数字图像处理 Matlab

图像噪声处理

图像噪声处理

- 基本概念
- 椒盐噪声的消除
  - 中值滤波
- 加性高斯白噪声的消除
  - 2D 卷积和DFT
- 周期性噪声的消除
  - 带阻滤波器和陷波滤波器

基本概念

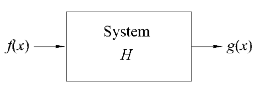
- 噪声的来源
  - 传感器（e.g.,热干扰或电干扰）
  - 环境条件（雨、雪等等）
- 为什么要消除噪声?
  - 不好看
  - 对压缩的坏影响
  - 对图像分析的坏影响

噪声模型

- 简单假定
  - 噪声与信号无关
- 噪声类型
  - 空间位置无关的噪声
    - 椒盐噪声
    - 加性高斯白噪声
  - 空间关联噪声
    - 周期性噪声

噪声去除技术

- 线性滤波
- 非线性滤波



Linear system

$$H[a_i f_i(x) + a_j f_j(x)] = a_i H[f_i(x)] + a_j H[f_j(x)] = a_i g_i(x) + a_j g_j(x)$$

椒盐噪声（脉冲噪声）

- 定义

图像中每个像素以概率 p/2（0<p<1）被白点（盐）或黑点（胡椒）污染

$$Y(i,j)=\begin{cases}255 & \text{with probability of p/2} \\ 0 & \text{with probability of p/2} \\ X(i,j) & \text{with probability of 1-p}\end{cases}$$

noisy pixels

clean pixels

$1 \le i \le H, 1 \le j \le W$

X: noise-free image, Y: noisy image

注意：在一些应用中，噪声点不仅仅是黑点或白点，这加大了消除椒盐噪声的难度

Numerical Example

P=0.1

128	128	128	128	128	128	128	128	128	128
128	128	128	128	128	128	128	128	128	128
128	128	128	128	128	128	128	128	128	128
128	128	128	128	128	128	128	128	128	128
128	128	128	128	128	128	128	128	128	128
128	128	128	128	128	128	128	128	128	128
128	128	128	128	128	128	128	128	128	128
128	128	128	128	128	128	128	128	128	128
128	128	128	128	128	128	128	128	128	128
128	128	128	128	128	128	128	128	128	128

X

128	128	255	0	128	128	128	128	128	128
128	128	128	128	0	128	128	128	128	0
128	128	128	128	128	128	128	128	128	128
128	128	0	128	128	128	128	128	128	128
128	128	128	128	128	128	128	128	128	128
128	128	128	128	128	128	128	128	128	128
128	128	128	128	128	128	128	128	128	128
0	128	128	128	128	255	128	128	128	128
128	128	128	128	128	128	128	128	128	255
128	128	128	128	128	128	128	255	128	128

Y

噪声等级 P=0.1 意味着噪声点（白或黑）的出现概率为0.1，图中红色数字

MATLAB Command

>Y = IMNOISE(X,'salt & pepper',p)

- 输入图像的像素值在[0,1]区间
- P缺省值为0.05
- 更改类型，可以产生其它噪声

椒盐噪声消除问题

Noisy image Y

$\rightarrow$ filtering algorithm $\rightarrow$ denoised image $\hat{X}$

如何让去噪后的图像 X 尽可能逼近原来的无噪声图像X?

中值运算

- 给定数字序列$\{y_1,\ldots,y_N\}$
  - 均值 Mean: average of N numbers
  - 最小值 Min: minimum of N numbers
  - 最大值 Max: maximum of N numbers
  - 中值 Median: half-way of N numbers

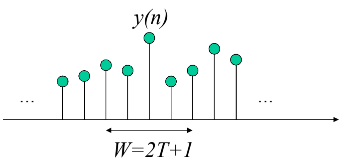
例如

$$\vec{y}=[50,0,52,255,54,55,56]$$

$$\text{sorted } \vec{y}=[0,50,52,\boxed{54},55,56,255]$$

$$median(\vec{y})=54$$

1D 中值滤波



$$\hat{x}(n)=median[y(n-T),\ldots,y(n),\ldots,y(n+T)]$$

MATLAB command: x=median(y(n-T:n+T));

