EL5123 --- Image Processing

2-D Fourier Transforms

Yao Wang
Polytechnic University, Brooklyn, NY 11201

With contribution from Zhu Liu, Onur Guleryuz, and Gonzalez/Woods, Digital Image Processing, 2ed

Lecture Outline

- Continuous Fourier Transform (FT)
  - 1D FT (review)
  - 2D FT
- Fourier Transform for Discrete Time Sequence (DTFT)
  - 1D DTFT (review)
  - 2D DTFT
- Linear Convolution
  - 1D, Continuous vs. discrete signals (review)
  - 2D
- Filter Design
- Computer Implementation

What is a transform?

- Transforms are decompositions of a function f(x) into some basis functions Ø(x, u). u is typically the freq. index.

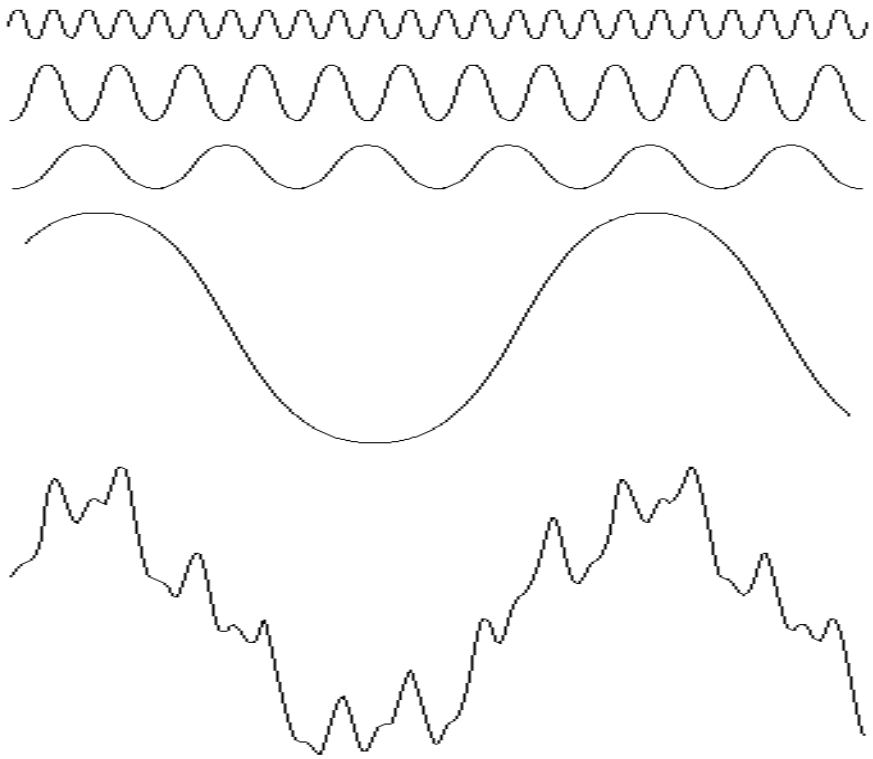
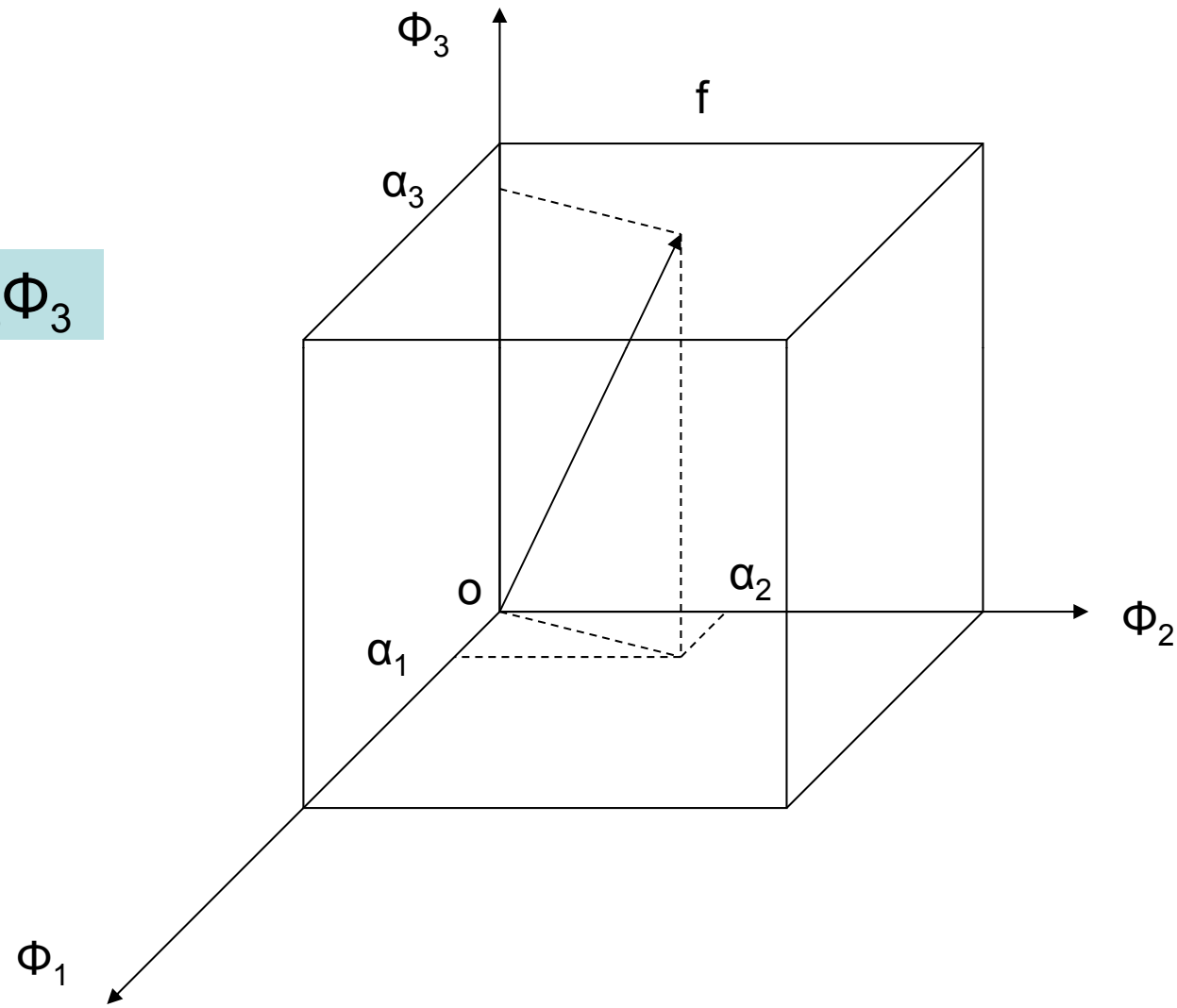


FIGURE 4.1 The function at the bottom is the sum of the four functions above it. Fourier's idea in 1807 that periodic functions could be represented as a weighted sum of sines and cosines was met with skepticism.

Illustration of Decomposition

$$f = \alpha_1\Phi_1+\alpha_2\Phi_2+\alpha_3\Phi_3$$



Decomposition

- Ortho-normal basis function

$$\int_{-\infty}^{\infty} \phi(x,u_1)\phi^*(x,u_2)dx = \begin{cases} 1, & u_1 = u_2 \\ 0, & u_1 \neq u_2 \end{cases}$$

- Forward

$$F(u) = < f(x), \phi(x,u) > = \int_{-\infty}^{\infty} f(x)\phi^*(x,u)dx$$

Projection of *f(x)* onto *ϕ(x,u)*

- Inverse

$$f(x) = \int_{-\infty}^{\infty} F(u)\phi(x,u)du$$

Representing *f(x)* as sum of *ϕ(x,u)* for all *u*, with weight *F(u)*

Fourier Transform

- Basis function

$$\phi(x,u) = e^{j2\pi ux}, \quad u \in (-\infty, +\infty).$$

- Forward Transform

$$F(u) = F\{f(x)\} = \int_{-\infty}^{\infty} f(x) e^{-j2\pi ux} dx$$

- Inverse Transform

$$f(x) = F^{-1}\{F(u)\} = \int_{-\infty}^{\infty} F(u) e^{j2\pi ux} du$$

Important Transform Pairs

$$f(x)=1 \quad \Leftrightarrow \quad F(u)=\delta(u)$$

$$f(x)=e^{j2\pi f_0 x} \quad \Leftrightarrow \quad F(u)=\delta(u-f_0)$$

$$f(x)=\cos(2\pi f_0 x) \quad \Leftrightarrow \quad F(u)=\frac{1}{2}\left(\delta(u-f_0)+\delta(u+f_0)\right)$$

$$f(x)=\sin(2\pi f_0 x) \quad \Leftrightarrow \quad F(u)=\frac{1}{2j}\left(\delta(u-f_0)-\delta(u+f_0)\right)$$

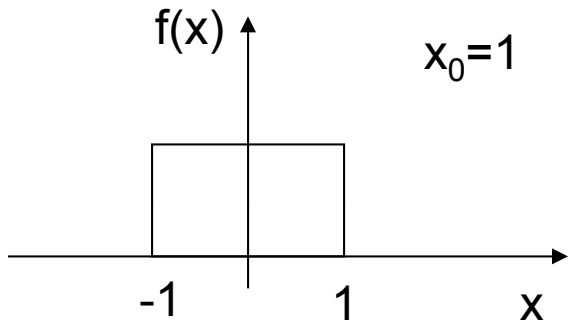
$$f(x)=\begin{cases} 1, & |x|<x_0 \\ 0, & otherwise \end{cases} \quad \Leftrightarrow \quad F(u)=\frac{\sin(2\pi x_0 u)}{\pi u}=2x_0\,\mathrm{sinc}(2x_0 u)$$

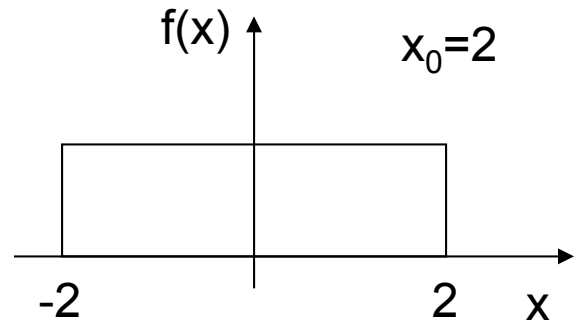
$$where, \mathrm{sinc}(t)=\frac{\sin(\pi t)}{\pi t}$$

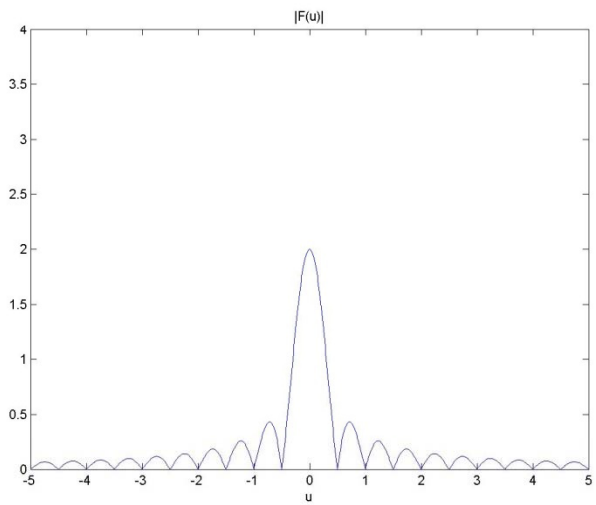
Derive the last transform pair in class

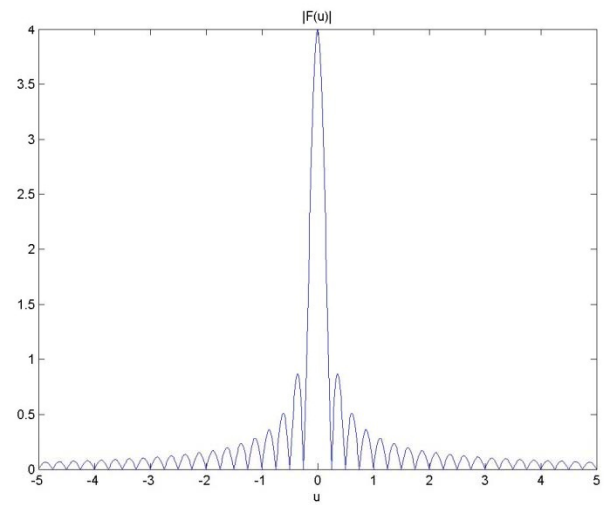
FT of the Rectangle Function

$$F(u) = \frac{\sin(2\pi x_0 u)}{\pi u} = 2x_0 \operatorname{sinc}(2x_0 u) \quad where, \operatorname{sinc}(t) = \frac{\sin(\pi t)}{\pi t}$$





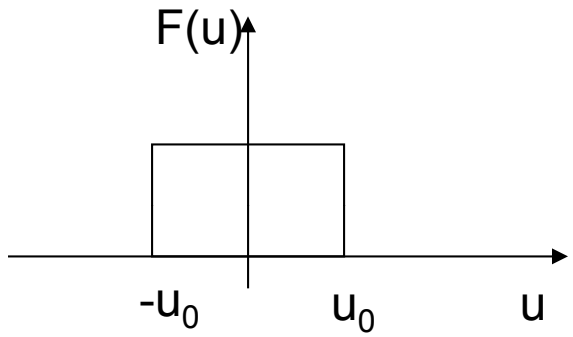




Note first zero occurs at u_0=1/(2 x_0)=1/pulse-width, other zeros are multiples of this.

