$

Pergeseran Kontras (Hist. Sliding)

```
clear; clc;  
img = imread('office_1.jpg'); img = rgb2gray(img);  
imgn = img + 100;  
subplot(2,2,1); imshow(img); title('Original Image');  
subplot(2,2,2); imshow(imgn); title('Shifted Image');  
subplot(2,2,3); imhist(img); title('Original Histogram');  
subplot(2,2,4); imhist(imgn); title('Shifted Histogram');
```



Pergeseran Kontras (Hist. Sliding)

In MATLAB

The `imadd` and `imsubtract` functions introduced earlier in the book can be used for histogram sliding.

```
J = imread('pout.tif');
I = im2double(J);
clear J
figure, subplot(3,2,1), imshow(I), title('Original Image')
subplot(3,2,2), imhist(I), axis tight, ...
    title('Original Histogram')

const = 0.1;
I2 = I + const;
subplot(3,2,3), imshow(I2), title('Original Image + 0.1')
subplot(3,2,4), imhist(I2), axis tight, ...
    title('Original Hist + 0.1')
```


Pergeseran Kontras (Hist. Sliding)

```
const = 0.5;
I3 = I + const;
bad_values = find(I3 > 1);
I3(bad_values) = 1;
subplot(3,2,5), imshow(I3), title('Original Image + 0.5')
subplot(3,2,6), imhist(I3), axis tight, ...
    title('Original Hist + 0.5')
```

In Matlab

- MATLAB's IPT has a built-in function to perform histogram stretching and shrinking (among other operations): `imadjust`.

Histogram Shrinking

- This technique—also known as *output cropping*—modifies the original histogram in such a way as to compress its dynamic grayscale range, $[r_{\min}, r_{\max}]$, into a narrower gray scale, between s_{\min} and s_{\max} . As a consequence, the resulting image contrast is reduced. Mathematically,

$$s = \left[\frac{s_{\max} - s_{\min}}{r_{\max} - r_{\min}} \right] (r - r_{\min}) + s_{\min}$$