Note: 中值运算是非线性的

Numerical Example

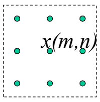
T=1:

$$\vec{y}=[50,0,52,255,54,55,56]$$

Boundary Padding $\vec{y}=[\mathbf{50},50,0,52,255,54,55,56,\mathbf{56}]$

$$\hat{\vec{x}}=[50,50,52,54,54,55,55]$$

2D 中值滤波



W: (2T+1)-by-(2T+1) window

$$\hat{x}(m,n) = median[y(m-T,n-T),\ldots,y(m-T,n+T),\ldots,$$
$$y(m,n),\ldots,y(m+T,n-T),\ldots,y(m+T,n+T)]$$

MATLAB command: x=medfilt2(y,[2*T+1,2*T+1]);

Numerical Example

225	225	225	226	226	226	226	226
225	225	255	226	226	226	225	226
226	226	225	226	0	226	226	255
255	226	225	0	226	226	226	226
225	255	0	225	226	226	226	255
255	225	224	226	226	0	225	226
226	225	225	226	255	226	226	228
226	226	225	226	226	226	226	226

y

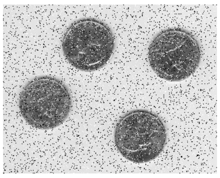
0	225	225	226	226	226	226	226
225	225	226	226	226	226	226	226
225	226	226	226	226	226	226	226
226	226	225	**225**	226	226	226	226
225	225	225	225	226	226	226	226
225	225	225	226	226	226	226	226
225	225	225	226	226	226	226	226
226	226	226	226	226	226	226	226

$\hat{x}$

Sorted: [0, 0, 0, 225, **225**, 225, 226, 226, 226]

Image Example

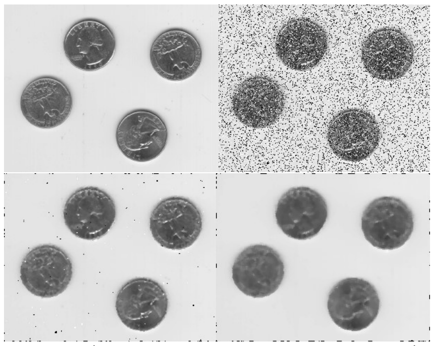
P=0.1



Noisy image Y

denoised image $\hat{x}$
3-by-3 window

Image Example (Con't)



clean

noisy (p=0.2)

3-by-3 window

5-by-5 window

优缺点

- 中值运算的优点是什么?
  - 对于脉冲噪声的噪声点 black (minimum) or white (maximum) , 取中值有效的消除了这些点。
- 中值运算的缺点是?
  - 当然，它会影响非噪声点。
  - 在中值滤波后，图像边界有明显的模糊现象。

增强中值滤波的一些想法

- 能否在不影响非噪声点的情况下，消除噪声点?
  - Yes, 如果能确定非噪声点的位置，或者能确定噪声点的位置。
- 如何检测噪声点?
  - 都是 black or white dots

带噪声检测的中值滤波

Noisy image Y

↓

Median filtering
x=medfilt2(y,[2*T+1,2*T+1]);

↓

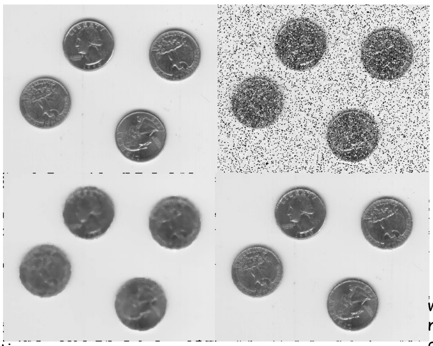
Noise detection
C=(y==0)|(y==255);

↓

Obtain filtering results
xx=c.*x+(1-c).*y;

Image Example

clean



noisy
(p=0.2)

w/o
noise
detection

with
noise
detection

Additive White Gaussian Noise

定义

Each pixel in an image is disturbed by a Gaussian random variable
With zero mean and variance σ^2

$$Y(i,j) = X(i,j) + N(i,j),$$
$$N(i,j) \sim N(0,\sigma^2), 1 \le i \le H, 1 \le j \le W$$

