Image Processing and Analysis

Lecture 5、Image Enhencement in Frequency Domain(I)

Weiqiang Wang
School of Computer Science and Technology, UCAS

October 18, 2023

Outline

2-D Fourier Transform

- Any function that periodically repeats itself can be expressed as the sum of sines and/or cosines of different frequencies, each multiplied by a different coefficient (Fourier series).
- Even functions that are not periodic (but whose area under the curve is finite) can be expressed as the integral of sines and/or cosines multiplied by a weighting function (Fourier transform).
- The frequency domain refers to the plane of the two dimensional discrete Fourier transform of an image.
- The purpose of the Fourier transform is to represent a signal as a linear combination of sinusoidal signals of various frequencies.

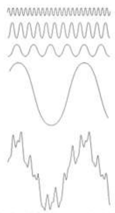


FIGURE 4.1 The function at the bottom is the sum of the four functions above it. Fourier's idea in 1807 that periodic functions could be represented as a weighted sum of sines and cosines was met with skepticism.

2-D Continuous Fourier Transform

- The one-dimensional Fourier transform and its inverse
  - Fourier transform
    $F(u) = \int_{-\infty}^{\infty} f(x) e^{-j2\pi ux} dx, where\ j = \sqrt{-1}$
  - Inverse Fourier transform:
    $f(x) = \int_{-\infty}^{\infty} F(u) e^{j2\pi ux}\, du \qquad e^{j\theta} = \cos\theta + j\sin\theta$
- The two-dimensional Fourier transform and its inverse
  - Fourier transform
    $F(u,v) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y) e^{-j2\pi(ux+vy)} dxdy$
  - Inverse Fourier transform:
    $f(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} F(u,v) e^{j2\pi(ux+vy)} dudv$

2-D Discrete Fourier Transform

- The one-dimensional Discrete Fourier transform (DFT) and its inverse
  - Fourier transform

$$F(u) = \frac{1}{M}\sum_{x=0}^{M-1} f(x)\, e^{-\frac{j2\pi ux}{M}} \quad for\ u = 0,1,2,\dots,M-1$$

  - Inverse Fourier transform:

$$f(x) = \sum_{u=0}^{M-1} F(u)\, e^{\frac{j2\pi ux}{M}} \quad for\ x = 0,1,2,\dots,M-1$$

- Since $e^{j\theta} = \cos\theta + j\sin\theta$, then DFT can be redefined as

$$F(u) = \frac{1}{M}\sum_{x=0}^{M-1} f(x)\left[cos\frac{2\pi ux}{M} - jsin\frac{2\pi ux}{M}\right]$$
$$for\ u = 0,1,2,\dots,M-1$$

  - Frequency (time) domain: the domain (values of u) over which the values of $F(u)$ range; because u determines the frequency of the components of the transform.
  - Frequency (time) component: each of the M terms of $F(u)$.

2-D Discrete Fourier Transform

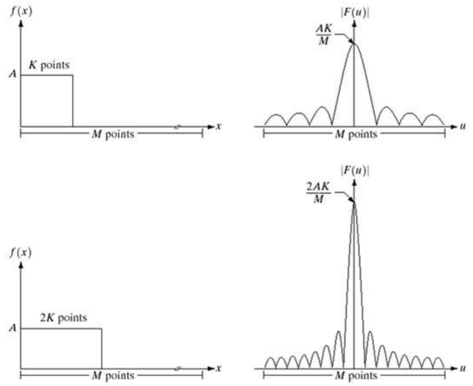
- $F(u)$ can be expressed in polar coordinates:

$$F(u) = |F(u)|e^{j\phi(u)}$$
$$where\ |F(u)| = [R(u)^2 + I(u)^2]^{\frac{1}{2}} \quad (magnitude\ or\ spectrum)$$
$$\phi(u) = tan^{-1}\left[\frac{I(u)}{R(u)}\right] \quad (phase\ angle\ or\ phase\ spectrum)$$

  - $I(u)$:the imaginary part of $F(u)$.
  - $R(u)$:the real part of $F(u)$.
- Power spectrum

$$P(u) = |F(u)|^2 = R^2(u) + I^2(u)$$

2-D Discrete Fourier Transform



a b
c d

FIGURE 4.2 (a) A discrete function of M points, and (b) its Fourier spectrum. (c) A discrete function with twice the number of nonzero points, and (d) its Fourier spectrum.

