12 – Penapisan Citra dalam Ranah Frekuensi

IF4073 Pemrosesan Citra Digital

Oleh: Rinaldi M

Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung
2026

Penapisan dalam ranah frekuensi

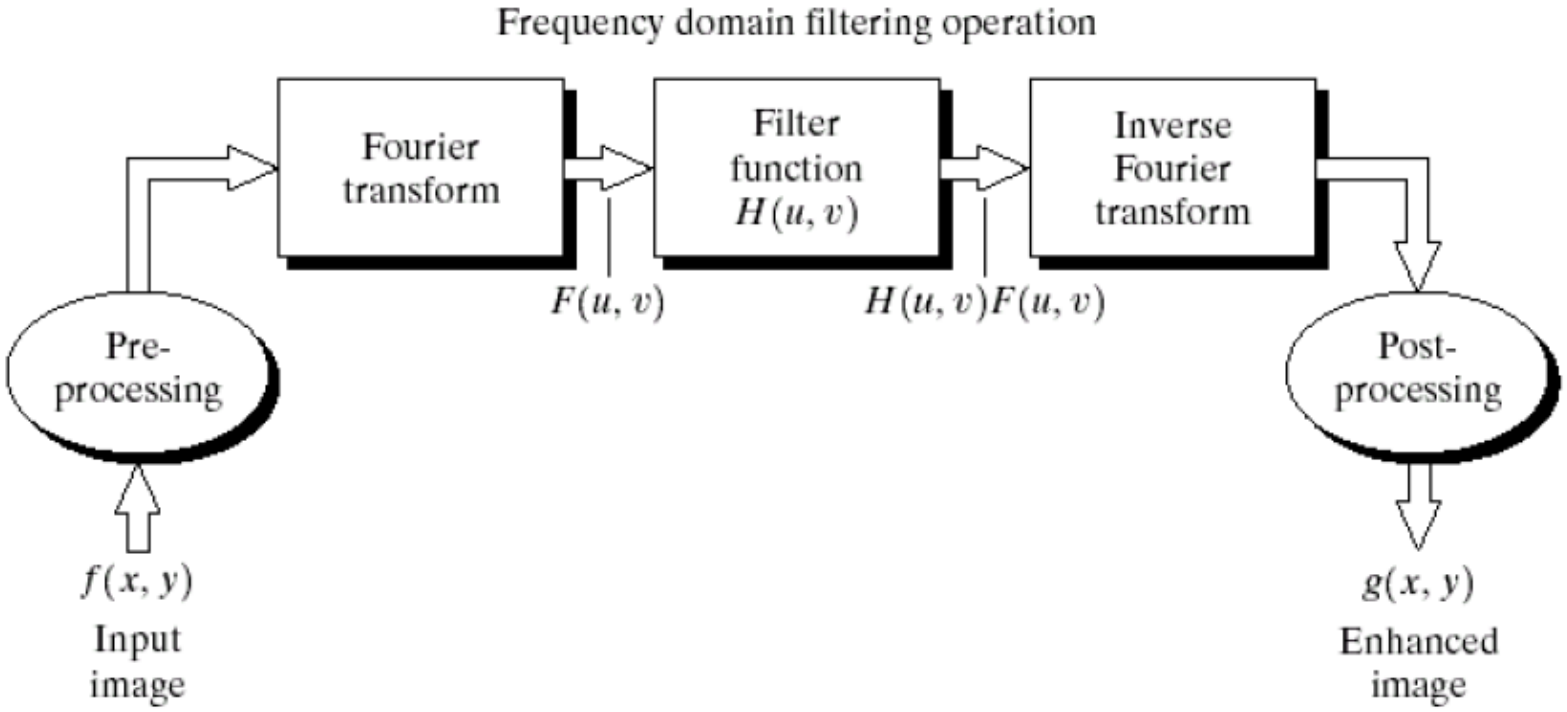


FIGURE 4.5 Basic steps for filtering in the frequency domain.

Hubungan antara operasi penapisan dalam ranah spasial dan ranah frekuensi:

$$g(x,y) = h(x,y)*f(x,y) \leftrightarrow G(u,v) = H(u,v)F(u,v)$$

dan sebaliknya: $g(x,y) = h(x,y)f(x,y) \leftrightarrow G(u,v) = H(u,v) * F(u,v)$

Catatan: ukuran matriks H(u,v) dan F(u,v) harus sama

```
>> A = magic(5)

A =
    17    24     1     8    15
    23     5     7    14    16
     4     6    13    20    22
    10    12    19    21     3
    11    18    25     2     9
```

```
>> Af = fft2(A)

Af =
   1.0e+02 *
   3.2500 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
  -0.0000 - 0.0000i   1.0113 - 0.3286i  -0.2023 - 0.0657i   0.0000 + 0.0000i   0.0000 - 0.0000i
   0.0000 + 0.0000i   0.0000 - 0.0000i  -0.3863 - 0.5317i  -0.0000 + 0.0000i   0.0773 - 0.1063i
   0.0000 + 0.0000i   0.0773 + 0.1063i  -0.0000 + 0.0000i  -0.3863 + 0.5317i   0.0000 + 0.0000i
  -0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 - 0.0000i  -0.2023 + 0.0657i   1.0113 + 0.3286i
```

```
>> B = ones(5)

B =
     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
```

```
>> Bf = fft2(B)

Bf =
    25     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
```

```
>> C = Af .* Bf

C =
        8125           0           0           0           0
           0           0           0           0           0
           0           0           0           0           0
           0           0           0           0           0
           0           0           0           0           0
```

```
>> C = Af .* Bf

C =
        8125         0         0         0         0
           0         0         0         0         0
           0         0         0         0         0
           0         0         0         0         0
           0         0         0         0         0
```

```
>> D = ifft2(C)

D =
   325   325   325   325   325
   325   325   325   325   325
   325   325   325   325   325
   325   325   325   325   325
   325   325   325   325   325
```

- Perhatikan bahwa penapisan dalam ranah spasial dan dalam ranah frekuensi keduanya berkoresponden.

$$f(x,y) * h(x,y) \Leftrightarrow H(u,v)F(u,v)$$

- Contoh: citra *pepper* akan dilakukan *sharpening* pada ranah spasial dan ranah frekuensi. Penapis yang digunakan adalah

