Image Processing and Analysis
Lecture 7、Image Restoration

Weiqiang Wang
School of Computer Science and Technology, UCAS

October 25, 2023

Outline

Image Degradation and Restoration

- The objective of restoration

  To improve a given image in some predefined sense.
- The difference between restoration and enhancement

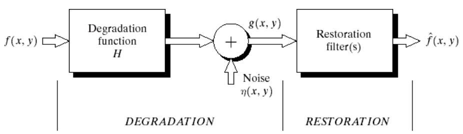


FIGURE 5.1 A model of the image degradation/ restoration process.

$$g(x,y) = H[f(x,y)] + \eta(x,y)$$
$$g(x,y) = h(x,y) * f(x,y) + \eta(x,y)$$
$$G(u,v) = H(u,v)F(u,v) + N(u,v)$$

Noise Models

- Adding Noise with Function imnoise
- Generating Spatial Random Noise with a Specified Distribution
- Periodic Noise
- Estimating Noise Parameters

Adding Noise with Function imnoise

- g = imnoise(f, type, parameters)
  f: input image, type and parameters will explain later.
  [note]: converts the input image to class double in the range [0,1] before adding noise to it.

  For Example
  - G=imnoise(f,'gussian',m,var)
  - G=imnoise(f,'localvar',V)
  - G=imnoise(f,'localvar',image_intensity,var)
  - G=imnoise(f,'salt&pepper',d))
  - G=imnoise(f,'speckle', var)
  - G=imnoise(f,'poisson')

Noise Models

Generating Random Noise with a Specified Distribution

- A famous result : if w is a uniformly distributed random varible in the interval (0,1), then we can obtained a random variable z with a specified CDF, F_z, by solving the equation

$$z = F_Z^{-1}(w)$$

- An Example: generate random numbers, z, *with a Rayleigh CDF*

$$F_Z(z) = \begin{cases} 1 - e^{-(z-a)^2/b} & if\ z \ge a \\ 0 & if\ z < a \end{cases}$$

  We can obtain z
  by solving the following equation

$$1 - e^{-(z-a)^2/b} = w$$

  get

$$z = a + \sqrt{-b\ln(1-w)}$$

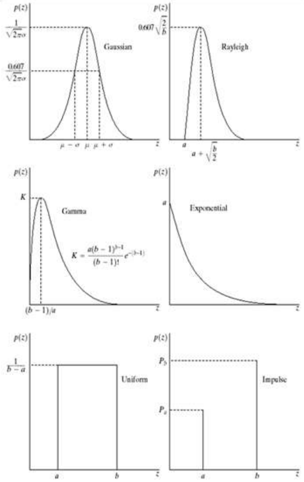
Generating Random Noise with a Specified Distribution

TABLE 5.1 Generation of random variables.

Name	PDF	Mean and Variance	CDF	Generator[†]
Uniform	$p_z(z) = \begin{cases} \frac{1}{b-a} & \text{if } a \le z \le b \\ 0 & \text{otherwise} \end{cases}$	$m = \frac{a+b}{2}, \quad \sigma^2 = \frac{(b-a)^2}{12}$	$F_z(z) = \begin{cases} 0 & z < a \\ \frac{z-a}{b-a} & a \le z \le b \\ 1 & z > b \end{cases}$	MATLAB function rand
Gaussian	$p_z(z) = \frac{1}{\sqrt{2\pi}b} e^{-(z-a)^2/2b^2}$ $-\infty < z < \infty$	$m = a, \quad \sigma^2 = b^2$	$F_z(z) = \int_{-\infty}^{z} p_z(v)\,dv$	MATLAB function randn
Salt & Pepper ??	$p_z(z) = \begin{cases} P_a & \text{for } z = a \\ P_b & \text{for } z = b \\ 0 & \text{otherwise} \end{cases}$ $b > a$	$m = aP_a + bP_b$ $\sigma^2 = (a-m)^2P_a + (b-m)^2P_b$	$F_z(z) = \begin{cases} 0 & \text{for } z < a \\ P_a & \text{for } a \le z < b \\ P_a + P_b & \text{for } b \le z \end{cases}$	MATLAB function rand with some additional logic
Lognormal	$p_z(z) = \frac{1}{\sqrt{2\pi}bz} e^{-[\ln(z)-a]^2/2b^2}$?? 0	$m = e^{a+(b^2/2)}, \sigma^2 = [e^{b^2} - 1]e^{2a+b^2}$	$F_z(z) = \int_0^z p_z(v)\,dv$	$z = ae^{bN(0,1)}$??
Rayleigh	$p_z(z) = \begin{cases} \frac{2}{b}(z-a)e^{-(z-a)^2/b} & z \ge a \\ 0 & z < a \end{cases}$	$m = a + \sqrt{\pi b/4}, \sigma^2 = \frac{b(4-\pi)}{4}$	$F_z(z) = \begin{cases} 1 - e^{-(z-a)^2/b} & z \ge a \\ 0 & z < a \end{cases}$	$z = a + \sqrt{b\ln[1 - U(0,1)]}$
Exponential	$p_z(z) = \begin{cases} ae^{-az} & z \ge 0 \\ 0 & z < 0 \end{cases}$	$m = \frac{1}{a}, \quad \sigma^2 = \frac{1}{a^2}$	$F_z(z) = \begin{cases} 1 - e^{-az} & z \ge 0 \\ 0 & z < 0 \end{cases}$	$z = -\frac{1}{a}\ln[1 - U(0,1)]$
Erlang	$p_z(z) = \frac{a^b z^{b-1}}{(b-1)!} e^{-az}$ $z \ge 0$	$m = \frac{b}{a}, \quad \sigma^2 = \frac{b}{a^2}$	$F_z(z) = \left[1 - e^{-az}\sum_{n=0}^{b-1}\frac{(az)^n}{n!}\right]$ $z \ge 0$	$z = E_1 + E_2 + \cdots + E_b$ (The E's are exponential random numbers with parameter a.)

