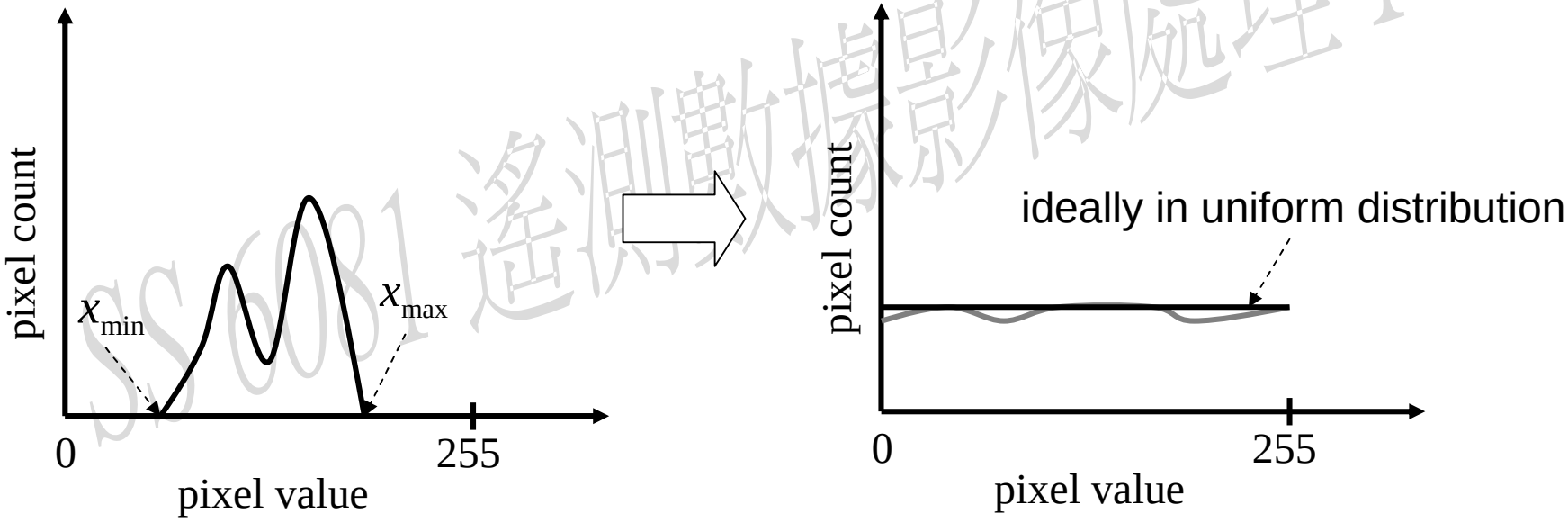
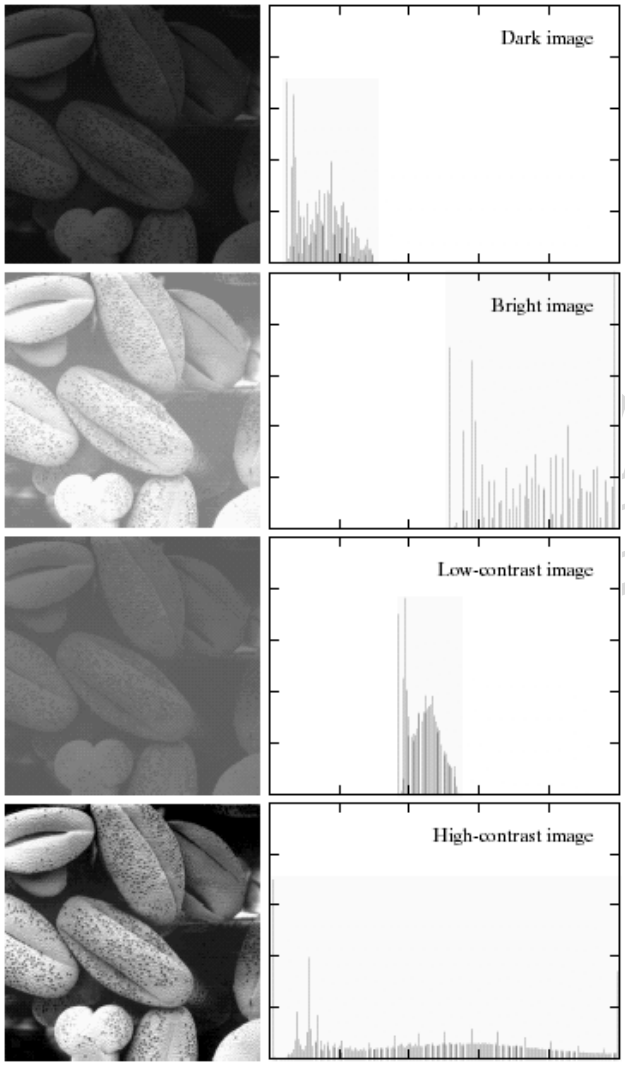
Histogram Equalization



Original histogram

Enhanced histogram

Histogram Equalization



The histogram of a digital image with gray levels in the range [0, L-1] is a discrete function

$$h(r_k) = n_k$$

where r_k is the kth gray level and n_k is the number of pixels in the image having gray level r_k. A normalized histogram is given by

$$P_r(r_k) = \frac{n_k}{n}, \text{ and } \sum_k P_r(r_k) = 1$$

a b

FIGURE 3.15 Four basic image types: dark, light, low contrast, high contrast, and their corresponding histograms. (Original image courtesy of Dr. Roger Heady, Research School of Biological Sciences, Australian National University, Canberra, Australia.)

Histogram Equalization

By the transformation: $s = T(r)$

Conditons:
(1)*T(r)* is single-valued and monotonically increasing
(2) $0 \le r \le 1$ for $0 \le T(r) \le 1$

The inverse transformation from s back to r is denoted

$$r = T^{-1}(s), 0 \le s \le 1$$

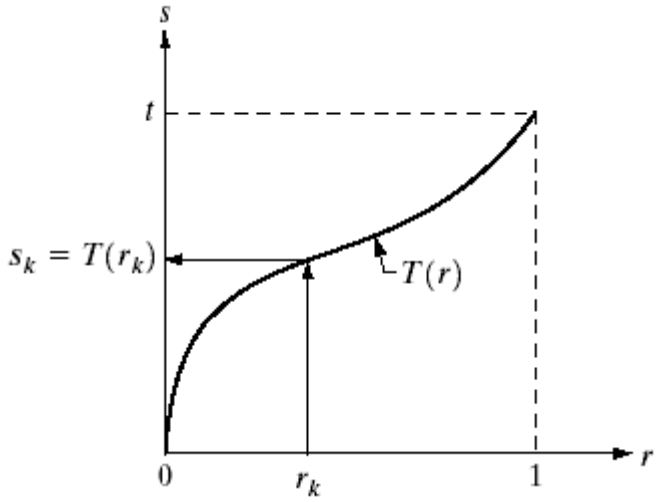


FIGURE 3.16 A gray-level transformation function that is both single valued and monotonically increasing.

Histogram Equalization

The gray levels in an image may be viewed as random variables in the interval [0, 1]. Let Pr(r) and Ps(s) denote the probability density functions of random variables r and s, respectively.

The probability density function Ps(s) of the transformed variable s can be obtained using a rather simple formula:

$$p_s(s) = p_r(r)\left|\frac{dr}{ds}\right|$$

A transformation function of particular importance in image processing has the form

$$s = T(r) = \int_0^r p_r(w)\,dw$$

Histogram Equalization

$$\frac{ds}{dr} = \frac{dT(r)}{dr}$$
$$= \frac{d}{dr}\left[\int_0^r p_r(w)\,dw\right]$$
$$= p_r(r)$$

$$p_s(s) = p_r(r)\left|\frac{dr}{ds}\right|$$

$$p_s(s) = p_r(r)\left|\frac{dr}{ds}\right|$$
$$= p_r(r)\left|\frac{1}{p_r(r)}\right|$$
$$= 1 \qquad 0 \le s \le 1$$

the form of Ps(s) given above as a *uniform* probability density function.

Discrete type

$$p_r(r_k) = \frac{n_k}{n} \qquad k = 0, 1, 2, \ldots, L-1$$

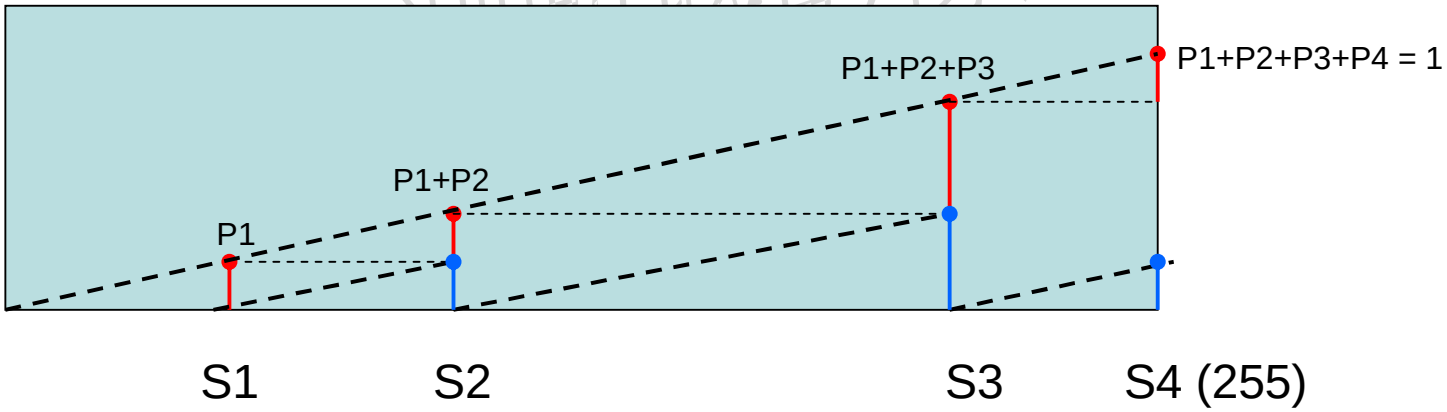
$$s_k = T(r_k) = \sum_{j=0}^{k} p_r(r_j)$$

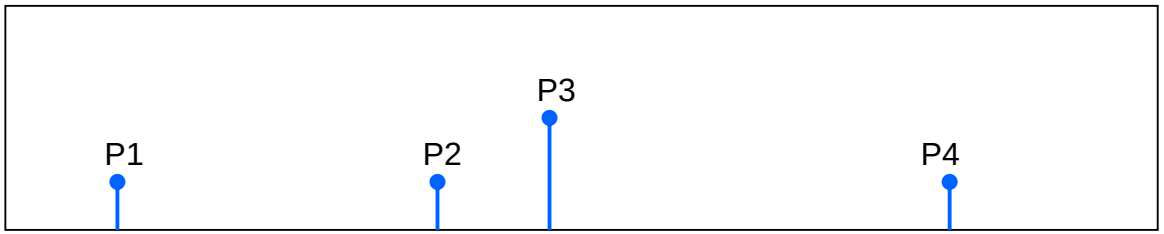
$$= \sum_{j=0}^{k} \frac{n_j}{n} \qquad k = 0, 1, 2, \ldots, L-1$$

Where n is the total number of pixels in the image, n_k is the number of pixels that have gray level r_k, and L is the total number of possible gray levels in the image.

$$slope = \frac{p_1}{S_1 - 0} = \frac{p_1 + p_2}{S_2 - 0} = \cdots = \frac{p_1 + p_2 + p_3 + p_4}{S_4 - 0} = \frac{1}{255}$$

$$slope = \frac{p_1}{S_1 - 0} = \frac{p_2}{S_2 - S_1} = \cdots = \frac{p_4}{S_4 - S_3} = \frac{1}{255}$$





same result

Example:

- Consider an 8-level 64 x 64 image with gray values (0, 1, …, 7). The normalized gray values are (0, 1/7, 2/7, …, 1). The normalized histogram :

$$s_k = T(r_k) = \sum_{j=0}^{k} p_r(r_j)$$

$$0 \le k \le L-1$$

k	r_k	n_j	$P(r_k) = n_k / n$
0	0	30	0.0036
1	1/7	50	0.0061
2	2/7	100	0.0122
3	3/7	1500	0.1829
4	4/7	2300	0.2804
5	5/7	4000	0.4878
6	6/7	200	0.0243
7	1	20	0.0024

- Applying the transformation, $s_k = T(r_k) = \sum_{j=0}^{k} p_r(r_j)$ we have

$$s_0 = \sum_{j=0}^{0} \frac{n_j}{n} = \frac{n_0}{n} = \frac{30}{8200} = 0.0037 \approx \frac{0}{7}$$

$$s_1 = \sum_{j=0}^{1} \frac{n_j}{n} = \frac{n_0 + n_1}{n} = \frac{30+50}{8200} = \frac{80}{8200} = 0.0098 \approx \frac{0}{7}$$

$$s_2 = \sum_{j=0}^{2} \frac{n_j}{n} = \frac{n_0 + n_1 + n_2}{n} = \frac{30+50+100}{8200} = \frac{180}{8200} = 0.022 \approx \frac{0}{7}$$

$$s_3 = \sum_{j=0}^{3} \frac{n_j}{n} = \frac{30+50+100+1500}{8200} = \frac{1680}{8200} = 0.205 \approx \frac{1}{7}$$

$$s_4 = \sum_{j=0}^{4} \frac{n_j}{n} = \frac{30+50+100+1500+2300}{8200} = \frac{3980}{8200} = 0.485 \approx \frac{3}{7}$$

$$s_5 = \sum_{j=0}^{5} \frac{n_j}{n} = \frac{30+50+100+1500+2300+4000}{8200} = \frac{7980}{8200} = 0.973 \approx \frac{7}{7}$$

$$s_6 = \sum_{j=0}^{6} \frac{n_j}{n} = \frac{30+50+100+1500+2300+4000+200}{8200} = \frac{8180}{8200} = 0.998 \approx \frac{7}{7}$$

$$s_7 = \sum_{j=0}^{7} \frac{n_j}{n} = \frac{30+50+100+1500+2300+4000+200+20}{8200} = \frac{8200}{8200} = 1 \approx \frac{7}{7}$$