X: noise-free image, Y: noisy image

Note: 与椒盐噪声不同，图像中的每一个像素都被AWGN污染了。

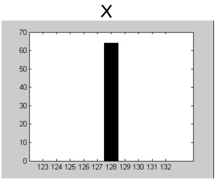
Numerical Example

128	128	128	128	128	128	128	128
128	128	128	128	128	128	128	128
128	128	128	128	128	128	128	128
128	128	128	128	128	128	128	128
128	128	128	128	128	128	128	128
128	128	128	128	128	128	128	128
128	128	128	128	128	128	128	128
128	128	128	128	128	128	128	128

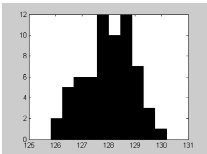
σ^2 =1

128	128	129	127	129	126	126	128
126	128	128	129	129	128	128	127
128	128	128	129	129	127	127	128
128	129	127	126	129	129	129	128
127	127	128	127	129	127	129	128
129	130	127	129	127	129	130	128
129	128	129	128	128	128	129	129
128	128	130	129	128	127	127	126

X



Y



MATLAB Command

>Y = IMNOISE(X,'gaussian',m,v)

or

>Y = X+m+randn(size(X))*v;

Note: rand()产生[0,1]区间均匀分布随机数

randn()产生正态分布随机数N(0,1)

图像去噪

Noisy image Y

filtering algorithm → denoised image $\hat{X}$

问题: 为什么不用中值滤波?
Hint: 噪声类型改变了。

复习

- 1D 线性滤波
  - 1D 卷积, 1D DFT, 频率域
- 2D 滤波
  - 2D 卷积, 2D DFT, 2D 频率域
- 各种平滑滤波器
  - 空间均值
  - 高斯滤波

1D Linear Filtering

$$f(n) \longrightarrow \boxed{h(n)} \longrightarrow g(n)$$

$$g(n)=\sum_{k=-\infty}^{\infty} h(k)f(n-k)=\underbrace{h(n)\otimes f(n)}_{线性卷积}=f(n)\otimes h(n)$$

- Linearity $a_1 f_1(n)+a_2 f_2(n)\rightarrow a_1 g_1(n)+a_2 g_2(n)$
- Time-invariant property $f(n-n_0)\rightarrow g(n-n_0)$

Fourier Series

forward $$F(w)=\sum_{-\infty}^{\infty} f(n)e^{-jwn}$$

inverse $$f(n)=\frac{1}{2\pi}\int_{-\pi}^{\pi} F(w)e^{jwn}dw$$

time-domain convolution $f(n)\otimes h(n)$

frequency-domain multiplication $F(w)H(w)$

Filter Examples

Low-pass (LP)

h(n)=[1,1]

↓

|h(w)|=2cos(w/2)

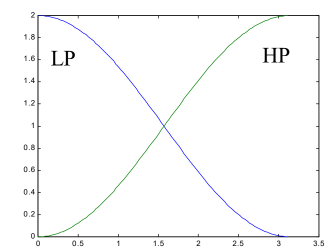
High-pass (LP)

h(n)=[1,-1]

↓

|h(w)|=2sin(w/2)

$|H(w)|$



w

1D Discrete Fourier Transform

forward transform $\quad$ inverse transform

$$y(k)=\sum_{n=0}^{N-1} x(n)W_N^{kn} \longleftrightarrow x(n)=\sum_{n=0}^{N-1} y(k)W_N^{-kn}$$

$$\boxed{W_N=\exp\{-\frac{j2\pi}{N}\}}$$

- Properties
  - periodic $y(k+N)=y(k)$

Proof: $y(k+N)=\sum_{n=0}^{N-1} x_n W_N^{(k+N)n}=\sum_{n=0}^{N-1} x_n W_N^{kn}=y(k)$

  - conjugate symmetric $y(N-k)=y^*(k)$

Proof: $y(N-k)=\sum_{n=0}^{N-1} x_n W_N^{(N-k)n}=\sum_{n=0}^{N-1} x_n W_N^{-kn}=y^*(k)$