[†] $N(0,1)$ denotes normal (Gaussian) random numbers with mean 0 and a variance of 1. $U(0,1)$ denotes uniform random numbers in the range (0, 1).

Generating Random Noise with a Specified Distribution



a b
c d
e f

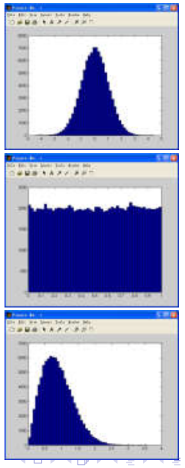
FIGURE 5.2 Some important probability density functions.

Generating Random Noise with a Specified Distribution

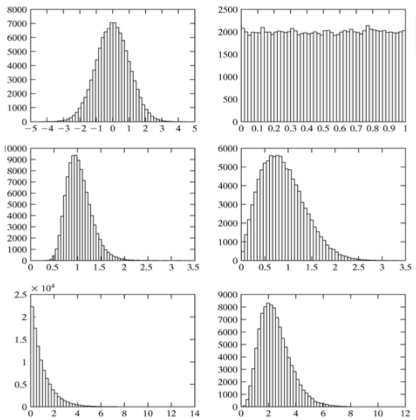
```
r=imnoise2('gaussian',100000,1,0,1);
hist(r,50);

ru=imnoise2('uniform',100000,1,0,1);
hist(ru,50);

rr=imnoise2('rayleigh',100000,1,0,1);
hist(rr,50);
```



Generating Random Noise with a Specified Distribution



a b
c d
e f

FIGURE 5.2 Histograms of random numbers: (a) Gaussian, (b) uniform, (c) lognormal, (d) Rayleigh, (e) exponential, and (f) Erlang. In each case the default parameters listed in the explanation of function `imnoise2` were used.

Generating Random Noise with a Specified Distribution

- Adding Noise with Function imnoise
- Generating Spatial Random Noise with a Specified Distribution
- Periodic Noise
- Estimating Noise Parameters

Periodic Noise

- Periodic Noise Signal

$$r(x,y) = A\sin(2\pi u_0(x+B_x)/M + 2\pi v_0(y+B_y)/N)$$

- Its FT

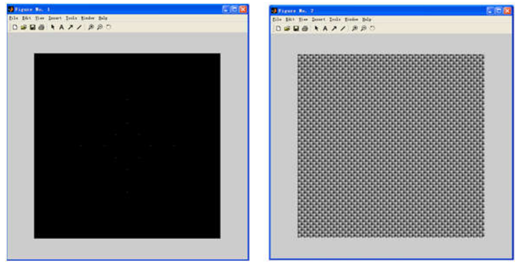
$$R(u,v) = j\frac{A}{2}[\exp\left(\frac{j2\pi u_0 B_x}{M}\right)\delta(u+u_0, v+v_0)$$

??

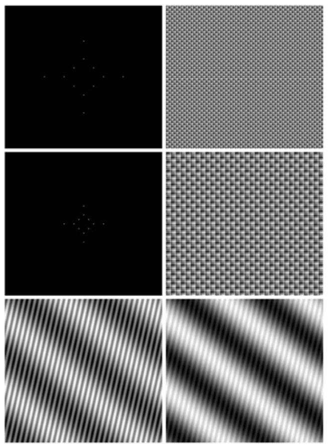
$$-\exp(\frac{j2\pi v_0 B_y}{N})\ \delta(u-u_0, v-v_0)]$$

Periodic Noise

- C=[0 64;0 128;32 32;64 0;128 0;-32 32];
- [r,R,S]=imnoise3(512,512,C);
- imshow(S,[]);
- figure,imshow(r,[]);



Periodic Noise



a b
c d
e f

FIGURE 5.3 (a) Spectrum of specified impulses. (b) Corresponding sine noise pattern. (c) and (d) A similar sequence. (e) and (f) Two other noise patterns. The dots in (a) and (c) were enlarged to make them easier to see.

Noise Models

- Adding Noise with Function imnoise
- Generating Spatial Random Noise with a Specified Distribution
- Periodic Noise
- Estimating Noise Parameters

Example: Images corrupted by Independent Noises

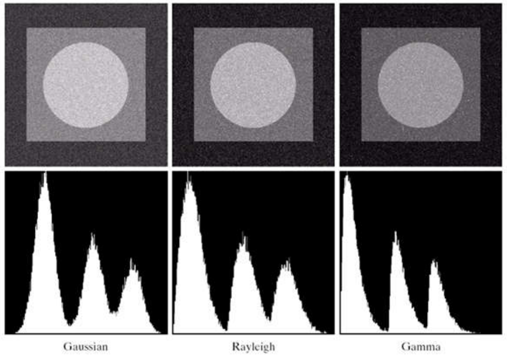


FIGURE 5.4 Images and histograms resulting from adding Gaussian, Rayleigh, and gamma noise to the image in Fig. 5.3.