Histogram Shrinking

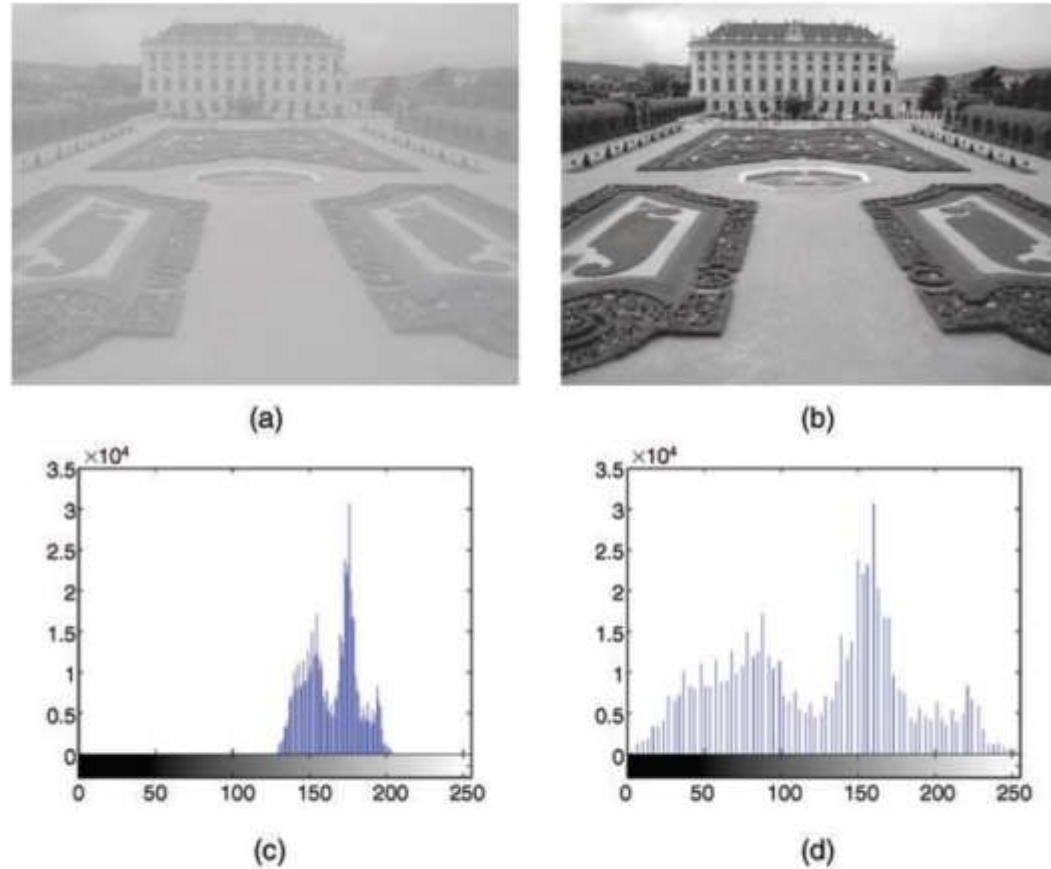


FIGURE 9.10 Example of using histogram stretching to improve contrast: (a) original image ($r_{\min} = 129$, $r_{\max} = 204$); (b) result of stretching using equation (9.12); (c and d) histograms corresponding to images in (a) and (b).

Histogram Stretching

- This technique—also known as *input cropping*—consists of a linear transformation that expands (stretches) part of the original histogram so that its nonzero intensity range $[r_{\min}, r_{\max}]$ occupies the full dynamic gray scale, $[0, L - 1]$.
- Mathematically, each input intensity value, r , is mapped to an output value, s , according to the following linear mapping function:

$$s = \frac{r - r_{\min}}{r_{\max} - r_{\min}} \times (L - 1)$$

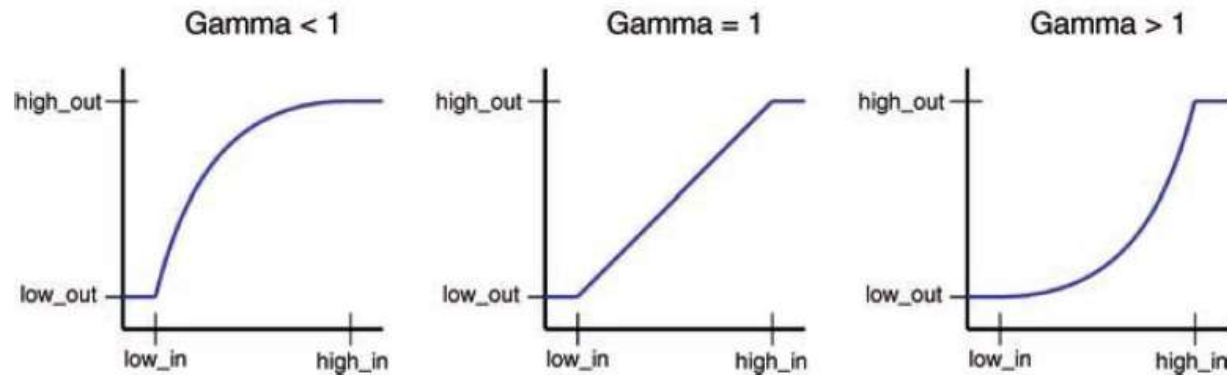
Histogram Streching

```
I = imread('westconcordorthophoto.png');
I_shrink = imadjust(I,stretchlim(I),[0.25 0.75]);

figure
subplot(2,2,1), imshow(I), title('Original Image')
subplot(2,2,2), imhist(I), axis tight, ...
    title('Original Histogram')
subplot(2,2,3), imshow(I_shrink), ...
    title('Shrunk Image')
subplot(2,2,4), imhist(I_shrink), axis tight, ...
    title('Shrunk Histogram')

X = reshape(I,1,prod(size(I)));
Y = reshape(I_shrink,1,prod(size(I_shrink)));
figure, plot(X,Y, '.')
xlim([0 255]); ylim([0 255]);
xlabel('Original Image');
ylabel('Adjusted Image');
```