IFT of Ideal Low Pass Signal

- What is f(x)?



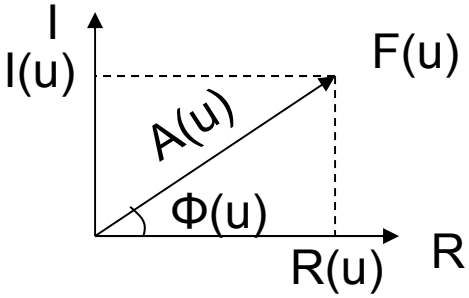
Representation of FT

- Generally, both f(x) and F(u) are complex
- Two representations
  - Real and Imaginary
  - Magnitude and Phase

$$F(u) = R(u) + jI(u)$$

$$F(u) = A(u)e^{j\phi(u)}, \quad where$$

$$A(u) = \sqrt{R(u)^2 + I(u)^2}, \phi(u) = \tan^{-1}\frac{I(u)}{R(u)}$$



- Relationship

$$R(u) = A(u)\cos\phi(u), \quad I(u) = A(u)\sin\phi(u)$$

- Power spectrum

$$P(u) = A(u)^2 = F(u) \times F(u)^* = |F(u)|^2$$

What if f(x) is real?

- Real world signals f(x) are usually real
- F(u) is still complex, but has special properties

$$F^*(u) = F(-u)$$

$$R(u) = R(-u), A(u) = A(-u), P(u) = P(-u) : \text{even function}$$

$$I(u) = -I(-u), \phi(u) = -\phi(-u) : \text{odd function}$$

Property of Fourier Transform

- Duality

$$f(t) \Leftrightarrow F(u)$$
$$F(t) \Leftrightarrow f(-u)$$

- Linearity

$$F\{a_1 f_1(x) + a_2 f_2(x)\} = a_1 F\{f_1(x)\} + a_2 F\{f_2(x)\}$$

- Scaling

$$F\{af(x)\} = aF\{f(x)\}$$

- Translation

$$f(x - x_0) \Leftrightarrow F(u)e^{-j2\pi x_0 u}, \quad f(x)e^{j2\pi u_0 x} \Leftrightarrow F(u - u_0)$$

- Convolution

$$f(x) \otimes g(x) = \int f(x-\alpha)g(\alpha)d\alpha$$

$$f(x) \otimes g(x) \Leftrightarrow F(u)G(u)$$

We will review convolution later!

Two Dimension Fourier Transform

- Basis functions

$$\phi(x,y;u,v) = e^{j(2\pi ux+2\pi vy)} = e^{j2\pi ux}e^{j2\pi vy}, \quad u,v \in (-\infty,+\infty).$$

- Forward – Transform

$$F(u,v) = F\{f(x,y)\} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)e^{-j2\pi(ux+vy)}dxdy$$

- Inverse – Transform

$$f(x,y) = F^{-1}\{F(u,v)\} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} F(u,v)e^{j2\pi(ux+vy)}dudv$$

- Property
  - All the properties of 1D FT apply to 2D FT

Example 1

$$f(x,y) = \sin 4\pi x + \cos 6\pi y$$

$$F\{\sin 4\pi x\} = \iint \sin 4\pi x e^{-j2\pi(ux+vy)} dxdy$$

$$= \int \sin 4\pi x e^{-j2\pi ux} dx \int e^{-j2\pi vy} dy$$

$$= \int \sin 4\pi x e^{-j2\pi ux} dx \delta(v)$$

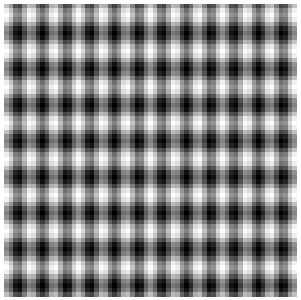
$$= \frac{1}{2j}(\delta(u-2) - \delta(u+2))\delta(v)$$

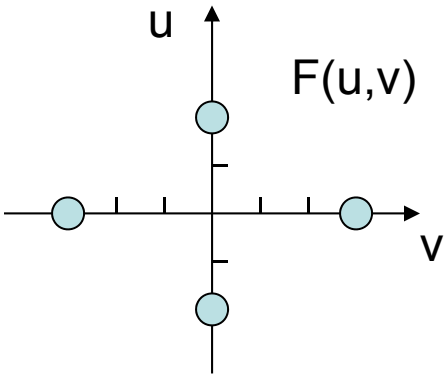
$$= \frac{1}{2j}(\delta(u-2,v) - \delta(u+2,v))$$

$$where \quad \delta(x,y) = \delta(x)\delta(y) = \begin{cases} \infty, & x = y = 0 \\ 0, & otherwise \end{cases}$$

$$Likewise, \quad F\{\cos 6\pi y\} = \frac{1}{2}(\delta(u,v-3) + \delta(u,v+3))$$

f(x,y)





Example 2

$$f(x,y) = \sin(2\pi x + 3\pi y) = \frac{1}{2j}\left(e^{j(2\pi x+3\pi y)} - e^{-j(2\pi x+3\pi y)}\right)$$

$$F\left\{e^{j(2\pi x+3\pi y)}\right\} = \iint e^{j(2\pi x+3\pi y)} e^{-j2\pi(xu+yv)} dxdy$$

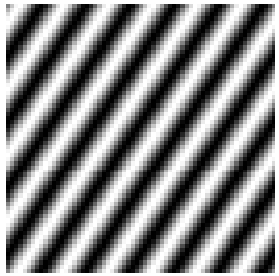
$$= \int e^{j2\pi x} e^{-j2\pi ux} dx \int e^{j3\pi y} e^{-j2\pi yv} dy$$

$$= \delta(u-1)\delta(v-\frac{3}{2}) = \delta(u-1, v-\frac{3}{2})$$

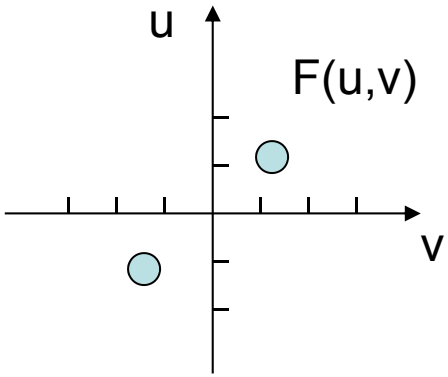
Likewise, $F\left\{e^{-j(2\pi x+3\pi y)}\right\} = \delta(u+1, v+\frac{3}{2})$

Therefore,

$$F\{\sin(2\pi x + 3\pi y)\} = \frac{1}{2j}\left(\delta(u-1, v-\frac{3}{2}) - \delta(u+1, v+\frac{3}{2})\right)$$



```
[X,Y]=meshgrid(-2:1/16:2,-2:1/16:2);
f=sin(2*pi*X+3*pi*Y);
imagesc(f); colormap(gray)
Truesize, axis off;
```



Important Transform Pairs

$$\sin(2\pi f_x x + 2\pi f_y y) \quad \Leftrightarrow \quad \frac{1}{2j}\Big(\delta(u - f_x, v - f_y) - \delta(u + f_x, v + f_y)\Big)$$

$$\cos(2\pi f_x x + 2\pi f_y y) \quad \Leftrightarrow \quad \frac{1}{2}\Big(\delta(u - f_x, v - f_y) + \delta(u + f_x, v + f_y)\Big)$$

2D rectangular function ⇔ 2D sinc function

Properties of 2D FT (1)

- Linearity

$$F\{a_1 f_1(x,y) + a_2 f_2(x,y)\} = a_1 F\{f_1(x,y)\} + a_2 F\{f_2(x,y)\}$$

- Translation

$$f(x-x_0, y-y_0) \quad \Leftrightarrow \quad F(u,v)e^{-j2\pi(x_0 u + y_0 v)},$$

$$f(x,y)e^{j2\pi(u_0 x + v_0 y)} \quad \Leftrightarrow \quad F(u-u_0, v-v_0)$$

- Conjugation

$$f^*(x,y) \quad \Leftrightarrow \quad F^*(-u,-v)$$

Properties of 2D FT (2)

- Symmetry

$$f(x,y)\ is\ real \quad \Leftrightarrow \quad |F(u,v)| = |F(-u,-v)|$$

- Convolution
  - Definition of convolution

$$f(x,y) \otimes g(x,y) = \iint f(x-\alpha, y-\beta) g(\alpha,\beta) d\alpha d\beta$$

  - Convolution theory

$$f(x,y) \otimes g(x,y) \quad \Leftrightarrow \quad F(u,v)G(u,v)$$

We will describe 2D convolution later!

Separability of 2D FT and Separable Signal

- Separability of 2D FT

$$F_2\{f(x,y)\} = F_y\{F_x\{f(x,y)\}\} = F_x\{F_y\{f(x,y)\}\}$$

  - where F_x, F_y are 1D FT along x and y.
  - one can do 1DFT for each row of original image, then 1D FT along each column of resulting image
- Separable Signal
  - $f(x,y) = f_x(x)f_y(y)$
  - $F(u,v) = F_x(u)F_y(v)$,
    - where $F_x(u) = F_x\{f_x(x)\}$, $F_y(u) = F_y\{f_y(y)\}$
  - For separable signal, one can simply compute two 1D transforms!

Example 1

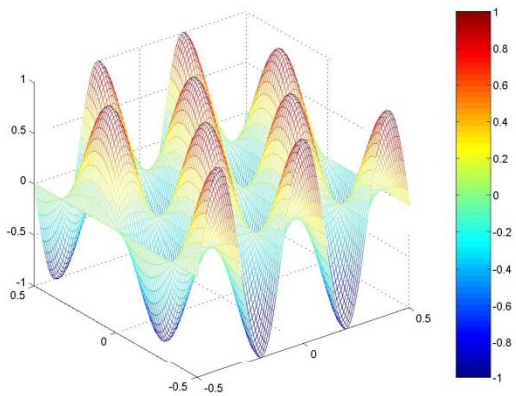
$$f(x,y) = \sin(3\pi x)\cos(5\pi y)$$

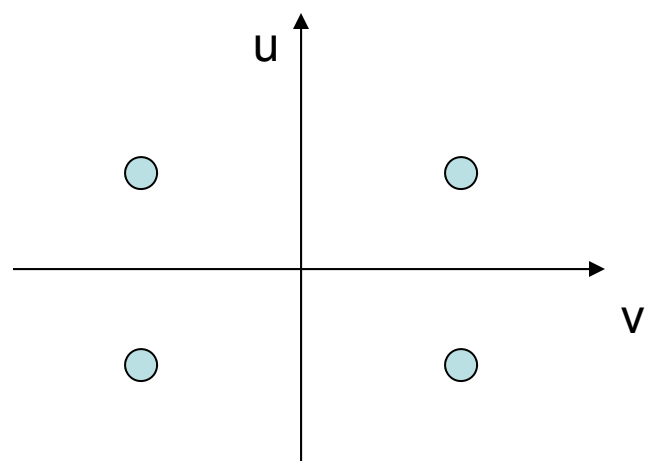
$$f_x(x) = \sin(3\pi x) \quad \Leftrightarrow \quad F_x(u) = \frac{1}{2j}\left(\delta(u-3/2) - \delta(u+3/2)\right)$$

$$f_y(y) = \cos(5\pi y) \quad \Leftrightarrow \quad F_y(v) = \frac{1}{2}\left(\delta(v-5/2) + \delta(v+5/2)\right)$$

$$F(u,v) = Fx(u)Fy(v)$$

$$= \frac{1}{4j}\left(\delta(u-\frac{3}{2}, v-\frac{5}{2}) - \delta(u+\frac{3}{2}, v-\frac{5}{2}) + \delta(u-\frac{3}{2}, v+\frac{5}{2}) - \delta(u+\frac{3}{2}, v+\frac{5}{2})\right)$$

