Image Processing and Visual Communications

Frequency Domain Image Restoration

Zhou Wang

Dept. of Electrical and Computer Engineering
University of Waterloo

Linear Filtering

- **Spatial Domain Filtering: Convolution**

$$x(m,n) \longrightarrow \boxed{h(m,n)} \longrightarrow y(m,n)$$

$$y(m,n) = \sum_{k,l=-\infty}^{\infty} h(k,l)x(m-k,n-l) = h(m,n) \otimes x(m,n)$$

- **Frequency Domain Filtering: Multiplication**

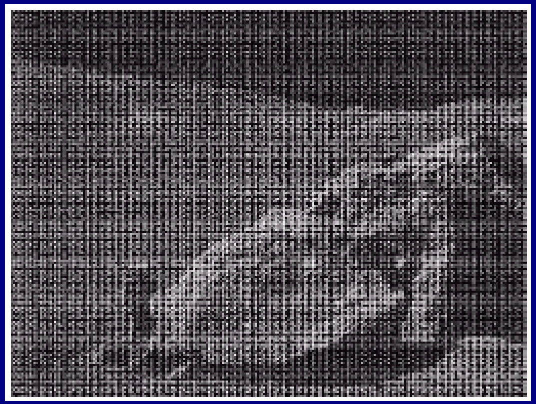
$$X(u,v) = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} x(m,n)e^{-j(um+vn)}$$

$$x(m,n) \otimes h(m,n) \quad \underset{F^{-1}}{\overset{F}{\rightleftarrows}} \quad X(u,v)H(u,v)$$