2-D Discrete Fourier Transform

- The two-dimensional Fourier transform and its inverse
  - Fourier transform (discrete case)DFT

$$F(u,v) = \frac{1}{MN}\Sigma_{x=0}^{M-1}\Sigma_{y=0}^{N-1} f(x,y)e^{-j2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)}$$
$$for\ u = 0,1,2,\dots,M-1, v = 0,1,2,\dots,N-1$$

  - Inverse Fourier transform:

$$f(x,y) = \Sigma_{u=0}^{M-1}\Sigma_{v=0}^{N-1} F(u,v)e^{j2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)}$$
$$for\ x = 0,1,2,\dots,M-1, y = 0,1,2,\dots,N-1$$

    - u,v: the transform or frequency variables
    - x,y: the spatial or image variables

2-D Discrete Fourier Transform

- We define the Fourier spectrum, phase angle, and power spectrum as follows:

$$|F(u,v)| = [R^2(u,v) + I^2(u,v)]^{\frac{1}{2}} \qquad (spectrum)$$

$$\phi(u,v) = tan^{-1}[\frac{I(u,v)}{R(u,v)}] \qquad (phase\ angle)$$

$$P(u,v) = |F(u,v)|^2 = R^2(u,v) + I^2(u,v) \qquad (power\ spectrum)$$

  - $I(u,v)$: the imaginary part of $F(u,v)$.
  - $R(u,v)$: the real part of $F(u,v)$.

Properties of 2-D DFT

- Time-shifting

$$\Im[f(x-x_0, y-y_0)] = F(u,v)e^{-j2\pi(\frac{ux_0}{M}+\frac{vy_0}{N})}$$

- Frequency shifting

$$\Im[f(x,y)e^{-j2\pi(\frac{u_0x}{M}+\frac{v_0y}{N})}] = F(u-u_0, v-v_0)$$

$$\Im[f(x,y)(-1)^{x+y}] = F(u-\frac{M}{2}, v-\frac{N}{2})$$

- Average and Symmetry

$$F(0,0) = \frac{1}{MN}\Sigma_{x=0}^{M-1}\Sigma_{y=0}^{N-1}f(x,y) \qquad (average)$$

$$F(u,v) = F^*(-u,-v) \qquad (conjugate\ symmetric)$$

$$|F(u,v)| = |F(-u,-v)| \qquad (symmetric)$$

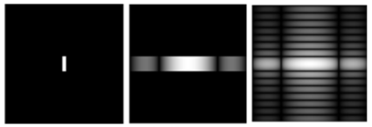
Properties of 2-D DFT (cont.)

- Separability

$$\begin{aligned} F(u,v) &= \Im[f(x,y)] \\ &= \Sigma_y[\frac{1}{M}\Sigma_x f(x,y)\exp\left(-j2\pi\frac{xu}{M}\right)]\exp(-j2\pi\frac{yv}{N}) \\ &= \frac{1}{N}\Sigma_y F(u,y)\exp\left(-j2\pi\frac{yv}{N}\right) \end{aligned}$$

The 2D DFT $F(u,v)$ can be obtained by

1. Taking the 1D DFT of every row of image $f(x,y)$,$F(u,y)$
2. The 1D DFT of every column of $F(u,y)$



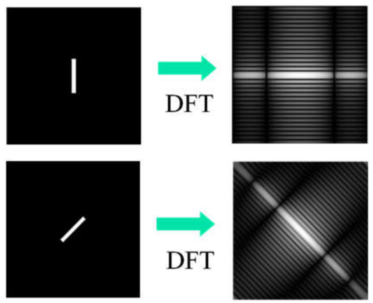
(a)f(x,y) (b) F(u,y) (c)F(u,v)

Properties of 2-D DFT (cont.)

