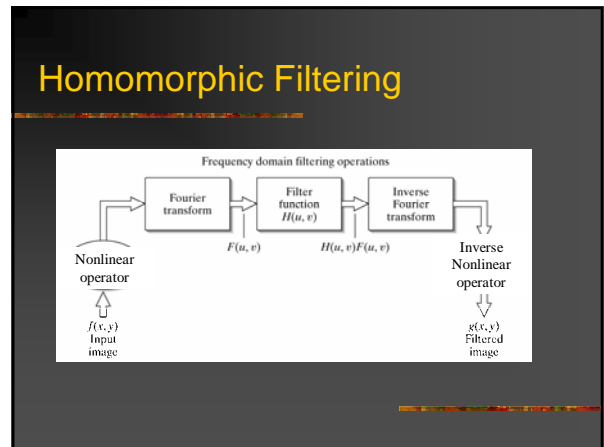
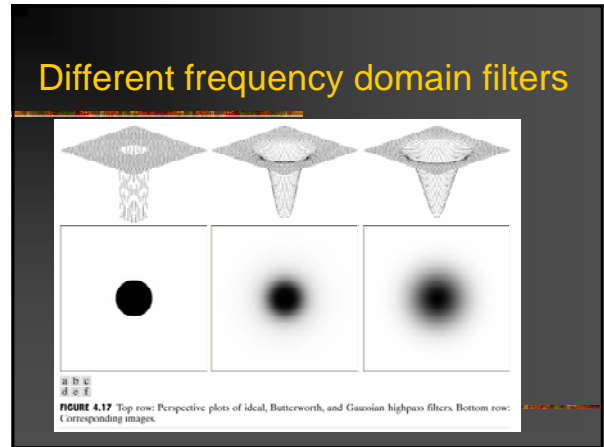
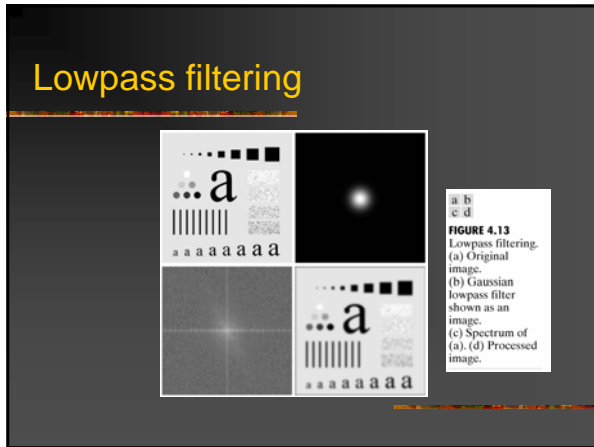
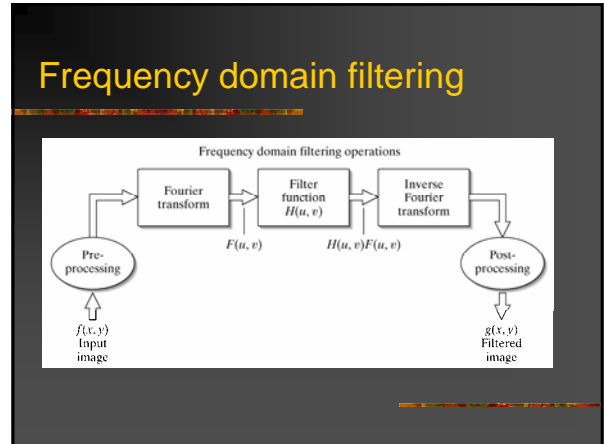
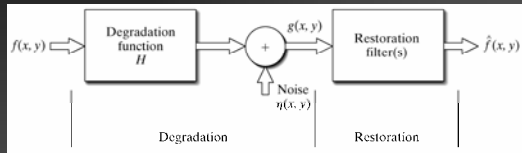


# CmpE 464 Image Processing

## Lecture 9 Image Restoration



## Image Restoration



## Image Restoration

□ Recover an image from

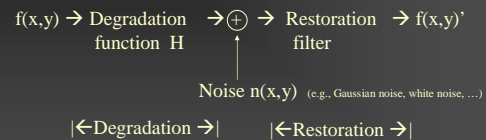
- Noise
- Degradation
- Distortion
- low-resolution

## Image Restoration

- (1) Recover an image that has been degraded by using a priori knowledge of the degradation phenomenon
- (2) The main task is to model the degradation, and apply the inverse process to recover the original image

E.g., remove additive noise; remove motion blurring

## Image degradation process



Spatial domain:  $g(x,y) = h(x,y) \otimes f(x,y) + n(x,y)$

Frequency domain:  $G(u,v) = H(u,v) \cdot F(u,v) + N(u,v)$

## Image Recovering Process

- Noise:
  - Gaussian noise
  - uniform noise
  - white noise
  - salt & pepper noise
  - periodic noise
  - ...