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

$$\sum = 1$$

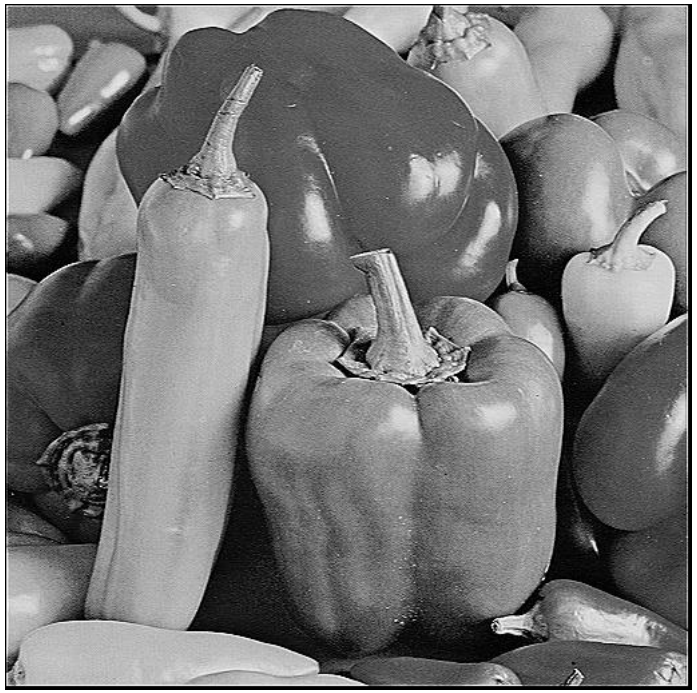
- Citra *f* dan penapis *h* keduanya ditransformasi ke dalam ranah frekuensi
- Sebelum ditransformasi, penapis *h* harus disamakan ukurannya dengan citra f dengan proses *zero padding*

- Penapisan dalam ranah spasial (penajaman tepi):

```
f=imread('lada-gray.bmp');
imshow(f), title('Original image');
h = [0 -1 0; -1 5 -1; 0 -1 0];
fsharp = uint8(convn(double(f), double(h)));
figure, imshow(fsharp), title('Sharp image');
```

Original image

Sharp image



- Penapisan dalam ranah frekuensi:

```
[f, map] = imread('lada-gray.bmp');
imshow(f, map), title('Original image');
h = [0 -1 0; -1 5 -1; 0 -1 0];
[M,N] = size(f);
hpad = zeros(M,N);
hpad(1:3,1:3) = h;
F = fft2(double(f));
H = fft2(double(hpad));
F2 = H.*F;
f2 = ifft2(F2);
g = real(f2);
figure, imshow(g, map), title('Sharp image');
```

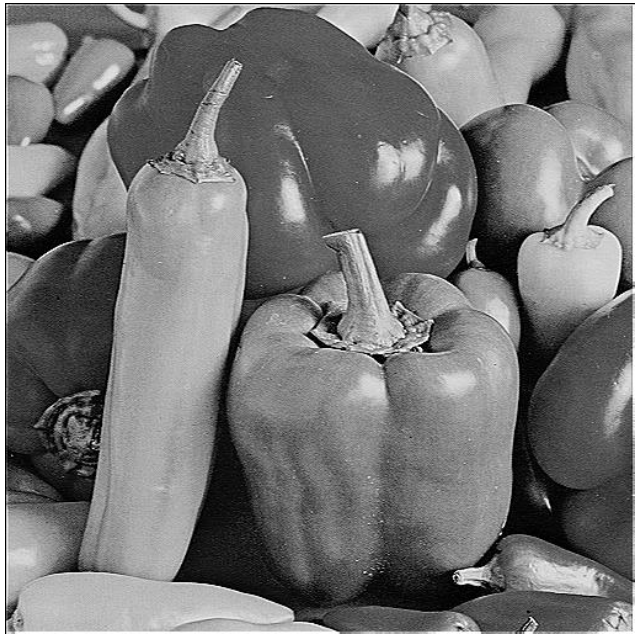
Keterangan:

```
H = fft2(double(h),M,N);
```

melakukan *zero padding* aar ukuran h menjadi M x N
lalu fft2

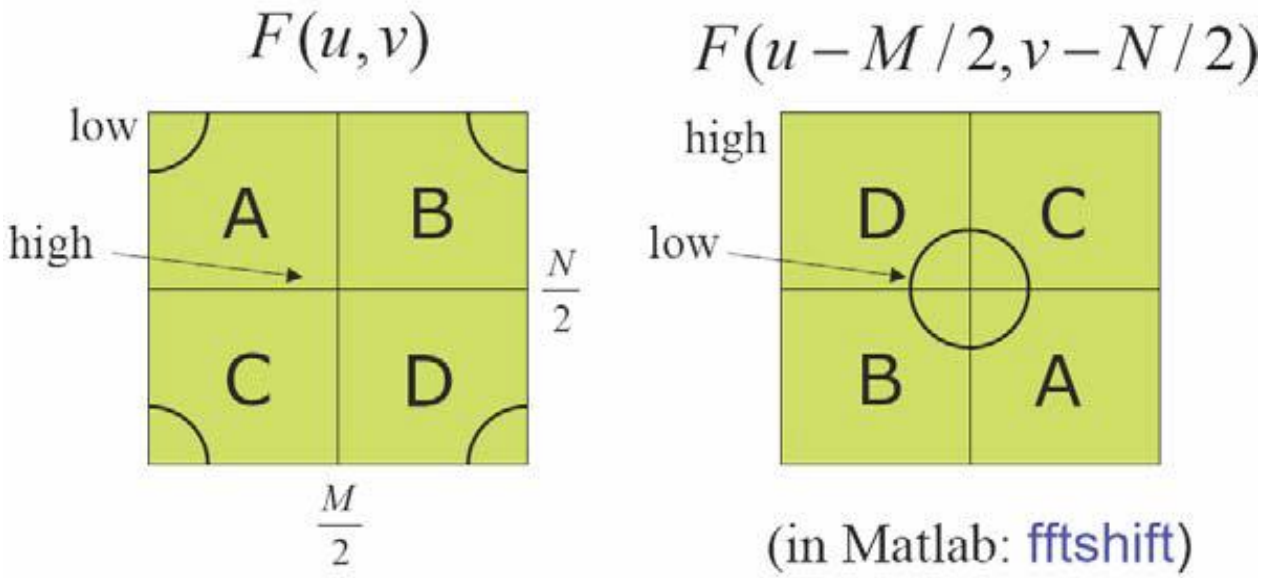
Original image

Sharp image



Tinjau Kembali Fourier Transform

$$f(x,y)(-1)^{x+y} \longleftrightarrow F(u-M/2, v-N/2)$$ → fftshift



Titik tengah → frekuensi horizontal = 0 dan
frekuensi vertikal = 0 → DC
Dekat titik tengah → frekuensi rendah
Semakin jauh dari tengah → frekuensi semakin tinggi
Bagian paling jauh dari tengah → frekuensi paling tinggi