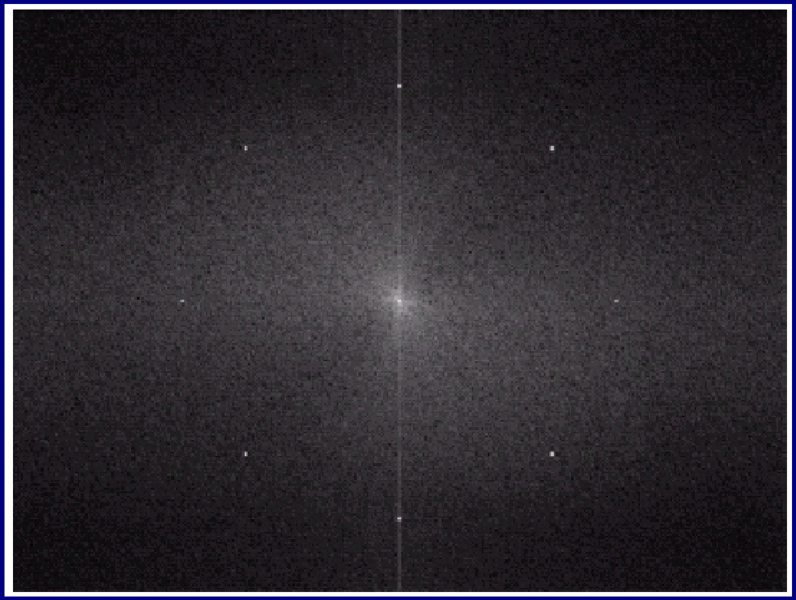
$H(u, v)$: low-pass

Linear Filtering: Periodic Noise Removal

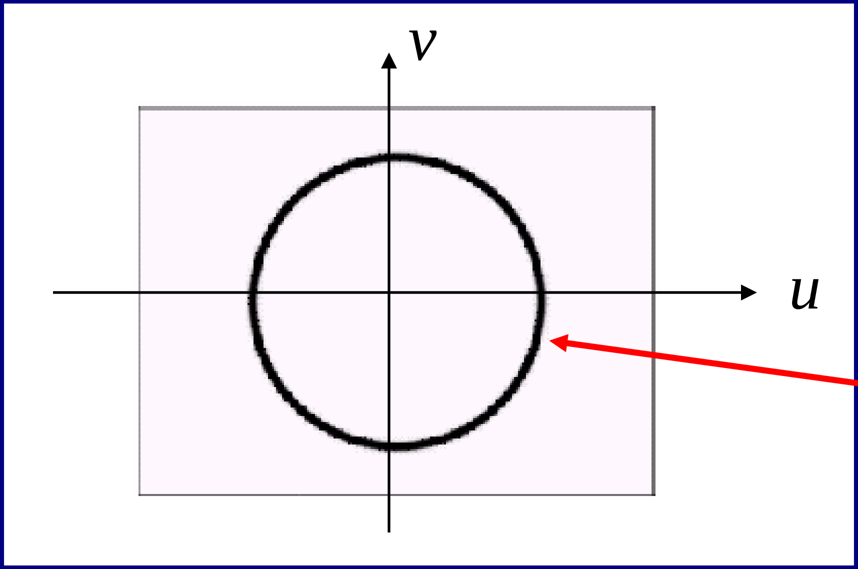
- **Frequency Domain Filtering**



spatial domain



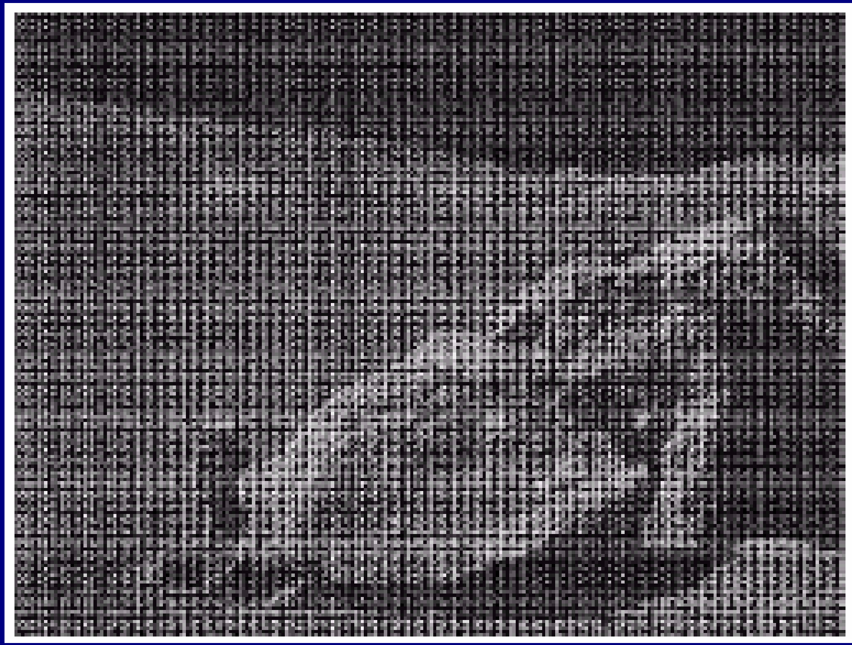
frequency domain



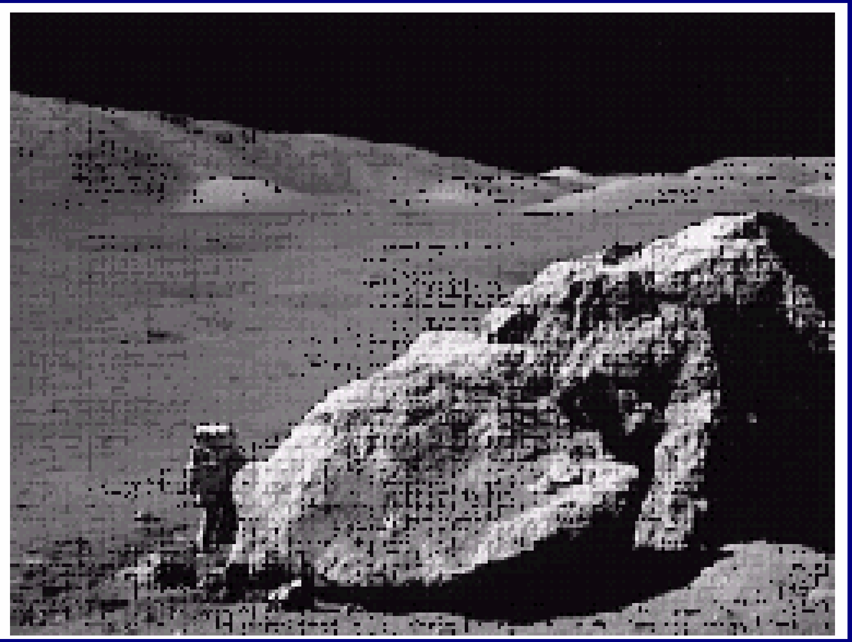
band rejection

Linear Filtering: Periodic Noise Removal

- **Frequency Domain Filtering**

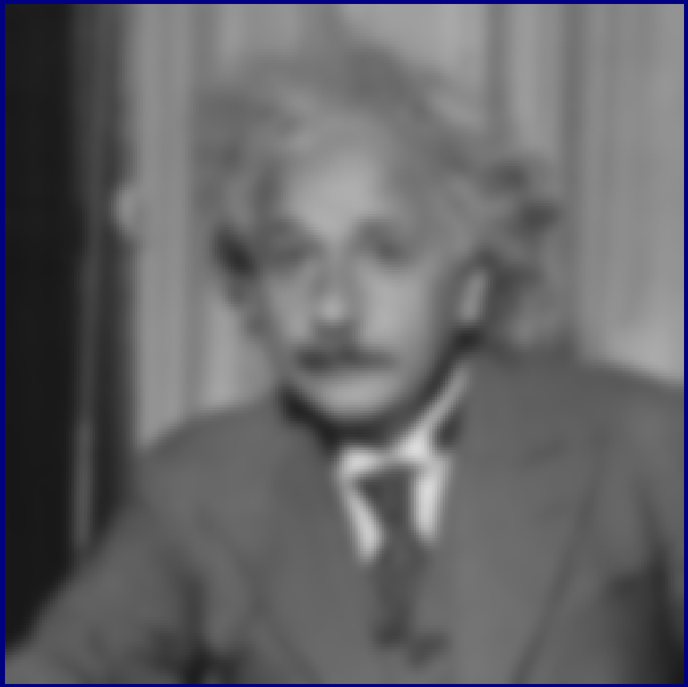


before



after

Blur



out-of-focus blur

motion blur

From Prof.
Xin Li

Question 1: How do you know they are blurred?
I've not shown you the originals!

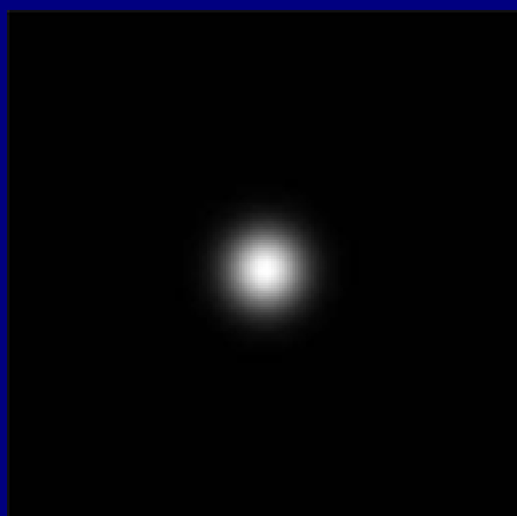
Question 2: How do I deblur an image?

