Introduction to Image Processing – Lab04 Report

107062230 張力仁

Proj05-01: Noise Generators

(1) Implementation

Gaussian Noise:

$noise\ =\ sigma\ .*\ randn(M,N)\ +\ mu;$

$output\_s\ =\ uint8(255\ *\ mat2gray(double(input\_s)\ +\ noise));$

Impulse Noise:

$output\_s\ =\ input\_s;$

$prob\ =\ rand(size(input\_s));$

$output\_s(prob\ <=\ Ps)\ =\ 255;$ % add salt

$output\_s((prob\ >\ Ps)\ \&\ (prob\ <=\ Ps\ +\ Pp))\ =\ 0;$ % add pepper

(2) Results

Fig 5.7 (a)

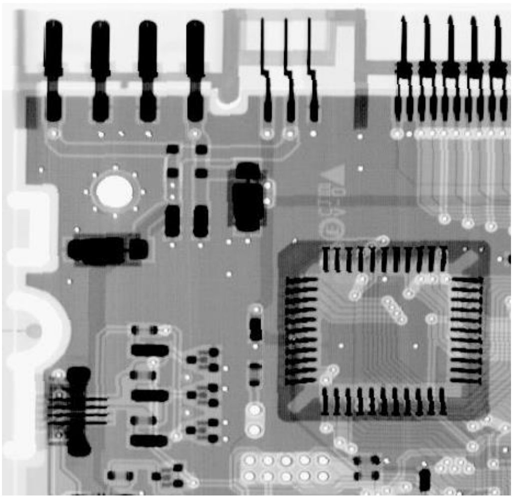


Fig 5.7 (b)

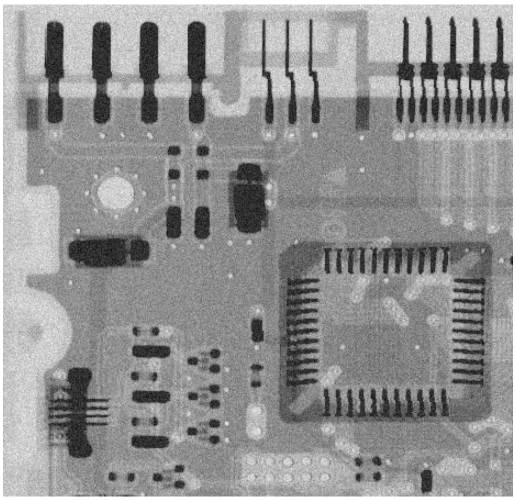


Fig 5.8 (a)

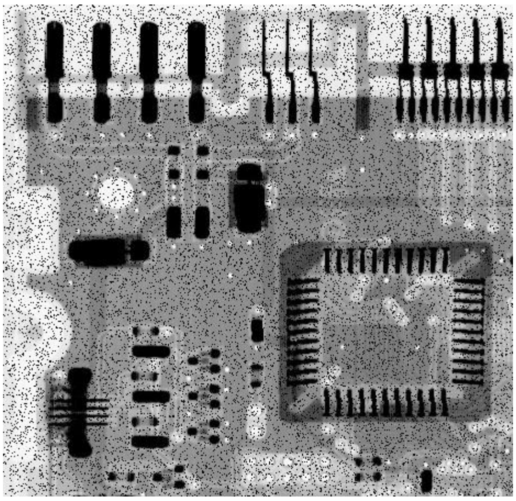


Fig 5.8 (b)