`fftshift` memindahkan frekuensi rendah, yaitu (u,v) = (0,0) ke tengah
Tujuannya agar spektrum frekuensi menjadi lebih mudah diinterpretasikan secara visual

Simulasi FFT2, IFFT2, dan FFTShift pada sebuah matriks

```
>> A =
magic(3)

A =
     8     1     6
     3     5     7
     4     9     2
```

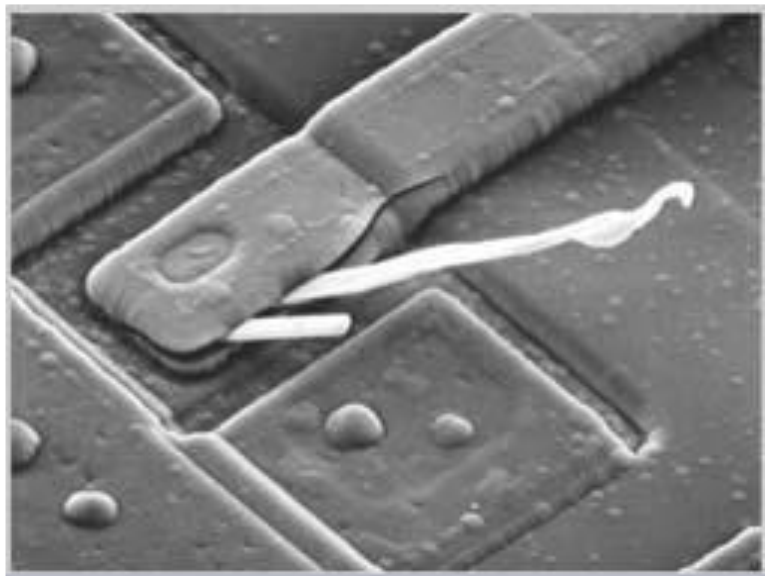
```
>> B = fft2(A)

B =
  45.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i  13.5000 + 7.7942i   0.0000 - 5.1962i
   0.0000 - 0.0000i   0.0000 + 5.1962i  13.5000 - 7.7942i
```

amplitudo frekuensi (0,0)

```
>> B = abs(B)

B =
  45.0000         0         0
   0.0000   15.5885    5.1962
   0.0000    5.1962   15.5885
```

```
>> B = fftshift(B)

B =
  15.5885    0.0000    5.1962
         0   45.0000         0
    5.1962    0.0000   15.5885
```

amplitudo frekuensi (0,0)



Image

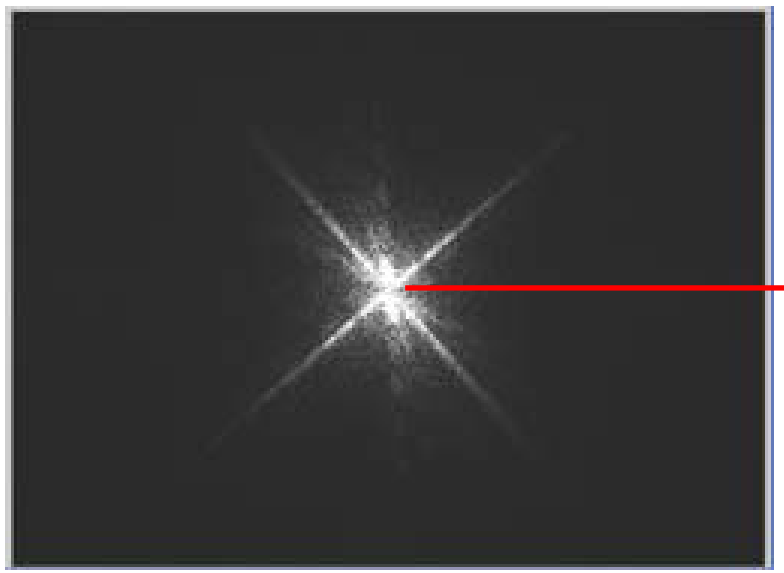
low

low

low

low

Sebelum fftshift

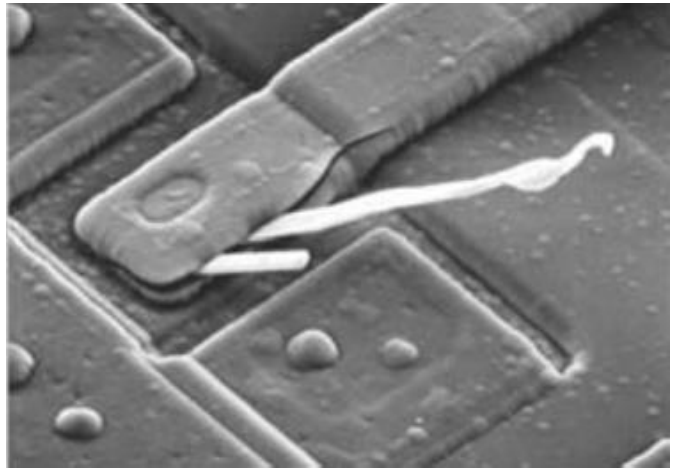


low

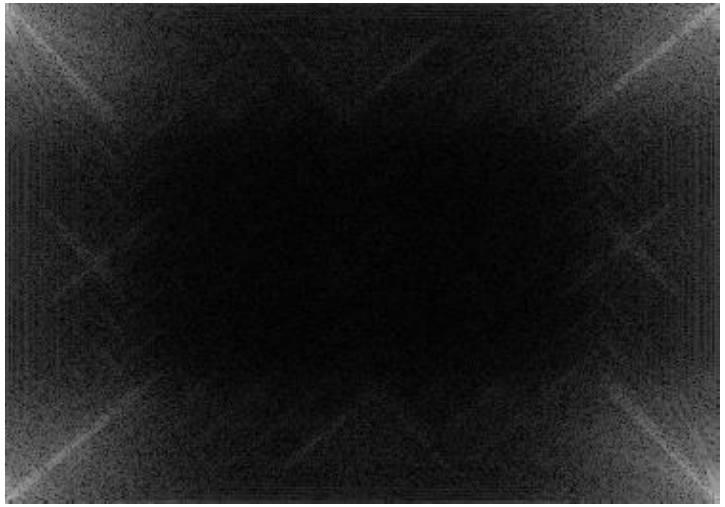
Setelah fftshift

```
f = imread('damage.bmp');
imshow(f); title('Original image');
% Ukuran citra
[M,N] = size(f);

% Konversi ke double
f = im2double(f);

% Fourier transform
F = fft2(fp);

S1 = log(1 + abs(F));
figure, imshow(S1,[]); title('Fourier Spectrum');

% Geser ke tengah');
F = fftshift(F);

figure;
S2 = log(1 + abs(F));
imshow(S2,[]); title('Fourier Spectrum after fftshift');
```