Linear Blur Model

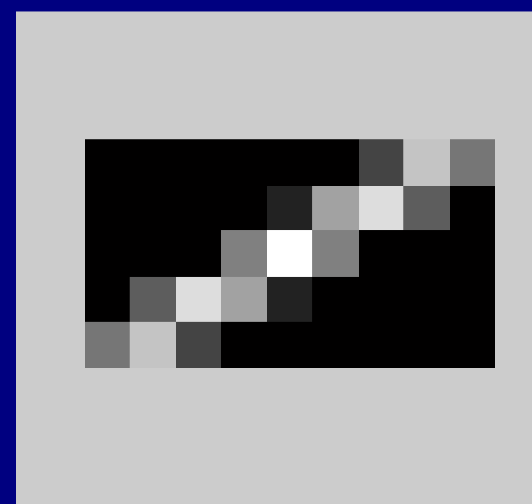
- **Spatial domain**

$$x(m,n) \longrightarrow \boxed{h(m,n)} \longrightarrow y(m,n)$$

blurring filter



Gaussian blur



motion blur

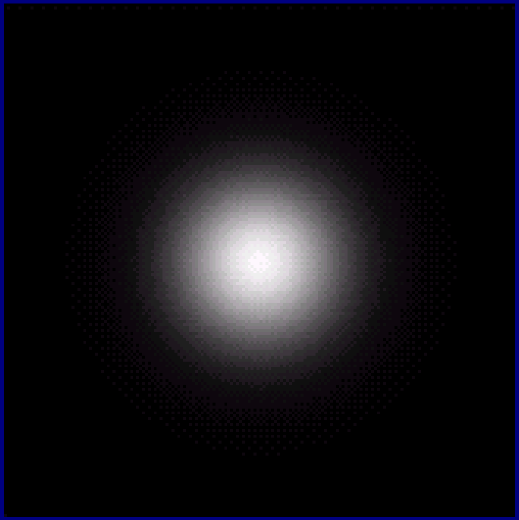
From Prof. Xin Li

Linear Blur Model

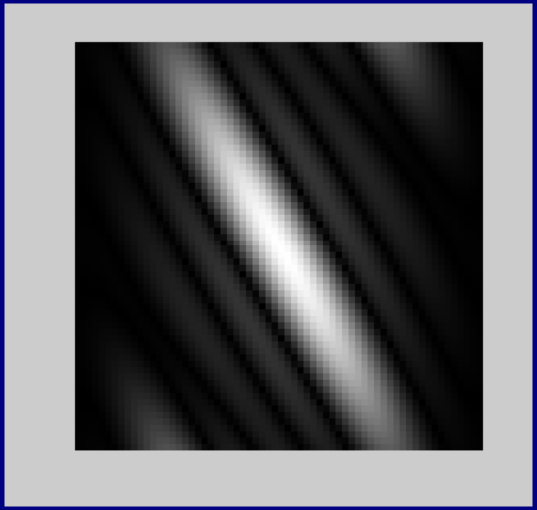
- **Frequency (2D-DFT) domain**

$$X(u,v) \longrightarrow \boxed{H(u,v)} \longrightarrow Y(u,v)$$

blurring filter



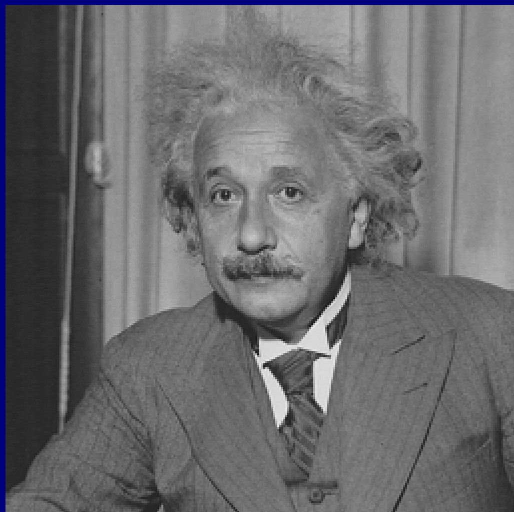
Gaussian blur



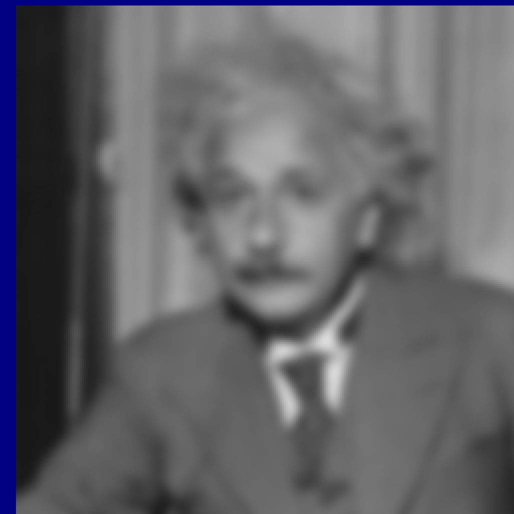
motion blur

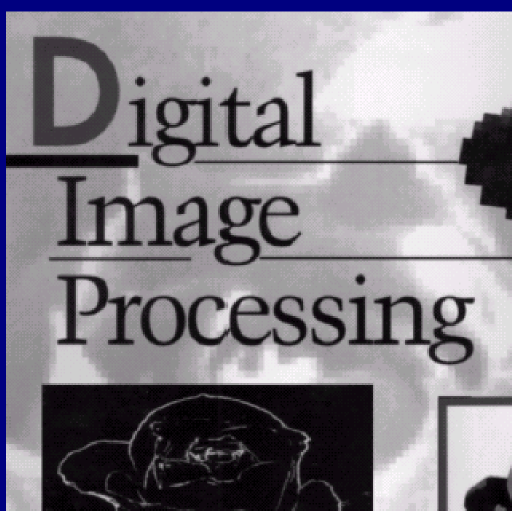
From Prof. Xin Li

Blurring Effect

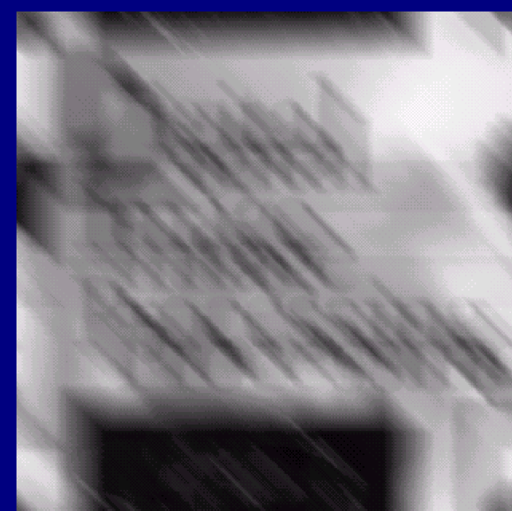


Gaussian blur





motion blur



From [Gonzalez & Woods]

Image Restoration: Deblurring/Deconvolution

$$x(m,n) \longrightarrow \boxed{h(m,n)} \longrightarrow y(m,n) \longrightarrow \boxed{g(m,n)} \longrightarrow \hat{x}(m,n)$$

blurring filter

deblurring/ deconvolution filter

- **Non-blind deblurring/deconvolution**

  Given: observation $y(m,n)$ and blurring function $h(m,n)$

  Design: $g(m,n)$, such that the **distortion** between $x(m,n)$ and $\hat{x}(m,n)$ is minimized