Example: Images corrupted by Independent Noises

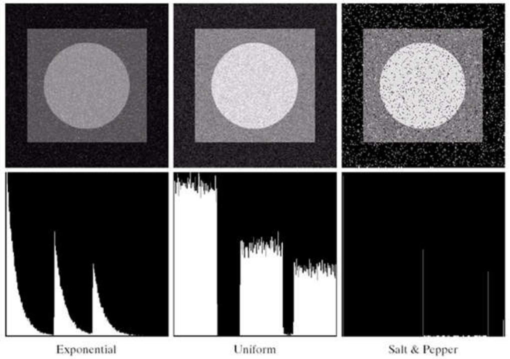
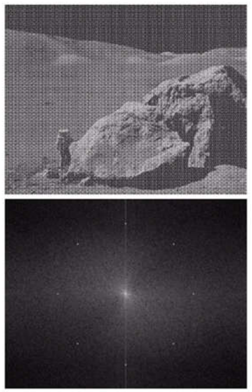


FIGURE 5.4 *(Continued)* Images and histograms resulting from adding exponential, uniform, and impulse noise to the image in Fig. 5.3.

Example: Image corrupted by Periodic Noise

FIGURE 5.5 (a) Image corrupted by sinusoidal noise. (b) Spectrum (each pair of conjugate impulses corresponds to one sine wave). (Original image courtesy of NASA.)



Estimating Noise Parameters

- The parameters of periodic noise typically are estimated by inspection of Fourier spectrum of the image. The frequency spikes can be detected by visual analysis.
- Another approach is to attempt to infer the periodicity of noise components directly from the image, but this is possible only in simplistic cases.
- Automated analysis is possible in situations in which the noise spikes are either exceptionally pronounced, or when some knowledge is available about the general location of the frequency components of the interference.
- The parameters of noise PDFs may be known partially from sensor specifications, but it is often necessary to estimate them for a particular imaging arrangement, e.g, a set of "flat" environments.
- When only images generated are available, it is frequently possible to estimate the parameters of the PDF from small patches of reasonably constant gray level.

Estimating Noise Parameters

- The simple way to use the data from the image strips is for calculating the mean and variance of the gray levels.
- The shape of the histogram identifies the closest PDF match. If the shape is approximately Gaussian, the mean and variance is all we need. For other shapes discussed in section 5.2.2, we use the mean and variance to solve for the parameters a and b.
- $\mu_n = \Sigma_{i=0}^{L-1}(z_i - m)^n p(z_i) \quad where \ \ m = \Sigma_{i=0}^{L-1} z_i p(z_i)$

when $n = 0$ $\quad \mu_0 = \Sigma_{i=0}^{L-1}(z_i - m)^0 p(z_i) = 1$

when $n = 1$ $\quad \mu_1 = \Sigma_{i=0}^{L-1}(z_i - m) p(z_i)$

$\quad = \Sigma_{i=0}^{L-1} z_i p(z_i) - m\Sigma_{i=0}^{L-1} p(z_i) = 0$

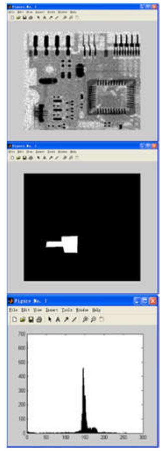
when $n = 2$ $\quad \mu_2 = \Sigma_{i=0}^{L-1}(z_i - m)^2 p(z_i) = Var(z)$

Estimating Noise Parameters

- Mean and central moments

  [v,unv]=statmoments(p,n)
- Select region of interest

  B=roipoly(f,c,r)
- f=imread('Fig0505(a)(ckt pepper only).tif');
- imshow(f)
- [B,c,r]=roipoly(f);
- Mouse select
- imshow(B)
- [p,npix]=histroi(f,c,r);
- bar(p,1)



Restoration in the Presence of Noise Only–Spatial Filtering

- If there is only noise, then

$$g(x,y) = f(x,y) + \eta(x,y)$$
$$G(u,v) = F(u,v) + N(u,v)$$

- Spatial Noise Filters
- Adaptive Spatial Filters

Mean Filters

- Arithmetic mean filter

$$\hat{f}(x,y) = \frac{1}{mn}\sum_{(s,t)\in S_{xy}} g(s,t)$$

- Harmonic mean filter

$$\hat{f}(x,y) = \frac{mn}{\Sigma_{(s,t)\in S_{xy}} \frac{1}{g(s,t)}}$$

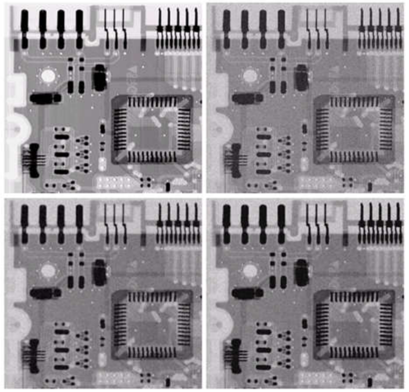
- Geometric mean filter

$$\hat{f}(x,y) = [\prod_{(s,t)\in S_{xy}} g(s,t)]^{\frac{1}{mn}}$$

- Contraharmonic mean filter

$$\hat{f}(x,y) = \frac{\Sigma_{(s,t)\in S_{xy}} g(s,t)^{Q+1}}{\Sigma_{(s,t)\in S_{xy}} g(s,t)^{Q}}$$

- The contraharmonic mean filter is well suited for eliminating the effects of salt-and pepper noise. For positive values of Q, the filter eliminates pepper noise. For negative values of Q, it eliminates salt noise. It cannot do both simultaneously.



a b
c d

FIGURE 5.7 (a) X-ray image. (b) Image corrupted by additive Gaussian noise. (c) Result of filtering with an arithmetic mean filter of size 3 × 3. (d) Result of filtering with a geometric mean filter of the same size. (Original image courtesy of Mr. Joseph E. Pascente, Lixi, Inc.)