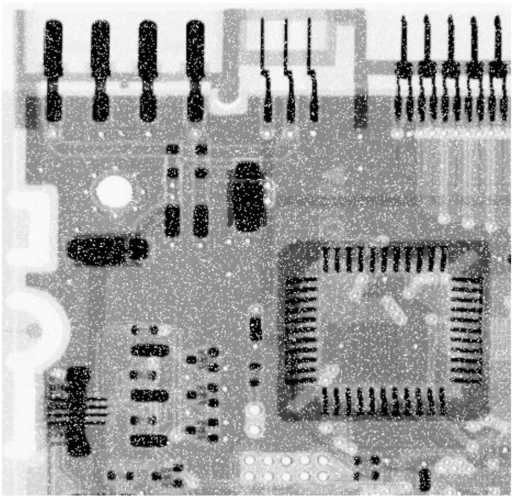
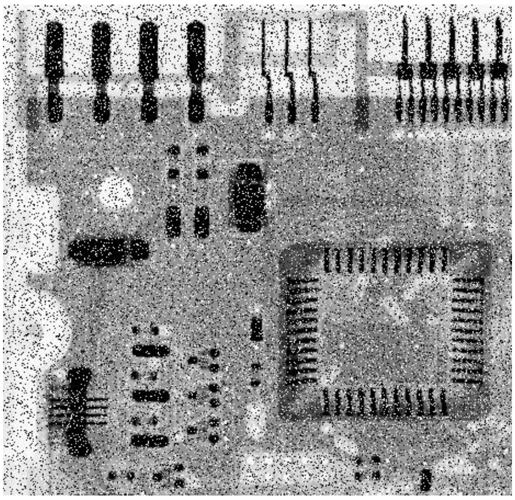


Fig 5.10 (a)



Proj05-03: Periodic Noise Reduction Using a Notch Filter

(1) Implementation

Add Sinusoidal Noise:

$X = repmat(u0 * (0{:}M-1)' \,/\, M, 1, N);$ % matrix of row indices

$Y = repmat(v0 * (0{:}N-1) \,/\, N, M, 1);$ % matrix of column indices

$noise = A * sin(2 * pi * (X + Y));$

$output\_s = input\_s + noise;$

Notch Filtering:

$D1 = (repmat(((0{:}M-1)' - M/2 - u0).\hat{}2, 1, N) + repmat(((0{:}N-1) - N/2 - v0).\hat{}2, M, 1)).\hat{}(1/2);$

$D2 = (repmat(((0{:}M-1)' - M/2 + u0).\hat{}2, 1, N) + repmat(((0{:}N-1) - N/2 + v0).\hat{}2, M, 1)).\hat{}(1/2);$

$Notch = ones(M, N);$

$Notch((D1 <= D0) \,|\, (D2 <= D0)) = 0;$

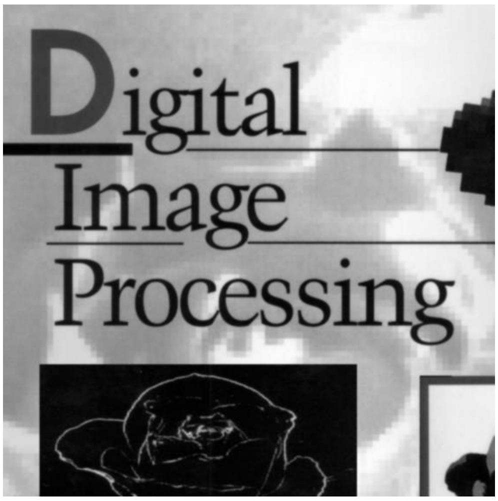
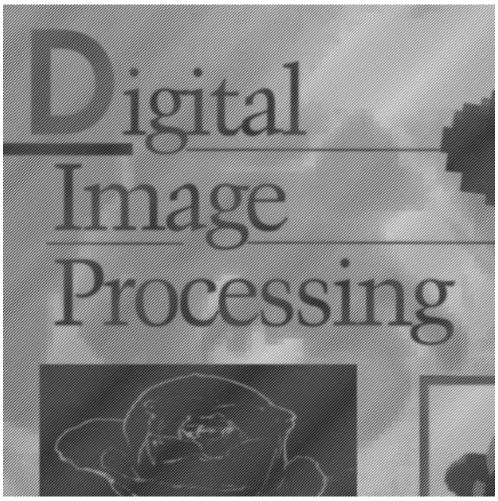
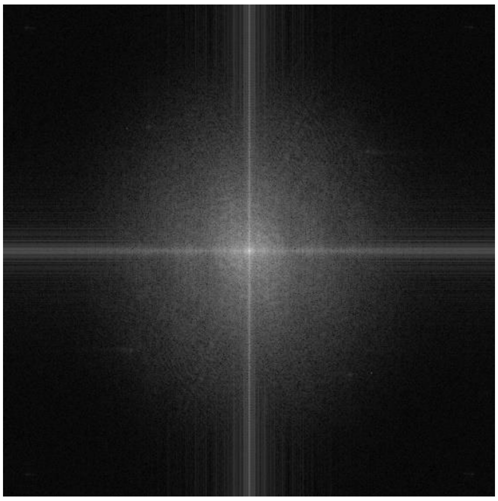
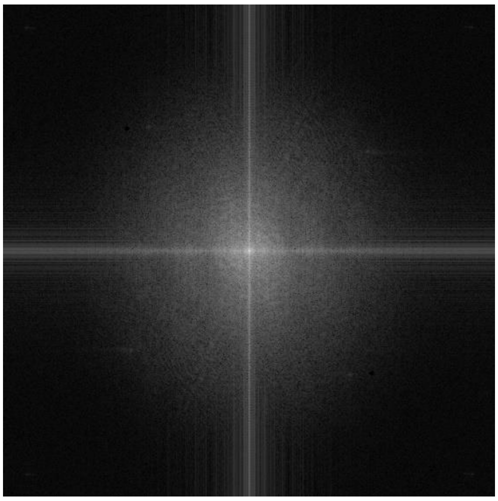
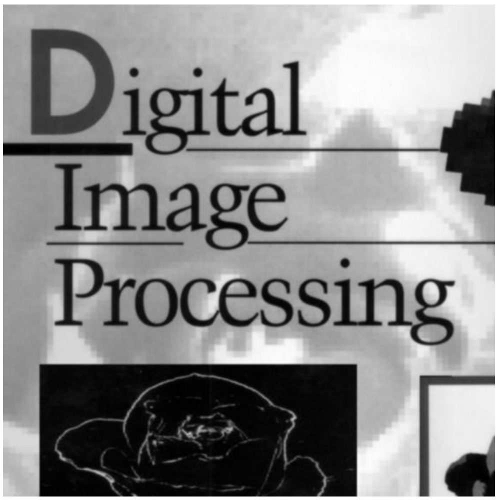
$output\_f = Notch \,.{*}\, input\_f;$