- Rotation

let $x = r\cos\theta,\ y = r\sin\theta,\ u = \omega\cos\varphi,\ v = \omega\sin\varphi$

$$f(r, \theta + \theta_0) \Leftrightarrow F(\omega, \varphi + \theta_0)$$



Properties of 2-D DFT (cont.)

- Periodicity

$$f(x, y) = f(x + M, y) = f(x, y + N) = f(x + M, y + N)$$

$$F(u, v) = F(u + M, v) = F(u, v + N) = F(u + M, v + N)$$

- Linearity

$$\Im(af(x,y) + bg(x,y)) = a\Im(f(x,y)) + b\Im(g(x,y))$$

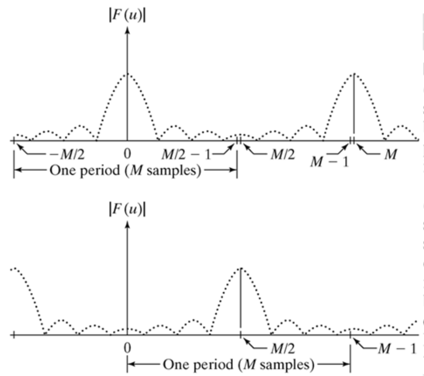
- Differentiation

$$\Im\left(\frac{\partial^n f(x,y)}{\partial x^n}\right) = (j2\pi u)^n \Im(f(x,y)) = (j2\pi u)^n F(u,v)$$

$$\Im((-j2\pi u)^n f(x,y)) = \frac{\partial^n F(u,v)}{\partial u^n}$$

$$\Im(\nabla^2 f(x,y)) = -4\pi^2(u^2+v^2)F(u,v)$$

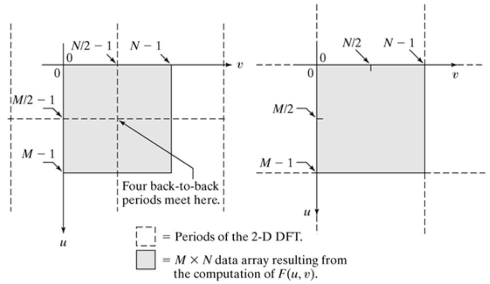
2-D Discrete Fourier Transform



a
b

FIGURE 4.1 (a) Fourier spectrum showing back-to-back half periods in the interval $[0, M-1]$. (b) Centered spectrum in the same interval, obtained by multiplying $f(x)$ by $(-1)^x$ prior to computing the Fourier transform.

2-D Discrete Fourier Transform (cont.)



a b

FIGURE 4.2 (a) $M \times N$ Fourier spectrum (shaded), showing four back-to-back quarter periods contained in the spectrum data. (b) Spectrum obtained by multiplying $f(x, y)$ by $(-1)^{x+y}$ prior to computing the Fourier transform. Only one period is shown shaded because this is the data that would be obtained by an implementation of the equation for $F(u, v)$.

Properties of 2-D DFT (cont.)

- Convolution

$$\Im(f(x,y) * g(x,y)) = F(u,v)G(u,v)$$

$$\Im\left(f(x,y)g(x,y)\right) = F(u,v) * G(u,v)$$

- Correlation

$$\Im\left(f(x,y) \circ g(x,y)\right) = F^*(u,v)G(u,v)$$

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

$$\Im\left(f^*(x,y)g(x,y)\right) = F(u,v) \circ G(u,v)$$

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

- Similarity

$$\Im\left(f(ax,by)\right) = \frac{1}{|ab|}F\left(\frac{u}{a},\frac{v}{b}\right)$$