Mean Filters(cont.)

- The harmonic mean filter works well for salt noise, but fails for peper noise. It does well for Gaussian noise

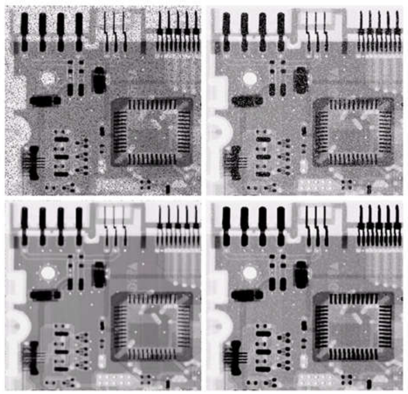
$$\hat{f}(x,y) = \frac{mn}{\Sigma_{(s,t)\in S_{xy}} \frac{1}{g(s,t)}}$$

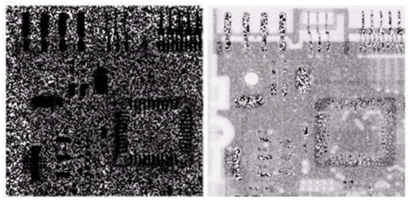
- The contraharmonic mean filter can effectively eliminate salt noise when Q is a negtive value, and works well for pepper noise when Q is a positive value. But it cannot do both simulteneously

$$\hat{f}(x,y) = \frac{\Sigma_{(s,t)\in S_{xy}} g(s,t)^{Q+1}}{\Sigma_{(s,t)\in S_{xy}} g(s,t)^{Q}}$$

a b
c d

FIGURE 5.8 (a) Image corrupted by pepper noise with a probability of 0.1. (b) Image corrupted by salt noise with the same probability. (c) Result of filtering (a) with a 3 × 3 contraharmonic filter of order 1.5. (d) Result of filtering (b) with $Q = -1.5$.





a b

FIGURE 5.9 Results of selecting the wrong sign in contraharmonic filtering. (a) Result of filtering Fig. 5.8(a) with a contraharmonic filter of size 3 × 3 and $Q = -1.5$. (b) Result of filtering 5.8(b) with $Q = 1.5$.

Order-statistics Filters

- Median filter

$$\hat{f}(x,y) = \underset{(s,t)\in S_{xy}}{median}\, g(s,t)$$

- Midpoint filter

$$\hat{f}(x,y) = \frac{1}{2}\left[\max_{(s,t)\in S_{xy}} g(s,t) + \min_{(s,t)\in S_{xy}} g(s,t)\right]$$

- Max and min filter

$$\hat{f}(x,y) = \max_{(s,t)\in S_{xy}} g(s,t)$$

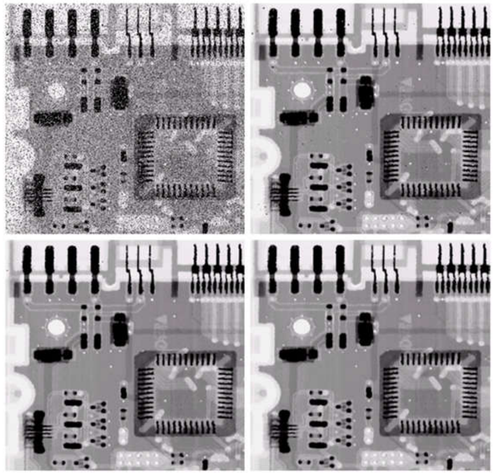
$$\hat{f}(x,y) = \min_{(s,t)\in S_{xy}} g(s,t)$$

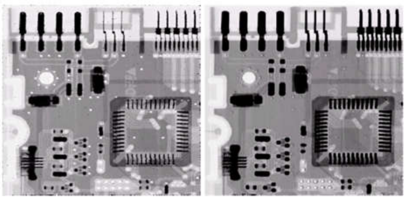
- Alpha-trimmed mean filter

$$\hat{f}(x,y) = \frac{1}{mn-d} \sum_{(s,t)\in S_{xy}} g_r(s,t)$$

a b
c d

FIGURE 5.10 (a) Image corrupted by salt-and-pepper noise with probabilities $P_a = P_b = 0.1$. (b) Result of one pass with a median filter of size 3 × 3. (c) Result of processing (b) with this filter. (d) Result of processing (c) with the same filter.





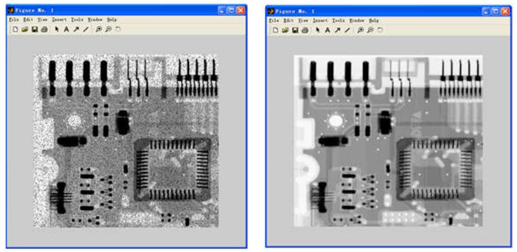
a b

FIGURE 5.11 (a) Result of filtering Fig. 5.8(a) with a max filter of size 3 × 3. (b) Result of filtering 5.8(b) with a min filter of the same size.

Spatial Noise Filters

%Filter pepper noise is to use contraharmonic filter with a positive value

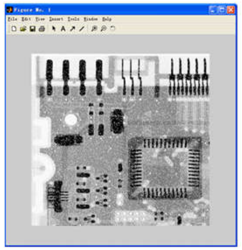
- gp=imread('Fig0505(a)(ckt pepper only).tif');
  %corrupted by pepper noise only with propability 0.1
- fp=spfilt(gp,'chmean',3,3,1.5);
- imshow(gp),figure,imshow(fp)



Spatial Noise Filters(cont.)