Compute PSNR:

$[M, N] = size(input1\_s);$

$psnr = 10 * log10(M * N * 1\hat{}2 \,/\, sum((input1\_s - input2\_s).\hat{}2, \,'all'));$

(2) Results

Fig 5.26 (a)	With noise in spatial domain
	
With noise in frequency domain	Notch filter
	 (two circles in diagonal axis)
Filtered result in frequency domain	Filtered result in spatial domain
	

PSNR:

```
>> main
PSNR: 88.887413 db
fx >>
```

In my experiment, A, $u0$, and $v0$ are set to 0.8, $M/4-1$, and $N/4-1$, respectively.

Proj05-04: Parametric Wiener Filter

(1) Implementation

Add Motion Blur:

$U = repmat((0{:}M-1)', 1, N);$ % matrix of row indices

$V = repmat((0{:}N-1), M, 1);$ % matrix of column indices

% u and v are centered since the input image in frequency domain is centered

$COMP = pi * ((U - M/2) * a + (V - N/2) * b);$

$H = (T \,./\, COMP) \,.{*}\, sin(COMP) \,.{*}\, exp(-1j * COMP);$

% When COMP is 0, according to Eq 5-74 in the 4/e textbook, the NaN coming from division by 0 is replaced with T.

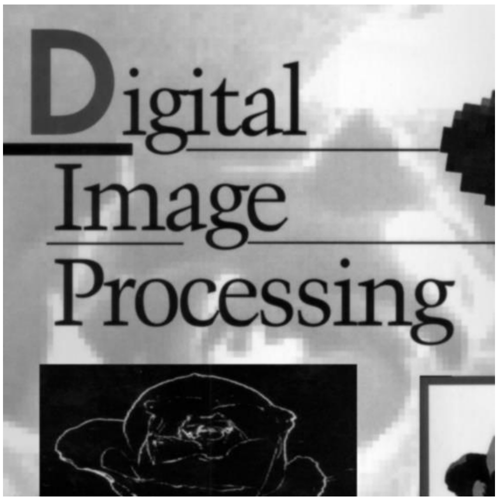
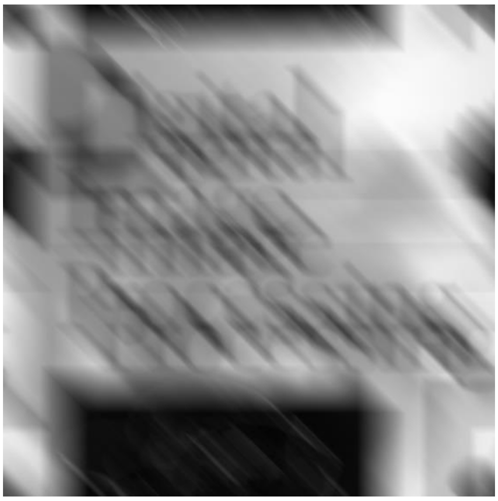
$H\big(isnan(H)\big) = T;$