- **Blind deblurring/deconvolution**

  Given: observation $y(m,n)$

  Design: $g(m,n)$, such that the **distortion** between $x(m,n)$ and $\hat{x}(m,n)$ is minimized

Deblurring: Inverse Filtering

$$x(m,n) \longrightarrow \boxed{h(m,n)} \longrightarrow y(m,n) \longrightarrow \boxed{g(m,n)} \longrightarrow \hat{x}(m,n)$$

blurring filter

inverse filter

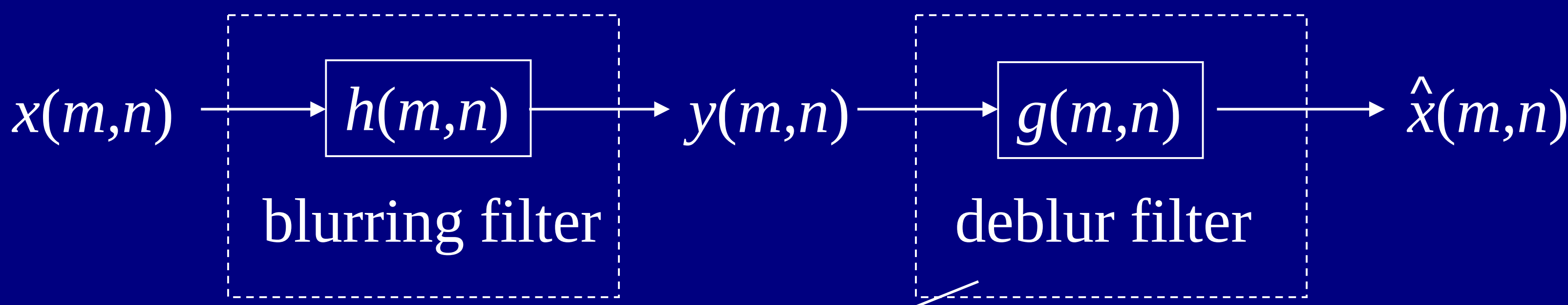
$$X(u,v)\, H(u,v) = Y(u,v)$$

$$X(u,v) = \frac{Y(u,v)}{H(u,v)} = \frac{1}{H(u,v)}\, Y(u,v)$$

$$\boxed{G(u,v) = \frac{1}{H(u,v)}}$$

Exact recovery!

Deblurring: Pseudo-Inverse Filtering



Inverse filter: $G(u,v) = \dfrac{1}{H(u,v)}$

What if at some (u,v), $H(u,v)$ is 0 (or very close to 0) ?

Pseudo-inverse filter:

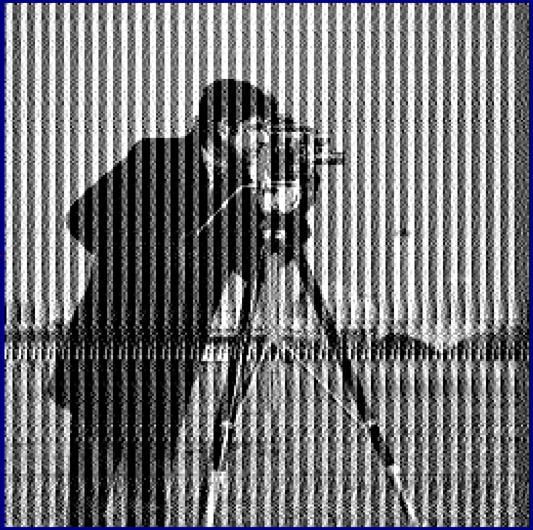
$$G(u,v) = \begin{cases} \dfrac{1}{H(u,v)} & |H(u,v)| > \delta \\ 0 & |H(u,v)| \le \delta \end{cases}$$

small threshold

Inverse and Pseudo-Inverse Filtering

$$G(u,v) = \frac{1}{H(u,v)}$$

blurred image

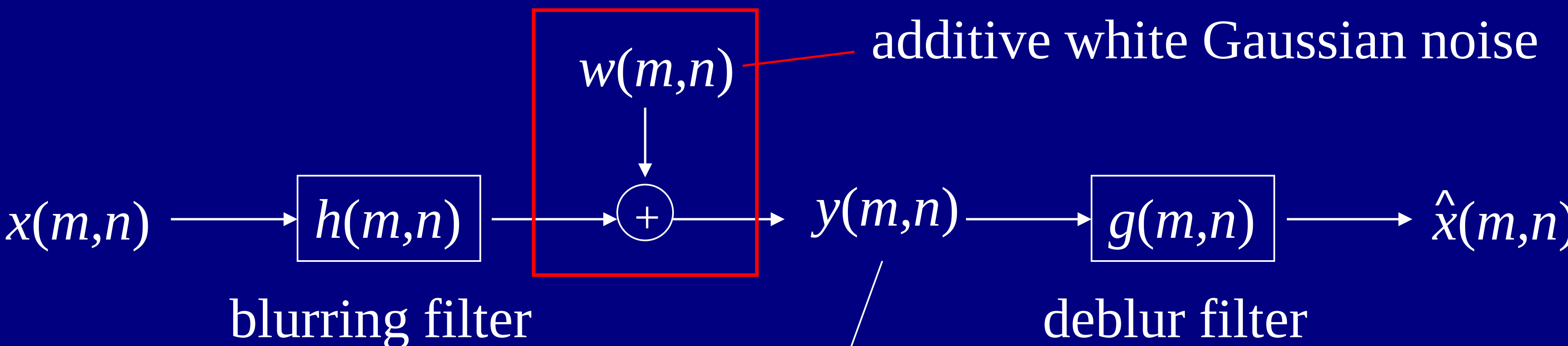


$$G(u,v) = \begin{cases} \dfrac{1}{H(u,v)} & |H(u,v)| > \delta \\ 0 & |H(u,v)| \le \delta \end{cases}$$