%Filter salt noise is to use contraharmonic filter with a negative value

- gs=imread('Fig0505(b)(ckt salt only).tif');
- fs=spfilt(gs,'chmean',3,3,-1.5);
- imshow(gs),figure,imshow(fs)

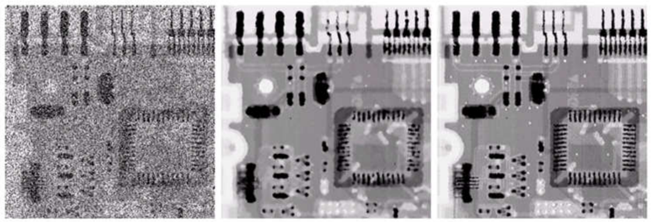



Adaptive Spatial Median Filters

- *Notations*:
  Z_{min} = *minimum intensity value in* S_{xy}
  Z_{max} = *maximum intensity value in* S_{xy}
  Z_{med} = *median intensity value in* S_{xy}
  Z_{xy} = *intensity value at coordinates* x, y
- Algorithm (adaptive median algorithm):
  - Level A:
    If $Z_{min} < Z_{med} < Z_{max}$, go to
    level B else increase the window size
    if window size $\leq S_{max}$, repeat level A
  - else output Z_{med}
    Level B:
    If $Z_{min} < Z_{xy} < Z_{max}$, output Z_{xy}
    else output Z_{med}

Adaptive Spatial Median Filters(cont.)

- f=adpmedian(g,Smax)

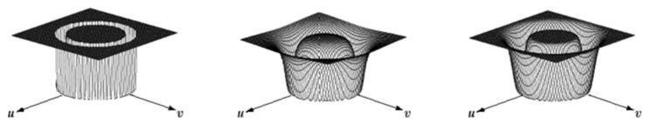


a b c

FIGURE 5.14 (a) Image corrupted by salt-and-pepper noise with probabilities $P_a = P_b = 0.25$. (b) Result of filtering with a 7 × 7 median filter. (c) Result of adaptive median filtering with $S_{max} = 7$.

Periodic Noise Reduction by Frequency Domain Filtering

- Bandreject Filters



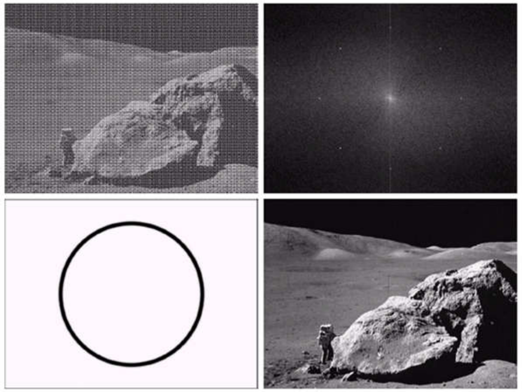
a b c

FIGURE 5.15 From left to right, perspective plots of ideal, Butterworth (of order 1), and Gaussian bandreject filters.

- $H(u,v) = \begin{cases} 1 & D(u,v) < D_0 - \frac{W}{2} \\ 0 & D_0 - \frac{W}{2} < D(u,v) < D_0 + \frac{W}{2} \\ 1 & D(u,v) > D_0 + \frac{W}{2} \end{cases}$
- $H(u,v) = \dfrac{1}{1 + \left[\dfrac{D(u,v)W}{D^2(u,v) - D_0^2}\right]^{2n}}$
- $H(u,v) = 1 - \exp[-\frac{1}{2}(\frac{D^2(u,v) - D_0^2}{D(u,v)W})^2]$

Periodic Noise Reduction by Frequency Domain Filtering

- Application of Bandreject Filters



a b
c d

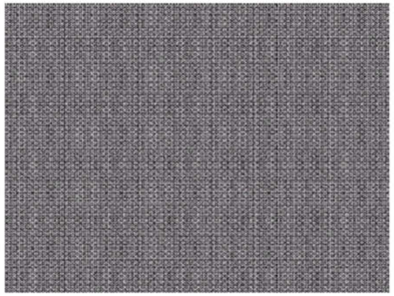
FIGURE 5.16 (a) Image corrupted by sinusoidal noise. (b) Spectrum of (a). (c) Butterworth bandreject filter (white represents 1). (d) Result of filtering. (Original image courtesy of NASA.)

Periodic Noise Reduction by Frequency Domain Filtering

- Bandpass Filters

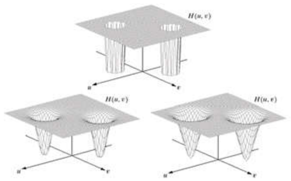
$$H_{bp}(u,v) = 1 - H_{br}(u,v)$$

FIGURE 5.17 Noise pattern of the image in Fig. 5.16(a) obtained by bandpass filtering.



Periodic Noise Reduction by Frequency Domain Filtering(3)

- Notch Filters



a
b c

FIGURE 5.18 Perspective plots of (a) ideal, (b) Butterworth (of order 2), and (c) Gaussian notch (reject) filters.