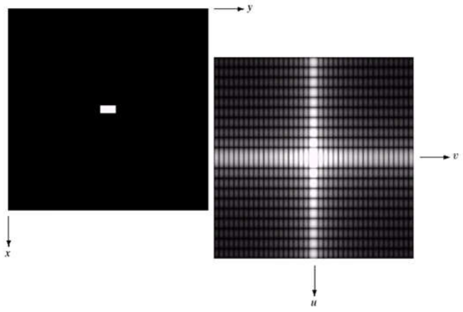
Some useful FT pairs

- $\delta(x,y) \;\Leftrightarrow\; 1$
- $A2\pi\sigma^2 exp(-2\pi^2\sigma^2(x^2+y^2)) \;\Leftrightarrow\; Aexp(-\dfrac{(u^2+v^2)}{2\sigma^2})$
- $\cos(2\pi u_0 x \;+\; 2\pi v_0 y) \;\Leftrightarrow\; \dfrac{1}{2}[\delta(u+u_0, v+v_0)+\delta(u-u_0, v-v_0)]$
- $sin(2\pi u_0 x \;+\; 2\pi v_0 y) \;\Leftrightarrow\; \dfrac{1}{2}j\;[\delta(u+u_0, v+v_0)-\delta(u-u_0, v-v_0)]$

2-D Discrete Fourier Transform

a b

FIGURE 4.3 (a) Image of a 20×40 white rectangle on a black background of size 512×512 pixels. (b) Centered Fourier spectrum shown after application of the log transformation given in Eq. (3.2-2). Compare with Fig. 4.2.



2-D DFT in Matlab

- The DFT and its inverse are obtained in practice using a Fast Fourier Transform(FFT) algorithm. The FFT of an $M \times N$ image array f is obtained in the toolbox with function fft2,which has the simple syntax:

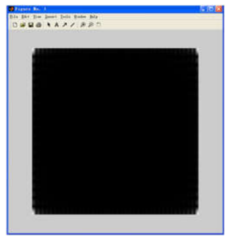
  F=fft2(f)

  This function returns a Fourier transform that is also of size $M \times N$, *with the origin of the data at the top left, and with four quarter periods meeting at the center of the frequency rectangle.*
- The Fourier spectrum is obtained by using function abs:

  S=abs(F)

2-D DFT in Matlab(cont.)

- f=imread(‘Fig0403(a)(image).tif’);
- F=fft2(f);
- S=abs(F);
- imshow(S,[])

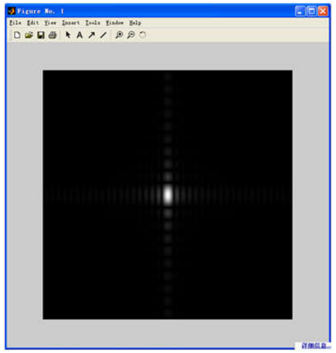


2-D DFT in Matlab(cont.)

- Fc=fftshift(F);
- imshow(abs(Fc),[])

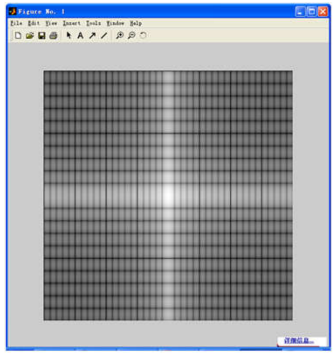
  The net result of using fftshift is the same as if the input image had been multiplied by $(-1)^{x+y}$ *prior to computing the transform.*

  Note,however,that the two processes are not interchangeable.

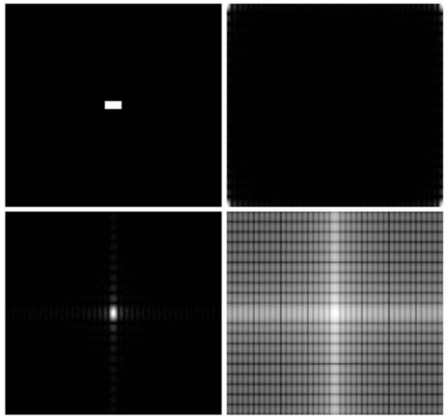


2-D DFT in Matlab(cont.)

- S2=log(1+abs(Fc));
- imshow(S2,[])



2-D DFT in Matlab(cont.)



a b
c d

FIGURE 4.3 (a) A simple image. (b) Fourier spectrum. (c) Centered spectrum. (d) Spectrum visually enhanced by a log transformation.

2-D DFT in Matlab(cont.)