$\delta = 0.1$

Adapted from Prof. Xin Li

More Realistic Distortion Model



$$Y(u,v) = X(u,v)\,H(u,v) + W(u,v)$$

- **What happens when an inverse filter is applied?**

$$\hat{X}(u,v) = Y(u,v)G(u,v) = \frac{X(u,v)H(u,v) + W(u,v)}{H(u,v)}$$

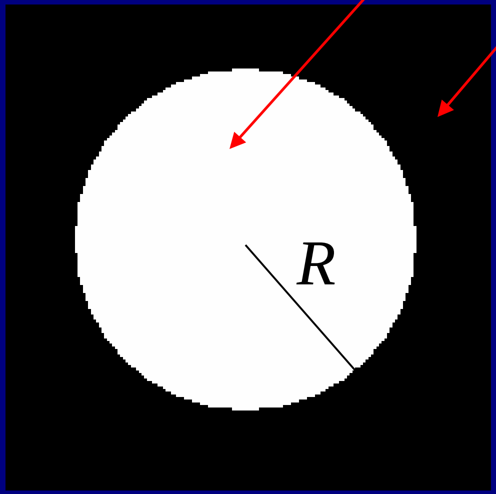
$$= X(u,v) + \frac{W(u,v)}{H(u,v)}$$

close to zero at high frequencies

Radially Limited Inverse Filtering

Radially limited inverse filter:

$$G(u,v)=\begin{cases}\dfrac{1}{H(u,v)} & \sqrt{u^2+v^2}\le R\\ 0 & \sqrt{u^2+v^2}>R\end{cases}$$



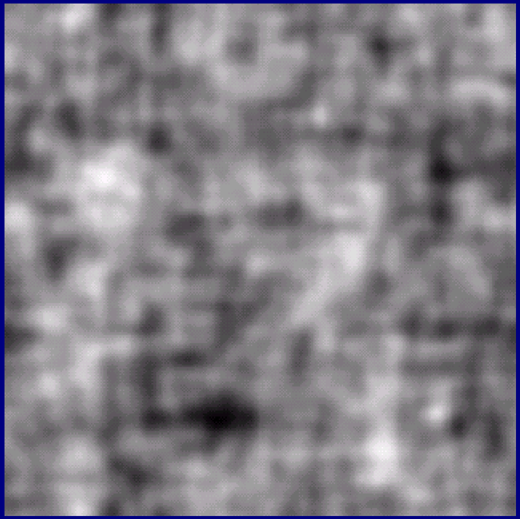
- **Motivation**
  - Energy of image signals is concentrated at low frequencies
  - Energy of noise uniformly is distributed over all frequencies
  - Inverse filtering of image signal dominated regions only

Radially Limited Inverse Filtering

Image size: 480x480

Original

Blurred

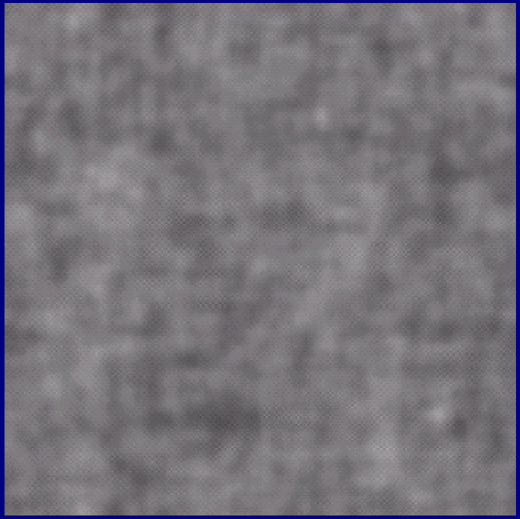


Inverse filtered

Radially limited inverse filtering:

$R = 40$

$R = 70$



$R = 85$

From [Gonzalez & Woods]

Wiener (Least Square) Filtering

Wiener filter: $$G(u,v) = \frac{H^*(u,v)}{|H(u,v)|^2 + K}$$

$$K = \frac{\sigma_W^2}{\sigma_X^2}$$

noise power

signal power

- Optimal in the least MSE sense, i.e.
  G(*u*, *v*) is the best possible linear filter that minimizes

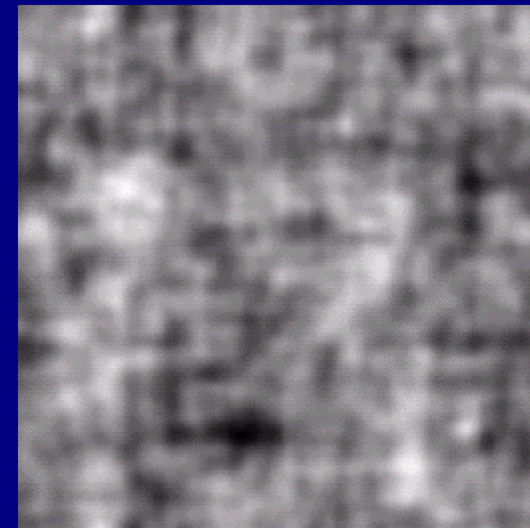
$$error\ energy = E\left\{\left|\hat{X}(u,v) - X(u,v)\right|^2\right\}$$

- Have to estimate signal and noise power

Wiener Filtering

Blurred image

Inverse filtering



Radially limited inverse filtering
$R = 70$

Wiener filtering

From [Gonzalez & Woods]