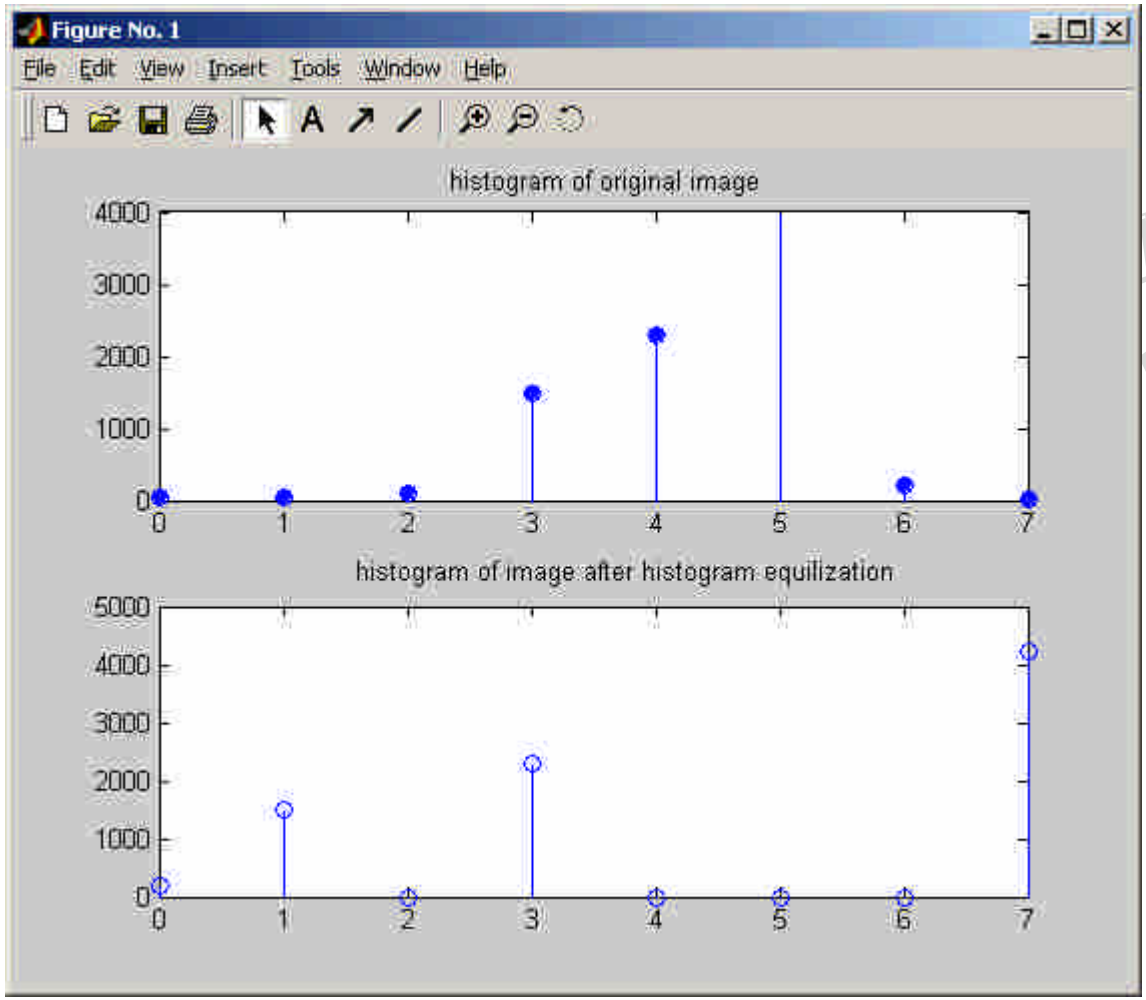
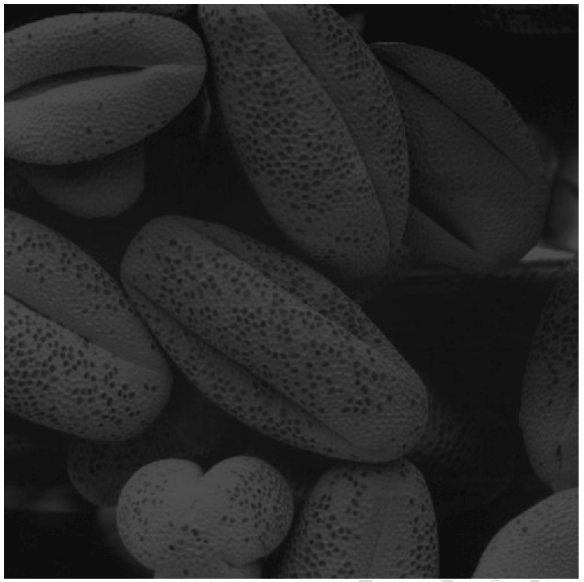
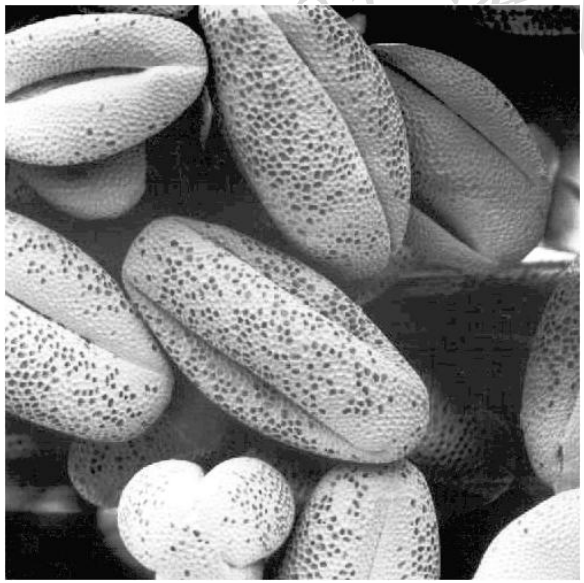


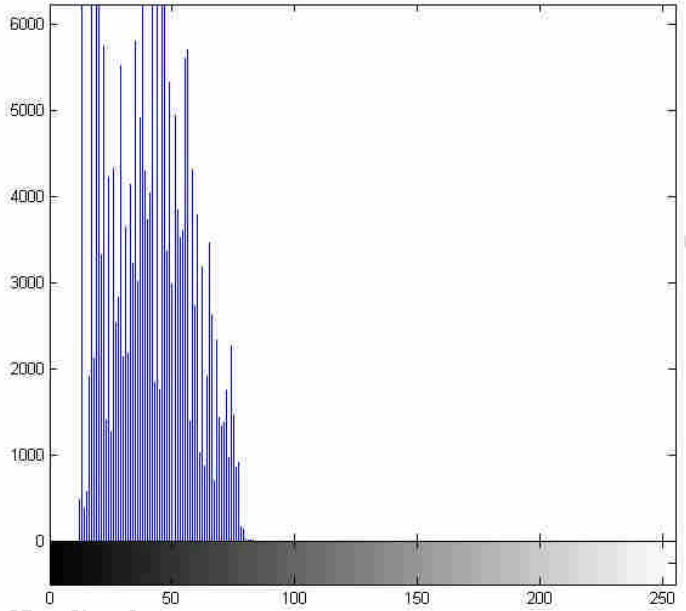
Figure No. 1
File Edit View Insert Tools Window Help
histogram of original image
4000
3000
2000
1000
0
0 1 2 3 4 5 6 7
histogram of image after histogram equilization
5000
4000
3000
2000
1000
0
0 1 2 3 4 5 6 7

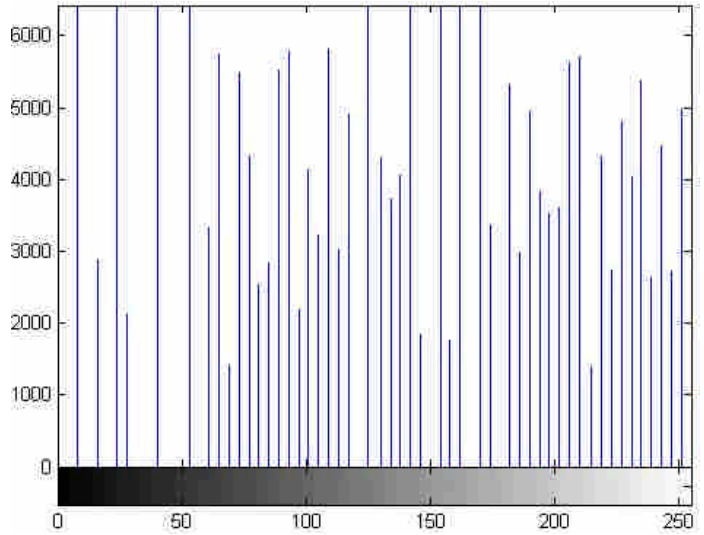
k	n_j	$\sum_{j=0}^{k} n_j$	$\sum_{j=0}^{k} \frac{n_j}{n}$	$s_k = (L-1)\sum_{j=0}^{k} \frac{n_j}{n}$
0	30	30	0.0037	0.0256
1	50	80	0.0098	0.0683
2	100	180	0.0220	0.1537
3	1500	1680	0.2050	1.4241
4	2300	3980	0.4854	3.3976
5	4000	7980	0.9732	6.8122
6	200	8180	0.9976	6.9830
7	20	8200	1.0000	7.0000

Example



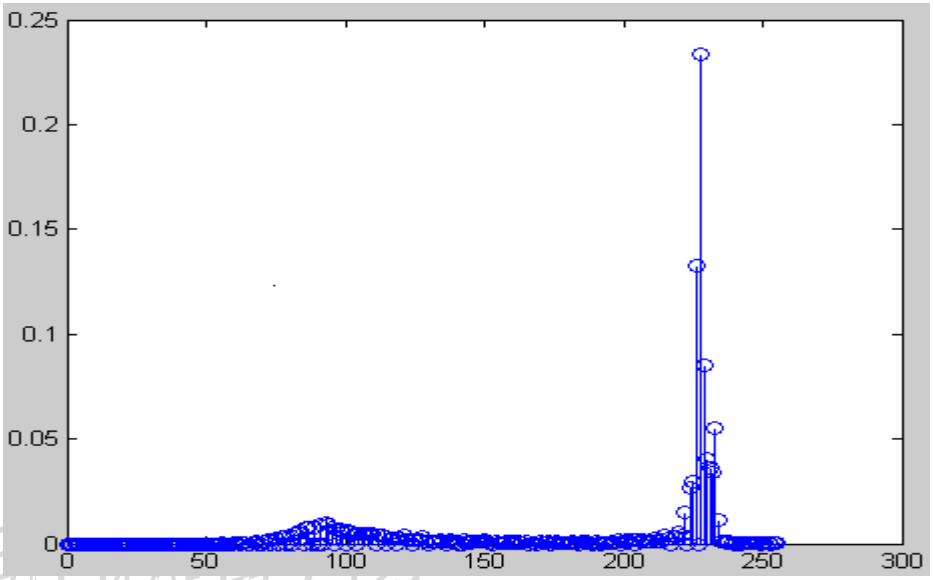


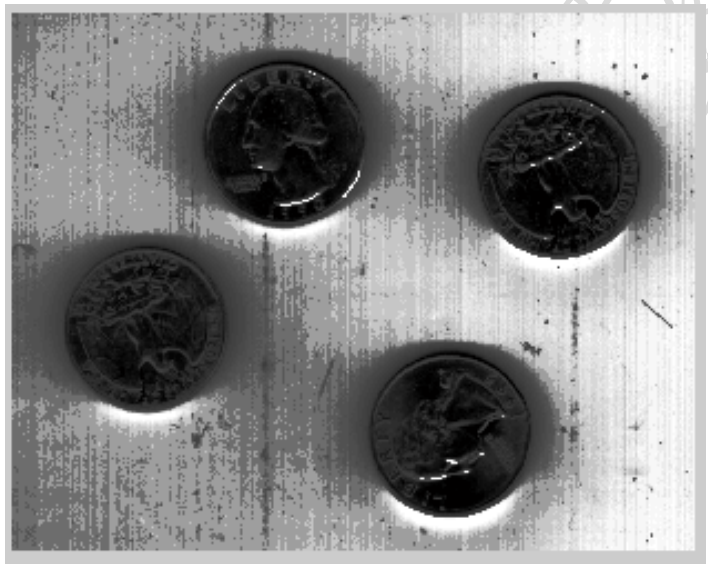

6000
5000
4000
3000
2000
1000
0
0
50
100
150
200
250

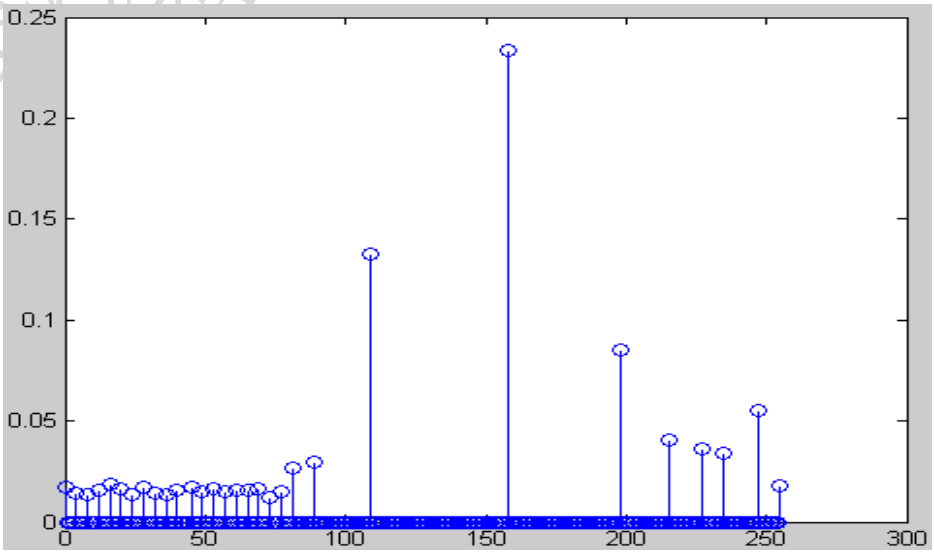

6000
5000
4000
3000
2000
1000
0
0
50
100
150
200
250

Histogram Equalization

- Histogram equalization may not always produce desirable results, particularly if the given histogram is very narrow. It can produce false edges and regions. It can also increase image “graininess” and “patchiness.”