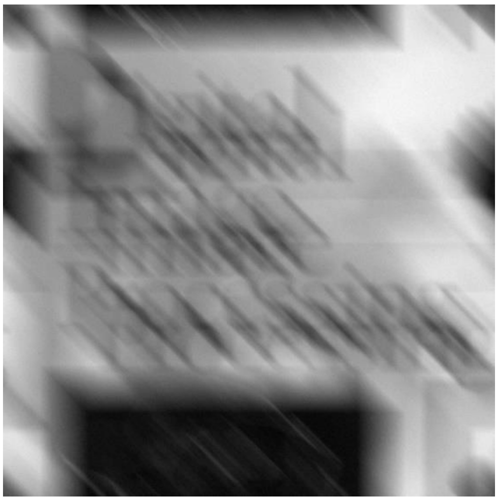
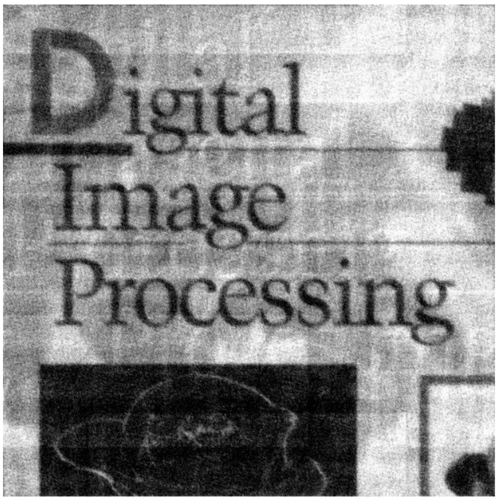
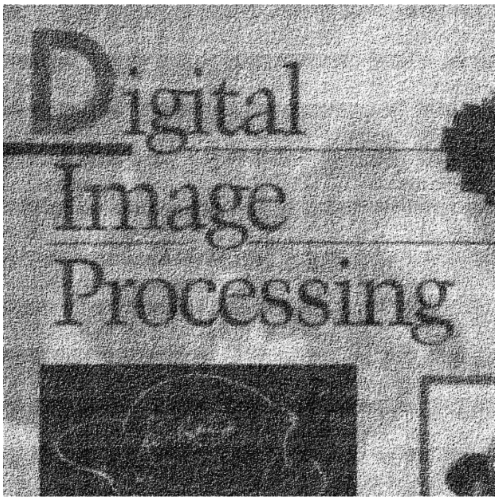
$output\_f = H \,.{*}\, input\_f;$

Wiener Filtering:

$output\_f = 1 \,./\, H \,.{*}\, (abs(H).\verb|^|2 \,./\, (abs(H).\verb|^|2 + K)) \,.{*}\, input\_f;$

(2) Results

Fig 5.26 (a)	Fig 5.26 (b)
	

<table>
<tr><td>Fig 5.26 (b) + Gaussian noise of 0 mean and variance of 10</td><td>Wiener filtering with $K = 0.01$</td></tr>
<tr><td></td><td>
</td></tr>
<tr><td>Wiener filtering with $K = 0.001$</td><td>Wiener filtering with $K = 0.0001$</td></tr>
<tr><td>
</td><td>
</td></tr>
</table>

(3) Discussion

```
>> main
PSNR (K = 0.010000): 17.059881
PSNR (K = 0.001000): 14.728026
PSNR (K = 0.000100): 11.195855
>>
```

Wiener filtering with $K = 0.001$ gets better visual effect, but the one with $K = 0.01$ has higher PSNR (in dB).