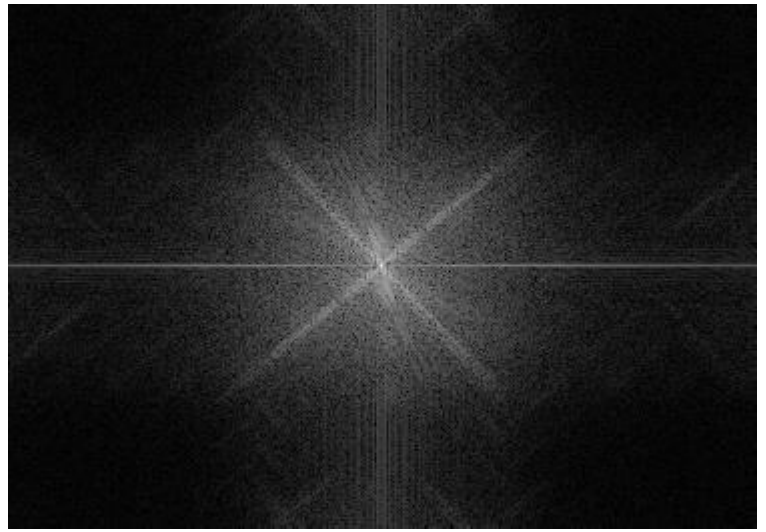
Original image



Fourier Spectrum



Fourier Spectrum after fftshift

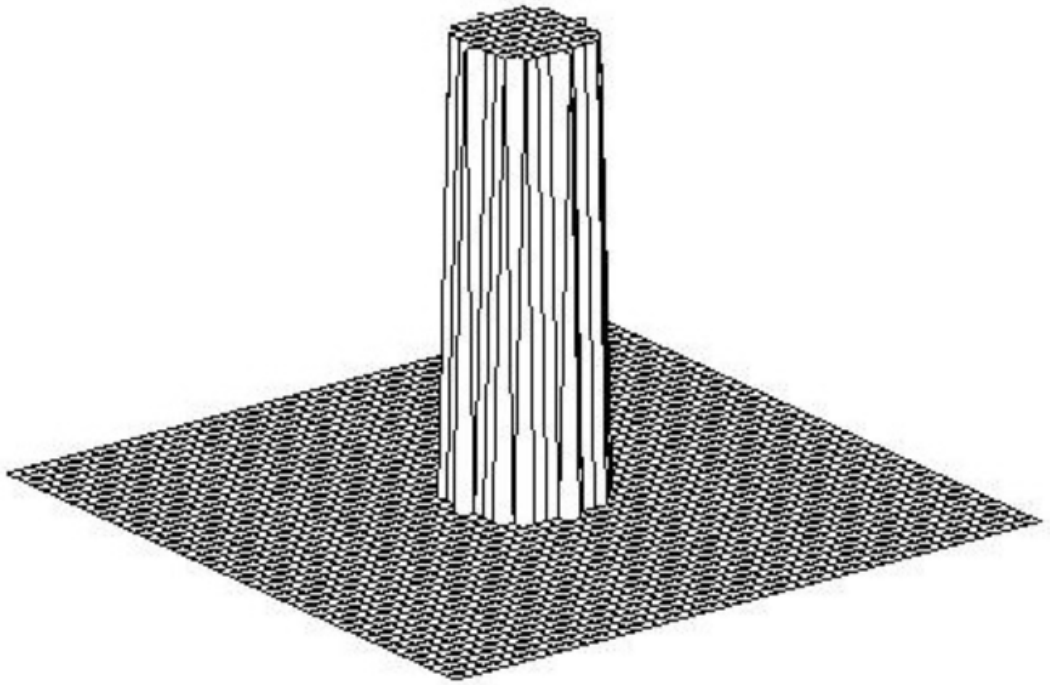
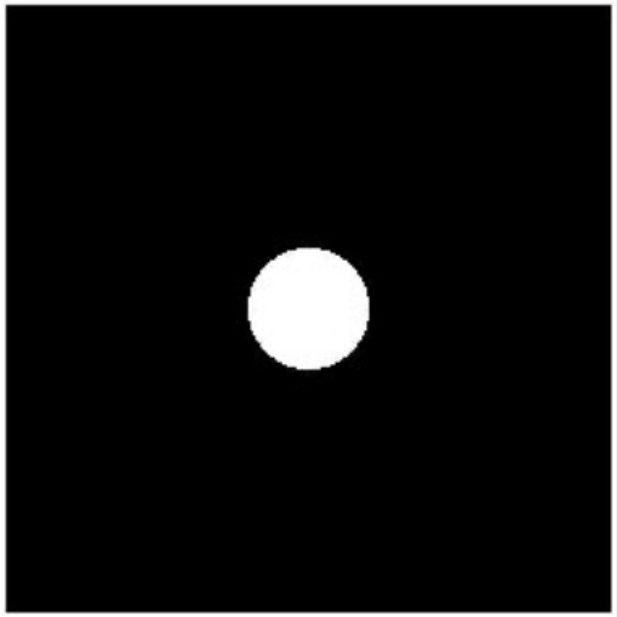


Lowpass filter (LPF) dalam ranah frekuensi

- Penapis lolos rendah (*lowpass filter*) bertujuan menekan (meredam) komponen berfrekuensi tinggi dan membiarkan komponen berfrekuensi rendah relatif tidak berubah.

- Menghasilkan efek *blurring* (atau *smoothed image*)

- Tiga buah LPF yang utama:
  1. *Ideal lowpass filter* (ILPF)
  2. *Gaussian lowpass filter* (GLPF)
  3. *Butterworth lowpass filter* (BLPF)

Lowpass Filter	Formula	Mesh	Image
Ideal	$H(u,v)=\begin{cases}1 & \text{if } D(u,v)\le D_0\\ 0 & \text{if } D(u,v)>D_0\end{cases}$		