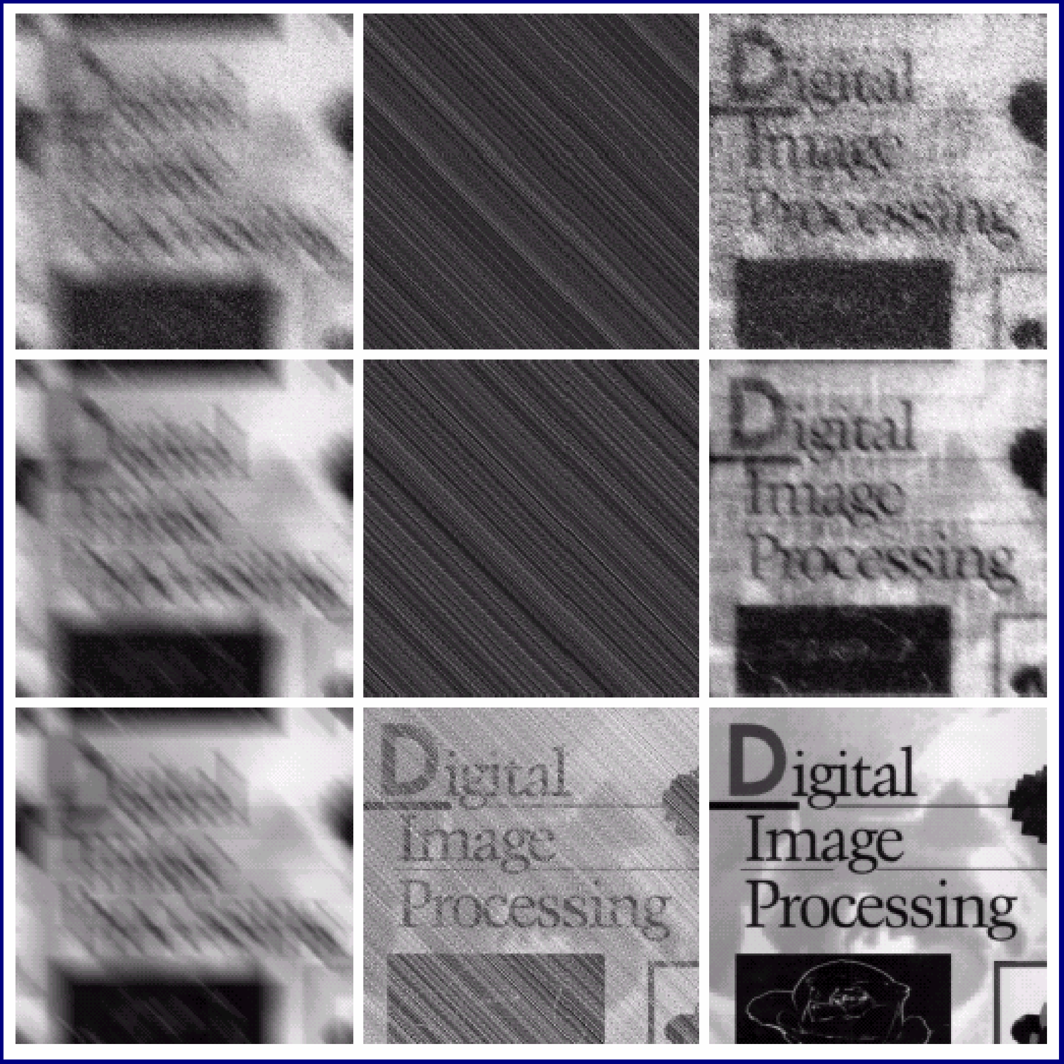
Inverse vs. Wiener Filtering

distorted | inverse filtering | Wiener filtering

motion blur
+
noise

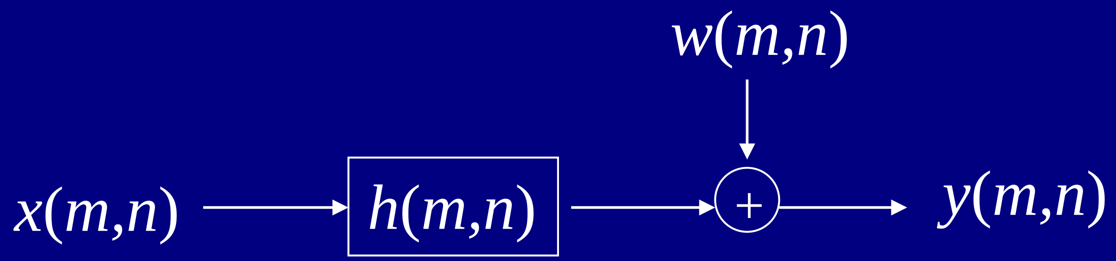
less noise

less noise



From [Gonzalez & Woods]

Wiener Image Denoising



- What if no blur, but only noise, i.e. $h(m,n)$ is an impulse or $H(u, v) = 1$?

Wiener filter:

$$G(u,v) = \frac{H^*(u,v)}{|H(u,v)|^2 + K}$$

where

$$K = \frac{\sigma_W^2}{\sigma_X^2}$$

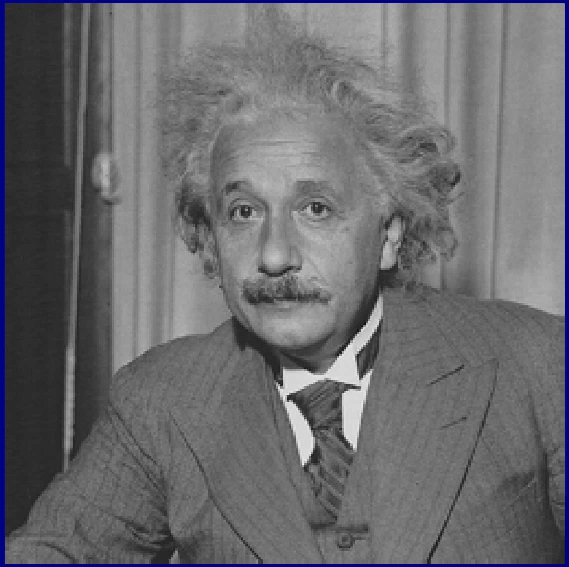
for $H(u,v) = 1$

Wiener denoising filter:

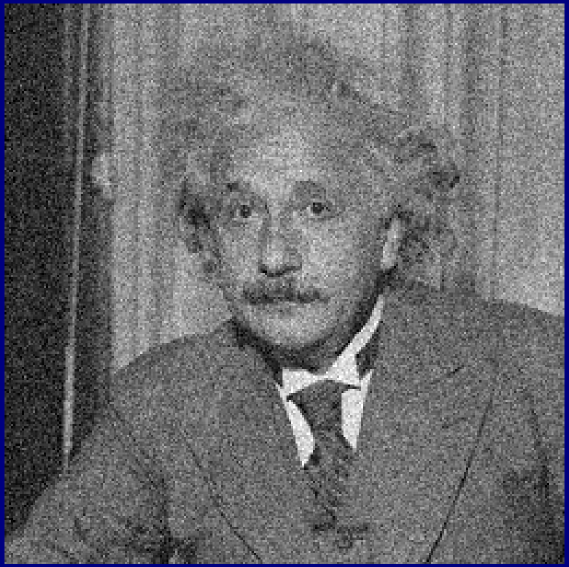
$$G(u,v) = \frac{1}{1+K} = \frac{1}{1+\sigma_W^2/\sigma_X^2} = \frac{\sigma_X^2}{\sigma_X^2 + \sigma_W^2}$$