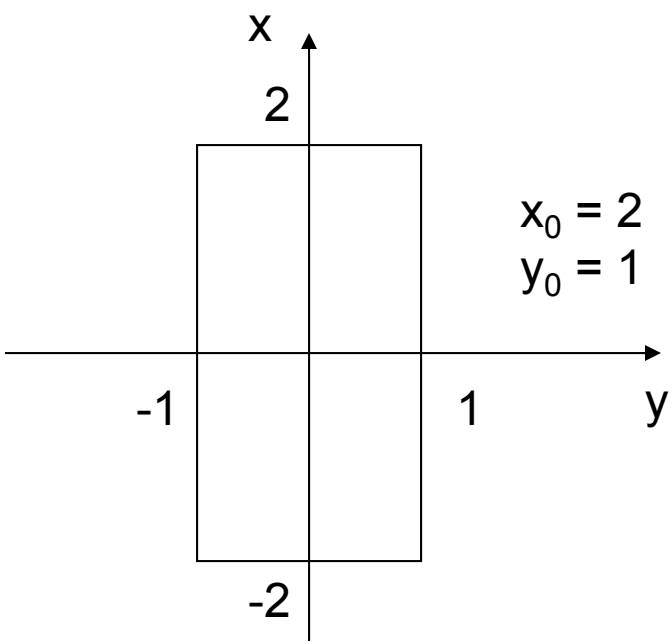


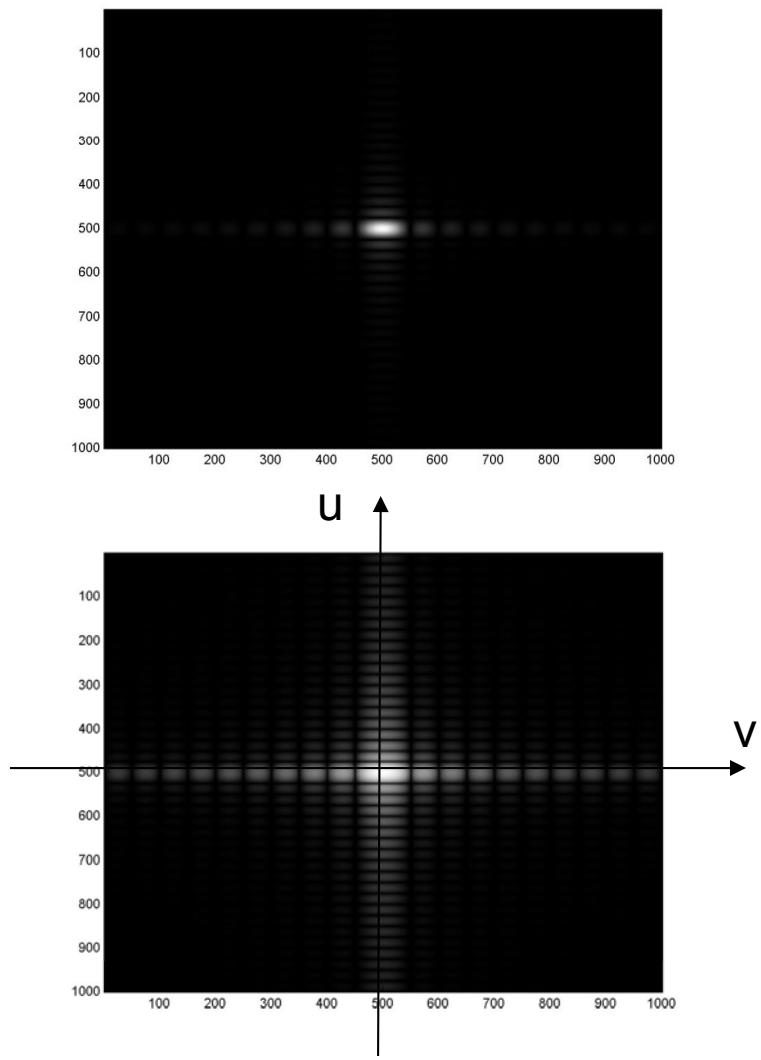


Example 2

$$f(x,y)=\begin{cases}1, & |x|\le x_0, |y|\le y_0\\ 0, & otherwise\end{cases} \Rightarrow$$

$$F(u,v)=4x_0y_0\,\mathrm{sinc}(2x_0u)\,\mathrm{sinc}(2y_0v)$$





w/ logrithmic mapping

Rotation

- Let $x = r\cos\theta, \quad y = r\sin\theta, \quad u = \rho\cos\omega, \quad v = \rho\sin\omega.$
- 2D FT in polar coordinate (r, θ) and (ρ, Ø)

$$F(\rho,\phi) = \int_0^{\infty}\int_0^{2\pi} f(r,\theta)e^{-j2\pi(r\cos\theta\,\rho\cos\phi + r\sin\theta\,\rho\sin\phi)} r\,dr\,d\theta$$
$$= \iint f(r,\theta)e^{-j2\pi r\rho\cos(\theta-\phi)} r\,dr\,d\theta$$

- Property

$$f(r,\theta+\theta_0) \quad \Leftrightarrow \quad F(\rho,\phi+\theta_0)$$

Example of Rotation

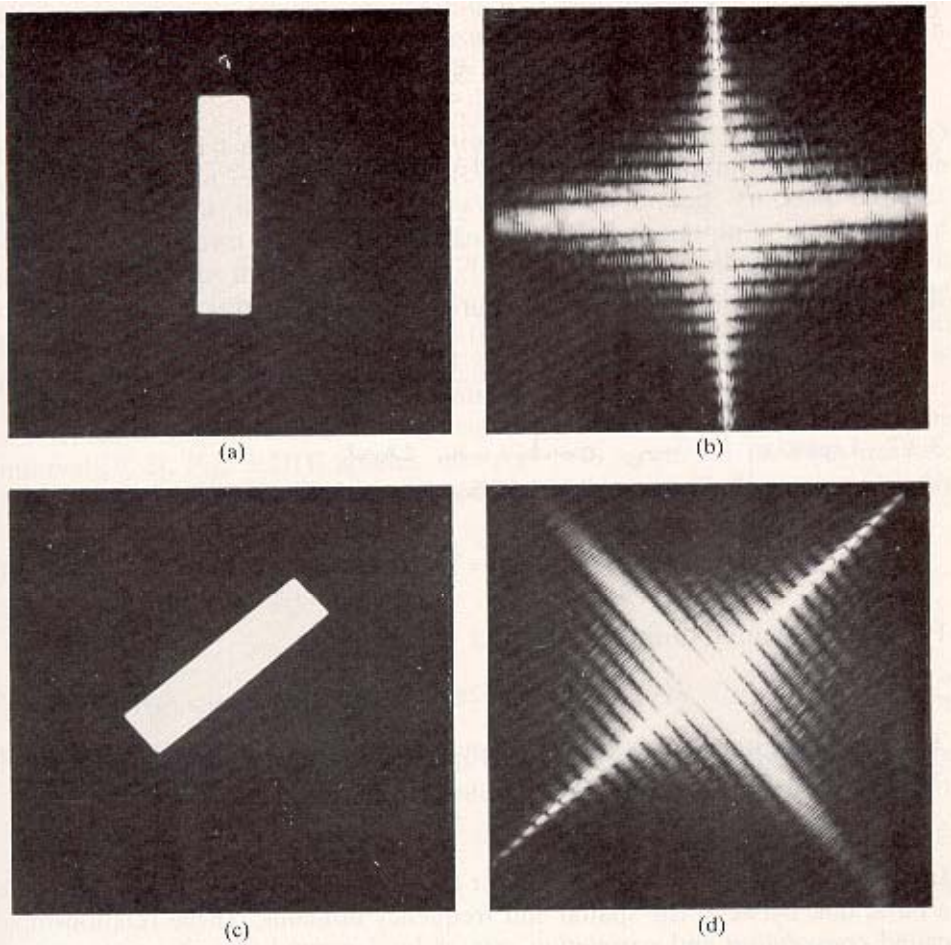


Figure 3.10 *Rotational properties of the Fourier transform: (a) a simple image; (b) spectrum; (c) rotated image; (d) resulting spectrum.*

Fourier Transform For Discrete Time Sequence (DTFT)

- One Dimensional DTFT
  - f(n) is a 1D discrete time sequence
  - Forward Transform

$$F(u) = \sum_{n=-\infty}^{\infty} f(n) e^{-j2\pi un}$$

F(u) is periodic in u, with period of 1

  - Inverse Transform

$$f(n) = \int_{-1/2}^{1/2} F(u) e^{j2\pi un} du$$

Properties unique for DTFT

- Periodicity
  - F(u) = F(u+1)
  - The FT of a discrete time sequence is only considered for u ϵ (-½ , ½), and u = ±½ is the highest discrete frequency
- Symmetry for real sequences

$$f(n) = f^*(n) \quad \Leftrightarrow \quad F(u) = F^*(-u)$$

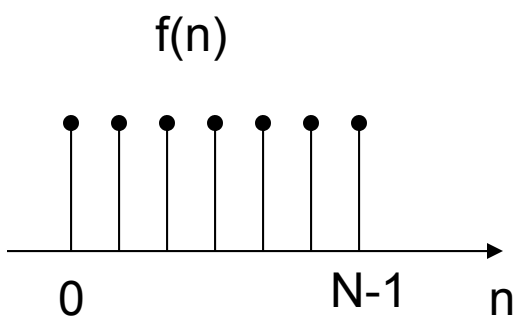
$$\Rightarrow \quad |F(u)| = |F(-u)|$$

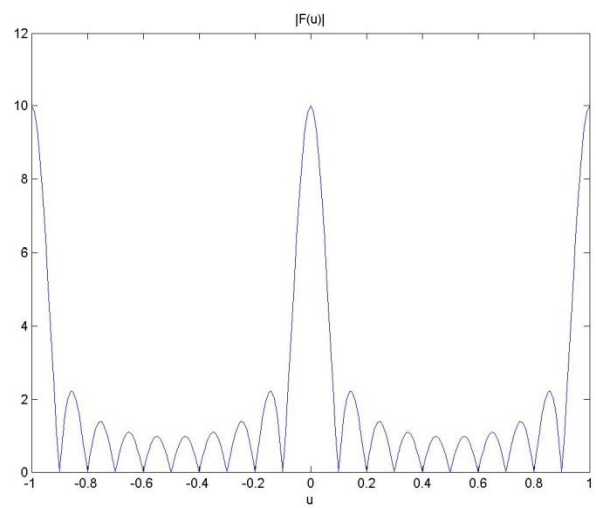
$$\Rightarrow \quad |F(u)| \; is \; symmetric$$

Example

$$f(n) = \begin{cases} 1, & n = 0,1,\ldots,N-1; \\ 0, & others \end{cases}$$

$$F(u) = \sum_{n=0}^{N-1} e^{-j2\pi nu} = \frac{1-e^{-j2\pi uN}}{1-e^{-j2\pi u}} = e^{-j\pi(N-1)u}\,\frac{\sin 2\pi u(N/2)}{\sin 2\pi u(1/2)}$$





N=10

There are N/2 zeros in (0, ½], 1/N apart

Two Dimensional DTFT

- Let f(m,n) represent a 2D sequence
- Forward Transform

$$F(u,v) = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} f(m,n) e^{-j2\pi(mu+nv)}$$

- Inverse Transform

$$f(m,n) = \int_{-1/2}^{1/2} \int_{-1/2}^{1/2} F(u,v) e^{j2\pi(mu+nv)} du dv$$

- Properties
  - Periodicity, Shifting and Modulation, Energy Conservation

Periodicity

$$F(u,v) = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} f(m,n) e^{-j2\pi(mu+nv)}$$

- *F(u,v)* is periodic in u, v with period 1, i.e., for all integers k, l:
  - F(u+k, v+l) = F(u, v)
- To see this consider