0.25
0.2
0.15
0.1
0.05
0
0
50
100
150
200
250
300




0.25
0.2
0.15
0.1
0.05
0
0
50
100
150
200
250
300

Histogram specification

$$r \xrightarrow{T} s \qquad \text{(equalized)}$$

$$s = T(r) = \int_0^r p_r(w)dw$$

$$v \xleftarrow{G} z \qquad \text{(desired level)}$$

$$v = G(z) = \int_0^Z p_Z(w)dw$$

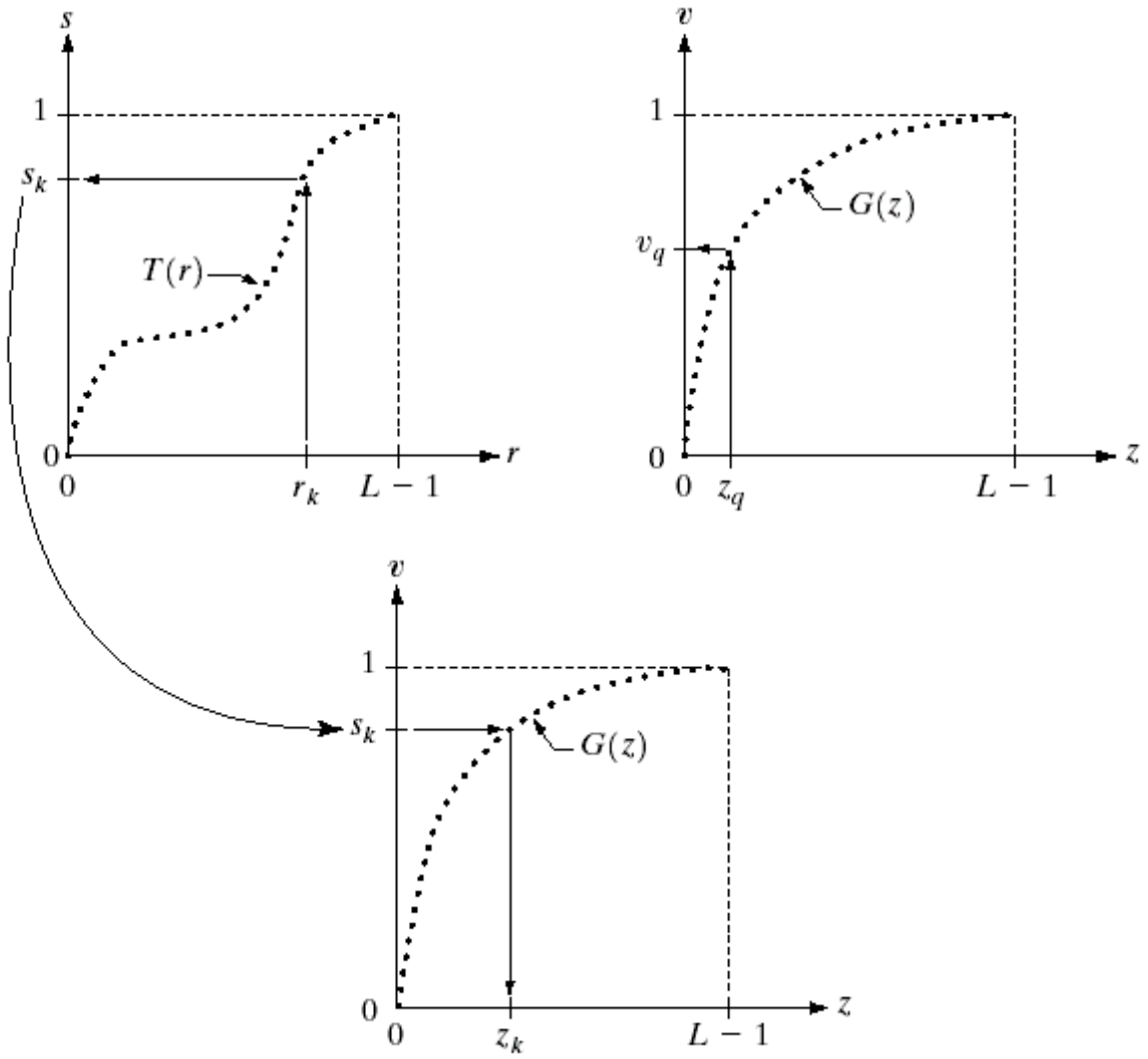
$$z = G^{-1}[s] = G^{-1}[v] = G^{-1}[T(r)]$$

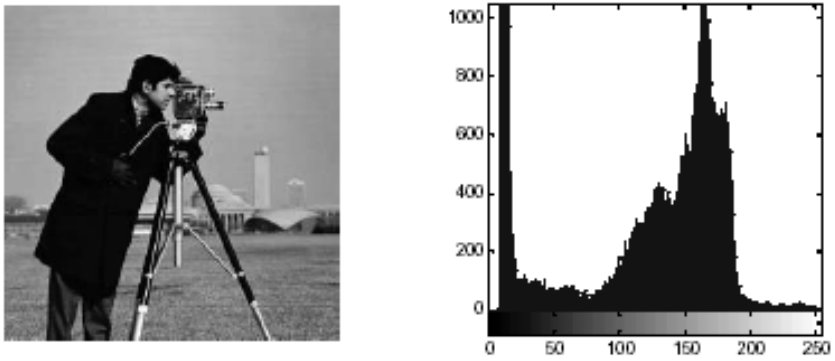
$p_r(r)$: original *p.d.f*

$p_z(z)$: desired *p.d.f*

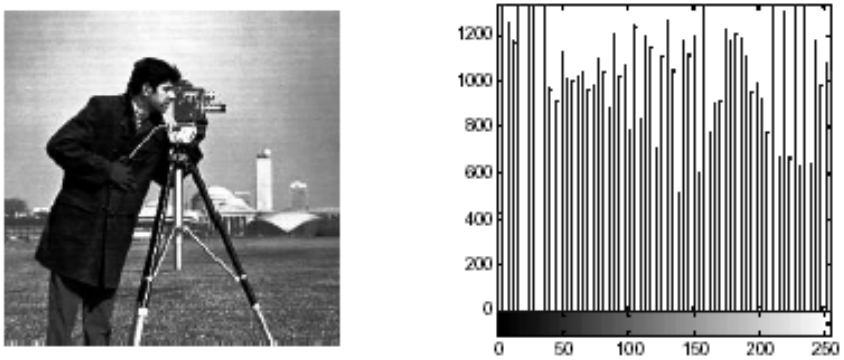
a b
c

FIGURE 3.19
(a) Graphical interpretation of mapping from r_k to s_k via $T(r)$. (b) Mapping of z_q to its corresponding value v_q via $G(z)$. (c) Inverse mapping from s_k to its corresponding value of z_k.

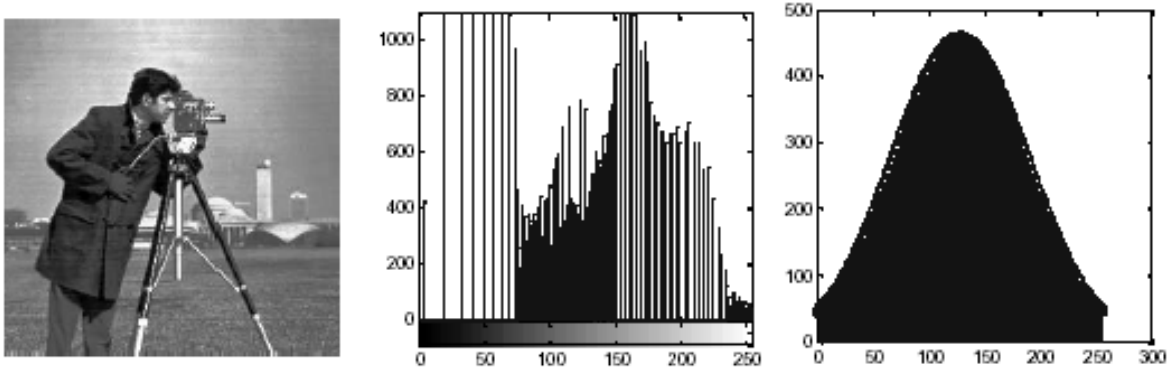




Original image and its histogram



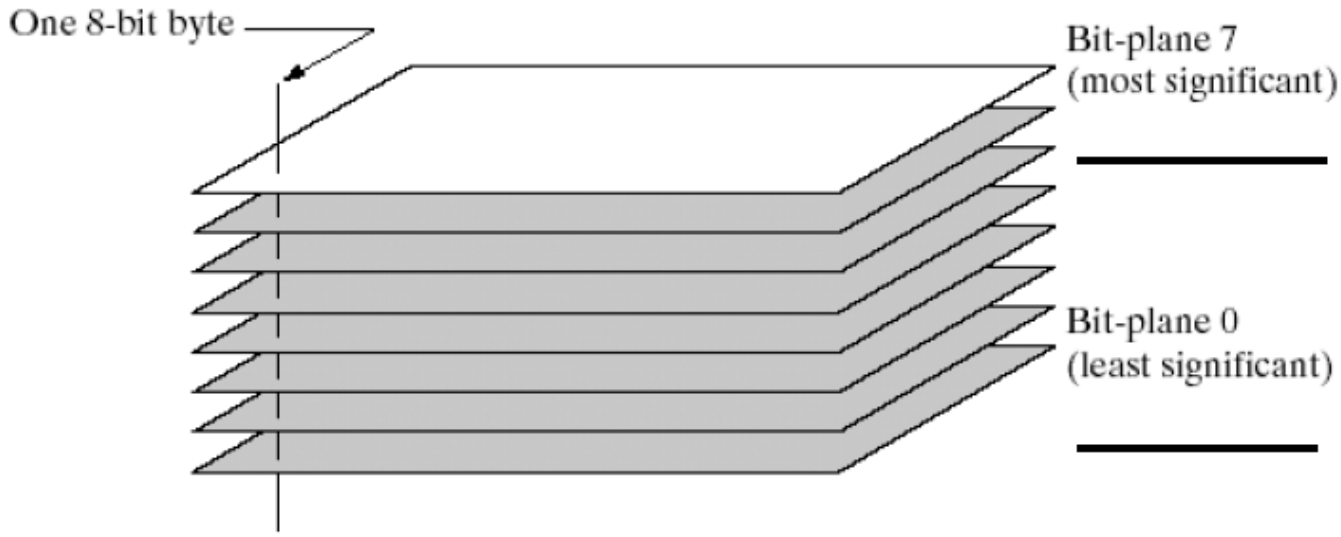
Histogram equalized image, Actual histogram of output



Histogram specified image, Actual Histogram, and Specified Histogram

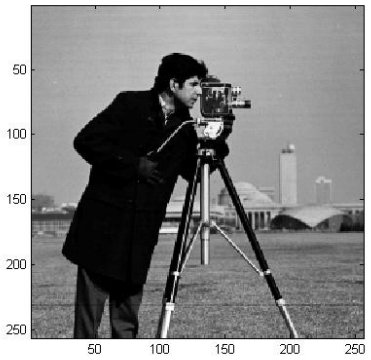
Bit-plane slicing

- Bit-plane slicing :highlighting the contributions of specific bits

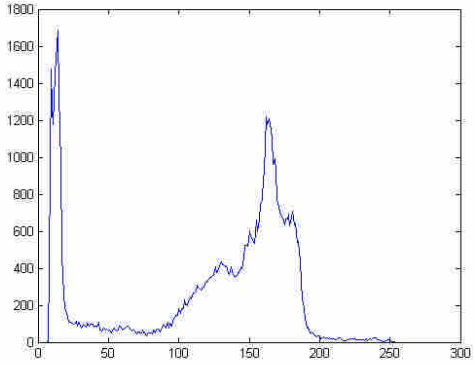


Example

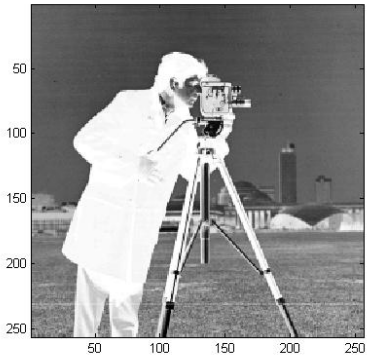
```
A=imread('cameraman.tif');
A=double(A);
for i=1:8
B(:,:,i)=uint8(A>=2^(8-i));
A=A-2^(8-i).*double(B(:,:,i));
end
figure
for i=1:8
subplot(3,3,i)
imagesc(B(:,:,i))
end
colormap(gray)
subplot(3,3,9)
A=imread('cameraman.tif');
A=double(A);
imagesc(A)
```