Matrix Representation of 1D DFT

$$\mathbf{A}_{N\times N}=\begin{bmatrix} a_{11} & \cdots & \cdots & a_{1N} \\ \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ a_{N1} & \cdots & \cdots & a_{NN} \end{bmatrix}=\mathbf{F}$$

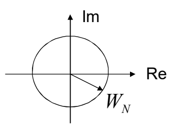
$$a_{kl}=\frac{1}{\sqrt{N}}W_N^{kl},$$

$$W_N=e^{-j\frac{2\pi}{N}}, W_N^N=1$$

$$y_k=\sum_{l=1}^{N} a_{kl}x_l$$

⇓

DFT: $$y_k=\frac{1}{\sqrt{N}}\sum_{l=1}^{N} x_l W_N^{kl}$$



Properties of DFT matrix*

symmetry $\mathbf{F}=\mathbf{F}^T$ $\quad$ Proof: $a_{kl}=a_{lk}$

unitary $\mathbf{F}^{-1}=\mathbf{F}^{T*}=\mathbf{F}^*$

Proof: If we denote $\mathbf{P}=\mathbf{F}\cdot\mathbf{F}^{T*}$ then we have

$$p_{kl}=\sum_{n=1}^{N} a_{kn}a_{nl}^*=\frac{1}{N}\sum_{n=1}^{N} e^{-\frac{j2\pi(k-l)n}{N}}=\frac{1}{N}\frac{1-e^{-j2\pi(k-l)n}}{1-e^{-j2\pi(k-l)n/N}}=\begin{cases}1 & k=l\\ 0 & k\neq l\end{cases}$$

$$\boxed{a_{kl}=\frac{1}{\sqrt{N}}e^{-j2\pi kl/N}} \quad \boxed{\sum_{n=0}^{N-1} r^n=\frac{1-r^{N-1}}{1-r},(r\neq 1)}$$

⇒ $\mathbf{P}=\mathbf{F}\cdot\mathbf{F}^{T*}=\mathbf{I}$ (identity matrix)

Fast Fourier Transform (FFT)*

- Invented by Tukey and Cooley in 1965
- Basic idea: divide-and-conquer

$$y_k = \sum_{n=0}^{N-1} x_n W_N^{kn} = \sum_{n=2m} x_{2m} W_N^{k2m} + \sum_{n=2m-1} x_{2m-1} W_N^{k(2m-1)}$$

N-point DFT

$$= \sum_{n=2m} x_{2m} W_{N/2}^{km} + W_N^{-k} \sum_{n=2m-1} x_{2m-1} W_{N/2}^{km}$$

N/2-point DFT   N/2-point DFT

- Reduce the complexity of N-point DFT from $O(N^2)$ to $O(N\log_2 N)$

Filtering in the Frequency Domain

$$f(n) \rightarrow \boxed{h(n)} \rightarrow g(n) \qquad F(k) \rightarrow \boxed{H(k)} \rightarrow G(k)$$

$$g(n) = f(n) \otimes h(n) \xrightarrow{\text{DFT}} G(k) = F(k)H(k)$$

convolution in the time domain is equivalent to multiplication in the frequency domain

FFT Under MATLAB

```
>> help fft
 FFT Discrete Fourier transform.
    FFT(X) is the discrete Fourier transform (DFT) of vector X.  For
    matrices, the FFT operation is applied to each column. For N-D
    arrays, the FFT operation operates on the first non-singleton
    dimension.

    FFT(X,N) is the N-point FFT, padded with zeros if X has less
    than N points and truncated if it has more.

    FFT(X,[],DIM) or FFT(X,N,DIM) applies the FFT operation across the
    dimension DIM.

    For length N input vector x, the DFT is a length N vector X,
    with elements
                     N
       X(k) =       sum  x(n)*exp(-j*2*pi*(k-1)*(n-1)/N), 1 <= k <= N.
                    n=1
    The inverse DFT (computed by IFFT) is given by
                     N
       x(n) = (1/N) sum  X(k)*exp( j*2*pi*(k-1)*(n-1)/N), 1 <= n <= N.
                    k=1
```