• PDF (probability density function) – measure of random variable z.  
For example: Gaussian noise:

$$p(z) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(z-\mu)^2}{2\sigma^2}}$$

## Image Recovering Process

- In the noise-only case
  - using spatial filtering
$$g(x,y) = f(x,y) + n(x,y)$$

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

Note:

- In the case of periodic noise,  $N(u,v)$  can be estimated from  $G(u,v)$
- Additive noise can be removed by spatial filter

## Image Recovering Process

- In the noise-only case (cont'd)

E.g.,

- arithmetic mean filter:  
Because the local variation is smoothed, the image will be blurred.
- geometric mean filter
- median filter
- Max/min filter

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

## Image Recovering Process

- In the noise-only case (cont'd)

E.g., (cont'd)

- geometric mean filter

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

- contra-harmonic mean filter for removing pepper-and-salt noise

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

- median filter (less blurring, good for impulse noise)

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

- Max/min filter

## Image Recovering Process

- In the noise-only case (cont'd)

E.g., (cont'd)

- Max/min filter

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

Remove pepper noise

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

Remove salt noise

- Midpoint filter

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

Remove randomly distributed noise, e.g., Gaussian or uniform noise

## Image Recovering Process

- recovering from degradation

Degradation function:

$$G(u, v) = H(u, v)F(u, v) + N(u, v)$$

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

$$\hat{F}(u, v) = F(u, v) + \frac{N(u, v)}{H(u, v)}$$

Note: given the degradation model  $H(u, v)$ , we can estimate  $F(u, v)$ .  
Because of the unknown  $N(u, v)$ ,  $\hat{F}(u, v)$  is an estimate value of  $F(u, v)$ .

## Image Recovering Process

- recovering from degradation (cont'd)

Wiener filter (minimum mean square error filter):

- find an estimate  $\hat{f}$ , such that  $e^2$  is minimized  
where  $e^2 = E\{(f - \hat{f})^2\}$

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

Where:  $H(u, v)$ : degradation function

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

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

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

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

## Image Recovering Process

- recovering from degradation (cont'd)

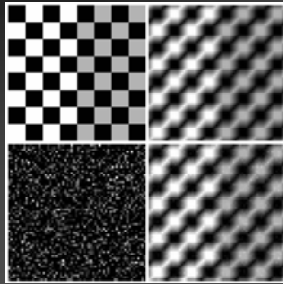
Wiener filter (minimum mean square error filter):

- Special case: white noise  
The spectrum  $|N(u, v)|^2 = \text{constant}$

- In most cases, the power spectrum  $|F(u, v)|^2$  is unknown, the following approximation is commonly used (where  $K$  is a specified constant)

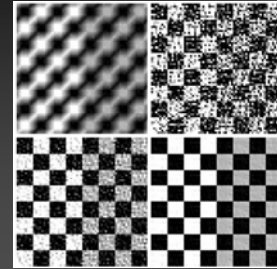
$$\hat{F}(u, v) = \left[ \frac{1}{H(u, v)} \frac{|H(u, v)|^2}{|H(u, v)|^2 + K} \right] G(u, v)$$

## Image Blur



a b  
c d  
**FIGURE 5.7**  
(a) Original image. (b) Image blurred using fspecial with len = 7, and theta = -45 degrees. (c) Noise image. (d) Sum of (b) and (c).

## Image Blur +Noise



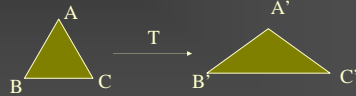
a b  
c d  
**FIGURE 5.8**  
(a) Blurred, noisy image. (b) Result of inverse filtering. (c) Result of Wiener filtering using a constant ratio. (d) Result of Wiener filtering using autocorrelation functions.

## Geometric distortion correction

- Distortion



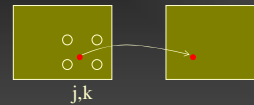
- Geometric transform



$$T = \begin{bmatrix} a1 & b1 & c1 \\ a2 & b2 & c2 \\ 0 & 0 & 1 \end{bmatrix}$$

## Geometric distortion correction

- Gray level interpolation (bilinear interpolation)

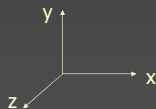


Interpolate the value at (j,k) position by using four neighbor pixels

- Super-resolution using multiple images
- Super-resolution using image examples or database

## Points in 3D Space

- A 3D point (x,y,z) – x,y, and z coordinates
- We use column vectors to represent points
- Homogeneous coordinates of a 3D point (x,y,z,1)
- Transformation will be performed using 4x4 matrix