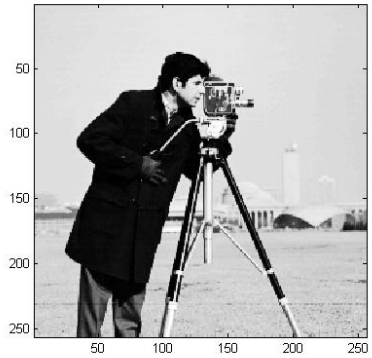
A (Original image)



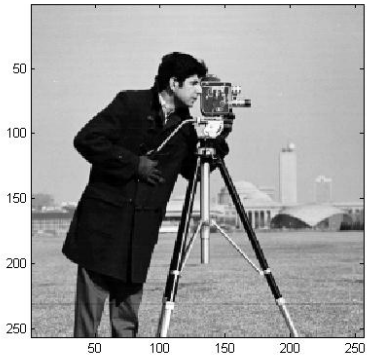
plot(x,y) A



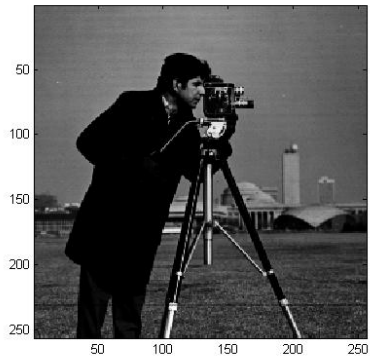
B (negative)



C

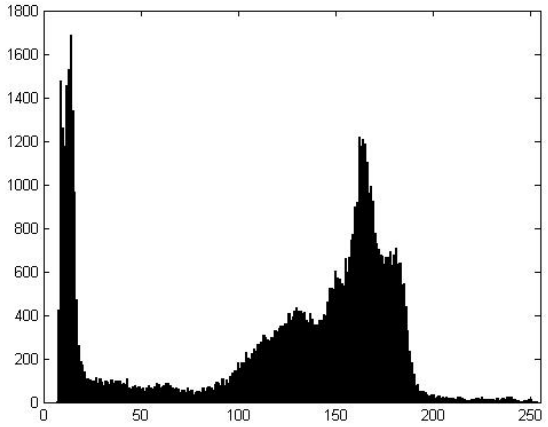


D

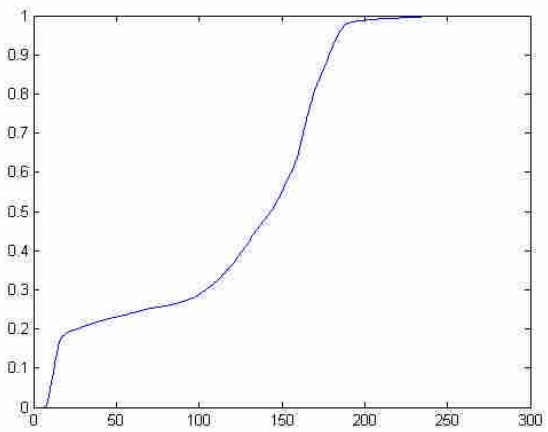


E

```
A=imread('cameraman.tif');
imagesc(A)
colormap(gray)
A=double(A);
x=0:255;
[y,x]=hist(A(:),x);
figure
plot(x,y)
B=255-double(A);
figure
imagesc(B)
colormap(gray)
C=A*2.*(A<=100)+(200+55/155
*(A-100)).*(A>100);
figure
imagesc(C)
colormap(gray)
D=sqrt(A*255);
imagesc(D)
E=A.^2/255;
imagesc(E)
```

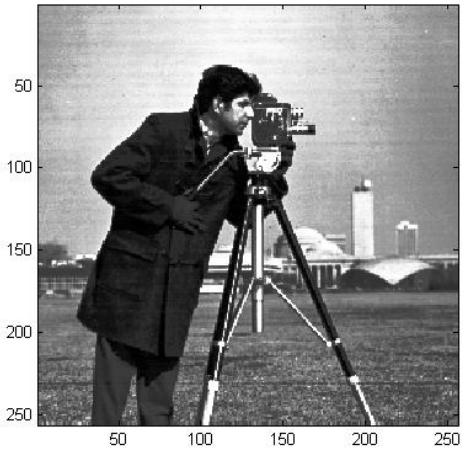


bar(x,y)

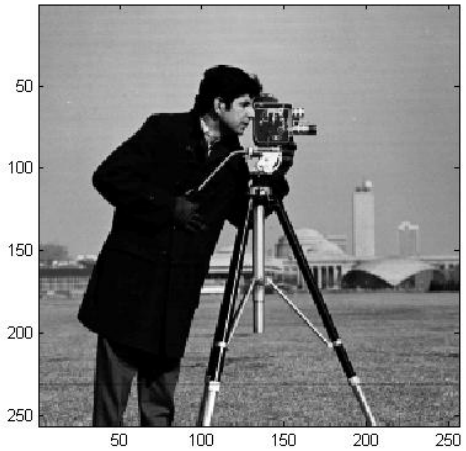


plot(x,z/(256^2))

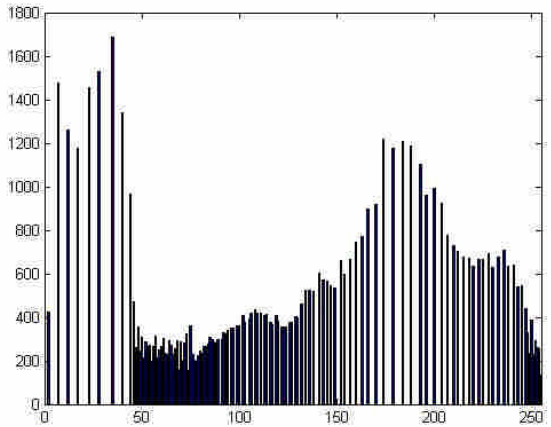
```
bar(x,y)
axis([0 255 0 1800])
for i=1:256
z(i)=sum(y(1:i));
end
figure
plot(x,z/(256^2))
for i=1:256
F(A==(i-1))=z(i)/(256^2)*255;
end
figure
imagesc(reshape(F,256,256))
colormap(gray)
[y,x]=hist(F(:),x);
bar(x,y)
```



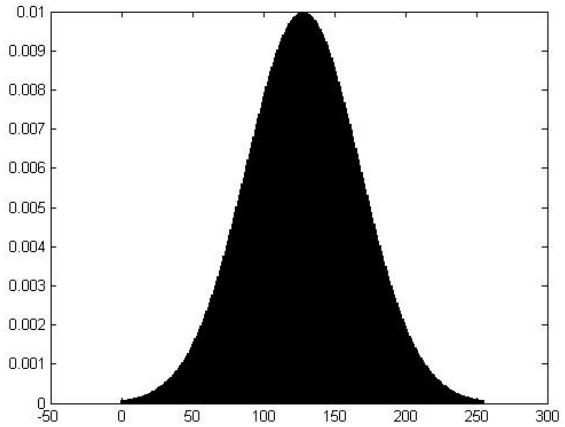
F (histogram equalization)



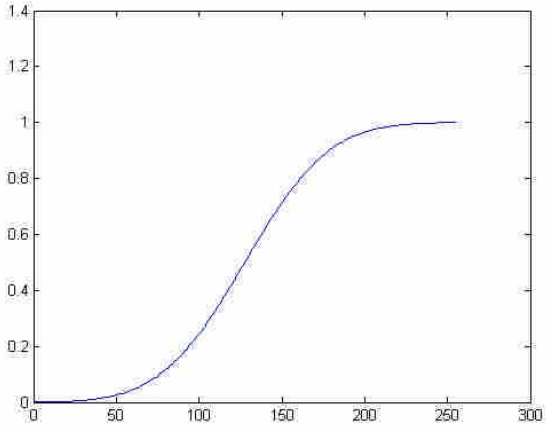
Original image



bar(x,y)

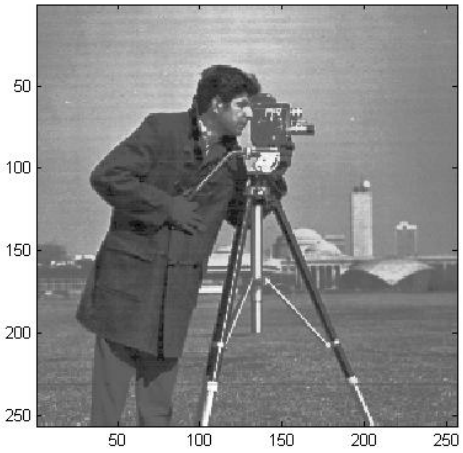


bar(x,y1)

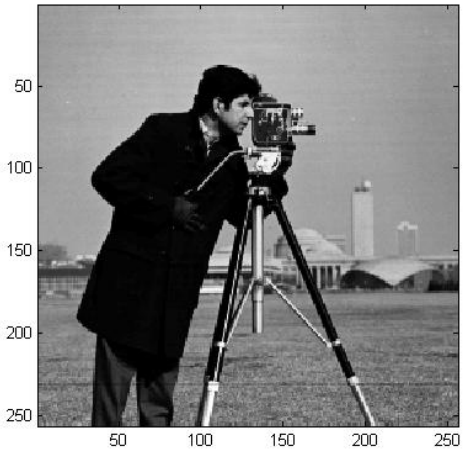


plot(x,z1)

```
y1=exp(-0.5*(x-128).^2/40^2);
y1=y1/sum(y1);
bar(x,y1)
for i=1:256
z1(i)=sum(y1(1:i));
end
plot(x,z1)
for i=1:256
[Y,m]=max(z1>z(i)/(256^2));
F(A==(i-1))=m;
end
figure
imagesc(reshape(F,256,256))
colormap(gray)
[y,x]=hist(F(:),x);
bar(x,y)
```

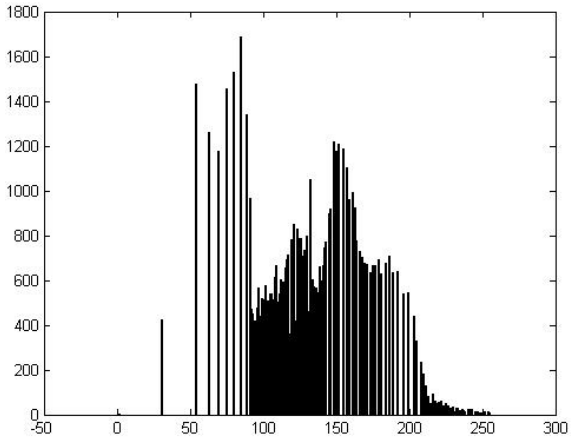


F (histogram specification)



Original image

bar(x,y)



密碼學

wiki

- **密碼學**（在西歐語文中之源於希臘語*kryptós*，「隱藏的」，和*gráphein*，「書寫」）是研究如何隱密地傳遞資訊的學門。在現代特別指對資訊以及其傳輸的數學性研究，常被認為是數學和計算機科學的分支，和資訊理論也密切相關。著名的密碼學者Ron Rivest解釋道：「密碼學是關於如何在敵人存在的環境中通訊」，自工程學的角度，這相當於密碼學與純數學的異同。密碼學是資訊安全等相關議題，如認證、存取控制的核心。密碼學的首要目的是隱藏訊息的涵義，並不是隱藏訊息的存在。密碼學也促進了電腦科學，特別是在於電腦與網路安全所使用的技術，如存取控制與資訊的機密性。密碼學已被應用在日常生活：包括自動櫃員機的晶片卡、電腦使用者存取密碼、電子商務等等。

- 在近代以前，**密碼學**只考慮到訊息的機密性（confidentiality）：如何將可理解的訊息轉換成難以理解的訊息，並且使得有秘密訊息的人能夠逆向回復，但缺乏秘密訊息的攔截者或竊聽者則無法解讀。近數十年來，這個領域已經擴展到涵蓋身分認證（或稱鑒權）、訊息完整性檢查、數位簽章、互動證明、安全多方計算等各類技術。
- 其實在西元前，秘密書信已用於戰爭之中。西洋「史學之父」希羅多德（Herodotus）的《歷史》（The Histories）當中記載了一些最早的秘密書信故事。西元前5世紀，希臘城邦為對抗奴役和侵略，與波斯發生多次衝突和戰爭。於西元前480年，波斯秘密結了強大的軍隊，準備對雅典（Athens）和斯巴達（Sparta）發動一次突襲。希臘人狄馬拉圖斯（Demaratus）在波斯的蘇薩城（Susa）裏看到了這次集結，便利用了一層蠟把木板上的字遮蓋住，送往並告知了希臘人波斯的圖謀。最後，波斯海軍覆沒於雅典附近的沙拉米斯灣（Salamis Bay）。
- 由於古時多數人並不識字，最早的秘密書寫的形式只用到紙筆或等同物品，隨著識字率提高，就開始需要真正的密碼學了。最古典的兩個加密技巧是是：
- 移位式（Transposition cipher）：將字母順序重新排列，例如『help me』變成『ehpl em』；與
- 替代式（substitution cipher）：有系統地將一組字母換成其他字母或符號，例如『fly at once』變成『gmz bu podf』（每個字母用下一個字母取代）。