Histogram Modification



```
I_shrink = imadjust(I,stretchlim(I),[0.25 0.75],2);  
X = reshape(I,1,prod(size(I)));  
Y = reshape(I_shrink,1,prod(size(I_shrink)));  
figure  
subplot(2,2,1), imshow(I), title('Original Image')  
subplot(2,2,2), imhist(I), axis tight, ...  
    title('Original Histogram')  
subplot(2,2,3), imshow(I_shrink), title('Adjusted Image')  
subplot(2,2,4), imhist(I_shrink), axis tight, ...  
    title('Adjusted Histogram')  
figure, plot(X,Y,'.'), xlim([0 255]), ylim([0 255])
```

Histogram Stretching

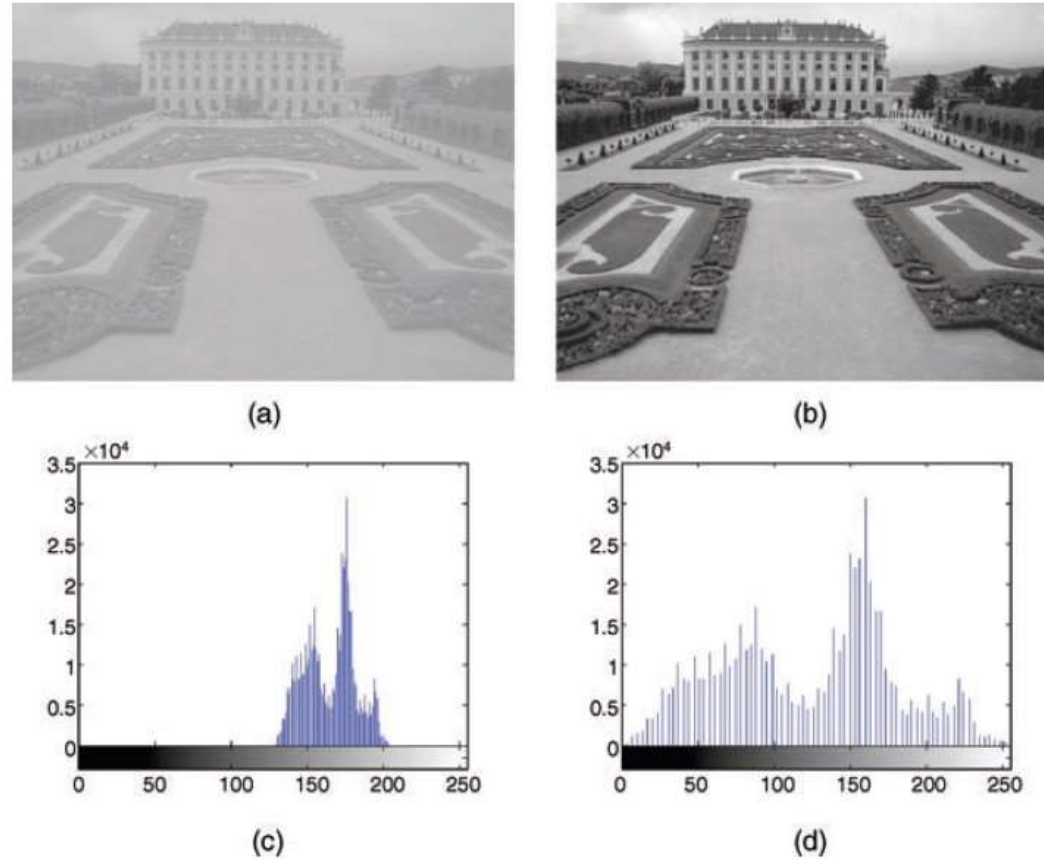


FIGURE 9.10 Example of using histogram stretching to improve contrast: (a) original image ($r_{\min} = 129$, $r_{\max} = 204$); (b) result of stretching using equation (9.12); (c and d) histograms corresponding to images in (a) and (b).