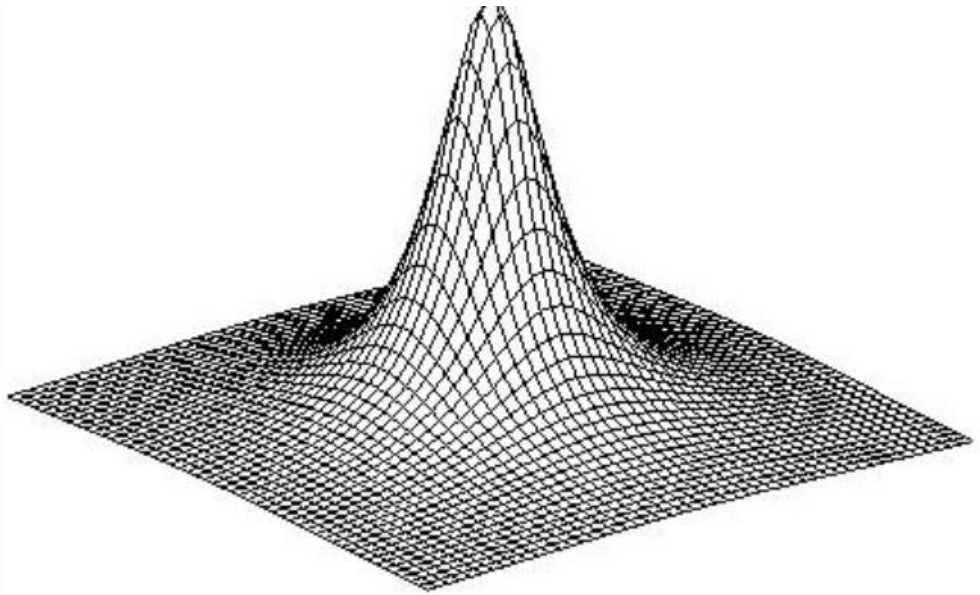
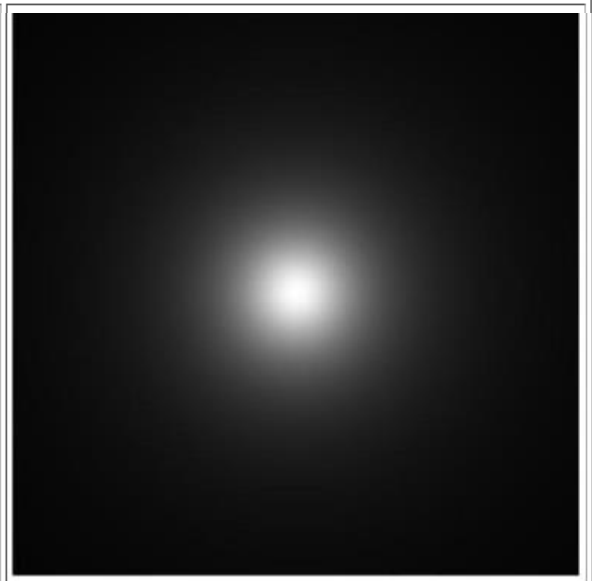
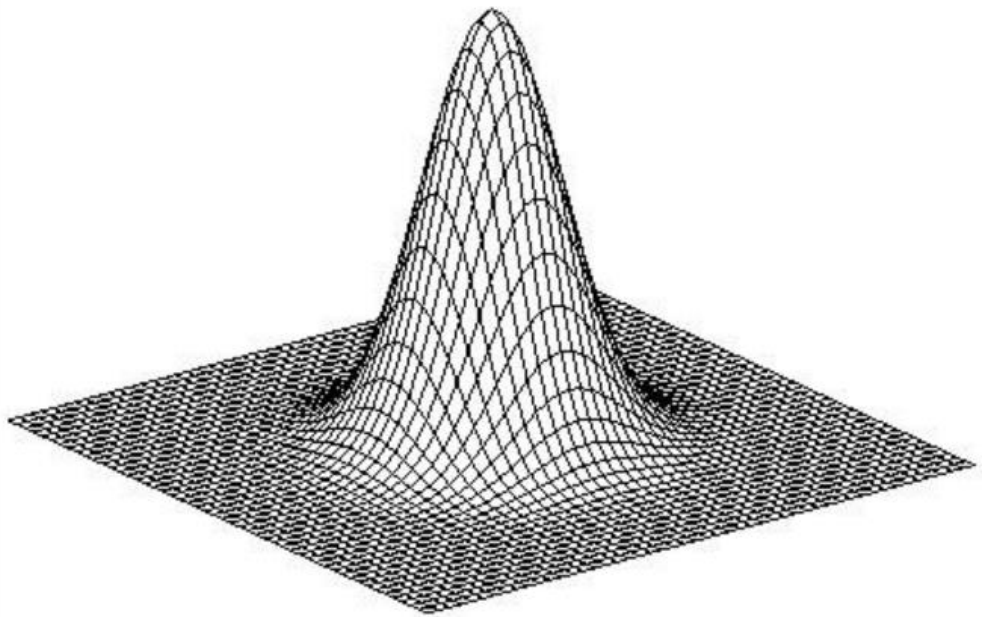
Where $D(u,v)$ is the distance from point (u,v) to the center of the frequency rectangle

If the center is at $(M/2,N/2)$

$$D(u,v)=\sqrt{(u-M/2)^2+(v-N/2)^2}$$

- *D0* : cutoff frequency

Sumber: https://www.cs.uregina.ca/Links/class-info/425-nova/Lab5/index.html

Butterworth	$H(u,v) = \frac{1}{1+\left[D(u,v)/D_0\right]^{2n}}$		
Gaussian	$H(u,v) = e^{-D^2(u,v)/2D_0^2}$		

Pertimbangan nilai frekuensi *cutoff* (D0)

- Frekuensi *cutoff* D0 dari ILPF ideal menentukan banyaknya komponen frekuensi yang dilewatkan oleh penapis (filter).

- Makin kecil nilai D0, makin banyak komponen gambar yang dihilangkan oleh filter (karena banyak 0 di dalam H(u,v)).

- Secara umum, nilai D0 dipilih sedemikian rupa sehingga sebagian besar komponen yang diinginkan dilewatkan, sementara sebagian besar komponen yang tidak diinginkan dihilangkan.