- 這兩種單純的方式都不足以提供足夠的機密性。凱撒密碼是最經典的替代法，據傳由古羅馬帝國的皇帝凱撒所發明，用在與遠方將領的通訊上，每個字母被往後位移三格字母所取代。
- 加密旨在確保通訊的秘密性，例如間諜、軍事將領、外交人員間的通訊，同時也有宗教上的應用。舉例來說，早期基督徒使用密碼學模糊他們寫作的部份觀點以避免遭受迫害。666或部分更早期的手稿上的616是新約基督經啟示錄所指的野獸的數字，常用來暗指專迫害基督徒的古羅馬皇帝尼祿（Nero）。史上也有部份希伯來文密碼的記載。古印度愛經中也提及愛侶可利用密碼來通信。隱寫術也出現在古代，希羅多德記載將訊息刺青在奴隸的頭皮上，較近代的隱寫術使用隱形墨水、縮影術（en:microdots）或數位浮水印來隱藏訊息。

- 除了應用於軍事外，西元四世紀婆羅門學者跋舍耶那（:en:Vatsyayana） 所書的《愛慾經》（:en:Kama-Sutra）4中曾提及到用代替法加密資訊。書中第45項是秘密書信（:en:mlecchita-vikalpa） ，用以幫助婦女隱瞞她們與愛郎之間的關係。其中一種方法是把字母隨意配對互換，如套用在羅馬字母中，可有得出下表：
- A B C D E F G H I J K L M Z Y X W V U T S R Q P O N 由經典加密法產生的密碼文很容易洩漏關於明文的統計資訊，以現代觀點其實很容易被破解。阿拉伯人津帝（en:al-Kindi）便提及到如果要破解加密資訊，可在一篇至少一頁長的文章中數算出每個字母出現的頻率，在加密信件中也數算出每個符號的頻率，然後互相對換，這是頻率分析的前身，此後幾乎所有此類的密碼都馬上被破解。但經典密碼學現在仍未消失，經常出現在謎語之中（見en:cryptogram）。這種分析法除了被用在破解密碼法外，也常用於考古學上。在破解古埃及象形文字（en:Hieroglyphs）時便運用了這種解密法。

What is the book written without using the letter e?

- A 1939 novel called Gadsby by Ernest Vincent Wright (1872-1939) was written without using the letter e.
- The novel runs 267 pages and has about 50,000 words.

- If youth, throughout all history, had a champion to stand up for it; to show a doubting world that a child can think; and, possibly, do it practically; you wouldn't constantly run across folks today who claim that "a child don't know anything." A child's brain starts functioning at birth; and has, amongst its many infant convolutions, thousands of dormant atoms, into which God has put a mystic possibility for noticing an adult's act, and figuring out its purport.

- **中世紀至第二次世界大戰**
- 本質上所有的密碼仍然受到上述的破密法的危害，直到阿伯提（en:Leon Battista Alberti）約在1467年發明了多字元加密法（en:polyalphabetic cipher），阿伯提的創新在於對訊息的不同部分使用不同的代碼，他同時也發明了可能是第一個自動加密器，一個實現他部分想法的轉輪。多字元加密法最典型的例子是維瓊內爾加密法（en:Vigenere cipher）：加密重複使用到一個關鍵字（en:key word），用哪個字母取代端視輪替到關鍵字的哪個字母而定。儘管如此，多字元加密法仍然受到頻率分析法的部分危害，不過這直到十九世紀中期才被巴貝奇（en:Charles Babbage）發現。比較近代的著名的例子可數中世紀蘇格蘭的瑪麗女王（Mary Stuart, Queen of Scotland）、第一次世界大戰德國的齊默爾曼電報（Zimmerman Telegram）和第二次世界大戰的「謎」（Enigma）。

- **蘇格蘭的瑪麗女王**
- 西元1578年，瑪麗女王被伊莉莎白女王軟禁。在1586年1月6日瑪麗收到一批秘密信件，得悉了安東尼.貝平頓（Anthony Babington）的計劃。安東尼和幾個同黨在密謀營救瑪麗，並計劃行刺伊莉莎白女王。他們的信件被轉成密碼，並藏在啤酒桶的木塞以掩人耳目。但卻被英格蘭大臣華興翰（Walsingham）的從中截獲、複製、還信入塞，並由菲力普・馬尼斯（Philip van Marnix）破解信件。信件破解後，華興翰使菲力普摹擬瑪麗的筆跡引誘安東尼行動，把叛逆者一網成擒，審判並處死瑪麗女王。問題在於錯誤地使用脆弱的加密法會製造虛假的安全錯覺：安東尼對他們的通訊方式太過有信心，令他的加密方法過於簡單，輕易被敵人破解。

- **第一次世界大戰**
- 1914年8月25日德國的馬格德堡巡洋艦（Magdeburg）在芬蘭灣（Gulf of Finland）擱淺，俄國搜出多份德國的文件及兩本電碼本，一本被送往英國的「40號房間」（Room 40）進行密碼分析。同時，無線電的發明亦使得截獲密信易如反掌。由於德國通往美國的電纜在大戰開始時被剪斷了，德國借用了美國的海底電纜發電報到華盛頓，但電纜經過了英國，1917年1月17日齊默爾曼電報被「40號房間」截獲。同年2月23日，密電內容揭開了，內容指德國將在1917年2月1日開始『無限制潛艇戰』，用潛艇攻擊戰時包括中立國在內的海上商運船。為了阻止美國因此參戰，德國建議墨西哥入侵美國，並承諾幫助墨西哥從美國手中奪回德克薩斯、新墨西哥和亞利桑那三州。德國還要墨西哥說服日本共同進攻美國，德國將提供軍事和資金援助。密電內容揭開後，美國在4月16日向德國宣戰。

- **第二次世界大戰**
- 德國汲取了第一次大戰的教訓，發展出以機械代替人手的加密方法。雪畢伍斯（Arthur Scherbius）發明了「謎」（ENIGMA，恩尼格瑪密碼機），用於軍事和商業上。「謎」主要由鍵盤、編碼器和燈板組成。三組編碼器合、加上接線器和其他配件，合共提供了種一億億種編碼的可能性。1925年，「謎」開始有系列生產，在20年間，德國軍方購入了3萬多台「謎」，亦難倒了「40號房」，成為德國在二次大戰的重要工具。波蘭位於德國東面，俄國的西面，一直受到威脅，故成立了波蘭密碼局（Biuro Szyfrow）以獲取情報。波蘭從漢斯-提羅・施密德（Hans-Thilo Schmidt）處得到諜報，由年輕的數學家馬理安・瑞傑斯基（Marian Rejewski）解譯，用了一年時間編纂目錄，並在1930年代製造了「炸彈」（bomba），漸漸掌握瞭解「謎」的技術。
- 1938年12月德國加強了「謎」的安全性，令波蘭失去了情報。「謎」成為了希特勒（Hitler）閃電戰略的核心，每天更改的加密排列維繫了強大快速的攻擊。 1939年4月27日德國撤銷與波蘭的互不侵犯條約，波蘭才不得不把決定「炸彈」這個構想與英、法分享，合力破解新的「謎」。1939年9月1日，德國侵擊波蘭，大戰爆發。英國得到了波蘭的解密技術後，40號房間除了原有的語言和人文學家，還加入了數學家和科學家，後來更成立了政府代碼暨密碼學校（Government code and Cipher School），5年內人數增至7000人。1940至1942年是加密和解密的拉鋸戰，成功的解碼提供了很多寶貴的情報。例如在1940年得到了德軍進攻丹麥和挪威的作戰圖，以及在不列顛戰役（Battle of Britain）事先獲得了空襲情報，化解了很多危機。但「謎」卻並未被完全破解，加上「謎」的網路很多，令德國一直在大西洋戰役中佔上風。最後英國在「順手牽羊」的行動中在德國潛艇上俘獲「謎」的密碼簿，破解了「謎」。英國以各種虛假手段掩飾這件事，免得德國再次更改密碼，並策劃摧毀了德國的補給線，縮短了大西洋戰役。
- 許多物理裝置被用來輔助加密，例如古希臘斯巴達的密碼棒（en:scytale），這是一個協助置換法的圓柱體，可將資訊內字母的次序調動，利用了字條纏繞木棒的方式，把字母進行位移，收信人要使用相同直徑的木棒才能得到還原的資訊。在歐洲中世紀時期，密碼欄（en:cipher grille）用在某類隱寫術上。多字元加密法出現後，更多樣的輔助工具出現，如阿伯提發明的密碼盤（en:cipher disk）、特里特米烏斯發明的表格法（en:tabula recta）、以及美國總統湯瑪士傑佛遜（en:Thomas Jefferson）發明的多圓柱（en:Bazeries約在1900年再次獨立發明改進）。廿世紀早期，多項加解密機械被發明且被註冊專利，包括最有名的轉輪機（en:rotor machines），第二次世界大戰德軍所用，別名『謎』（恩尼格瑪密碼機），其加密法是在第一次世界大戰後針對當時破密術所做最好的設計。

Arithmetic Operations

- Image Addition

$$G_{sum}(i, j) = G_1(i, j) + G_2(i, j)$$

  – Force the value into 0~255

$$G_{sum}(i, j) = [G_1(i, j) + G_2(i, j)] / 2$$

Image Averaging

- Noise is any random phenomenon that contaminates an image.
- Noise is typically modeled as an additive process

$$g(m,n) = f(m,n) + N(m,n)$$

Noisy image — Noise free image — Noise

Image Averaging

- The noise N(m,n) at each pixel (m,n) is modeled as a random variable.
- Usually, N(m,n) has Zero-mean and the noise values at different pixels are uncorrelated.
- Suppose we have M observations $\{g_i(m,n)\}$, $i=1,2,\ldots,K$, we can (partially) mitigate the effect of noise by "averaging"

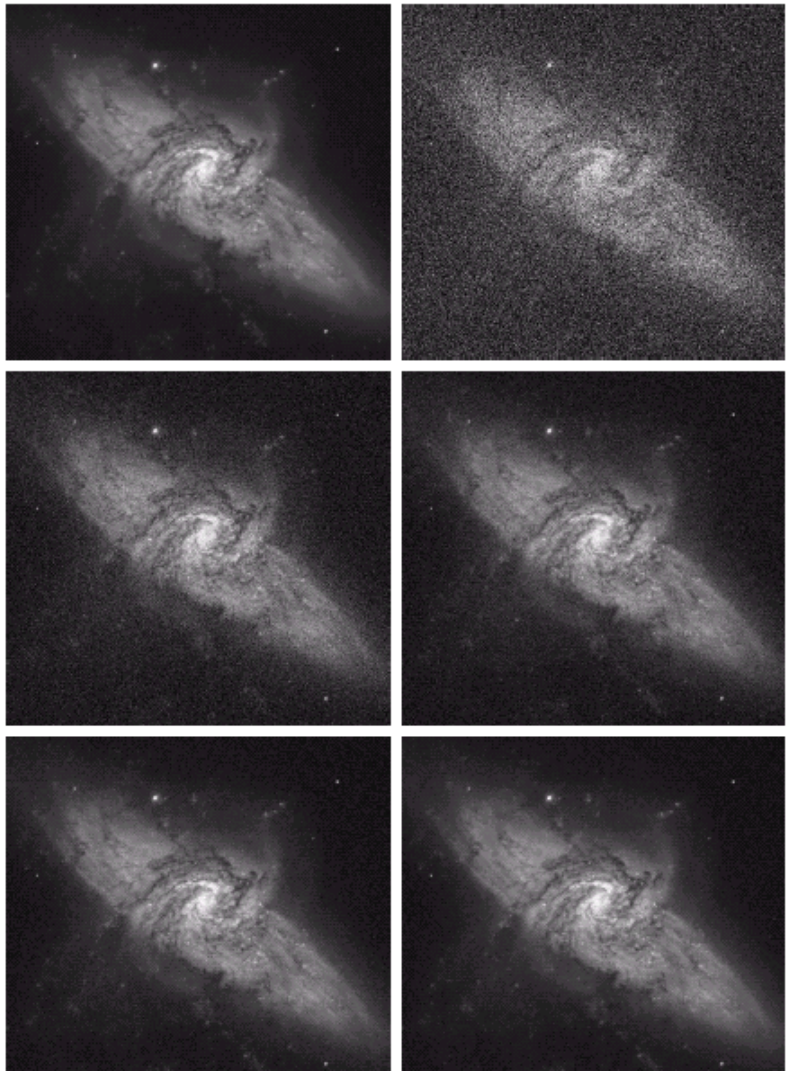
$$\bar{g}(m,n) = \frac{1}{K}\sum_{i=1}^{K} g_i(m,n)$$

then it follows that

$$E[\bar{g}(m,n)] = f(m,n) \qquad Var[\bar{g}(m,n)] = \frac{1}{K}Var[N(m,n)]$$

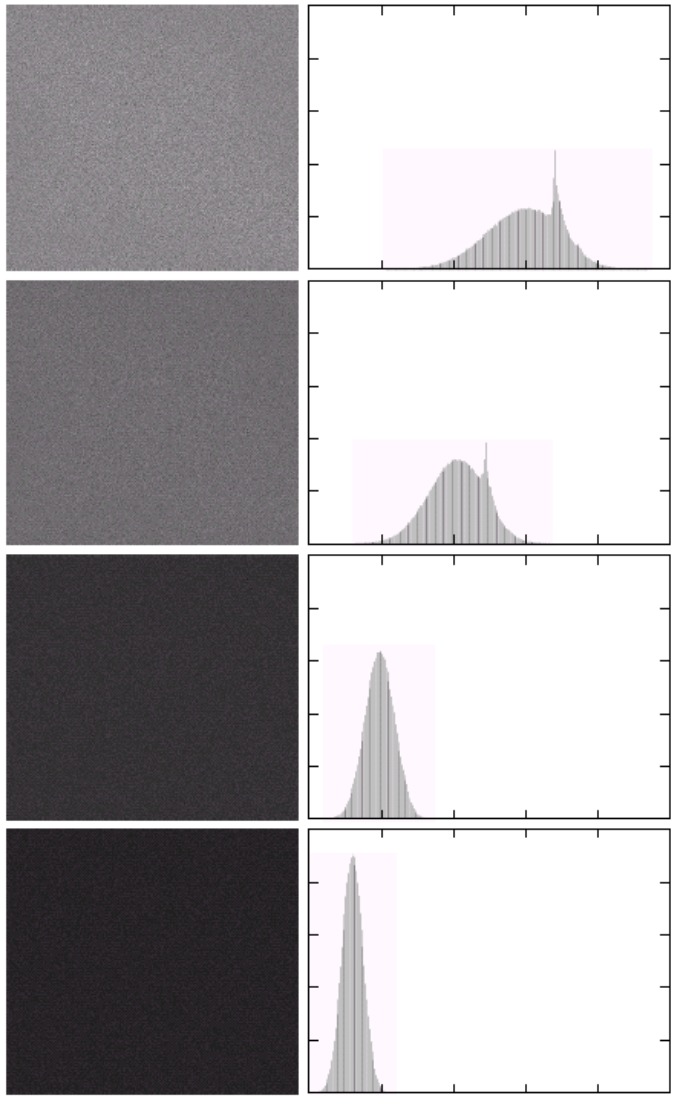
- Therefore, as the number on observations increases (M→infinite), the effect of noise trends to zero.