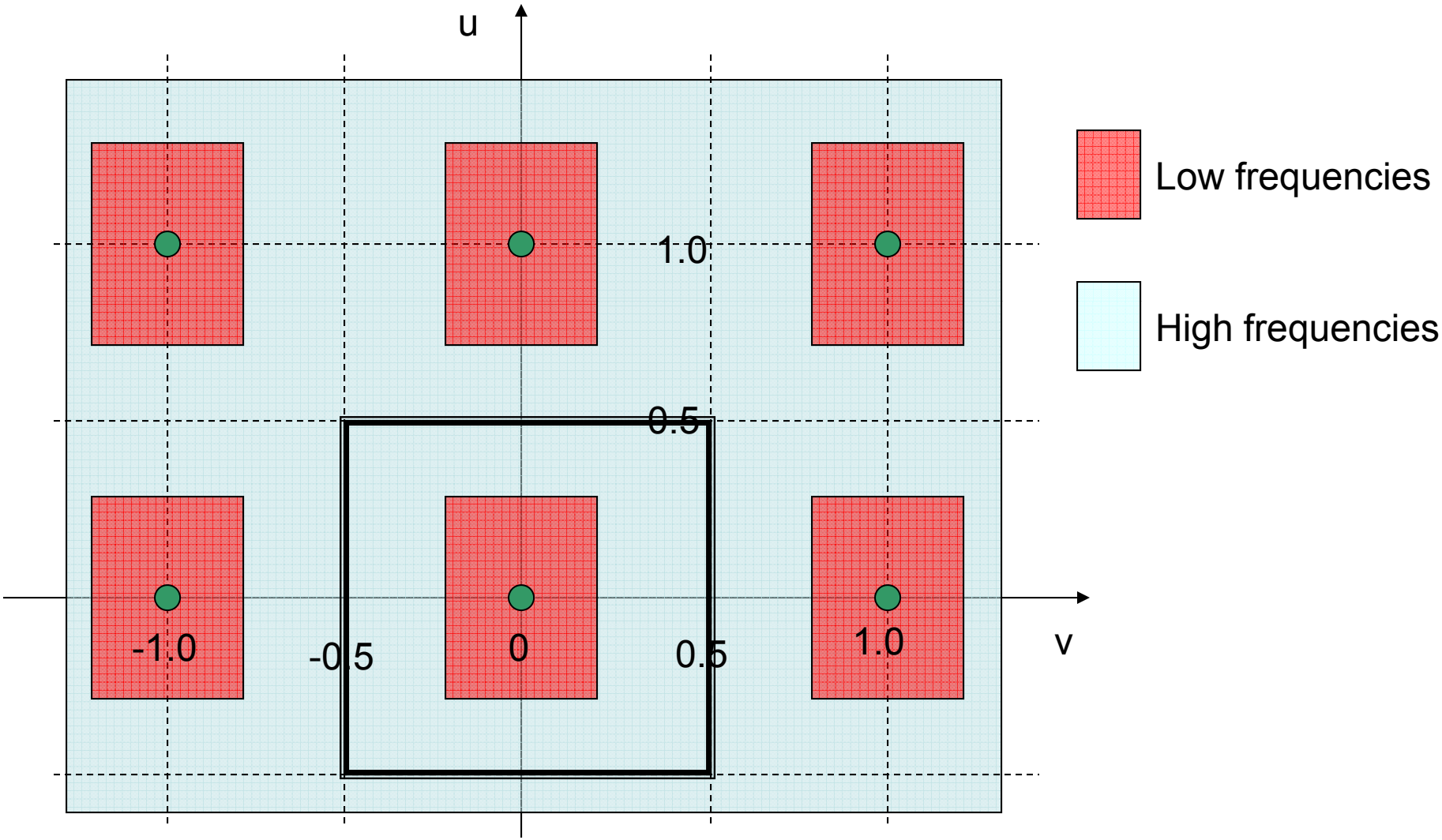
$$e^{-j2\pi(m(u+k)+n(v+l))}$$

$$= e^{-j2\pi(mu+nv)} e^{-j2\pi(mk+nl)}$$

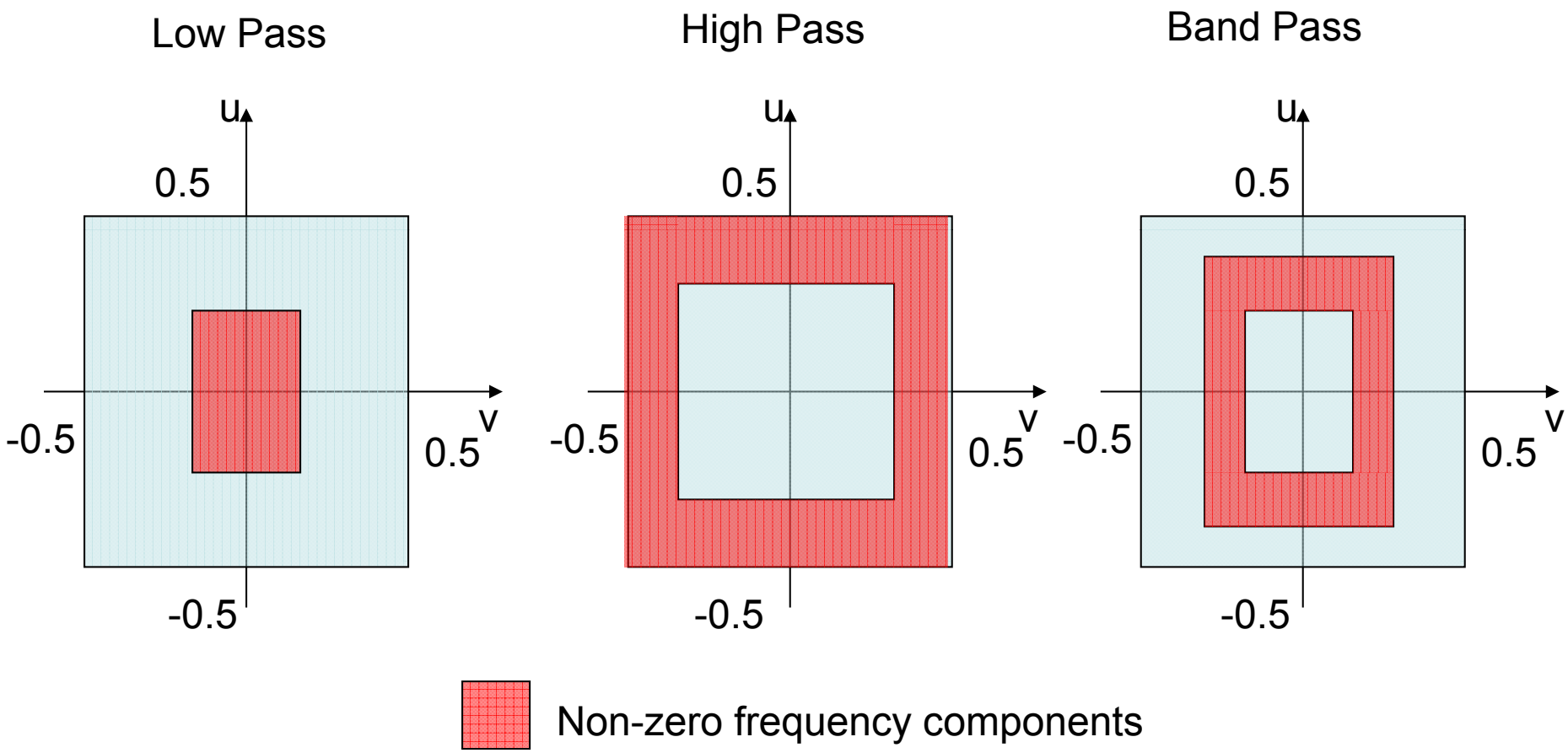
$$= e^{-j2\pi(mu+nv)}$$

In MATLAB, frequency scaling is such that 1 represents maximum freq u,v=1/2.

Illustration of Periodicity



Fourier Transform Types



Shifting and Modulation

- Shifting

$$F\{f(m-m_0,n-n_0)\} = \sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty} f(m-m_0,n-n_0)e^{-j2\pi(mu+nv)}$$

$$\underline{\underline{k=m-m_0, l=n-n_0}} \quad = \sum_{k=-\infty}^{\infty}\sum_{l=-\infty}^{\infty} f(k,l)e^{-j2\pi((k+m_0)u+(l+n_0)v)}$$

$$= e^{-j2\pi(m_0u+n_0v)}\sum_{k=-\infty}^{\infty}\sum_{l=-\infty}^{\infty} f(k,l)e^{-j2\pi(ku+lv)}$$

$$f(m-m_0,n-n_0) \quad \Leftrightarrow \quad e^{-j2\pi(m_0u+n_0v)}F(u,v)$$

- Modulation

$$e^{j2\pi(mu_0+nv_0)}f(m,n) \quad \Leftrightarrow \quad F(u-u_0,v-v_0)$$

Energy Conservation

- Inner Product

$$< f, g >= \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} f(m,n) g^*(m,n)$$

$$= \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} f(m,n) \left[\int_{-0.5}^{0.5} \int_{-0.5}^{0.5} G^*(u,v) e^{-j2\pi(mu+nv)} du dv \right]$$

$$= \int_{-0.5}^{0.5} \int_{-0.5}^{0.5} \left[\sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} f(m,n) e^{-j2\pi(mu+nv)} \right] G^*(u,v) du dv$$

$$= \int_{-0.5}^{0.5} \int_{-0.5}^{0.5} F(u,v) G^*(u,v) du dv =< F, G >$$

- Energy Conservation

$$\sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} |f(m,n)|^2 = \int_{-0.5}^{0.5} \int_{-0.5}^{0.5} |F(u,v)|^2 du dv$$

Delta Function

- Fourier transform of a delta function

$$F(u,v) = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} \delta(m,n) e^{-j2\pi(mu+nv)} = 1$$

$$\delta(m,n) \Leftrightarrow 1$$

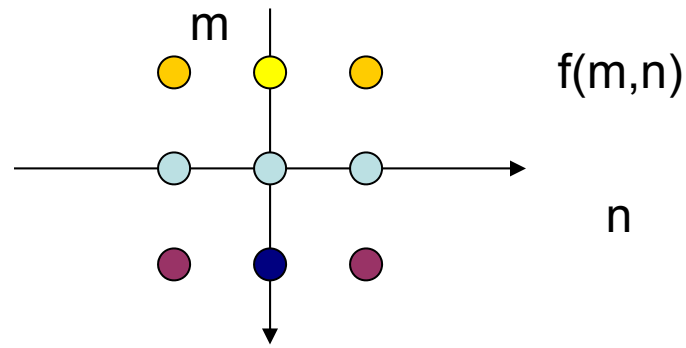
- Inverse Fourier transform of a delta function

$$f(m,n) = \int_{-0.5}^{0.5} \int_{-0.5}^{0.5} \delta(u,v) e^{j2\pi(mu+nv)} du dv = 1$$

$$1 \Leftrightarrow \delta(u,v)$$

Example

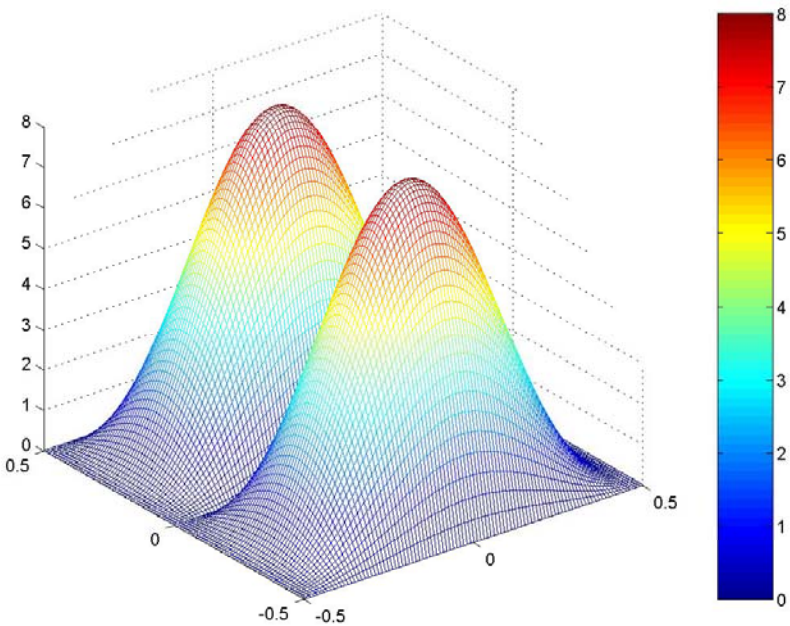
$$f(m,n) = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