Langkah-Langkah penapisan dengan LPF:

1. Tentukan parameter *padding*, biasanya untuk citra f(x,y) berukuran M x N, umumnya parameter *padding* P dan Q adalah P = 2M and Q = 2N.
2. Bentuklah citra *padding* fp(x,y) berukuran P X Q dengan menambahkan pixel-pixel bernilai nol pada f(x, y).
3. Lakukan transformasi Fourier pada fp(x, y) lalu dilanjutkan dengan pemusatan (fftshift)

   Kode Matlab: `F = fft2(fp); F = fftshift(F);`
4. Bangkitkan fungsi penapis LPF H berukuran P x Q
5. Kalikan F dengan H

   Kode Matlab: `G=H.*F;`
6. Lakukan invers fftshift terhadap G (Koe Matlab: `G = ifftshift(G)`)
7. Ambil bagian real dari inverse FFT terhadap G:

   Kode Matlab: `g=real(ifft2(G));`
8. Potong bagian kiri atas sehingga menjadi berukuran citra semula

   `g=g(1:size(f,1), 1:size(f,2));`

Catatan:

Langkah *zero padding* (1 dan 2) adalah opsional, bertujuan agar citra f(x,y) dan penapis H(u,v) memiliki ukuran yang sama

M

original image

N

padded image

Penjelasan:

1. *Zero padding* 2 kali ukuran citra, $P = 2M$dan $Q = 2N$, digunakan untuk menghindari efek *circular convolution* dan memberikan representasi frekuensi yang lebih rapat. Dengan *zero padding* 2x ukuran citra, FFT menjadi ekuivalen dengan *linear convolution* pada daerah yang diinginkan.
2. Contoh *linear convolution*:

$$\text{Citra f} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \quad \text{filter h} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Ketika mengkonvolusi pixel 1, nilai-nilai pada efek menggantung (?) diiisi 0:

$$\begin{matrix} ? & ? & ? \\ ? & 1 & 2 \\ ? & 4 & 5 \end{matrix} \quad \rightarrow \quad \begin{matrix} 0 & 0 & 0 \\ 0 & 1 & 2 \\ 0 & 4 & 5 \end{matrix}$$ → hasil konvolusi pixel 1 = 12

Tetapi pada *circular convolution*, nilai-nilai pada efek yang menggantung diisi dengan nilai *wrap-around* (informasi dari kanan muncul di kiri, dan informasi dari bawah muncul di atas):

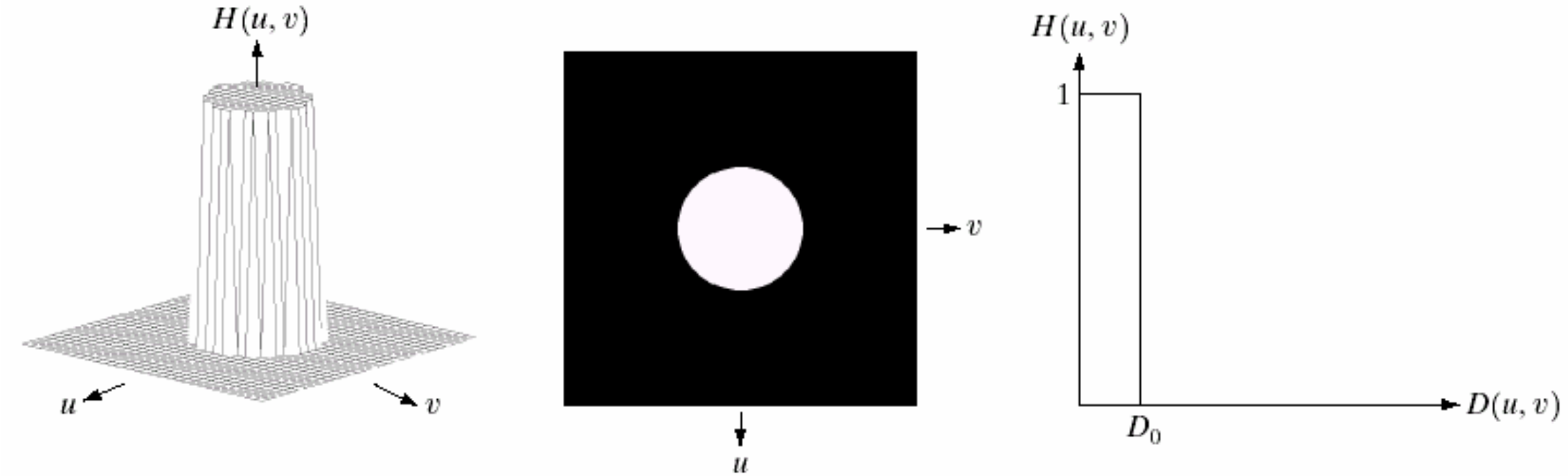
$$\begin{matrix} ? & ? & ? \\ ? & 1 & 2 \\ ? & 4 & 5 \end{matrix} \quad \rightarrow \quad \begin{matrix} 7 & 8 & 9 \\ 3 & 1 & 2 \\ 6 & 4 & 5 \end{matrix}$$ -→ hasil konvolusi pixel 1 = 45

3. Dengan *zero padding* menjadi P = 2M dan Q = 2N :

1	2	3	0	0	0
4	5	6	0	0	0
7	8	9	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0

Kemudian FFT dilakukan pada ukuran 6×6. Karena ruang yang tersedia cukup besar, hasil konvolusi tidak mengalami *overlap/wrap-around* pada daerah hasil yang kita perlukan.

Ideal Low Pass Filter (ILPF)



a b c

FIGURE 4.10 (a) Perspective plot of an ideal lowpass filter transfer function. (b) Filter displayed as an image. (c) Filter radial cross section.

```
% Penapisan dalam ranah frekuensi dengan Ideal Lowpass Filter (ILPF)

f = imread('cameraman.bmp');

% Ukuran citra
[M,N] = size(f);

% Konversi ke double
f = im2double(f);

% ========================================================
% Step 1: Zero padding menjadi 2M x 2N
% ========================================================
P = 2*M;
Q = 2*N;

% ========================================================
% Step 2: Membentuk citra padding
% ========================================================
fp = zeros(P,Q);
fp(1:M,1:N) = f;

figure; imshow(f); title('Original Image');

figure; imshow(fp); title('Padded Image');
```

```
% ======================================================
% Step 3: Fourier Transform
% ======================================================

F = fft2(fp);
F = fftshift(F);

S2 = log(1 + abs(F));

figure; imshow(S2,[]); title('Fourier Spectrum');

% ======================================================
% Step 4: Membentuk Ideal Low-Pass Filter
% ======================================================
D0 = 50;

% Koordinat frekuensi
u = -P/2:(P/2-1);
v = -Q/2:(Q/2-1);

[V,U] = meshgrid(v,u);

% Jarak dari pusat frekuensi
D = sqrt(U.^2 + V.^2);

% Ideal LPF
H = double(D <=D0);     % H(i,j)=1 jika D(i,j)?D0, H(i,j)=0 jika D(i,j)>D0
```

```
% Tampilkan mask
figure; imshow(H,[]); title('Ideal LPF Mask');

figure; mesh(H); title('Ideal LPF Surface');

% ========================================================
% Step 5: Kalikan spektrum dengan filter
% ========================================================

LPF_f = H .* F;

% ========================================================
% Step 6: Inverse Fourier Transform
% ========================================================

LPF_f1 = ifftshift(LPF_f);
LPF_f2 = real(ifft2(LPF_f1));

figure;
imshow(LPF_f2,[]);
title('Output Image after Inverse 2D DFT');

% ========================================================
% Step 7: Buang zero padding
% ========================================================

LPF_f2 = LPF_f2(1:M,1:N);
figure; imshow(LPF_f2,[]); title('Output Image');
```

Catatan: *Zero padding* tidak selalu harus dilakukan (opsional), program di atas tetap benar jika tidak dilakukan *zero padding*, yaitu P = M, dan Q = N.