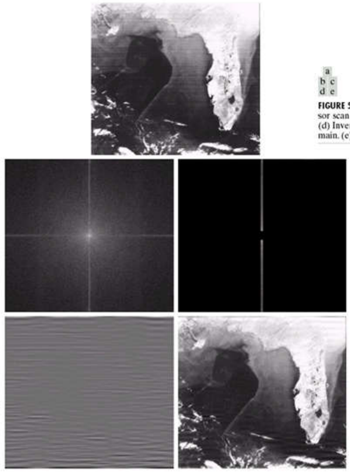
- $H(u,v) = \begin{cases} 0 & D_1(u,v) \le D_0 \ or \ D_2(u,v) \le D_0 \\ 1 & else \end{cases}$
- $H(u,v) = \dfrac{1}{1 + \left[\dfrac{D_0^2}{D_1(u,v)D_2(u,v)}\right]^n}$
- $H(u,v) = 1 - \exp[-\frac{1}{2}(\frac{D_1(u,v)D_2(u,v)}{D_0^2})]$

where

$$D_1(u,v) = [\left(u - \frac{M}{2} - u_0\right)^2 + \left(v - \frac{N}{2} - v_0\right)^2]^{0.5}$$

$$D_2(u,v) = [\left(u - \frac{M}{2} + u_0\right)^2 + \left(v - \frac{N}{2} + v_0\right)^2]^{0.5}$$

Removal of Periodic Noise by Notch Filters



a
b c
d e

FIGURE 5.19 (a) Satellite image of Florida and the Gulf of Mexico (note horizontal sensor scan lines). (b) Spectrum of (a). (c) Notch pass filter shown superimposed on (b). (d) Inverse Fourier transform of filtered image, showing noise pattern in the spatial domain. (e) Result of notch reject filtering. (Original image courtesy of NOAA.)

Optimum Notch Filtering

- When several interference components are present, the methods mentioned before are not always acceptable since too much image information is removed during the filtering.
- The method discussed here is optimum, in the sense that it minimizes the local variances of the restored estimate $\hat{f}(x,y)$.
- First, obtain an initial estimate of noises by

$$N(u,v) = F_N(u,v)G(u,v)$$
$$\eta(x,y) = \Im^{-1}(F_N(u,v)G(u,v))$$

where $F_N(u,v)$ is constructed to pass only the components associated with the interference pattern.

- Let

$$\hat{f}(x,y) = g(x,y) - w(x,y)\eta(x,y)$$

We will determine the modulation function $w(x,y)$, to minimize the local variances of $\hat{f}(x,y)$.

Optimum Notch Filtering(Cont.)

- Objective: $\min \sigma^2 = \frac{1}{(2a+1)(2b+1)} \sum_{s=-a}^{a} \sum_{t=-b}^{b} [\hat{f}(x+s,y+t) - \bar{f}]^2$

where $\bar{f} = \frac{1}{(2a+1)(2b+1)} \sum_{s=-a}^{a} \sum_{t=-b}^{b} \hat{f}(x+s,y+t)$

- Further

$$\sigma^2 = \frac{1}{(2a+1)(2b+1)} \sum_{s=-a}^{a} \sum_{t=-b}^{b} ([g(x+s,y+t) - w(x+s,y+t)\cdot\eta(x+s,y+t)] - [\overline{g(x,y)} - \overline{w(x,y)\eta(x,y)}])^2$$

- Let $w(x+s, y+t) = w(x,y)$
- We have

$$\sigma^2 = \frac{1}{(2a+1)(2b+1)} \sum_{s=-a}^{a} \sum_{t=-b}^{b} ([g(x+s,y+t) - w(x,y)\cdot\eta(x+s,y+t)] - [\overline{g(x,y)} - w(x,y)\overline{\eta(x,y)}])^2$$

Optimum Notch Filtering(Cont.)

- To minimize σ^2, we solve:

$$\frac{\partial \sigma^2}{\partial w(x,y)} = 0$$

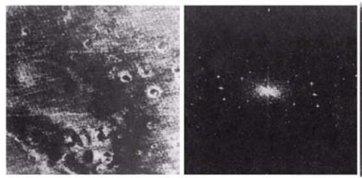
- The result is

$$w(x,y) = \frac{\overline{g(x,y)\eta(x,y)} - \bar{g}(x,y)\bar{\eta}(x,y)}{\overline{\eta^2}(x,y) - \bar{\eta}^2(x,y)}$$

Optimum Notch Filtering(Cont.)

a b

FIGURE 5.20 (a) Image of the Martian terrain taken by *Mariner 6*. (b) Fourier spectrum showing periodic interference. (Courtesy of NASA.)



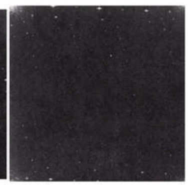
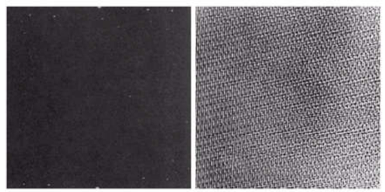


FIGURE 5.21 Fourier spectrum (without shifting) of the image shown in Fig. 5.20(a). (Courtesy of NASA.)



a b

FIGURE 5.22 (a) Fourier spectrum of $N(u,v)$, and (b) corresponding noise interference pattern $\eta(x,y)$. (Courtesy of NASA.)

:E 5.23 Processed image. (Courtesy of NASA.)

Estimating the Degradation Function

- Estimation by image observation

  Let $G_s(u,v)$ denote the observed subimage, and $\hat{F}_s(u,v)$ denotes the estimate of the original subimage, and assuming the noise is negligible because of our choice of a strong-signal area, we have

$$H_s(u,v) = \frac{G_s(u,v)}{\hat{F}_s(u,v)}$$

- Then we can deduce the complete function $H(u,v)$ from $H_s(u,v)$

  Estimation by experimentation

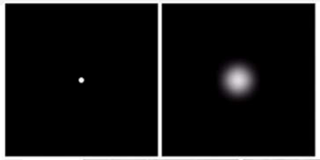
$$H(u,v) = \frac{G(u,v)}{A}$$

- Estimation by modeling

  A degradation model proposed by Hufnagel et al.[1964] is based on the phyiscal characteristics of atmosphere turbulence,

$$H(u,v) = \exp[-k(u^2+v^2)]^{5/6}]$$

Illustration of the Atmospheric Turbulence Model



a b

FIGURE 5.24 Degradation estimation by impulse characterization. (a) An impulse of light (shown magnified). (b) Imaged (degraded) impulse.

a b
c d

FIGURE 5.25 Illustration of the atmospheric turbulence model. (a) Negligible turbulence. (b) Severe turbulence, $k = 0.0025$. (c) Mild turbulence, $k = 0.001$. (d) Low turbulence, $k = 0.00025$. (Original image courtesy of NASA.)