## Extending to 3D

- Homogeneous coordinates in 3D

- $[x,y,z,1]^T$  (x,y,z,w)

- Matrices of this form:

$$\begin{bmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} ax+by+cz+d \\ ex+fy+gz+h \\ ix+jy+kz+l \\ 1 \end{bmatrix}$$

- 4x4 Matrices instead of 3x3 for 3D

## Translation

$$T(d,h,l) = \begin{bmatrix} 1 & 0 & 0 & d \\ 0 & 1 & 0 & h \\ 0 & 0 & 1 & l \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & 0 & d \\ 0 & 1 & 0 & h \\ 0 & 0 & 1 & l \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x+d \\ y+h \\ z+l \\ 1 \end{bmatrix}$$

## Scaling

$$S(a,f,k) = \begin{bmatrix} a & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & k & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} a & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & k & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} ax \\ fy \\ kz \\ 1 \end{bmatrix}$$

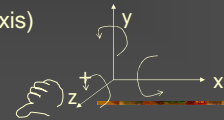
## What about Rotation?

- How can we convert this to 3D?

$$\begin{bmatrix} \cos \theta & -\sin \theta & 0 & 0 \\ \sin \theta & \cos \theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x \cos \theta - y \sin \theta \\ x \sin \theta + y \cos \theta \\ z \\ 1 \end{bmatrix}$$

## 3D Rotation

- 3D rotation is done around a rotation **axis**
- Fundamental rotations – rotate about x, y, or z axes
- Counter-clockwise rotation is referred to as positive rotation (when you look down negative axis)



## Rotation about Z axis

$$\begin{bmatrix} \cos \theta & -\sin \theta & 0 & 0 \\ \sin \theta & \cos \theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x \cos \theta - y \sin \theta \\ x \sin \theta + y \cos \theta \\ z \\ 1 \end{bmatrix}$$

## 3D transformation

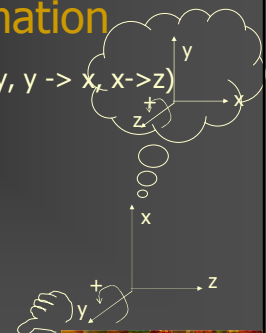
- Rotation about y ( $z \rightarrow y, y \rightarrow x, x \rightarrow z$ )

$$z' = z \cos(\theta) - x \sin(\theta)$$

$$x' = z \sin(\theta) + x \cos(\theta)$$

$$y' = y$$

$$\begin{bmatrix} \cos(\theta) & \sin(\theta) & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\theta) & \cos(\theta) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



## 3D transformation

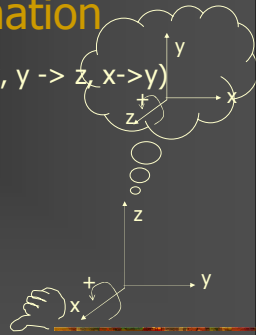
- Rotation about x ( $z \rightarrow x, y \rightarrow z, x \rightarrow y$ )

$$y' = y \cos(\theta) - z \sin(\theta)$$

$$z' = y \sin(\theta) + z \cos(\theta)$$

$$x' = x$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) & 0 \\ 0 & \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



## The 3 Rotation Matrices

$$R_z(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

## Skew or Shear

$$H_x(b) = \begin{bmatrix} 1 & b & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

## Matrix for Perspective Projection?

- We need division to do projection!
- But, matrix multiplication only does multiplication and addition
- What about:

$$M_{\text{per}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1/d & 0 \end{bmatrix}$$

$$M_{\text{per}} \cdot P = \begin{bmatrix} 1 & 0 & 0 & 0 & | & x & | & x \\ 0 & 1 & 0 & 0 & | & y & | & y \\ 0 & 0 & 1 & 0 & | & z & | & z \\ 0 & 0 & 1/d & 0 & | & 1 & | & z/d \end{bmatrix}$$

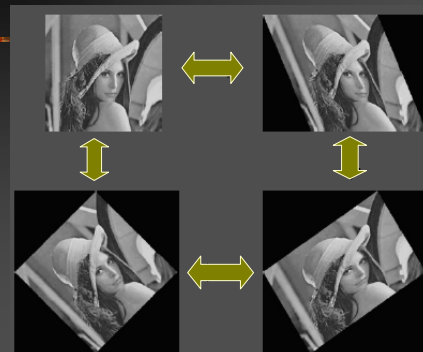
## i-pt Perspective Transformation

$$\text{1 - point} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1/d & 1 \end{bmatrix}$$

$$\text{2 - point} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & p & q & 1 \end{bmatrix}$$

$$\text{3 - point} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ p & q & r & 1 \end{bmatrix}$$

## Warping Example



## Warping Example