Histogram Shrinking

```
img_limits = stretchlim(I);  
I_stretch = imadjust(I,img_limits,[]);  
figure  
subplot(3,2,1), imshow(I), title('Original Image')  
subplot(3,2,2), imhist(I), axis tight, ...  
    title('Original Histogram')  
subplot(3,2,3), imshow(I_stretch), ...  
    title('Stretched Image')  
subplot(3,2,4), imhist(I_stretch), axis tight, ...  
    title('Stretched Histogram')
```

Histogram Shrinking

```
I_stretch2 = imadjust(I);  
subplot(3,2,5), imshow(I_stretch2), ...  
    title('Stretched Image')  
subplot(3,2,6), imhist(I_stretch2), axis tight, ...  
    title('Stretched Histogram')  
I_stretch_diff = imabsdiff(I_stretch, I_stretch2);  
figure, imshow(I_stretch_diff, [])  
min(I_stretch_diff(:))  
max(I_stretch_diff(:))
```

Blank Space

Lampiran

```
Command Window

>> [counts,center] = hist(reshape(imgeq,[1 100]), 8);
>> [counts;center]

ans =

    13.0000    15.0000         0    29.0000         0    19.0000         0    24.0000
    0.0923    0.2133    0.3343    0.4554    0.5764    0.6974    0.8185    0.9395

>> center(1)-center(2)

ans =

   -0.1210

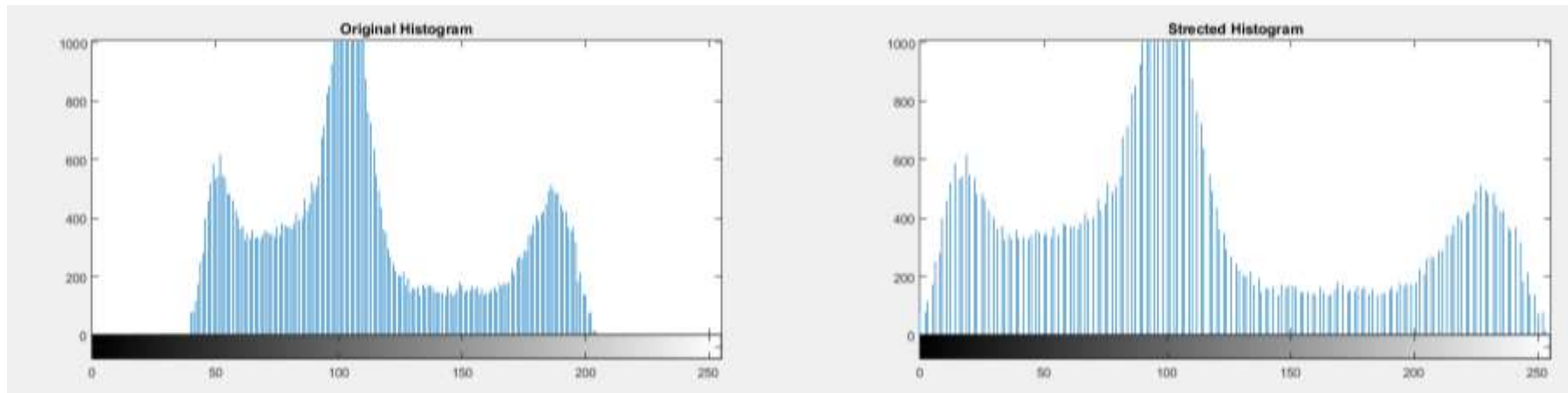
>> (max(max(imgeq)) - min(min(imgeq)))/8

ans =

    0.1210
```