- We point out that the inverse Fourier transform is computed using function if ifft2. which has the basic syntax

  f=ifft2(F)

- In practice, the output of ifft2 often has very small imaginary components resulting from round-off errors. Thus, it is good practice to extract the real part of the result.

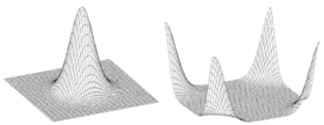
  f=real(ifft2(F))

Fundamental Concepts

- The foundation for linear filtering in both the spatial and frequency domains is the convolution theorem, which may be written as.

$$f(x,y) * h(x,y) \Leftrightarrow F(u,v)H(u,v)$$

- The idea in frequency domain filtering is to select a filter transfer function that modifies $F(u,v)$ in a specified manner.
- For example, the lowpass filter in figure 4.4



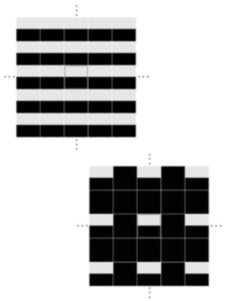
a b
FIGURE 4.4 Transfer functions of (a) a centered lowpass filter, and (b) the format used for DFT filtering. Note that these are frequency domain filters.

Fundamental Concepts(cont.)

- Based on the convolution theorem, we know that to obtain the corresponding filtered image in the spatial domain we simply compute the inverse Fourier transform of the product $H(u,v)F(u,v)$.
- Convolving periodic functions can cause interference of the nonzero periods if the periods are close with respect to the duration of the nonzero parts of the functions. This interference, called *wraparound error*, can be avoided by padding the functions with zeros.
- For example, the lowpass filter in figure 4.4

a b c
FIGURE 4.5 (a) A simple image of size 256 × 256. (b) Image lowpass-filtered in the frequency domain without padding. (c) Image lowpass-filtered in the frequency domain with padding. Compare the light portion of the vertical edges in (b) and (c).

Fundamental Concepts(cont.)

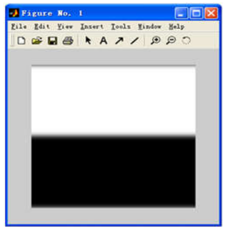


a b
FIGURE 4.6 (a) Implied, infinite periodic sequence of the image in Fig. 4.5(a). The dashed region represents the data processed by fft2. (b) The same periodic sequence after padding with 0s. The thin white lines in both images are shown for convenience in viewing; they are not part of the data.

FIGURE 4.7 Full padded image resulting from ifft2 after filtering. This image is of size 512 × 512 pixels.

Fundamental Concepts(cont.)

- Image lowpass-filtered in the frequency domain without padding
  - f=imread('Fig0405(a)(square original).tif');
  - [m n]=size(f)
  - F=fft2(f);
  - H=lpfilter('gaussian',m,n,10);
  - G=H.*F;
  - g=real(ifft2(G))
  - imshow(g,[])



Basic Steps in DFT Filtering

- 1. Obtain the padding parameters using function paddedsize:
  PQ=paddedsize(size(f));
- 2. Obtain the Fourier transform with padding:
  F=fft2(f,PQ(1),PQ(2));
- 3. Generate a filter function, H, of size PQ(1)×PQ(2) using any of the methods discussed in the remainder of this chapter.
  The filter must be in the format shown in Fig. 4.4(b). If it is centered instead, as in Fig. 4.4(a),let H = ifftshift (H) before using the filter.
- 4. Multiply the transform by the filter:G = H.*F;
- 5. Obtain the real part of the inverse FFT of G g=real(ifft2(G));
- 6. Crop the top, left rectangle to the original size:
  g=g(1:size(f,1),1:size(f,2));

Basic Steps in DFT Filtering(cont.)

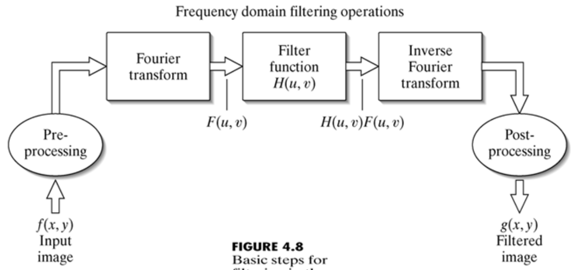


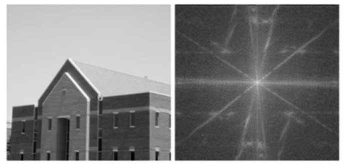
FIGURE 4.8 Basic steps for filtering in the frequency domain.