Image Averaging



a b
c d
e f

FIGURE 3.30 (a) Image of Galaxy Pair NGC 3314. (b) Image corrupted by additive Gaussian noise with zero mean and a standard deviation of 64 gray levels. (c)–(f) Results of averaging $K = 8, 16, 64$, and 128 noisy images. (Original image courtesy of NASA.)



a b

FIGURE 3.31
(a) From top to bottom: Difference images between Fig. 3.30(a) and the four images in Figs. 3.30(c) through (f), respectively. (b) Corresponding histograms.

Arithmetic Operations

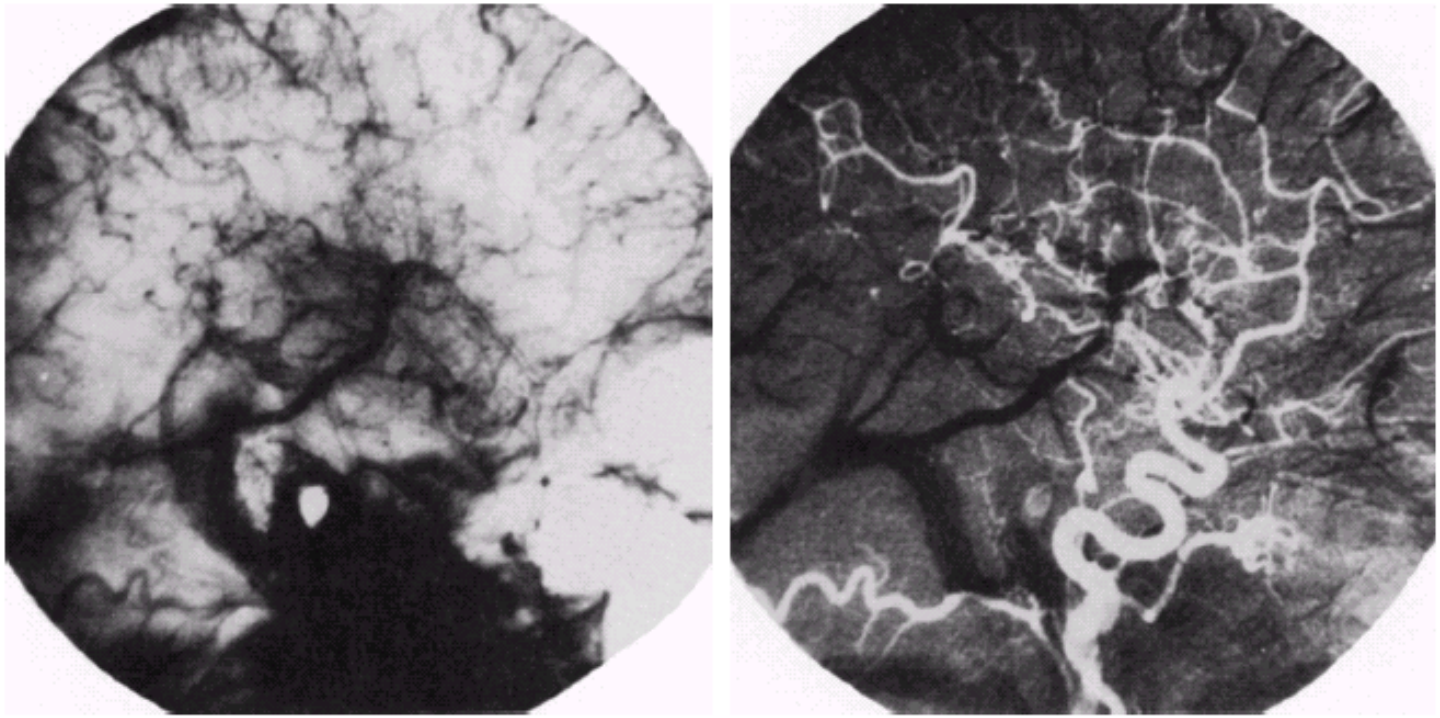
- Image Subtraction

$$G_{diff}(i, j) = G_1(i, j) - G_2(i, j)$$

  – Force the value into 0~255

$$G_{diff}(i, j) = [255 + G_1(i, j) - G_2(i, j)] / 2$$

Example



a b

FIGURE 3.29
Enhancement by image subtraction. (a) Mask image. (b) An image (taken after injection of a contrast medium into the bloodstream) with mask subtracted out.

Example: segmentation

original

After processing

Subtraction result

Spatial Filter

- Basics of Spatial Filter
- Smoothing Spatial Filters
- Sharpening Filters

Basics of Spatial Filter

- some neighborhood operations work with the values of the image pixels in the neighborhood and the corresponding values of a subimage that has the same dimensions as the neighborhood.

Basics of Spatial Filter

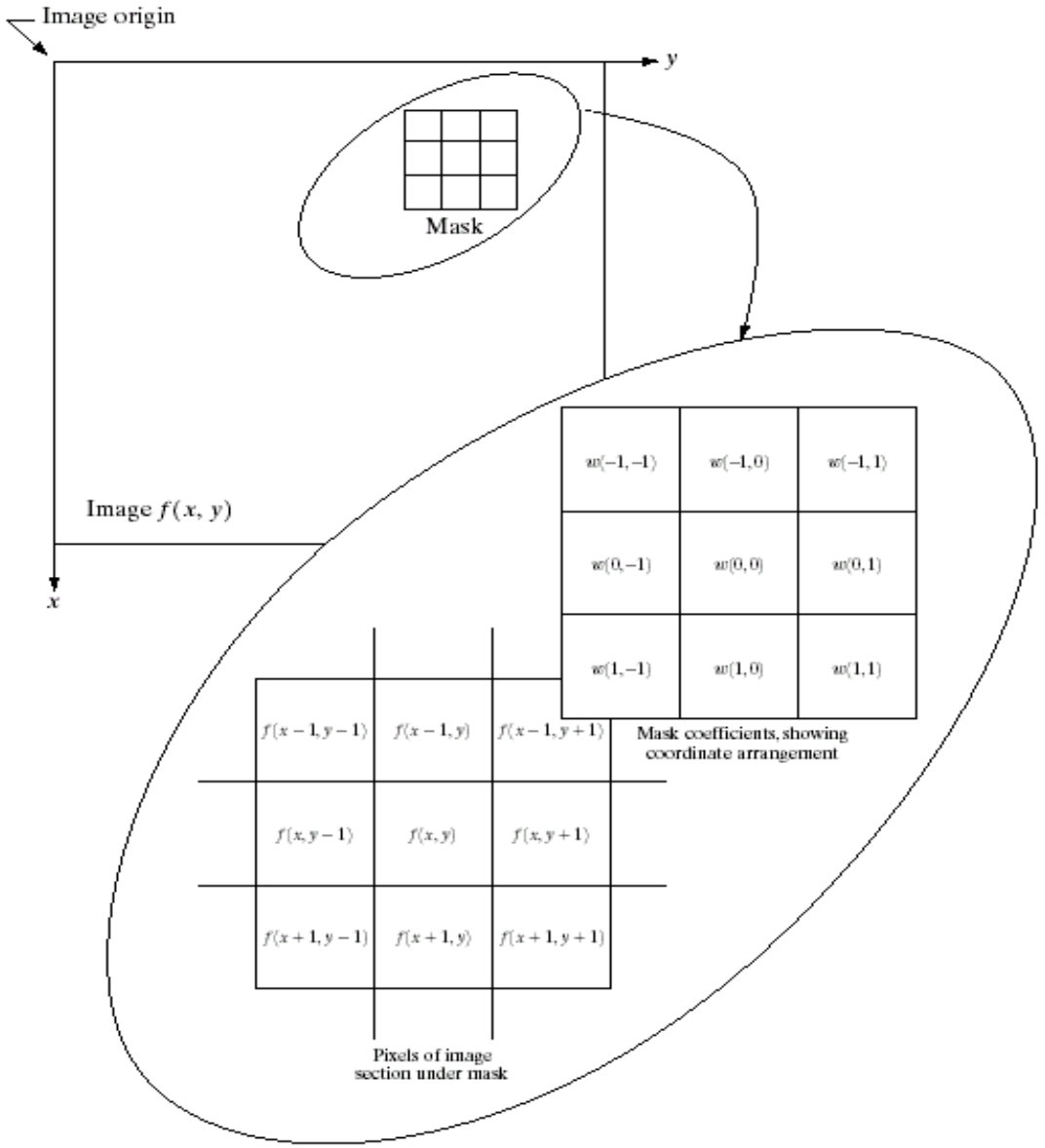


FIGURE 3.32 The mechanics of spatial filtering. The magnified drawing shows a 3×3 mask and the image section directly under it; the image section is shown displaced out from under the mask for ease of readability.

Basics of Spatial Filter

- In general, linear filtering of an image f of size M*N with a filter mask of size m*n is given by the expression:

$$g(x, y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s, t) f(x + s, y + t)$$

a=(m-1)/2 and *b=(n-1)/2*,*m* x *n* (odd numbers)

Basics of Spatial Filter

- The basic approach is to sum products between the mask coefficients and the intensities of the pixels under the mask at a specific location in the image:

$$R = w_1 z_1 + w_2 z_2 + \ldots + w_9 z_9$$

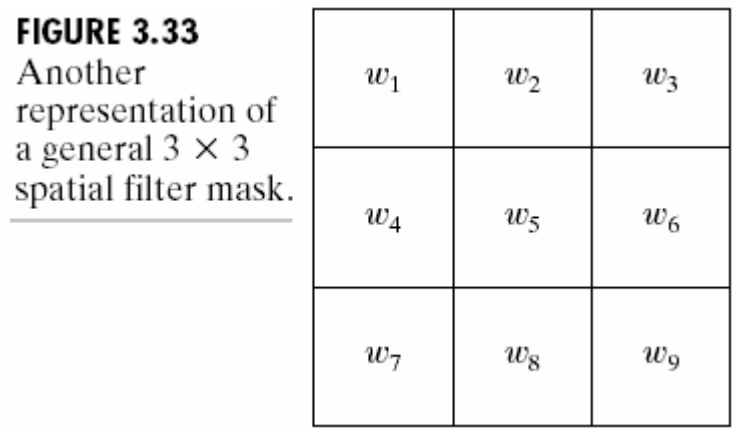


FIGURE 3.33 Another representation of a general 3 × 3 spatial filter mask.

(for a 3 x 3 filter)

Smoothing Spatial Filters

- Smoothing filters are used for blurring and for noise reduction.
  - Blurring is used in preprocessing steps, such as removal of small details from an image prior to object extraction, and bridging of small gaps in lines or curves
  - Noise reduction can be accomplished by blurring

Smoothing Spatial Filters

- Linear Smoothing Filters – averaging filters
- Nonlinear Smoothing Filters – median filters

Linear Smoothing Filters

- Linear spatial filter is simply the average of the pixels contained in the neighborhood of the filter mask.
- The idea is replacing the value of every pixel in an image by the average of the gray levels in the neighborhood defined by the filter mask.
- These filters sometimes are called *averaging filters o*r *lowpass filters*.