2D Linear Filtering

$$f(m,n) \rightarrow \boxed{h(m,n)} \rightarrow g(m,n)$$

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

2D convolution

MATLAB function: C = CONV2(A, B)

2D Filtering=Two Sequential 1D Filtering

- Just as we have observed with 2D transform, 2D (separable) filtering can be viewed as two sequential 1D filtering operations: one along row direction and the other along column direction
- The order of filtering does not matter $h(m,n) = h^1(m) \otimes h^1(n) = h^1(n) \otimes h^1(m)$

h^1 : 1D filter

Numerical Example

1D filter $h^1(m)=[1,1]$, $h^1(n)=[1,-1]$

$$h^1(m) \otimes h^1(n) = \begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix} \qquad h^1(n) \otimes h^1(m) = \begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix}$$

MATLAB command:

```
>h1=[1,1];h2=[1,-1];
>conv2(h1,h2)
>conv2(h2,h1)
```

Fourier Series (2D case)

$$F(w_1, w_2) = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} f(m,n) e^{-j(w_1 m + w_2 n)}$$

spatial-domain convolution

$$f(m,n) \otimes h(m,n)$$

frequency-domain multiplication

$$F(w_1, w_2) H(w_1, w_2)$$

Note that the input signal is discrete while its FT is a continuous function

Filter Examples

Low-pass (LP)

1D

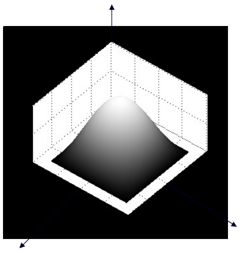
$h^1(n)=[1,1]$

$|h^1(w)|=2\cos(w/2)$

2D

$h(n)=[1,1;1,1]$

$|h(w_1,w_2)|=4\cos(w_1/2)\cos(w_2/2)$



2D DFT

Forward: $X(k,l) = \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} x(m,n) W_N^{-mk} W_N^{-nl}, 0 \le k,l \le N-1$

Inverse: $x(m,n) = \sum_{k=0}^{N-1} \sum_{l=0}^{N-1} X(k,l) W_N^{mk} W_N^{nl}, 0 \le m,n \le N-1$

- Properties
  - periodic $X(k+N, l+N) = X(k,l)$
  - conjugate symmetric $X(N-k, N-l) = X^*(k,l)$

2D Frequency Domain

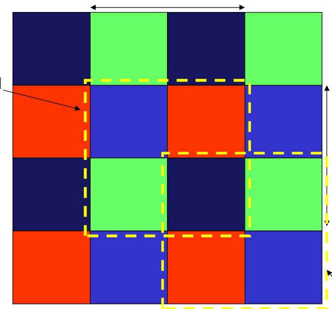
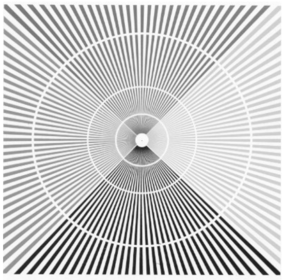
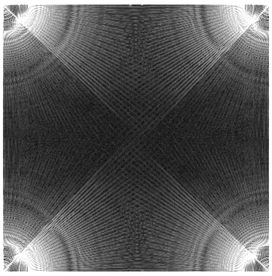


Image DFT Example

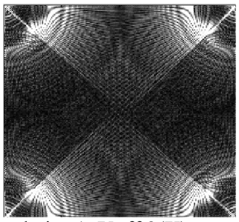


Original ray image **X**

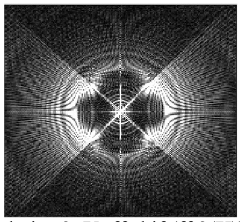


choice 1: **Y**=fft2(**X**)

Image DFT Example (Con't)