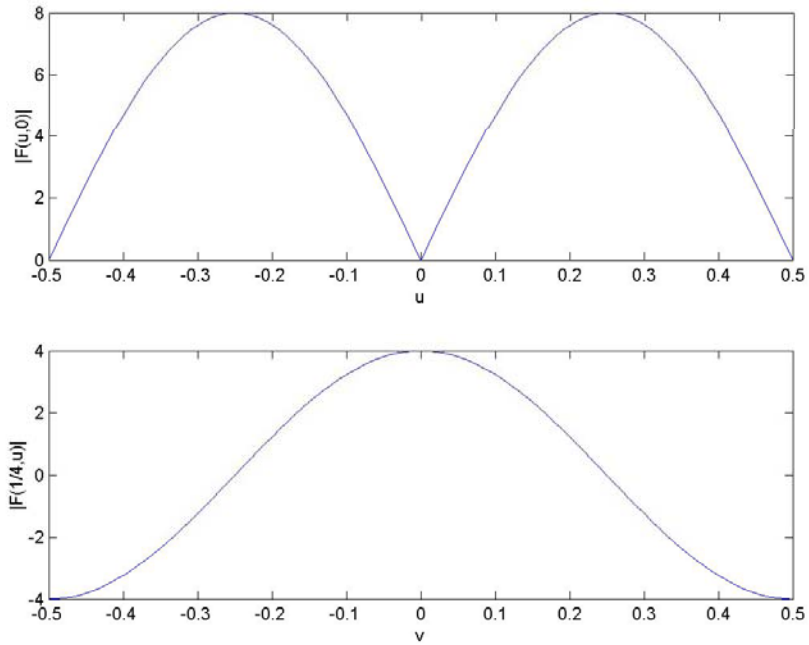


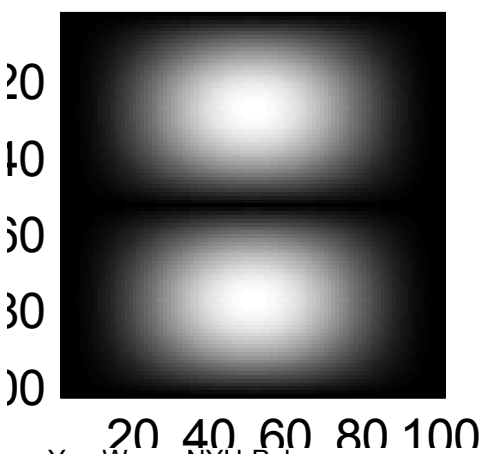
$$\begin{aligned} F(u,v) &= 1e^{-j2\pi(-1*u-1*v)} + 2e^{-j2\pi(-1*u+0*v)} + 1e^{-j2\pi(-1*u+1*v)} \\ &\quad -1e^{-j2\pi(1*u-1*v)} - 2e^{-j2\pi(1*u+0*v)} - 1e^{-j2\pi(1*u+1*v)} \\ &= j2\sin 2\pi u e^{j2\pi v} + j4\sin 2\pi u + j2\sin 2\pi u e^{-j2\pi v} \\ &= j4\sin 2\pi u(\cos 2\pi v + 1) \end{aligned}$$

Note: This signal is low pass in the horizontal direction, and band pass in the vertical direction.

Graph of F(u,v)







```
du = [-0.5:0.01:0.5];
dv = [-0.5:0.01:0.5];
Fu = abs(sin(2 * pi * du));
Fv = cos(2 * pi * dv);
F = 4 * Fu' * (Fv + 1);
mesh(du, dv, F);
colorbar;
Imagesc(F);
colormap(gray); truesize;
```

Using MATLAB freqz2:

```
f=[1,2,1;0,0,0;-1,-2,-1];
freqz2(f)
```

Linear Convolution

- Convolution of Continuous Signals
  - 1D convolution

$$f(x)*h(x)=\int_{-\infty}^{\infty} f(x-\alpha)h(\alpha)d\alpha=\int_{-\infty}^{\infty} f(\alpha)h(x-\alpha)d\alpha$$

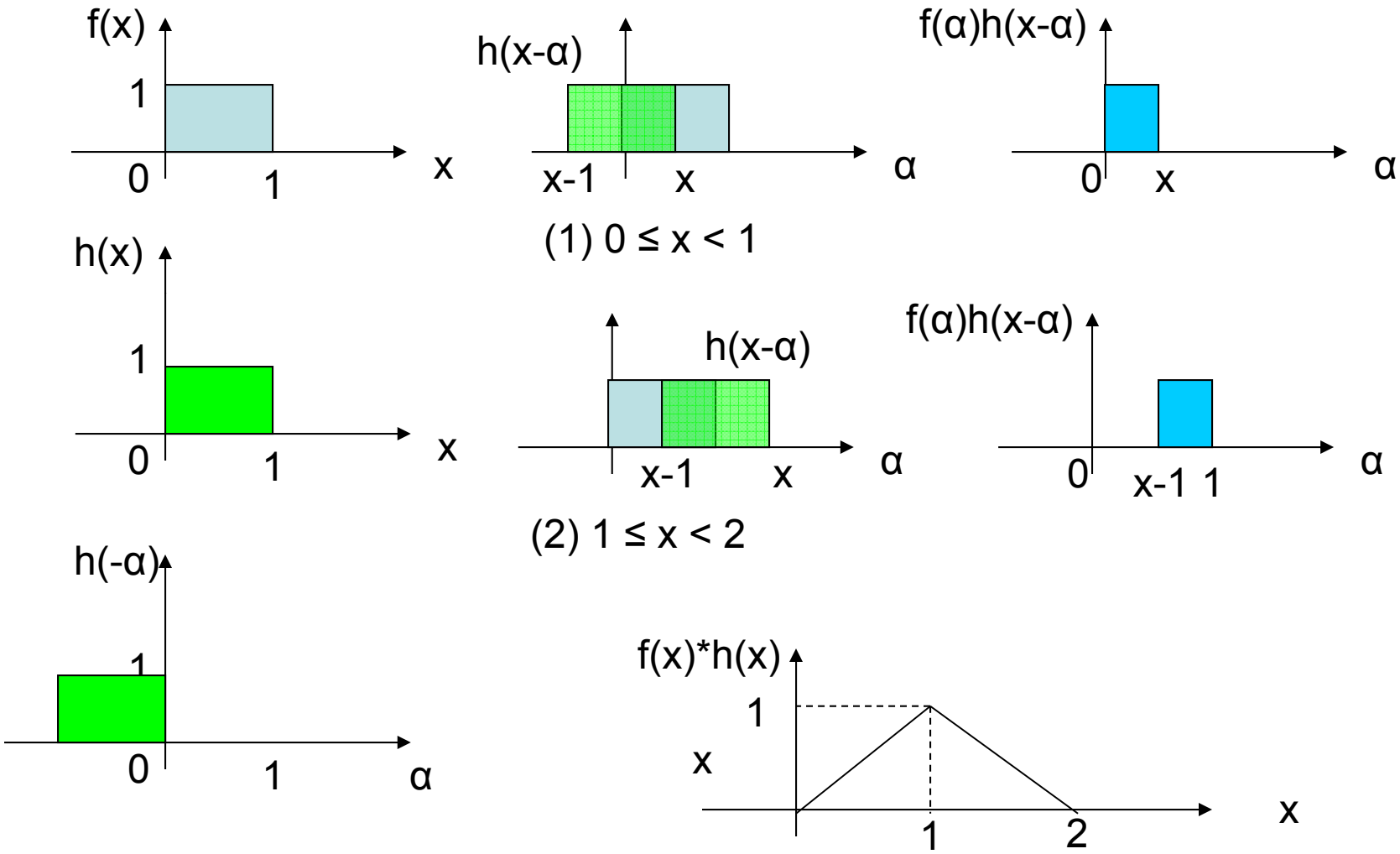
    - Equalities

$$f(x)*\delta(x)=f(x),\quad f(x)*\delta(x-x_0)=f(x-x_0)$$

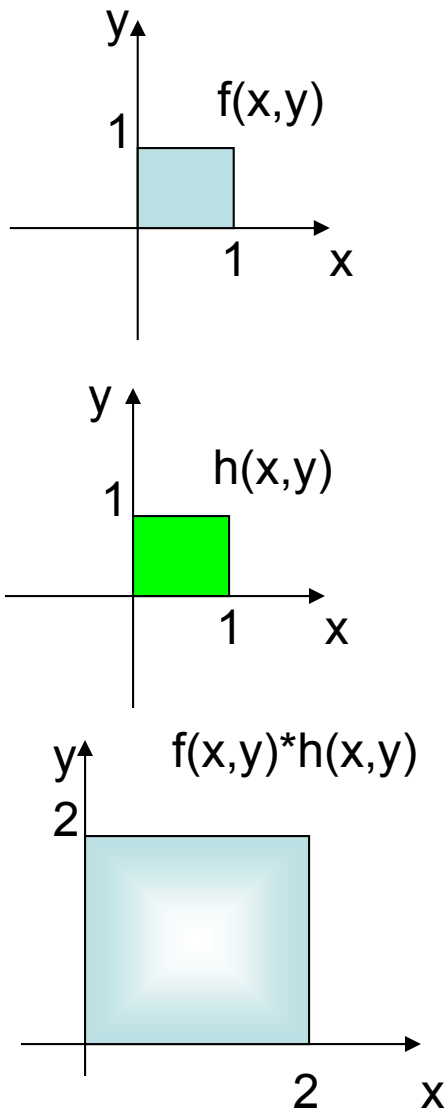
  - 2D convolution

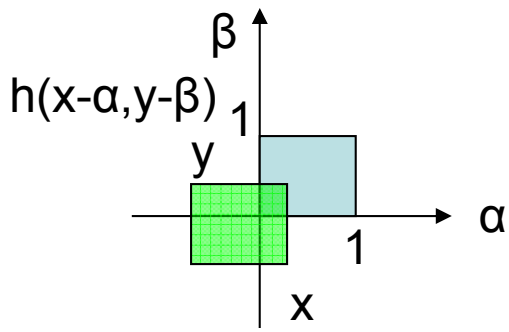
$$f(x,y)*h(x,y)=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x-\alpha,y-\beta)h(\alpha,\beta)d\alpha d\beta$$
$$=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(\alpha,\beta)h(x-\alpha,y-\beta)d\alpha d\beta$$

Examples of 1D Convolution

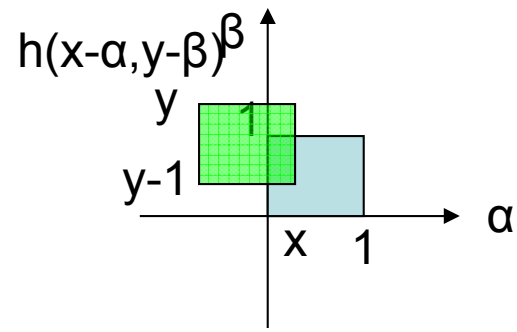


Example of 2D Convolution

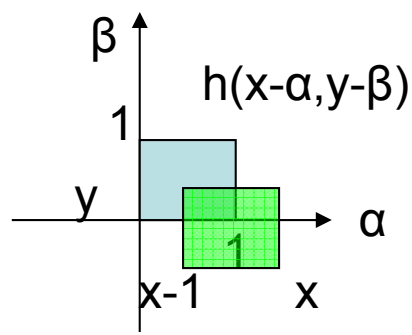




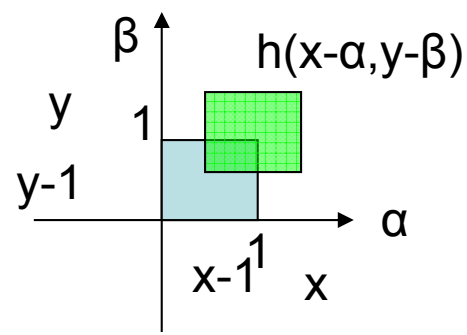
(1) 0<x≤1, 0<y≤1
g(x,y)=x*y



(2) 0<x≤1, 1<y≤2
g(x,y)=x*(2-y)



(3) 1<x≤2, 0<y≤1
g(x,y)=(2-x)*y



(4) 1<x≤2, 1<y≤2
g(x,y)=(2-x)*(2-y)