Linear Smoothing Filters

$$\frac{1}{9}\times\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

Standard average

$$\frac{1}{16}\times\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix}$$

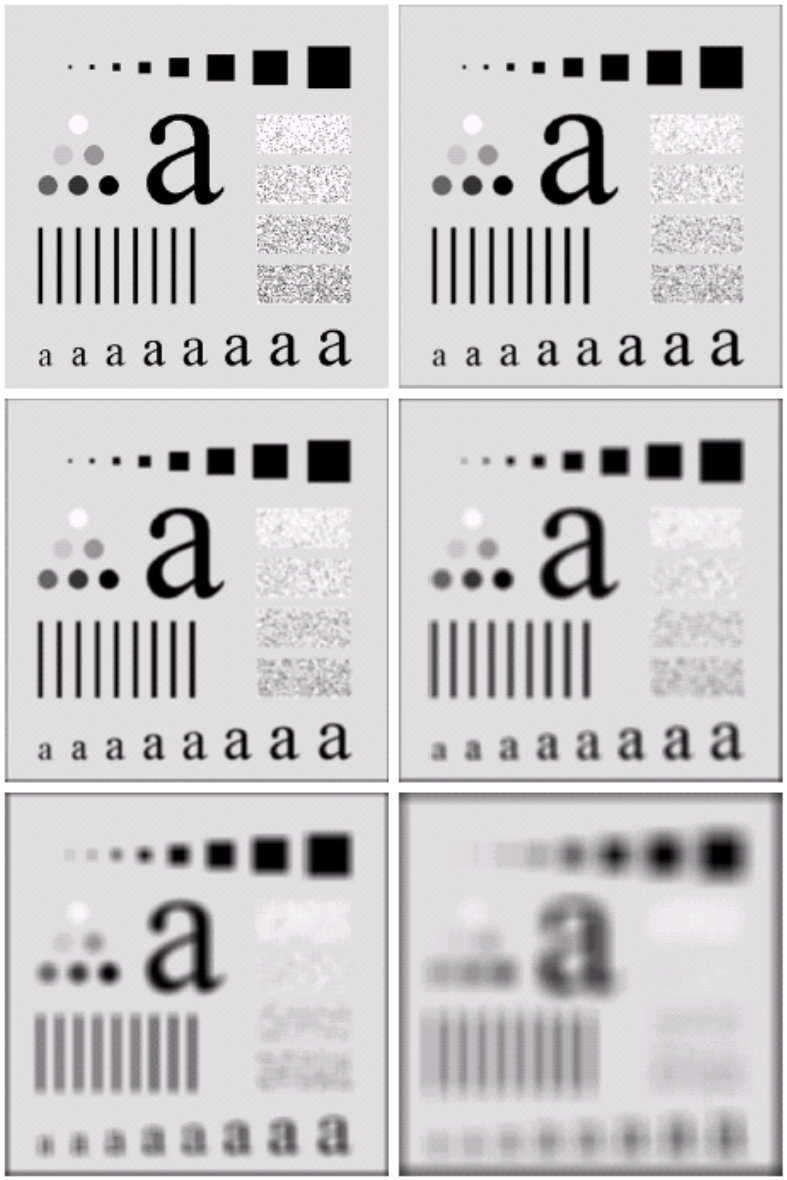
Weighted average

Linear Smoothing Filters

- The general implementation for filtering an MxN image with a weighted averaging filter of size mxn is given by the expression

$$g(x,y) = \frac{\sum_{s=-a}^{a}\sum_{t=-b}^{b} w(s,t)\, f(x+s, y+t)}{\sum_{s=-a}^{a}\sum_{t=-b}^{b} w(s,t)}$$

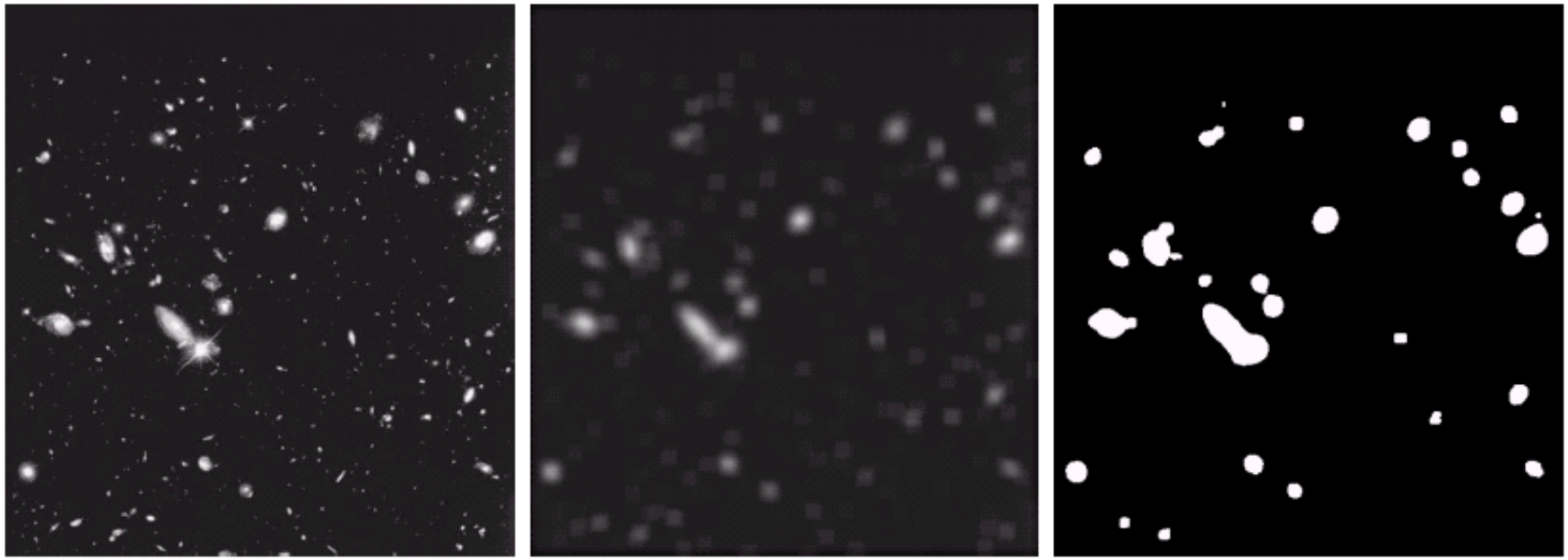
Image Enhancement in the Spatial Domain



a b
c d
e f

FIGURE 3.35 (a) Original image, of size 500 × 500 pixels. (b)–(f) Results of smoothing with square averaging filter masks of sizes $n = 3, 5, 9, 15$, and 35, respectively. The black squares at the top are of sizes 3, 5, 9, 15, 25, 35, 45, and 55 pixels, respectively; their borders are 25 pixels apart. The letters at the bottom range in size from 10 to 24 points, in increments of 2 points; the large letter at the top is 60 points. The vertical bars are 5 pixels wide and 100 pixels high; their separation is 20 pixels. The diameter of the circles is 25 pixels, and their borders are 15 pixels apart; their gray levels range from 0% to 100% black in increments of 20%. The background of the image is 10% black. The noisy rectangles are of size 50 × 120 pixels.

Image Enhancement in the Spatial Domain



a b c

FIGURE 3.36 (a) Image from the Hubble Space Telescope. (b) Image processed by a 15 × 15 averaging mask. (c) Result of thresholding (b). (Original image courtesy of NASA.)

Order-Statistics Filters

- Order-statistics filters are nonlinear spatial filters whose response is based on ordering (ranking) the pixels contained in the image area encompassed by the filter, and then replacing the value of the center pixel with the value determined by the ranking result.

- Best-known “median filter”
  - to force points with very distinct intensities to be more like their neighbors, such that don’t blur edges
  - effective for reducing impulse noise,

    salt-and-pepper noise

Median Filter

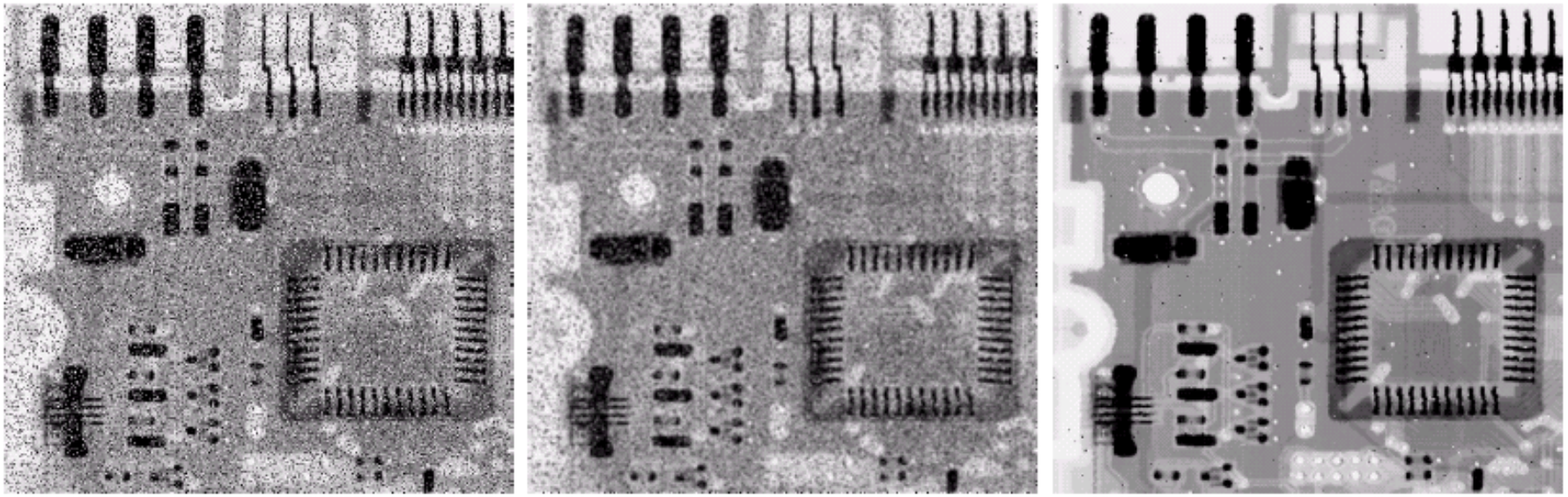
10	15	20
20	100	20
20	20	25

10, 15, 20, 20, 20, 20, 20, 25, 100

5th

- Sort the values of the pixel in our region
- In the MxN mask the median is MxN div 2 +1

Image Enhancement in the Spatial Domain



a b c

FIGURE 3.37 (a) X-ray image of circuit board corrupted by salt-and-pepper noise. (b) Noise reduction with a 3×3 averaging mask. (c) Noise reduction with a 3×3 median filter. (Original image courtesy of Mr. Joseph E. Pascente, Lixi, Inc.)

Sharpening Filters

- The principal objective of sharpening is to highlight fine detail in an image or to enhance detail that has been blurred, either in error or as an natural effect of a particular method of image acquisition.
  - smoothing → integration
  - Sharpening → differentiation

Derivatives

- Definition for a first derivative
  - Must be zero in flat segments
  - Must be nonzero at the onset of a gray-level step or ramp; and
  - Must be nonzero along ramps.
- A basic definition of the first-order derivative of a one-dimensional function f(x) is

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

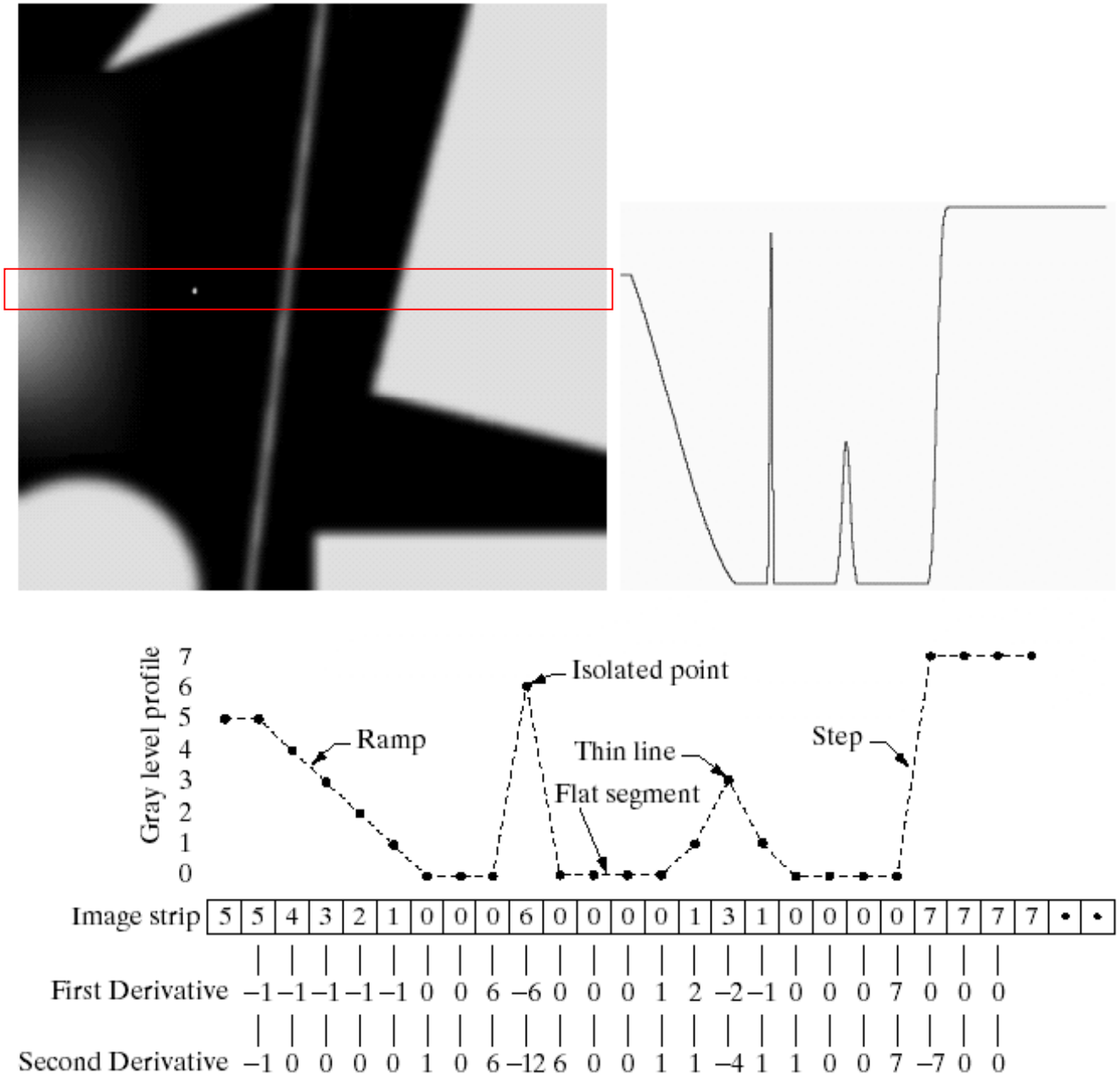
Derivatives

- Definition for a second derivative
  - Must be zero in flat areas;
  - Must be nonzero at the onset and end of a gray-level step or ramp;
  - Must be zero along ramps of constant slope
- We define a second-order derivative as the difference

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

a b
c

FIGURE 3.38
(a) A simple image. (b) 1-D horizontal gray-level profile along the center of the image and including the isolated noise point.
(c) Simplified profile (the points are joined by dashed lines to simplify interpretation).