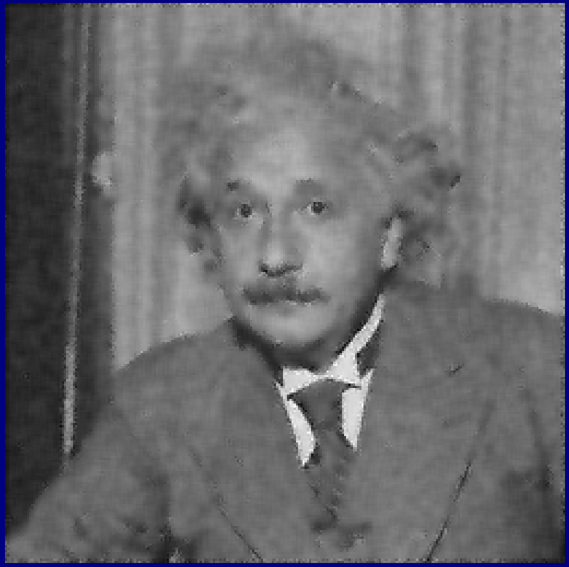
Typically applied locally in space

Wiener Image Denoising



adding noise
noise var = 400





local Wiener
denoising

Summary of Linear Image Restoration Filters

Inverse filter:

$$G(u,v) = \frac{1}{H(u,v)}$$

Pseudo-inverse filter:

$$G(u,v) = \begin{cases} \dfrac{1}{H(u,v)} & |H(u,v)| > \delta \\ 0 & |H(u,v)| \le \delta \end{cases}$$

Radially limited inverse filter:

$$G(u,v) = \begin{cases} \dfrac{1}{H(u,v)} & \sqrt{u^2+v^2} \le R \\ 0 & \sqrt{u^2+v^2} > R \end{cases}$$

Wiener filter:

$$G(u,v) = \frac{H^*(u,v)}{|H(u,v)|^2 + K}$$

where

$$K = \frac{\sigma_W^2}{\sigma_X^2}$$

Wiener denoising filter:

$$G(u,v) = \frac{\sigma_X^2}{\sigma_X^2 + \sigma_W^2}$$

Examples

- A blur filter $h(m,n)$ has a 2D-DFT given by

$$H(u,v) = \begin{bmatrix} 1 & -0.3-0.3j & 0 & -0.3+0.3j \\ -0.3-0.3j & 0.1j & 0 & 0.1 \\ 0 & 0 & 0 & 0 \\ -0.3+0.3j & 0.1 & 0 & -0.1j \end{bmatrix}$$

- Find the deblur filter $G(u,v)$ using
  1) The inverse filtering approach
  2) The pseudo-inverse filtering approach, with $\delta = 0.05$
  3) The pseudo-inverse filtering approach, with $\delta = 0.2$
  4) Wiener filtering approach, with $\sigma_X^2 = 625$ and $\sigma_W^2 = 125$

Examples

1) Inverse filter

$$G(u,v) = \frac{1}{H(u,v)} = \begin{bmatrix} 1 & -1.67+1.67j & Inf & -1.67-1.67j \\ -1.67+1.67j & -10j & Inf & 10 \\ Inf & Inf & Inf & Inf \\ -1.67-1.67j & 10 & Inf & 10j \end{bmatrix}$$

2) Pseudo-inverse filter, with $\delta = 0.05$

$$G(u,v) = \begin{cases} \dfrac{1}{H(u,v)} & |H(u,v)| > \delta \\ 0 & |H(u,v)| \le \delta \end{cases} = \begin{bmatrix} 1 & -1.67+1.67j & 0 & -1.67-1.67j \\ -1.67+1.67j & -10j & 0 & 10 \\ 0 & 0 & 0 & 0 \\ -1.67-1.67j & 10 & 0 & 10j \end{bmatrix}$$

Examples

3) Pseudo-inverse filter, with $\delta = 0.2$

$$G(u,v)=\begin{cases}\dfrac{1}{H(u,v)} & |H(u,v)|>\delta\\ 0 & |H(u,v)|\le\delta\end{cases}=\begin{bmatrix}1 & -1.67+1.67j & 0 & -1.67-1.67j\\ -1.67+1.67j & 0 & 0 & 0\\ 0 & 0 & 0 & 0\\ -1.67-1.67j & 0 & 0 & 0\end{bmatrix}$$

4) Wiener filter, with $\sigma_X^2 = 625$ and $\sigma_W^2 = 125$

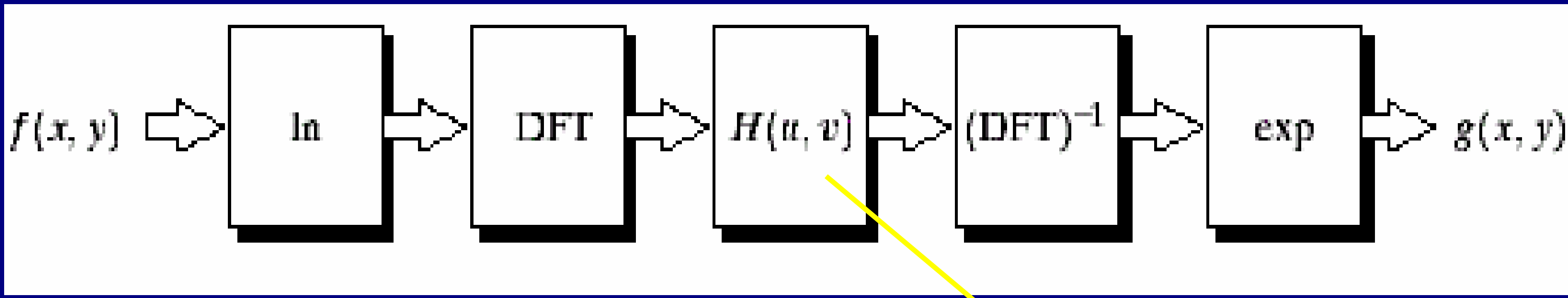
$$K=\frac{\sigma_W^2}{\sigma_X^2}=\frac{125}{625}=0.2$$

$$G(u,v)=\frac{H^*(u,v)}{|H(u,v)|^2+K}=\begin{bmatrix}0.83 & -0.79+0.79j & 0 & -0.79-0.79j\\ -0.79+0.79j & -0.48j & 0 & 0.48\\ 0 & 0 & 0 & 0\\ -0.79-0.79j & 0.48 & 0 & 0.48j\end{bmatrix}$$