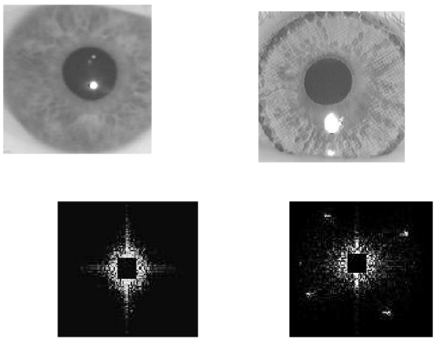
choice 1: **Y**=fft2(**X**)
Low-frequency at four corners



choice 2: **Y**=fftshift(fft2(**X**))
Low-frequency at the center

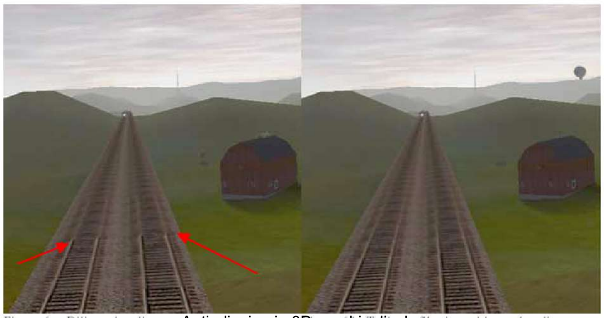
FFTSHIFT Shift zero-frequency component to center of spectrum.

2D DFT in Biometrics



natural　　　　fake

2D DFT in CG



Anti-aliasing in 3D graphic display

2D Filtering in the Frequency Domain

$$f(m,n) \rightarrow \boxed{h(m,n)} \rightarrow g(m,n) \qquad F(k,l) \rightarrow \boxed{H(k,l)} \rightarrow G(k,l)$$

$$g(m,n) = f(m,n) \otimes h(m,n) \xrightarrow{\text{DFT}} G(k,l) = F(k,l)H(k,l)$$

convolution in the spatial domain is equivalent to multiplication in the 2D frequency domain

空间域均值滤波器

$$h(m,n) = \frac{1}{W^2} \begin{array}{cc} \begin{bmatrix} 1 & \cdots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \cdots & 1 \end{bmatrix} & W \\ W & \end{array}$$

MATLAB code: >h=ones(W)/(W*W);
or
>h=fspecial('average',W);

Image Example

noisy

PSNR=20.2dB
(σ=25)

denoised

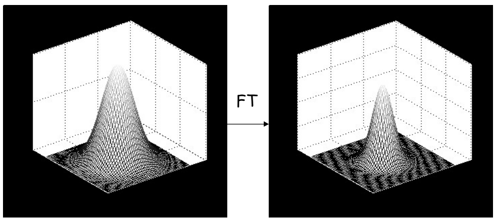
PSNR=23.8dB
(W=3)

denoised

PSNR=22.0dB
(W=5)

Matlab functions: imfilter, filter2

Gaussian Filter



$$h(m,n) = \exp\left(-\frac{m^2+n^2}{2\sigma^2}\right) \qquad H(w_1,w_2) = \exp\left(-\frac{w_1^2+w_2^2}{2\sigma^2}\right)$$

MATLAB code: >h=fspecial('gaussian', HSIZE,SIGMA);

Image Example

noisy

PSNR=20.2dB
(σ=25)

denoised

PSNR=24.4dB
(σ=1)

denoised

PSNR=22.8dB
(σ=1.5)

Matlab functions: imfilter, filter2

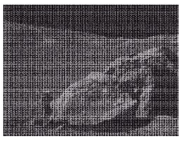
图像噪声处理

- 基本概念
- 椒盐噪声的消除
  - 中值滤波
- 加性高斯白噪声的消除
  - 2D 卷积和DFT
- 周期性噪声的消除
  - 带阻滤波器和陷波滤波器

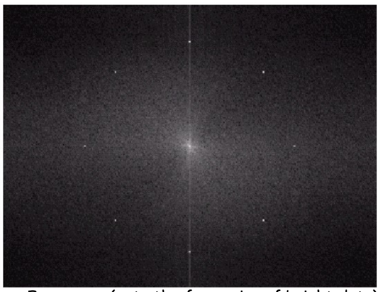
周期性噪声

- 来源: 电或电磁干扰
- 特性
  - 空间相关
  - 周期性 – 在频率域很容易观察到
- 处理方法
  - 在频率域，抑制噪声部分。

Image Example



spatial



Frequency (note the four pairs of bright dots)