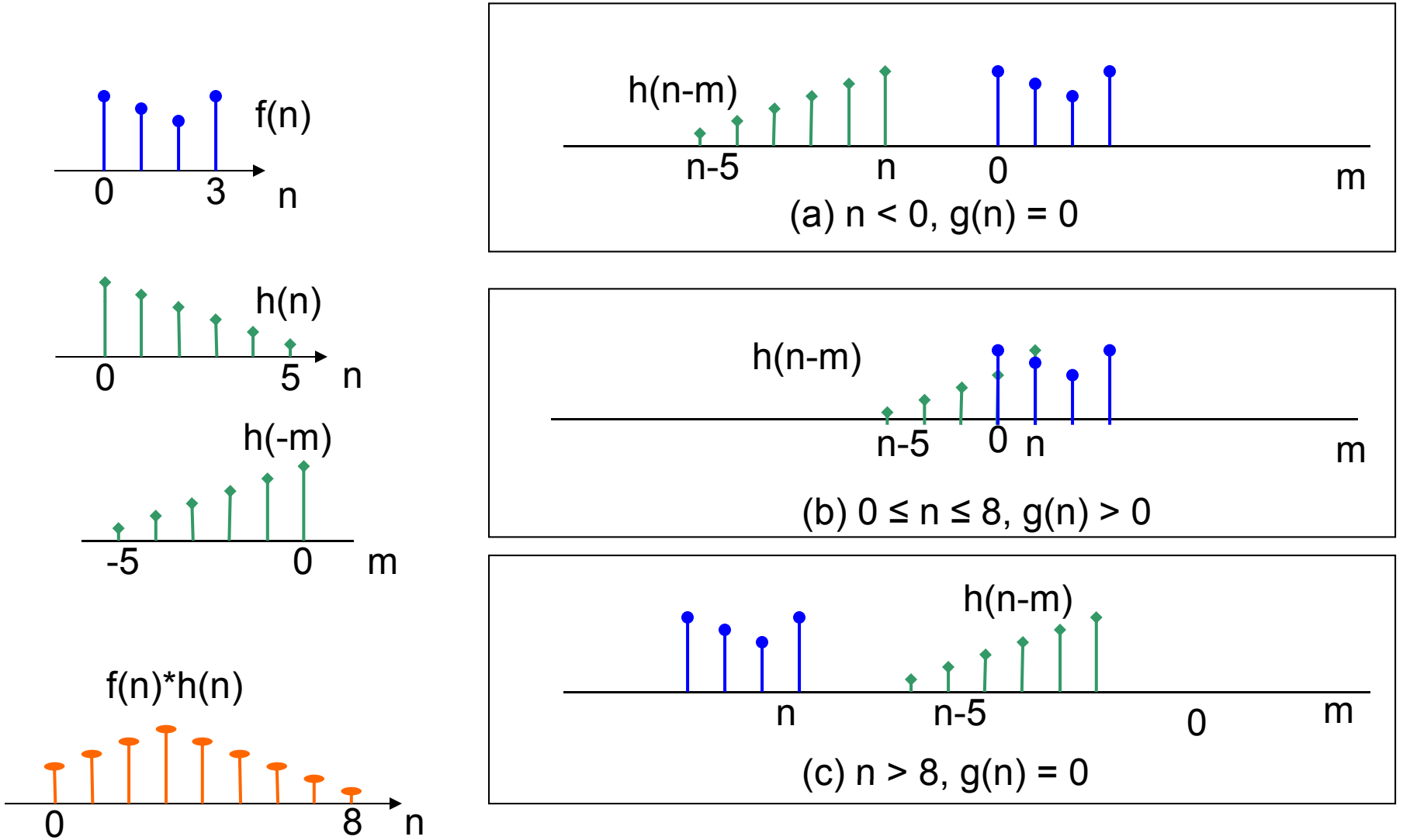
Convolution of 1D Discrete Signals

- Definition of convolution

$$f(n) * h(n) = \sum_{m=-\infty}^{\infty} f(n-m)h(m) = \sum_{m=-\infty}^{\infty} f(m)h(n-m)$$

- The convolution can be considered as the weighted average:
  - sample n-m is multiplied by h(m)
- Filtering: f(n) is the signal, and h(n) is the filter
  - Finite impulse response (FIR) filter
  - Infinite impulse response (IIR) filter
- If the range of f(n) is (n_0, n_1), and the range of h(n) is (m_0, m_1), then the range of f(n)*h(n) is (n_0+m_0, n_1+m_1)

Example of 1D Discrete Convolution



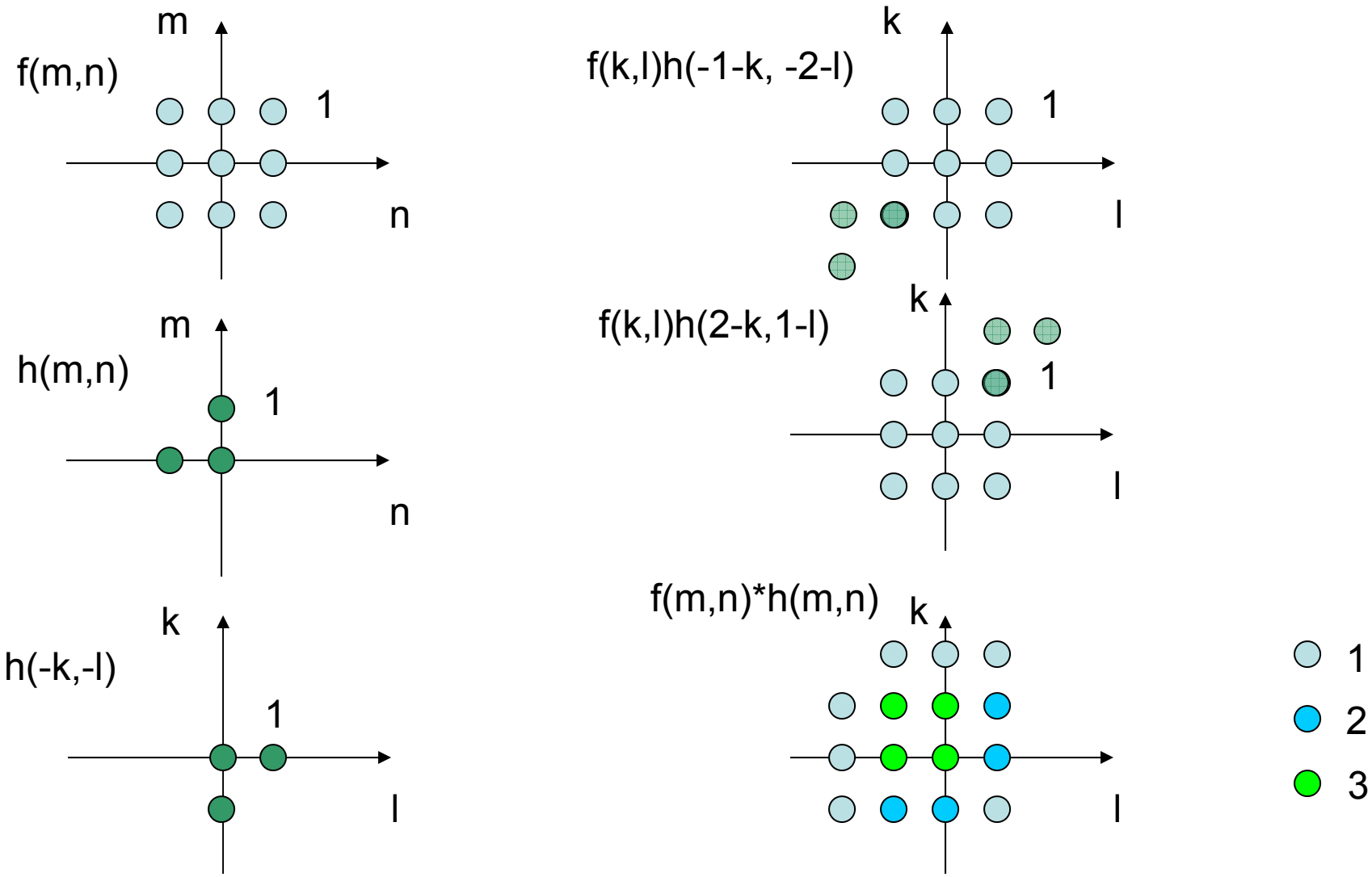
Convolution of 2D Discrete Signals

- Definition of convolution

$$f(m,n)*h(m,n)=\sum_{k=-\infty}^{\infty}\sum_{l=-\infty}^{\infty}f(m-k,n-l)h(k,l)$$
$$=\sum_{k=-\infty}^{\infty}\sum_{l=-\infty}^{\infty}f(k,l)h(m-k,n-l)$$

- Weighted average:
  - Pixel (m-k,n-l) is weighted by h(k,l)
- Range
  - If the range of f(m, n) is $m_0 \le m \le m_1$, $n_0 \le n \le n_1$
  - If the range of h(m, n) is $k_0 \le m \le k_1$, $l_0 \le n \le l_1$,
  - Then the range of f(m, n)*h(m, n) is
    $m_0 + k_0 \le m \le m_1 + k_1$, $n_0 + l_0 \le n \le n_1 + l_1$

Example of 2D Discrete Convolution



- Go through an example in class for an arbitrary 3x3 (non-symmetric) filter

Separable Filtering

- A filter is separable if $h(x, y)=h_x(x)h_y(y)$ or $h(m, n)=h_x(m)h_y(n)$.
- Matrix representation

$$H = h_x h_y^T$$

  – Where h_x and h_y are column vectors
- Example

$$H = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -1 \end{bmatrix} = h_x h_y^T$$

DTFT of Separable Filter

- Previous example

$$H = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -1 \end{bmatrix} = h_x h_y^T$$

- Recognizing that the filter is separable

$$\begin{bmatrix} 1 & 0 & -1 \end{bmatrix} <-> Fy(v) = 1e^{j2\pi v} + 0 + (-1)e^{-j2\pi v} = 2j\sin 2\pi v$$

$$\begin{bmatrix} 1 & 2 & 1 \end{bmatrix} <-> Fx(u) = 1e^{j2\pi u} + 2 + e^{-j2\pi u} = 2 + 2\cos 2\pi u$$

$$F(u,v) = Fx(u)Fy(v) = 4j(1+\cos 2\pi u)\sin 2\pi v$$

Separable Filtering

- If H(m,n) is separable, the 2D convolution can be accomplished by first applying 1D filtering along each row using $h_y(n)$, and then applying 1D filtering to the intermediate result along each column using the filter $h_x(n)$
- Proof

$$
\begin{aligned}
f(m,n) * h(m,n) &= \sum_k \sum_l f(m-k, n-l) h_x(k) h_y(l) \\
&= \sum_k \left(\sum_l f(m-k, n-l) h_y(l) \right) h_x(k) \\
&= \sum_k g_y(m-k, n) h_x(k) \\
&= (f(m,n) * h_y(n)) * h_x(m)
\end{aligned}
$$

- Go through an example of filtering using separable filter in class

Computation Cost

- Suppose
  - The size of the image is M*N.
  - The size of the filter is K*L.
- Non-separable filtering
  - Each pixel: K*L mul; K*L – 1 add.
  - Total: M*N*K*L mul; M*N*(K*L-1) add.
  - When M=N, K=L
    - M^2K^2 mul + $M^2(K^2-1)$ add.

Computation Cost

- Separable:
  - Each pixel in a row: L mul; L-1 add.
  - Each row: N*L mul; N*(L-1) add.
  - M rows: M*N*L mul; M*N*(L-1) add.
  - Each pixel in a column: K mul; K-1 add.
  - Each column: M*K mul; M*(K-1) add.
  - N columns: N*M*K mul; N*M*(K-1) add.
  - Total: M*N*(K+L) mul; M*N*(K+L-2) add.
  - When M=N, K=L:
    - $2M^2K$ mul; $2M^2(K-1)$ add.
    - Significant savings if K (and L) is large!