Semula:

```
% Penapisan dalam ranah frekuensi dengan ILPF

f = imread('cameraman.bmp');

% Ukuran citra
[M,N] = size(f);

% Konversi ke double
f = im2double(f);

% ========================================
% Step 1: Zero padding menjadi 2M x 2N
% ========================================
P = 2*M;
Q = 2*N;
```

Menjadi:

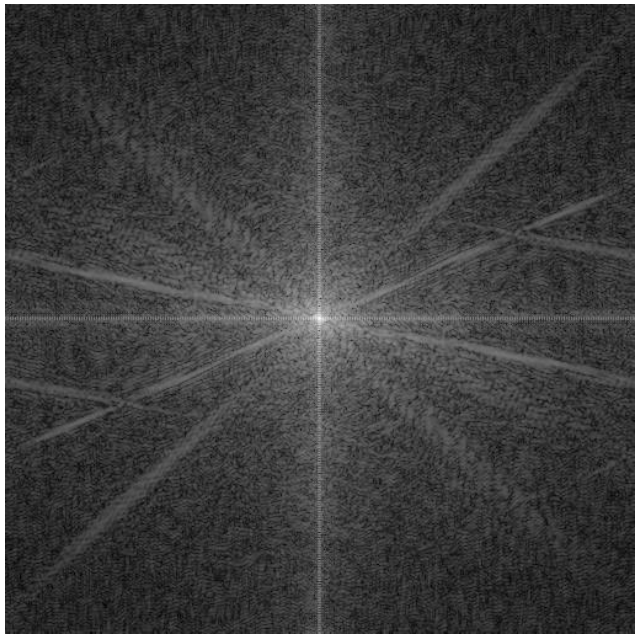
```
% Penapisan dalam ranah frekuensi dengan ILPF

f = imread('cameraman.bmp');

% Ukuran citra
[M,N] = size(f);

% Konversi ke double
f = im2double(f);

% ========================================
% Step 1: Zero padding menjadi 2M x 2N
% ========================================
P = M;
Q = N;
```