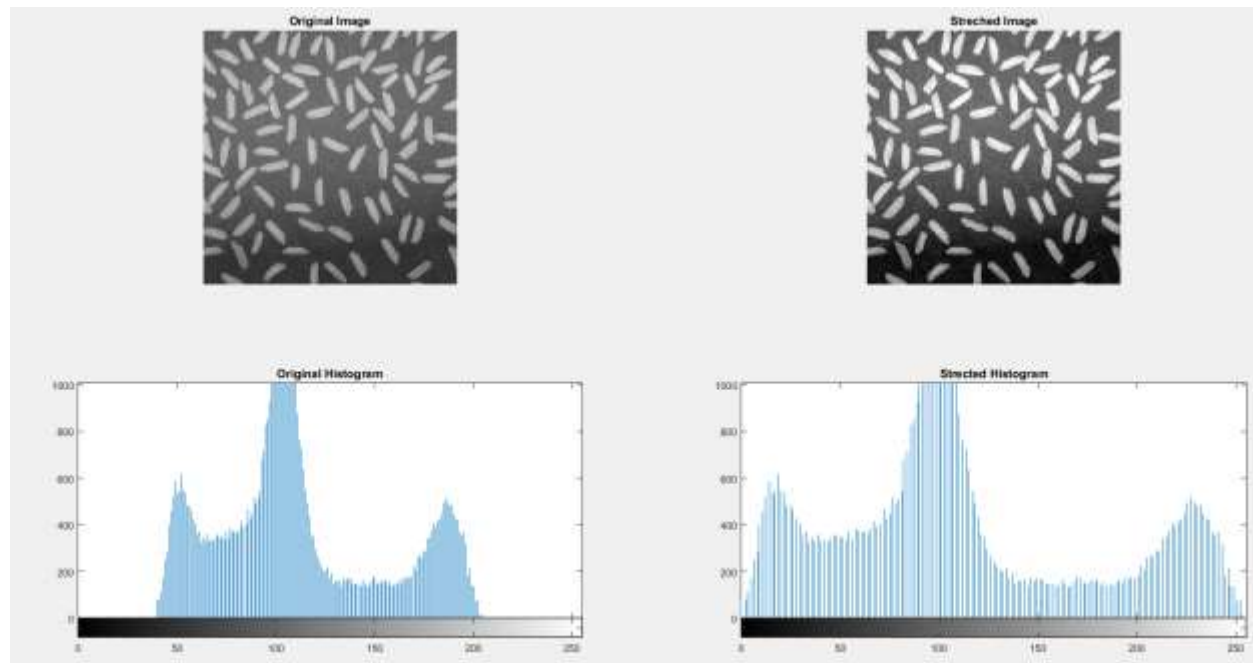
Peregangan Kontras

```
function imgn = contrastStrecthing(img, newMax, newMin)
    minImg = min(min(img));
    maxImg = max(max(img));
    delta = double((newMax - newMin))/double((maxImg - minImg));
    imgn = (img - minImg)*delta+newMin;
    imgn = uint8(imgn);
end
```



Peregangan Contrast

```
clear; clc;  
img = imread('rice.png');  
imgn = contrastStrechting(img, 255, 0);  
subplot(2,2,1); imshow(img); title('Original Image');  
subplot(2,2,2); imshow(imgn); title('Stretched Image');  
subplot(2,2,3); imhist(img); title('Original Histogram');  
subplot(2,2,4); imhist(imgn); title('Stretched Histogram');
```



Power Law (Gamma) Transformations

The power law transformation function is described by

$$s = c \cdot r^\gamma \quad (8.5)$$

where r is the original pixel value, s is the resulting pixel value, c is a scaling constant, and γ is a positive value. Figure 8.8 shows a plot of equation (8.5) for several values of γ .

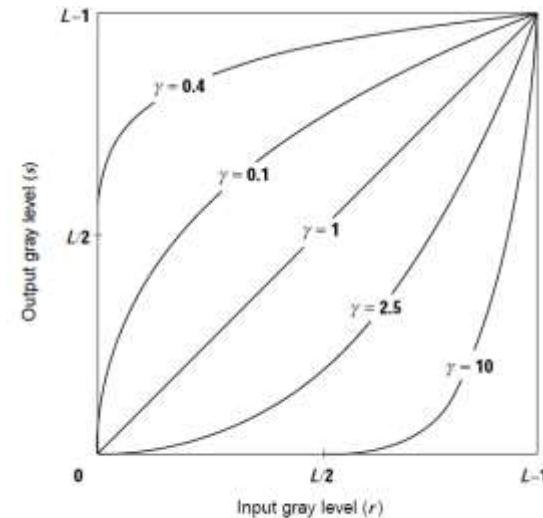


FIGURE 8.8 Examples of power law transformations for different values of γ .