Boundary Treatment

- When assuming the image values outside a given image are zero, the filtered values in the boundary are affected by assumed zero values adversely
- For better results, use symmetric extension (mirror image) to fill the outside image values

200	200	205	203	203
200	200	205	203	203
195	195	200	200	200
200	200	205	195	195
200	200	205	195	195

Actual image pixels

Extended pixels

Computer implementation of convolution

- Boundary treatment
  - An image of size MxN convolving with a filter of size KxL will yield an image of size (M+K-1,N+L-1)
  - If the filter is symmetric, the convolved image should have extra boundary of thickness K/2 on each side
  - Filtered values in the outer boundary of K/2 pixels depend on the extended pixel values
  - For simplicity, we can ignore the boundary problem, and process only the inner rows and columns of the image, leaving the outer K/2 rows and L/2 columns unchanged (if filter is low-pass) or as zero (if filter is high-pass)
- Renormalization:
  - The filtered value may not be in the range of (0,255) and may not be integers
  - Use two-pass operation
    - First pass: save directly filtered value in an intermediate floating-point array
    - Second pass: find minimum and maximum values of the intermediate image, renormalize to (0,255) and rounding to integers
      - F= round((F1-fmin)*255/(fmax-fmin))
    - To display the unnormalized image directly, use "imagesc( )" function

Sample Matlab Program

```
% readin bmp file
x = imread('lena.bmp');
[xh xw] = size(x);
y = double(x);
```

```
% define 2D filter
h = ones(5,5)/25;
[hh hw] = size(h);
hhh = (hh - 1) / 2;
hhw = (hw- 1) / 2;
```

```
% linear convolution, assuming the filter is non-separable (although this example filter is separable)
z = y; %or z=zeros(xh,xw) if not low-pass filter
for m = hhh + 1:xh - hhh,
            %skip first and last hhh rows to avoid boundary problems
  for n = hhw + 1:xw - hhw,
            %skip first and last hhw columns to avoid boundary problems
    tmpy = 0;
    for k = -hhh:hhh,
      for l = -hhw:hhw,
        tmpv = tmpv + y(m - k,n – l)* h(k + hhh + 1, l + hhw + 1);
                        %h(0,0) is stored in h(hhh+1,hhw+1)
      end
    end
    z(m, n) = tmpv;
%for more efficient matlab coding, you can replace the above loop with
            z(m,n)=sum(sum(y(m-hhh:m+hhh,n-hhw:n+hhw).*h))
  end
end
```

Results

$$h = \frac{1}{25}\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

f(m,n)

g(m,n)

- What if the filter is symmetric? Can you write a different MATLAB code that has reduced computation time?
- MATLAB built-in functions
  - 1D filtering: conv
  - 2D filtering: conv2
  - Computing DTFT: freqz, freqz2

Convolution Theorem

- Convolution Theorem

$$f * h \quad \Leftrightarrow \quad F \times H, \quad f \times h \quad \Leftrightarrow \quad F * H$$

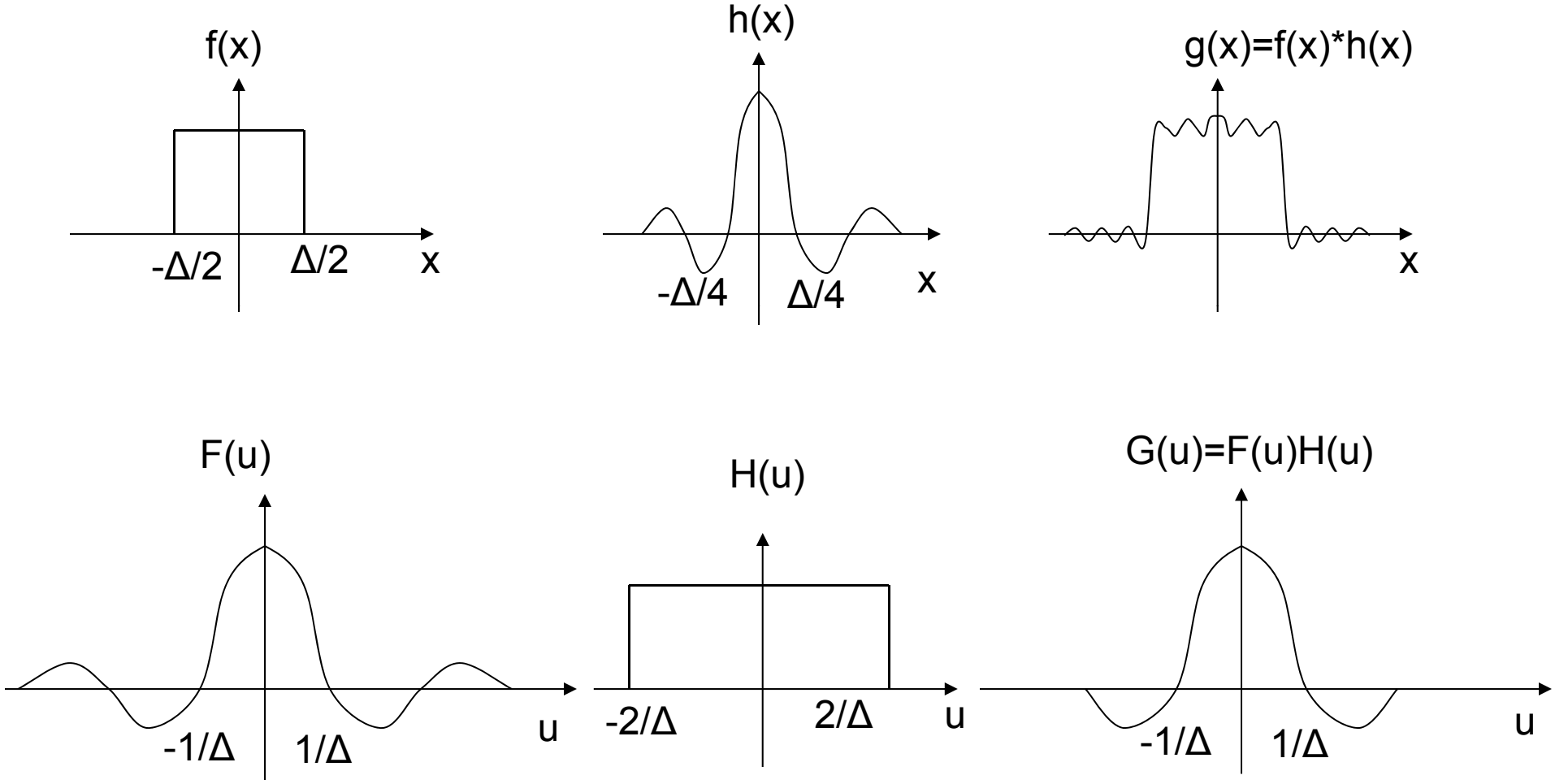
- Proof

$$g(m,n) = f(m,n) * h(m,n) = \sum_k \sum_l f(m-k, n-l) h(k,l)$$

FT on both sides

$$\begin{aligned}
G(u,v) &= \sum_{m,n} \sum_{k,l} f(m-k, n-l) h(k,l) e^{-j2\pi(mu+nv)} \\
&= \sum_{m,n} \sum_{k,l} f(m-k, n-l) e^{-j2\pi((m-k)u+(n-l)v)} h(k,l) e^{-j2\pi(ku+lv)} \\
&= \sum_{m,n} f(m-k, n-l) e^{-j2\pi((m-k)u+(n-l)v)} \sum_{k,l} h(k,l) e^{-j2\pi(ku+lv)} \\
&= \sum_{m',n'} f(m',n') e^{-j2\pi(m'u+n'v)} \sum_{k,l} h(k,l) e^{-j2\pi(ku+lv)} \\
&= F(u,v) \times H(u,v)
\end{aligned}$$

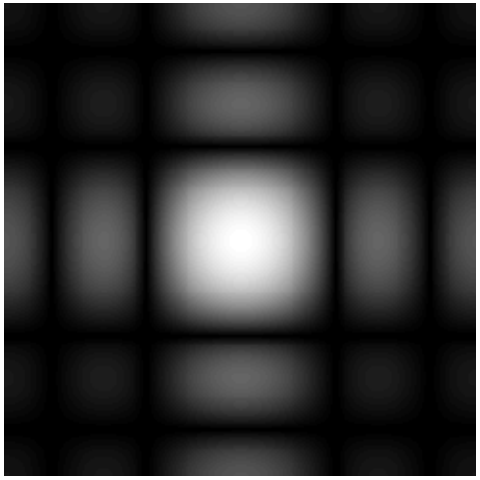
Explanation of Convolution in the Frequency Domain



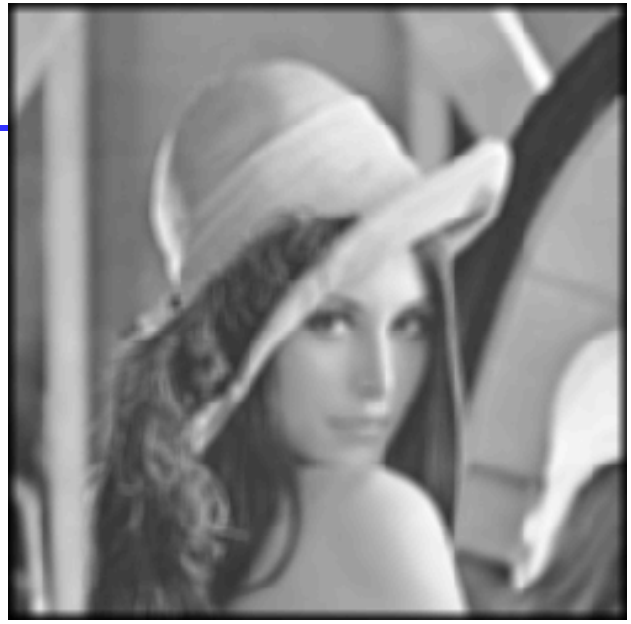
Example

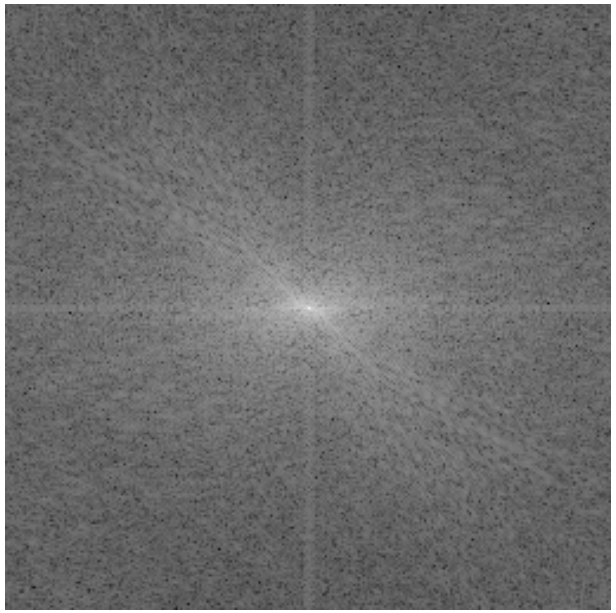
- Given a 2D filter, show the frequency response. Apply to a given image, show original image and filtered image in pixel and freq. domain

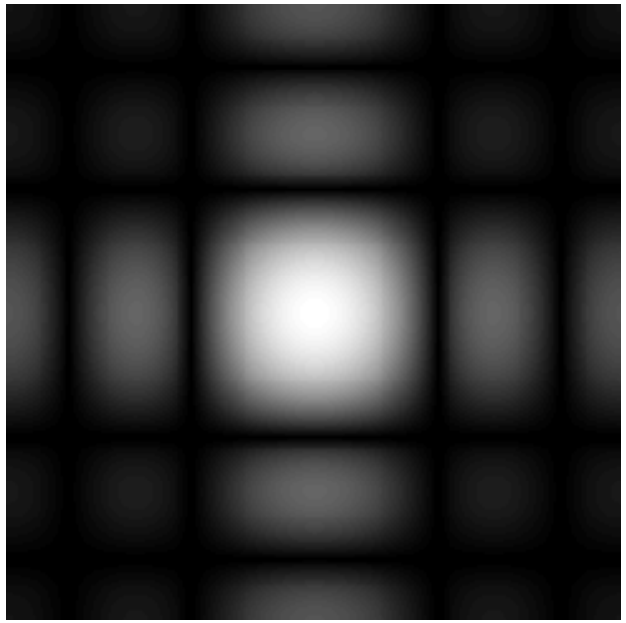
$$h = \frac{1}{25}\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