Obtaining Frequency Domain Filters from Spatial Filters

- Why obtains Frequency Domain Filters from Spatial Filters?
  - Efficiency
  - Meaningful comparisons
- How?
  - How to convert spatial filters into equivalent frequency domain filters
  - How to compare the results between spatial domain filtering using imfilter, and frequency domain filtering using freqz2

Obtaining Frequency Domain Filters from Spatial Filters(cont.)

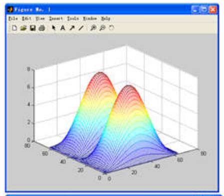
- >>f=imread('Fig0409(a)(bld).tif');
- >>F=fft2(f);
- >>S=fftshift(log(1+abs(F)));
- >>S=gscale(S);
- >>imshow(S)



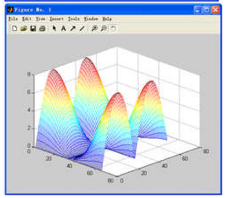
a b
FIGURE 4.9 (a) A gray-scale image. (b) Its Fourier spectrum.

Obtaining Frequency Domain Filters from Spatial Filters(cont.)

- h=fspecial(‘sobel’);
- H=freqz2(h);
- mesh(abs(H))

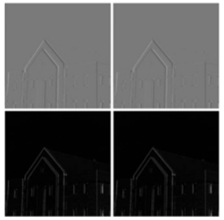


- H1=ifftshift(H);
- mesh(abs(H1))
- view(45,30)



Obtaining Frequency Domain Filters from Spatial Filters(cont.)

Spatial domain	Frequent domain
gs=imfilter(double(f),h);	PQ=paddedsize(size(f)); H=freqz2(h,PQ(1),PQ(2)); H1=ifftshift(H); gf=dftfilt(f,H1);



- d=abs(gs-gf);
- max(d(:))

  ans=

  5.4015e-012
- min(d(:))

  ans=

  0

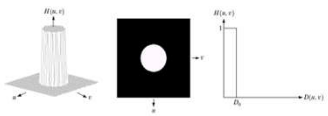
Generating Filters Directly in the Frequency Domain

- Ideal lowpass filter(ILPF)

$$H(u,v) = \begin{cases} 1 & if\ D(u,v) \le D_0 \\ 0 & if\ D(u,v) > D_0 \end{cases}$$

where $D(u,v)$: the distance from point (u,v) to the center of the frequency rectangle

$$D(u,v) = [(u-M/2)^2+(v-N/2)^2]^{\frac{1}{2}}$$



a b c

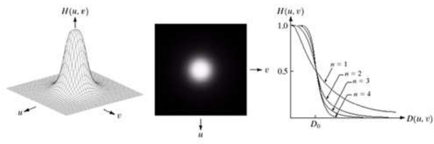
FIGURE 4.10 (a) Perspective plot of an ideal lowpass filter transfer function. (b) Filter displayed as an image. (c) Filter radial cross section.

- Create meshgrid arrays for using in implementing Filters in the frequency domain dftuv

Generating Filters Directly in the Frequency Domain(cont.)

- Butterworth Lowpass Filters (BLPFs) with order *n*

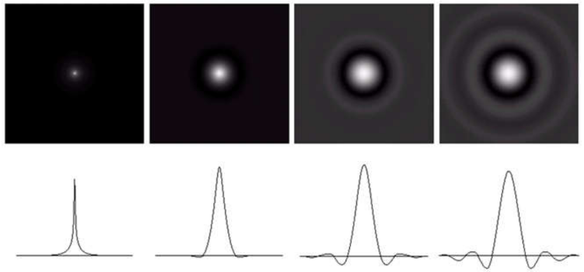
$$H(u,v) = \frac{1}{1+[D(u,v)/D_0]^{2n}}$$



a b c

FIGURE 4.14 (a) Perspective plot of a Butterworth lowpass filter transfer function. (b) Filter displayed as an image. (c) Filter radial cross sections of orders 1 through 4.