Observations

- The 1st-order derivative is nonzero along the entire ramp, while the 2nd-order derivative is nonzero only at the onset and end of the ramp.
- 1st make thick edge and 2nd make thin edge
- The response at and around the point is much stronger for the 2nd- than for the 1st-order derivative

The Gradient

- First Derivatives in image processing are implemented using the magnitude of the gradient.
- For a function *f*(x, y), the gradient of *f* at coordinates (x, y) is defined as the two-dimensional column *vector*

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

The magnitude of this vector is given by

$$\nabla f = mag(\nabla F) = \sqrt{G_x^2 + G_y^2} \approx |G_x| + |G_y|$$

G_x

-1	1

G_y

1
-1

Robert's Method

$G_x = (z_9\text{-}z_5)$ and $G_y = (z_8\text{-}z_6)$

$$\nabla f = \sqrt{(z_9 - z_5)^2 + (z_8 - z_6)^2}$$

Roberts Cross-Gradient Operators

$$\nabla f \approx |z_9 - z_5| + |z_8 - z_6|$$

Mask of even size are awkward to apply so the smallest filter mask should be 3x3.

Sobel's Method

$$\nabla f \approx |(z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)| + |(z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)|$$

a
b c
d e

FIGURE 3.44
A 3×3 region of an image (the z's are gray-level values) and masks used to compute the gradient at point labeled z_5. All masks coefficients sum to zero, as expected of a derivative operator.

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

Roberts Cross-Gradient

−1	0
0	1

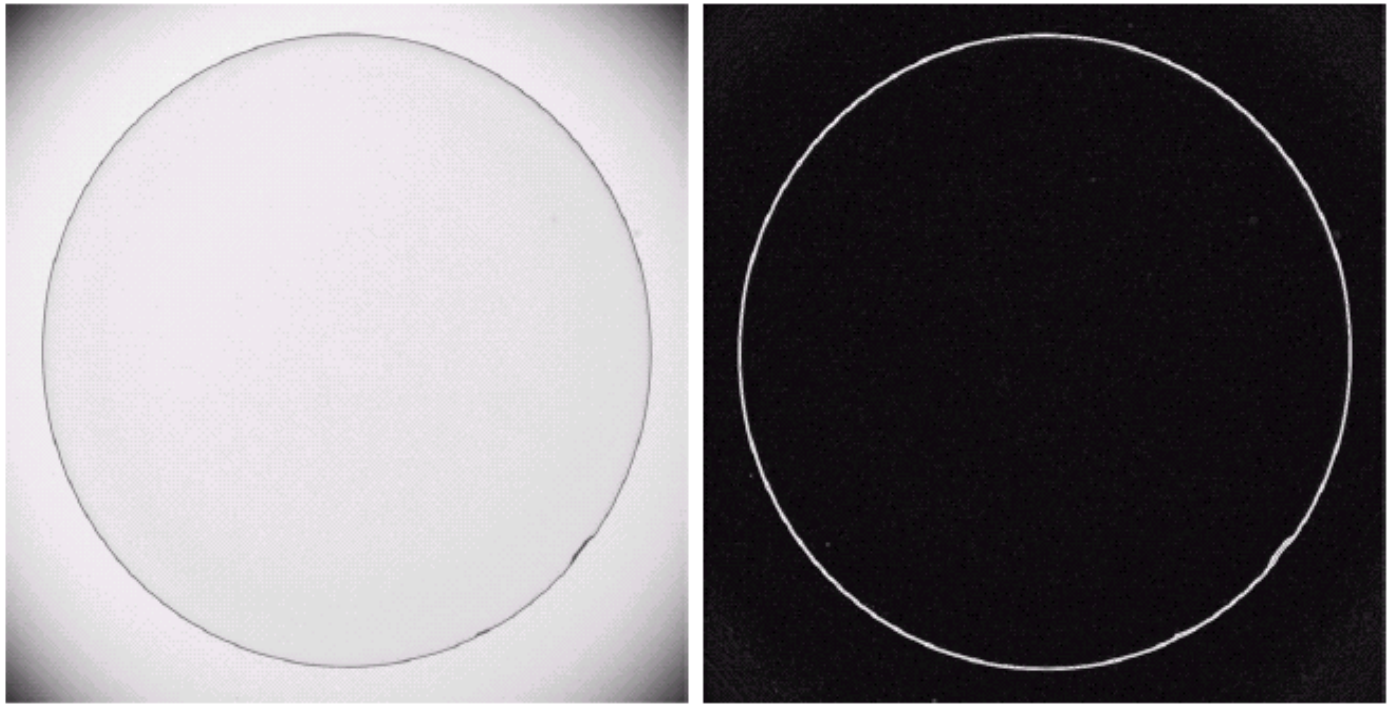
0	−1
1	0

Sobel Operators

−1	−2	−1
0	0	0
1	2	1

−1	0	1
−2	0	2
−1	0	1

Example



a b

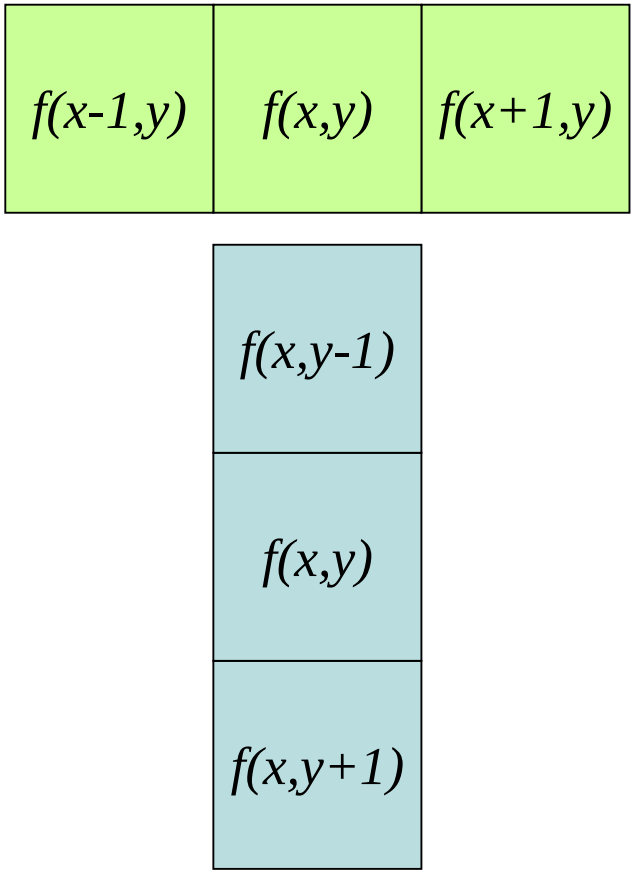
FIGURE 3.45
Optical image of contact lens (note defects on the boundary at 4 and 5 o'clock). (b) Sobel gradient. (Original image courtesy of Mr. Pete Sites, Perceptics Corporation.)

Laplacian

- 2^{nd} order derivative
- Isotropic filters: rotation invariant
- Linear operator
- Define as

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

Discrete version:



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

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

The digital implementation of the 2-Dimensional Laplacian is obtained by summing 2 components

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

$$\nabla^2 f = f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1) - 4f(x, y)$$

0	1	0
1	-4	1
0	1	0

1	0	1
0	-4	0
1	0	1

The basic way in which we use the Laplacian for image enhancement is as follows:

$$g(x,y)=\begin{cases} f(x,y)-\nabla^2 f(x,y) & \text{If the center coefficient is negative} \\ f(x,y)+\nabla^2 f(x,y) & \text{If the center coefficient is positive} \end{cases}$$

Where f(x,y) is the original image

$\nabla^2 f(x,y)$ is Laplacian filtered image

g(x,y) is the sharpen image

Simplification：

$$g(x,y)=f(x,y)-f(x+1,y)-f(x-1,y)-f(x,y+1)-f(x,y-1)+4f(x,y)$$

$$g(x,y)=5f(x,y)-f(x+1,y)-f(x-1,y)-f(x,y+1)-f(x,y-1)$$

a b
c d

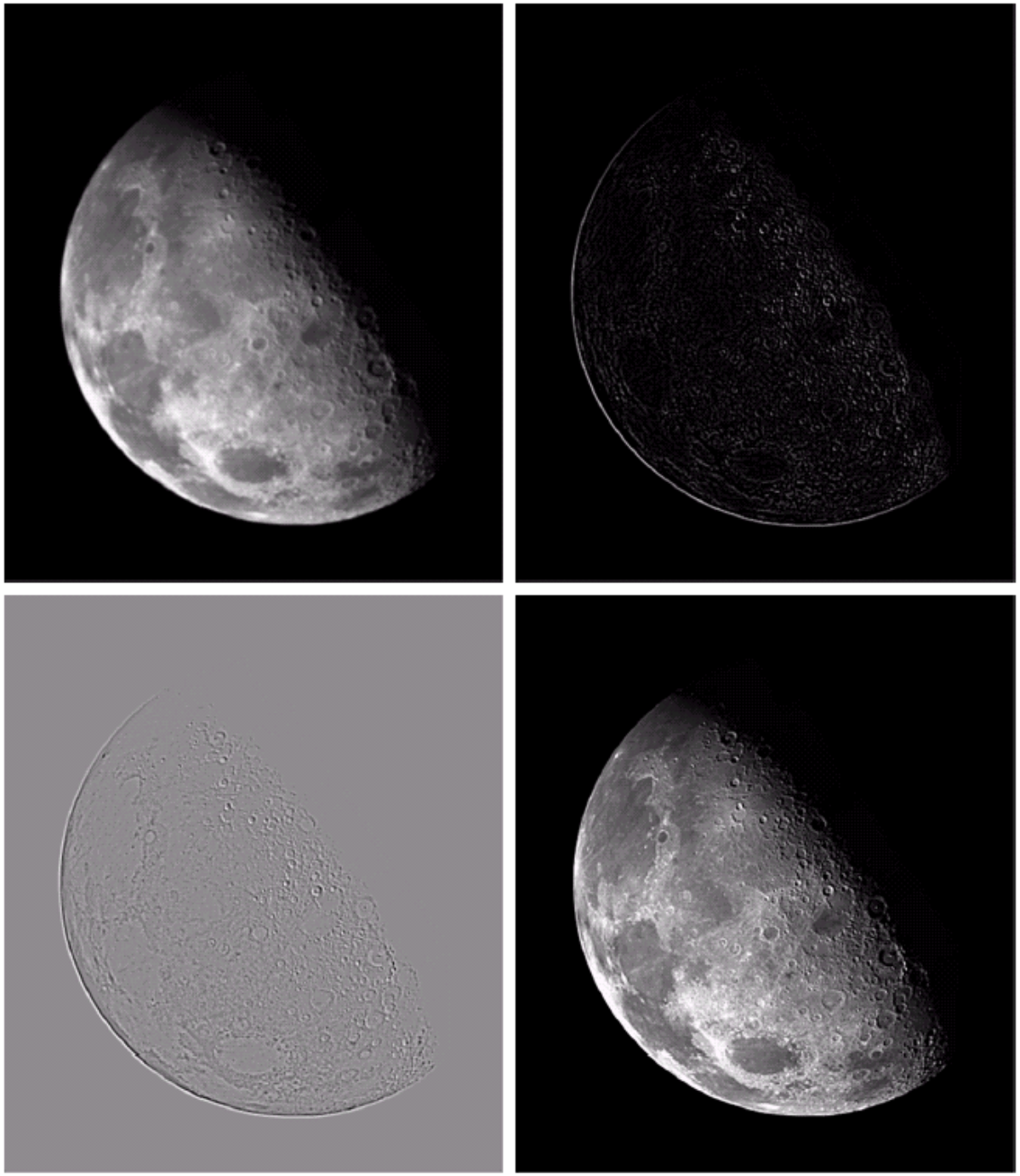


FIGURE 3.40
(a) Image of the North Pole of the moon.
(b) Laplacian-filtered image.
(c) Laplacian image scaled for display purposes.
(d) Image enhanced by using Eq. (3.7-5).
(Original image courtesy of NASA.)