带阻滤波器

$$H(w_1,w_2)=\begin{cases}0 & D-\frac{W}{2}\le\sqrt{w_1^2+w_2^2}\le D+\frac{W}{2}\\ 1 & otherwise\end{cases}$$

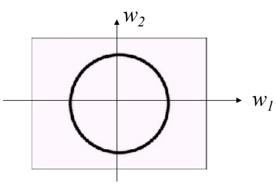
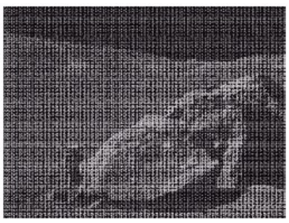
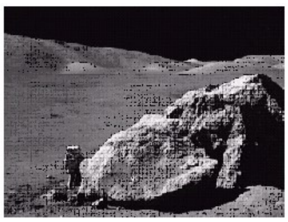


Image Example



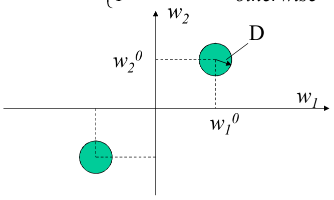
Before filtering



After filtering

陷波滤波器

$$H(w_1,w_2)=\begin{cases}0 & \sqrt{(w_1-w_1^0)^2+(w_2-w_2^0)^2}\le D \\ & or\sqrt{(w_1+w_1^0)^2+(w_2+w_2^0)^2}\le D \\ 1 & otherwise\end{cases}$$



陷波滤波器(续)

$$H(w_1,w_2)=\begin{cases}0 & |w_1|<\delta \\ 1 & otherwise\end{cases}$$

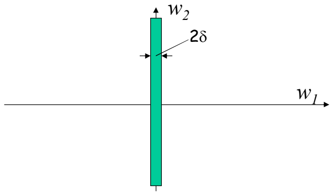
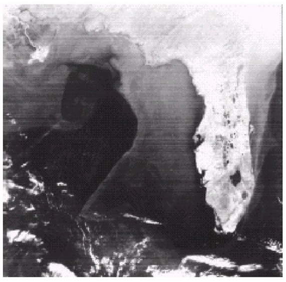
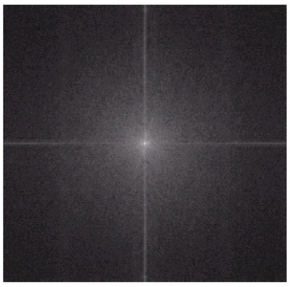


Image Example

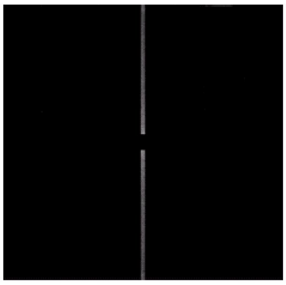


spatial

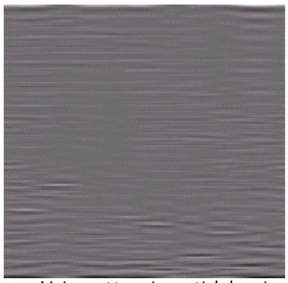


frequency

Image Example (Con't)

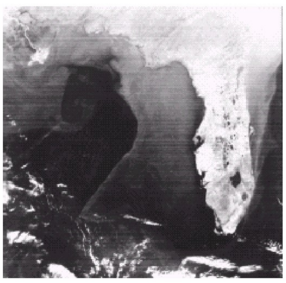


Notch filter in frequency domain

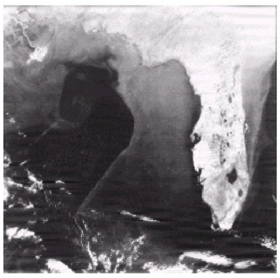


Noise pattern in spatial domain

Image Example (Con't)



before filtering



after filtering

总结

- 基本概念
  - 线性滤波，非线性滤波，中值滤波 (1D/2D)，2D 卷积/滤波，2D 频率域，带阻滤波器，陷波滤波器
- MATLAB 函数
  - imnoise,randn,rand,median,medfilt2,fft2, fftshift,fspecial,conv2,filter2,imfilter