Advanced Image Restoration

- **Adaptive Processing**
  - Spatial adaptive
  - Frequency adaptive
- **Nonlinear Processing**
  - Thresholding, coring …
  - Iterative restoration
- **Advanced Transformation / Modeling**
  - Advanced image transforms, e.g., wavelet …
  - Statistical image modeling
- **Blind Deblurring / Deconvolution**

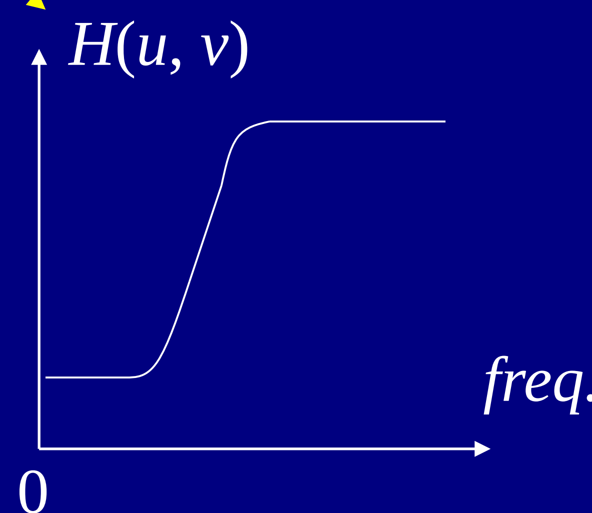
Frequency-domain Method: Homomorphic Filtering



$$f(x, y) = \underbrace{i(x, y)}_{\text{illumination}}\ \underbrace{r(x, y)}_{\text{reflectance}}$$

illumination
(slowly varying)
(low-frequency)

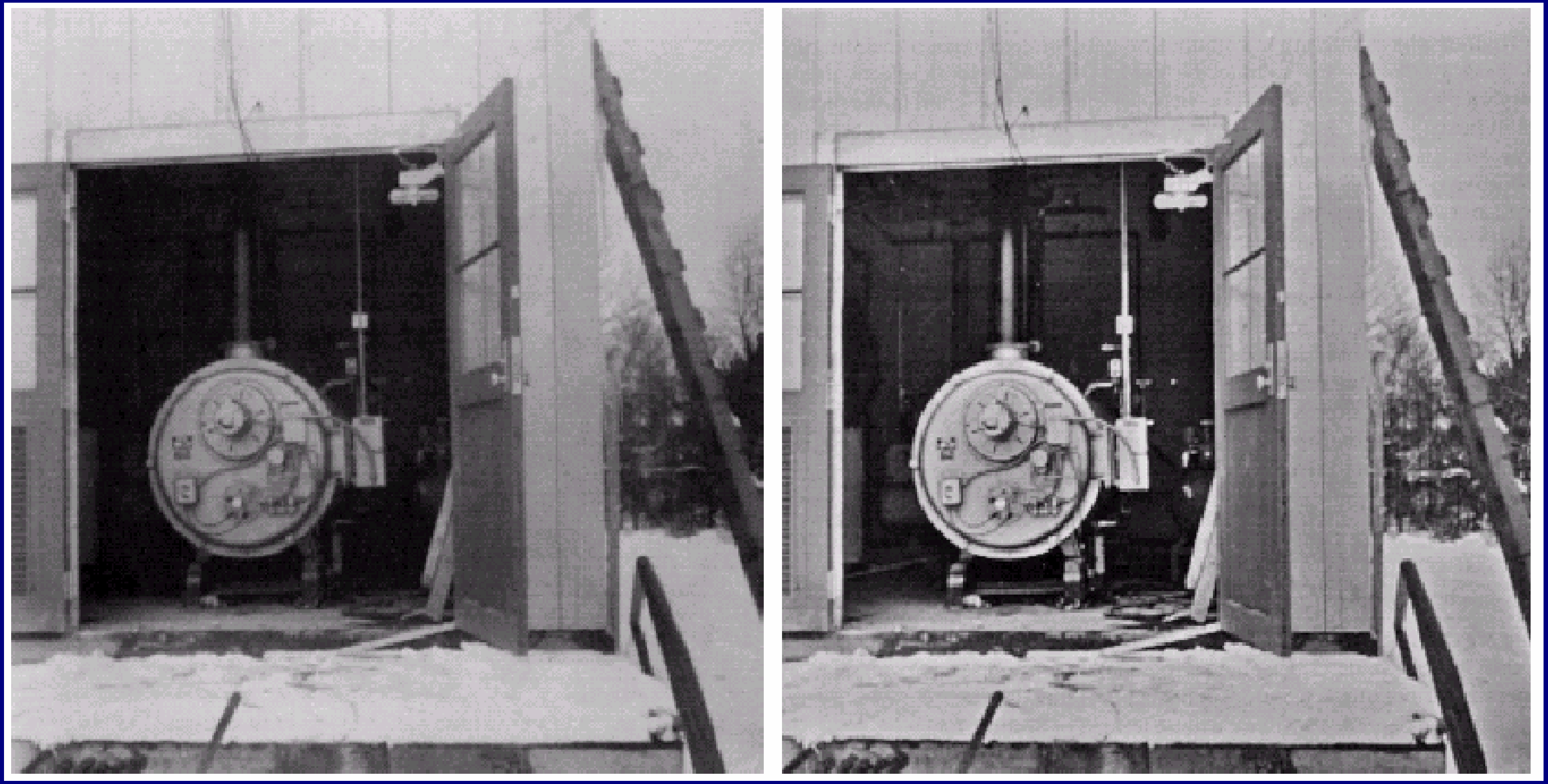
reflectance
(fastly varying)
(high-frequency)



$$\ln[f(x, y)] = \ln[i(x, y)] \underbrace{+}_{} \ln[r(x, y)]$$

Key: linear separation

Frequency-domain Method: Homomorphic Filtering



before after