Generating Filters Directly in the Frequency Domain(cont.)

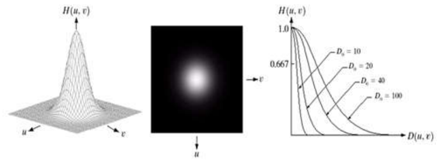


a b c d

FIGURE 4.16 (a)–(d) Spatial representation of BLPFs of order 1, 2, 5, and 20, and corresponding gray-level profiles through the center of the filters (all filters have a cutoff frequency of 5). Note that ringing increases as a function of filter order.

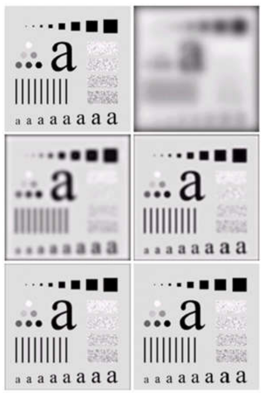
Gaussian Lowpass Filters (GLPFs)

- $H(u,v) = e^{-D^2(u,v)/2D_0^2}$



a b c

FIGURE 4.17 (a) Perspective plot of a GLPF transfer function. (b) Filter displayed as an image. (c) Filter radial cross sections for various values of D_0.



a b
c d
e f

FIGURE 4.15 (a) Original image. (b)–(f) Results of filtering with BLPFs of order 2, with cutoff frequencies at radii of 5, 15, 30, 80, and 230, as shown in Fig. 4.11(b). Compare with Fig. 4.12.

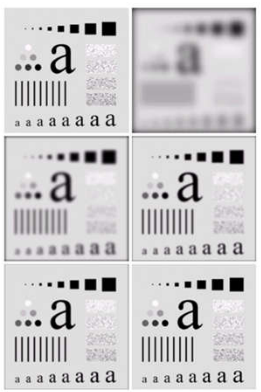


FIGURE 4.18 (a) Original image. (b)–(f) Results of filtering with Gaussian lowpass filters with cutoff frequencies set at radii values of 5, 15, 30, 80, and 230, as shown in Fig. 4.11(b). Compare with Figs. 4.12 and 4.15.

a b
c d
e f

Examples of Lowpass Filtering

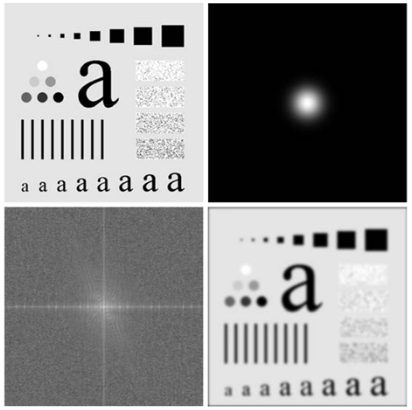
a b c

FIGURE 4.20 (a) Original image (1028 × 732 pixels). (b) Result of filtering with a GLPF with $D_0 = 100$. (c) Result of filtering with a GLPF with $D_0 = 80$. Note reduction in skin fine lines in the magnified sections of (b) and (c).

Generating Filters Directly in the Frequency Domain

- >>f=imread('Fig0413(a)(original_test_pattern).tif');
- >>PQ=paddedsize(size(f));
- >>[U V]=dftuv(PQ(1),PQ(2));
- >>D0=0.05*PQ(2);
- >>F=fft2(f,PQ(1),PQ(2));
- >>H=exp(-(U.^2+V.^2)/(2*(D0^2)));
- >>g=dftfilt(f,H);
- >>imshow(g,[]);

Generating Filters Directly in the Frequency Domain(cont.)



a b
c d

FIGURE 4.13 Lowpass filtering. (a) Original image. (b) Gaussian lowpass filter shown as an image. (c) Spectrum of (a). (d) Processed image.