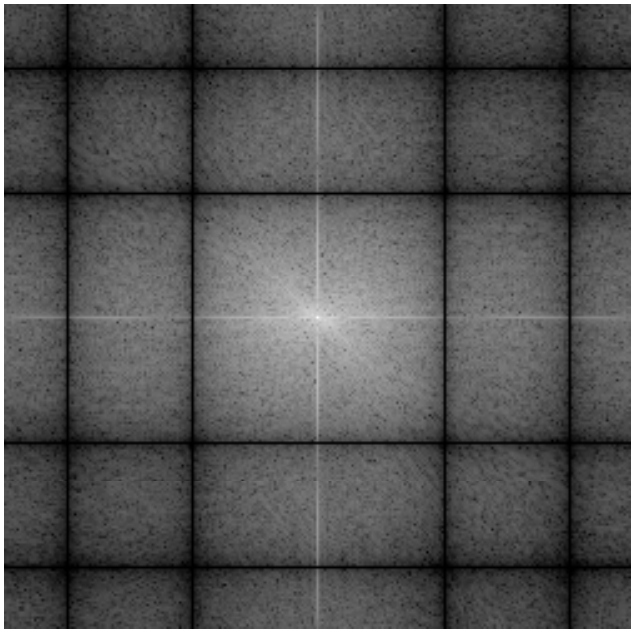


$$h = \frac{1}{25}\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$





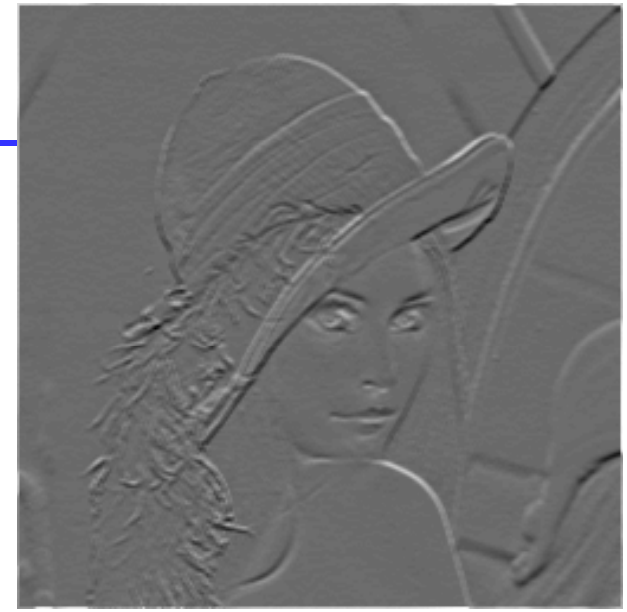


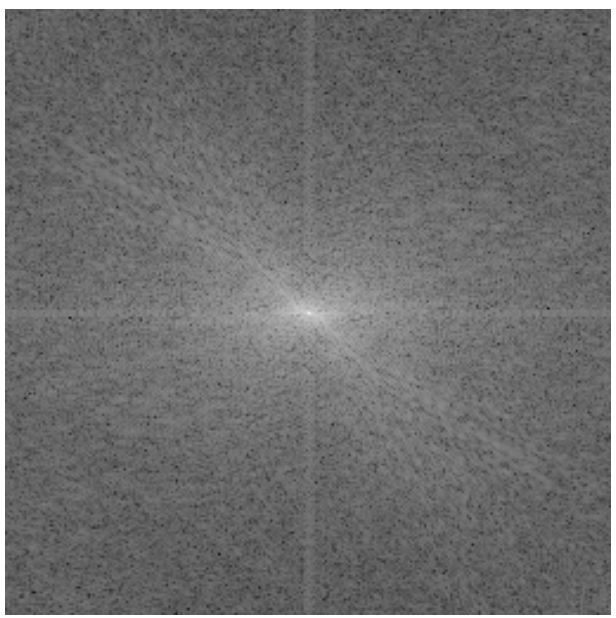


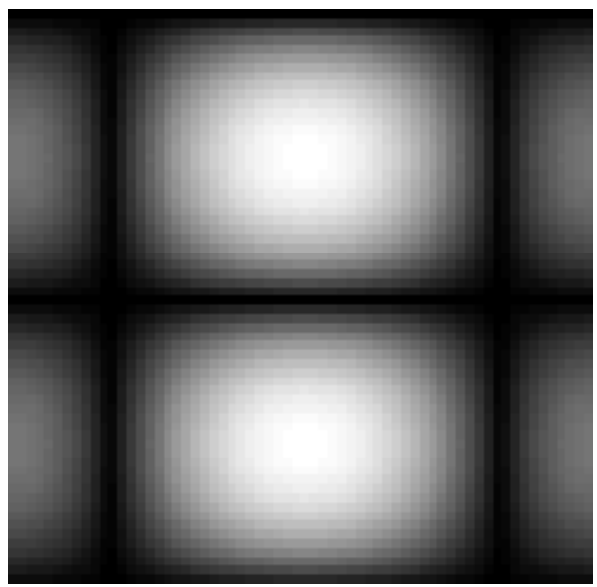
Matlab Program Used

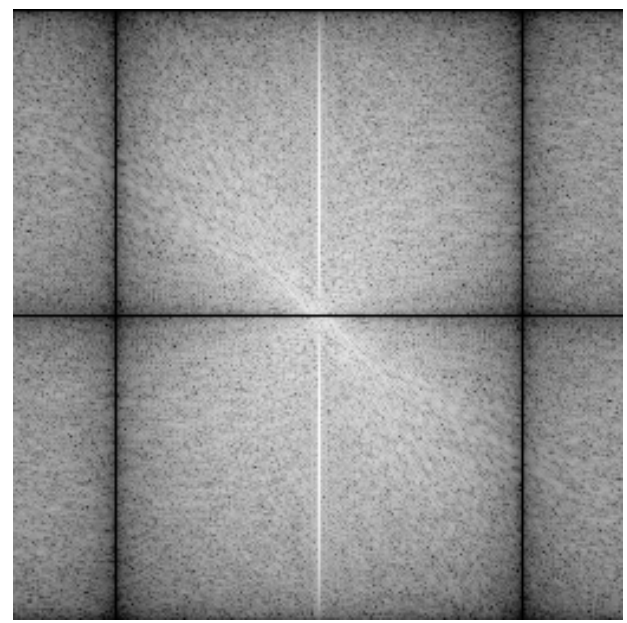
```
x = imread('lena256.bmp');
figure(1); imshow(x);
f = double(x);
ff=abs(fft2(f));
figure(2); imagesc(fftshift(log(ff+1))); colormap(gray);truesize;axis off;
h = ones(5,5)/9;
hf=abs(freqz2(h));
figure(3);imagesc((log(hf+1)));colormap(gray);truesize;axis off;
y = conv2(f, h);
figure(4);imagesc(y);colormap(gray);truesize;axis off;
yf=abs(fft2(y));
figure(5);imagesc(fftshift(log(yf+1)));colormap(gray);truesize;axis off;
```

$$H_1 = \frac{1}{3}\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$





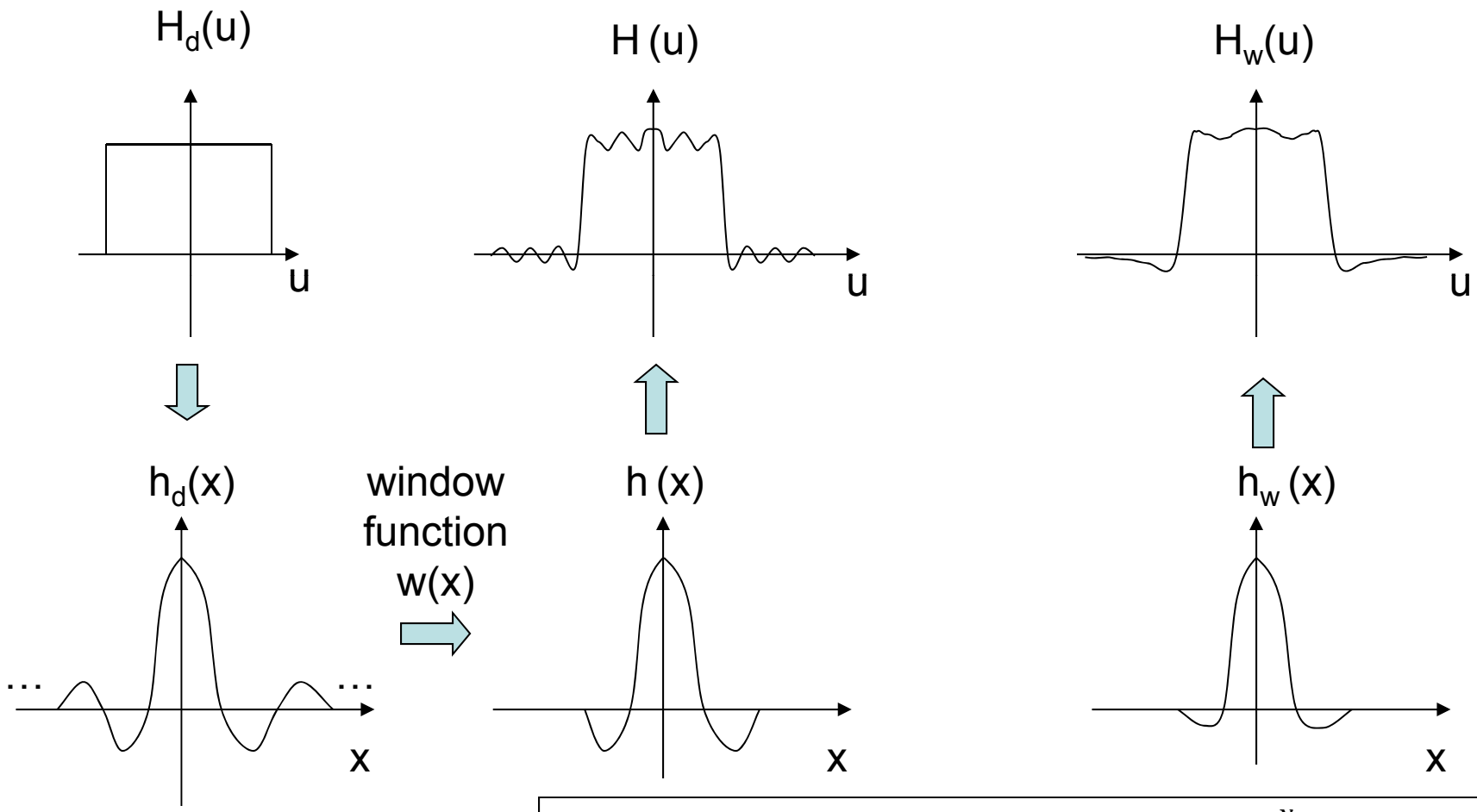




Filter Design

- Given desired frequency response of the filter $H_d(u,v)$
- Perform an inverse transform to obtain the desired impulse response $h_d(m,n)$.
- When $H_d(u,v)$ is band limited, $h_d(m,n)$ is infinitely long
- Truncate $h_d(m,n)$ to yield a realizable $h(m,n)$
- Will distort the original frequency response
- Better approach is to apply a well designed window function over the specified frequency response.

Use of Window Function in Filter Design



Hamming window $w(x) = 0.54 - 0.46\cos(2\pi \frac{x}{X}), \quad 0 < x < X$

Homework

1. Let $f(x,y) = \sin 2\pi f_0 (x+y), \quad h(x,y) = \dfrac{\sin(2\pi f_c x)\sin(2\pi f_c y)}{\pi^2 xy}$

   Find the convolved signal $g(x, y) = f(x, y) * h(x, y)$ for the following two cases:

   a) $f_0/2 < f_c < f_0$; and b) $f_0 < f_c < 2f_0$.

   Hint: do the filtering in the frequency domain. Explain what happened by sketching the original signal, the filter, the convolution process and the convolved signal in the frequency domain.

2. Repeat the previous problem for

$$h(x,y) = \begin{cases} f_c^2, & -\dfrac{1}{2f_c} < \{x,y\} < \dfrac{1}{2f_c} \\ 0, & otherwise \end{cases}$$

Homework (cntd)

3. For the three filters given below (assuming the origin is at the center):

   a) find their Fourier transforms (2D DTFT);

   b) sketch the magnitudes of the Fourier transforms. You should sketch by hand the DTFT as a function of u, when v=0 and when v=1/2; also as a function of v, when u=0 or ½. Also please plot the DTFT as a function of both u and v, using Matlab plotting function.

   c) Compare the functions of the three filters.

In your calculation, you should make use of the separable property of the filter whenever appropriate. If necessary, split the filter into several additive terms such that each term can be calculated more efficiently.

$$H_1 = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad H_2 = \frac{1}{24}\begin{bmatrix} 1 & -2 & 1 \\ -2 & 12 & -2 \\ 1 & -2 & 1 \end{bmatrix} \quad H_3 = \frac{1}{24}\begin{bmatrix} -1 & -2 & -1 \\ -2 & 12 & -2 \\ -1 & -2 & -1 \end{bmatrix}$$

Programming Assignment

1. Write a Matlab or C-program for implementing filtering of a gray scale image. Your program should allow you to specify the filter with an arbitrary size (in the range of k0->k1, l0->l1), read in a gray scale image, and save the filtered image into another file. For simplicity, ignore the boundary effect. That is, do not compute the filtered values for pixels residing on the image boundary. If the filter has a size of *M x N*, the boundary pixels are those in the top and bottom M/2 rows, and left and right most N/2 columns. You should properly normalize the filtered image so that the resulting image values can be saved as 8-bit unsigned characters. Apply the filters given in the previous problem to a test image. Observe on the effect of these filters on your image. Note: you cannot use the MATLAB conv2( ) function. In your report, include your MATLAB code, the original test image and the images obtained with the three filters. Write down your observation of the effect of the filters.
2. Optional: Now write a convolution program assuming the filter is separable. Your program should allow you to specify the horizontal and vertical filters, and call a 1D convolution sub-program to accomplish the 2D convolution. Note: you cannot use the MATLAB conv( ) function.
3. Learn the various options of conv2( ) function in MATLAB using online help manual.

Reading

- Gonzalez and Woods, Sec. 4.1 and 4.2, 4., 3.4, 3.5, 3.5