Image Blur due to Motion

- *Now, assume an image has been blurred by uniform linear motion.*

$$g(x,y) = \int_0^T f(x - x_0(t), y - y_0(t))\,dt$$

- *Then, its Fourier transform is*

$$G(u,v) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(x,y)\exp\big(-j2\pi(ux+vy)\big)\,dxdy$$

$$= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\left[\int_0^T f(x-x_0(t), y-y_0(t))dt\right]\exp(-j2\pi(ux+vy))dxdy$$

$$= \int_0^T [\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f\big(x-x_0(t), y-y_0(t)\big)\exp(-j2\pi(ux+vy))dxdy]\,dt$$

Image Blur due to Motion(Cont.)

$$= \int_0^T F(u,v)\exp(-j2\pi(ux_0(t)+vy_0(t)))dt$$

$$= F(u,v)\int_0^T \exp(-j2\pi(ux_0(t)+vy_0(t)))dt$$

- *Thus,*

$$H(u,v) = \int_0^T \exp(-j2\pi(ux_0(t)+vy_0(t)))dt$$

- if $x_0(t) = at/T$, $y_0(t) = 0$,then

$$H(u,v) = \int_0^T \exp(-j2\pi uat/T)dt = \frac{T}{\pi ua}\sin(\pi ua)\exp(-j\pi ua)$$

- if $y_0(t) = bt/T$ instead of 0, then

$$H(u,v) = \frac{T}{\pi(ua+vb)}\sin(\pi(ua+vb))\exp(-j\pi(ua+vb))$$

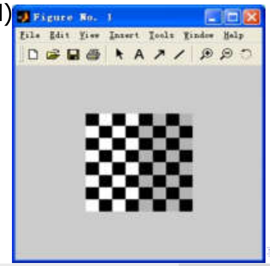
Using matlab to Model Image Blur due to Motion

Image blur due to motion can be modeled by the function fspecial in IPT PSF = fspecial('motion',len,theta)

- g = imfilter(f,PSF, 'circular');
- g = g+noise;

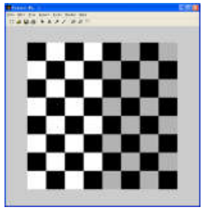
The test pattern generated by checkerboard is very useful, and its syntax is

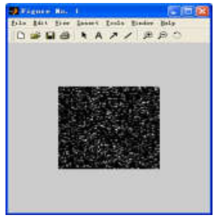
- C = checkerboard(NP,M,N)
- f=checkerboard(8);
  imshow(f)

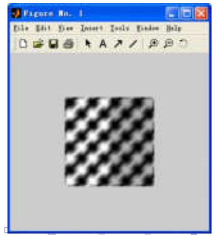


Using matlab to Model Image Blur due to Motion(cont.)

- PSF=fspecial('motion',7,45);
- gb=imfilter(f,PSF,'circular');
- noise=imnoise(zeros(size(f)),'gaussian',0,0.001);
- g=gb+noise;
- imshow(pixeldup(f,8),[])
- imshow(gb); imshow(noise,[])







Direct inverse filtering

- For an image degraded by a degradation function H, the simplest approach to restoration is direct inverse filtering
  $$\hat{F}(u,v) = \frac{G(u,v)}{H(u,v)}$$
- Further, we can obtain
  $$\hat{F}(u,v) = F(u,v) + \frac{N(u,v)}{H(u,v)}$$
- If the degradation has zero or very small values, then the ratio $N(u,v)/H(u,v)$ could easily dominate the estimation of $F(u,v)$. One way to overcome this is to limit the filter frequency to values near the origin.

Direct inverse filtering (cont.)

a b
c d

FIGURE 5.27 Restoring Fig. 5.25(b) with Eq. (5.7-1). (a) Result of using the full filter. (b) Result with H cut off outside a radius of 40; (c) outside a radius of 70; and (d) outside a radius of 85.

Wiener Filtering

- A wiener filter seeks an estimate that minimizes the statistical error function

$$e^2 = E\{(f-\hat{f})^2\}$$

- E is the expected value operator and f is the undergraded image.The solution to this expression in the frequency domain is

$$\hat{F}(u,v) = \left[\frac{1}{H(u,v)}\frac{|H(u,v)|^2}{|H(u,v)|^2 + S_\eta(u,v)/S_f(u,v)}\right]G(u,v)$$

where $H(u,v)$ = the degradation function

$|H(u,v)|^2 = H^*(u,v)H(u,v)$

$H^*(u,v)$ = the complex conjugate of $H(u,v)$

$S_\eta(u,v) = |N(u,v)|^2$ = the power spectrum of the noise

$S_f(u,v) = |F(u,v)|^2$ = the power spectrum of the undegraded image

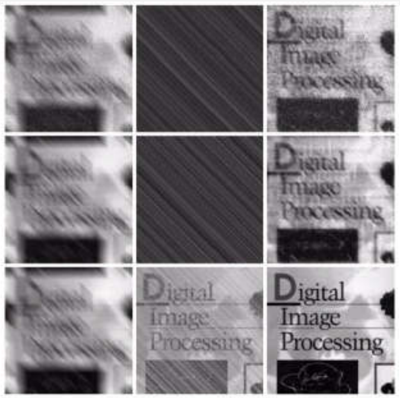
Wiener Filtering(cont.)

$$\hat{F}(u,v) = [\frac{1}{H(u,v)}\frac{|H(u,v)|^2}{|H(u,v)|^2 + K}]G(u,v)$$

a b c

FIGURE 5.28 Comparison of inverse- and Wiener filtering. (a) Result of full inverse filtering of Fig. 5.25(b). (b) Radially limited inverse filter result. (c) Wiener filter result.