Power Law (Gamma) Transformations

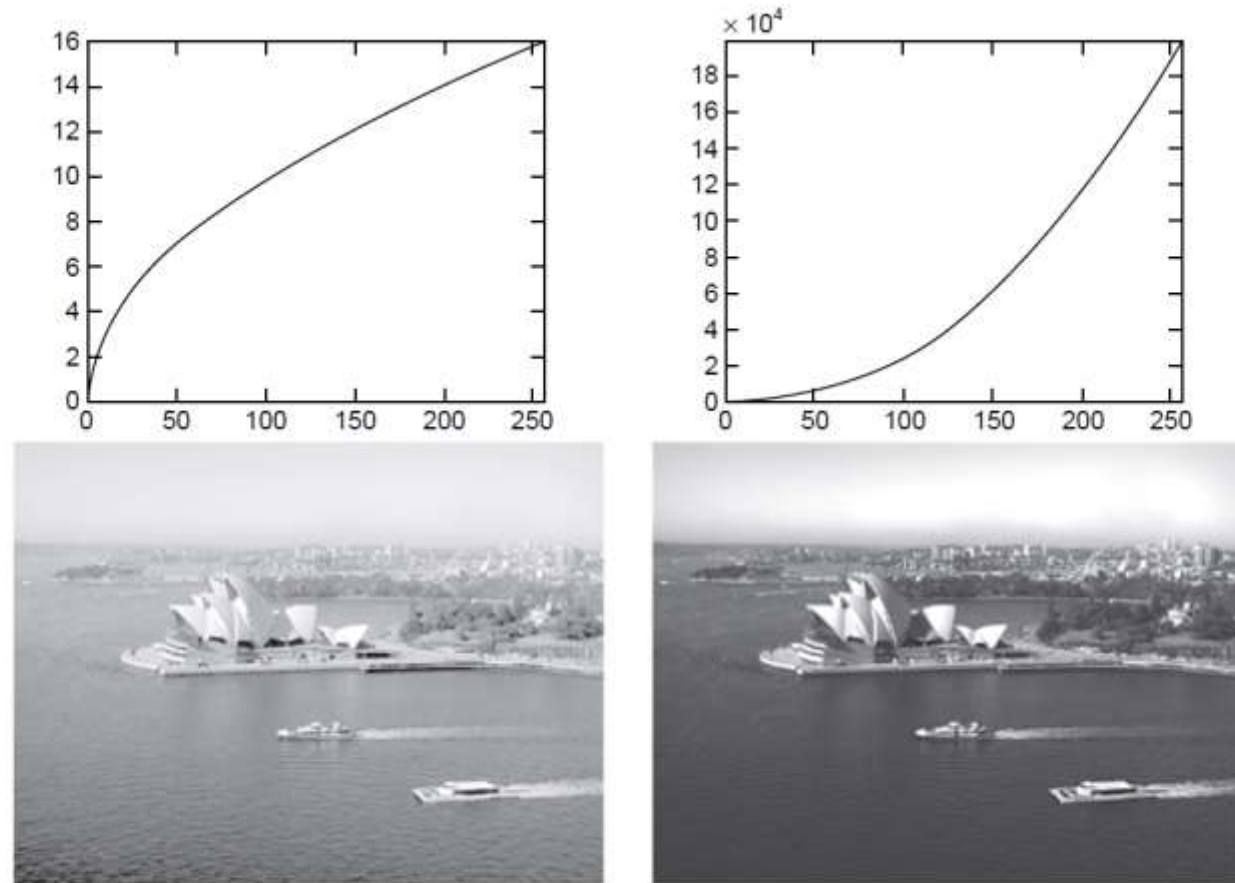


FIGURE 8.9 Examples of gamma correction for two different values of γ : 0.5 (left) and 2.2 (right).