Wiener Filtering(cont.)



a b c
d e f
g h i

FIGURE 5.29 (a) Image corrupted by motion blur and additive noise. (b) Result of inverse filtering. (c) Result of Wiener filtering. (d)–(f) Same sequence, but with noise variance one order of magnitude less. (g)–(i) Same sequence, but noise variance reduced by five orders of magnitude from (a). Note in (h) how the deblurred image is quite visible through a "curtain" of noise.

Wiener Filtering in Matlab(cont.)

- fr=deconvwnr(g,PSF)
- fr=deconvwnr(g,PSF,NSPR)
- fr=deconvwnr(g,PSF,NACORR,FACORR)

$$|F(u,v)|^2 = \Im[f(x,y)\circ f(x,y)]$$

- If the restored image exhibits ringing, it sometimes helps to use function edgetaper prior to calling deconvwnr

```
J=edgetaper(I,PSF)
```

Wiener filtering(cont.)

- PSF=fspecial('motion',7,45);
- g=imread('Fig0507(d).tif');
- fr1=deconvwnr(g,PSF);
- Sn=abs(fft2(noise)).^2;
- nA=sum(Sn(:))/prod(size(noise));
- Sf=abs(fft2(f)).^2;
- fA=sum(Sf(:))/prod(size(f));
- R=nA/fA; fr2=deconvwnr(g,PSF,R);
- NCORR=fftshift(real(ifft2(Sn)));
- ICORR=fftshift(real(ifft2(Sf)));
- fr3=deconvwnr(g,PSF,NCORR,ICORR);

Constrained Least Squares Filtering

- The problem of having to know something about the degradation function H is common to all methods discussed in this chapter.
- The Wiener filtering method presents additional difficulties: the power spectra of undegraded image and noise must be known. A constant estimate of the ratio of the power spectra can achieve excellent results sometimes, but does not mean it is always a suitable solution.
- The constrained least squares filtering method only requires knowledge of the mean and variance of noise.
- If we use the matrix to model the degradation procedure, we have

$$g(x,y) = h(x,y) * f(x,y) + \eta(x,y)$$

$$g = Hf + \eta$$

Constrained Least Squares Filtering(Cont.)

- Central to the method is the issue of sensitivity of H to noise. One way to alleviate the problem is to base the restoration on a measure of smoothness.
- The Laplacian (the second derivative of an image) seems to be a good choice.

$$minC = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1}[\nabla^2 f(x,y)]^2$$

Subject to: $\| g - H\hat{f} \| = \|\eta\|$

- The frequency solution of the problem is

$$\hat{F}(u,v) = \left[\frac{H^*(u,v)}{|H(u,v)|^2 + \gamma|P(u,v)|^2}\right]G(u,v)$$

and

$$p(x,y) = \begin{pmatrix} 0, & -1, & 0 \\ -1, & 4, & -1 \\ 0, & -1, & 0 \end{pmatrix}$$

Constrained Least Squares Filtering(Cont.)

- Let $r = g - H\hat{f}$ and $\varphi(r) = r^T r = ||r||^2$
- It can be proved that $\varphi(r)$is a monotonically increasing function of γ So we can adjust γ, so that:

$$||r||^2 = ||\eta||^2 \pm a$$

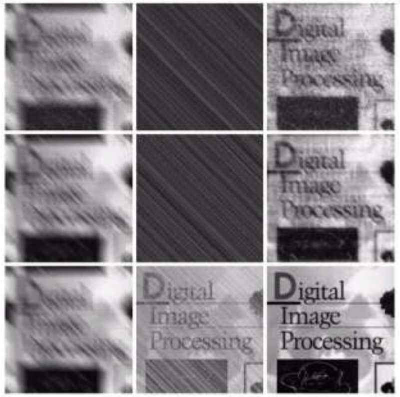
- How to evaluate the $||\eta||^2$ (???)

$$||\eta||^2 = MN[\sigma_\eta^2 + m_\eta^2]$$

$$\sigma_\eta^2 = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}[\eta(x,y) - m_\eta]^2$$

$$m_\eta = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}\eta(x,y)$$

Constrained Least Squares Filtering(Cont.)



a b c
d e f
g h i

FIGURE 5.29 (a) Image corrupted by motion blur and additive noise. (b) Result of inverse filtering. (c) Result of Wiener filtering. (d)–(f) Same sequence, but with noise variance one order of magnitude less. (g)–(i) Same sequence, but noise variance reduced by five orders of magnitude from (a). Note in (h) how the deblurred image is quite visible through a "curtain" of noise.

Constrained Least Squares Filtering(Cont.)

a b c

FIGURE 5.30 Results of constrained least squares filtering. Compare (a), (b), and (c) with the Wiener filtering results in Figs. 5.29(c), (f), and (i), respectively.