Tetap benar

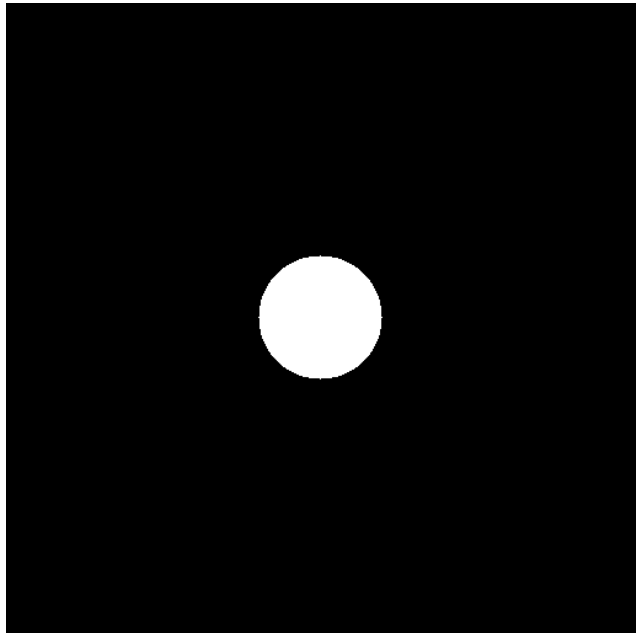
original image

padded image

Fourier spectrum



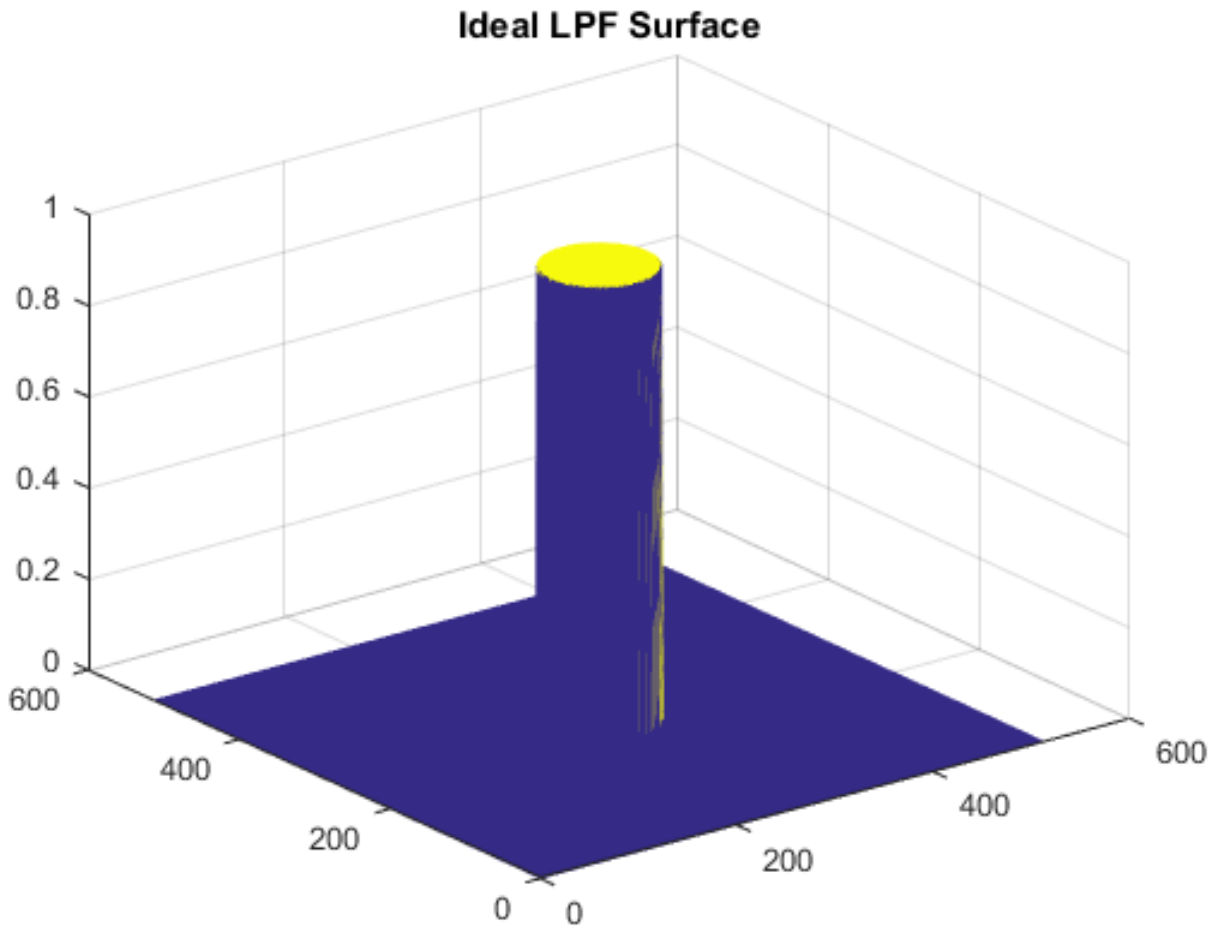
LPF Ideal Mask

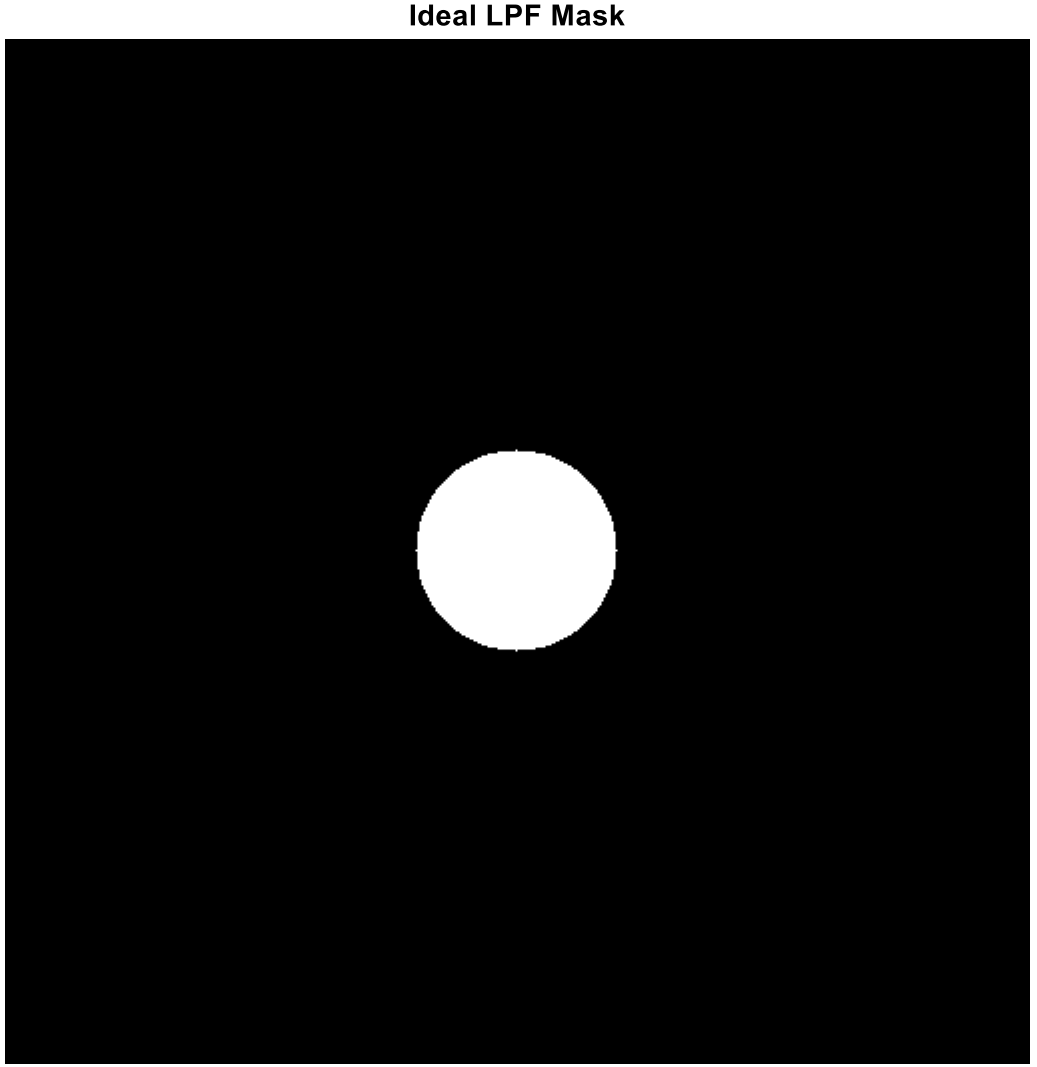


output image after inverse 2D DFT

output image

Hasil run program dengan ILPF, D0 = 50


Ideal LPF Surface
1
0.8
0.6
0.4
0.2
0
600
400
200
0
0
200
400
600

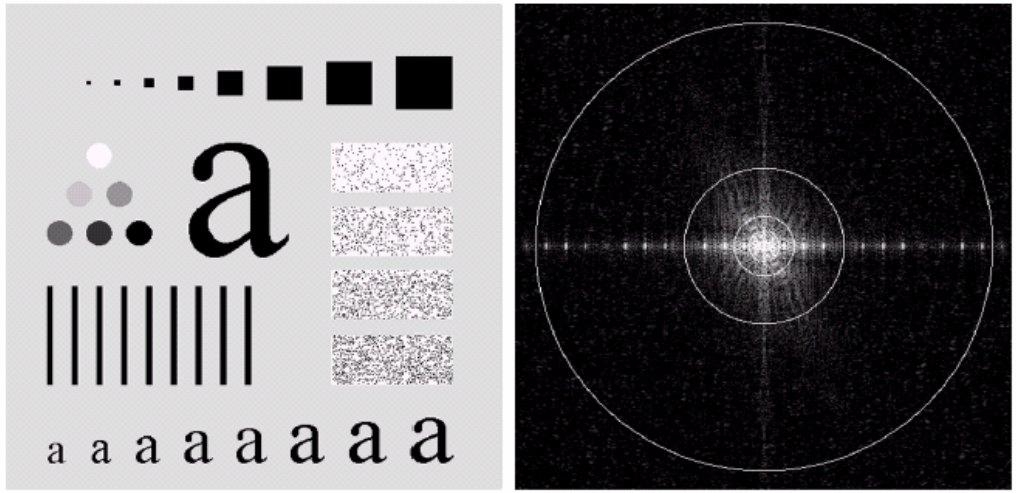

Ideal LPF Mask

Output Image

Kelemahan hasil ILPF

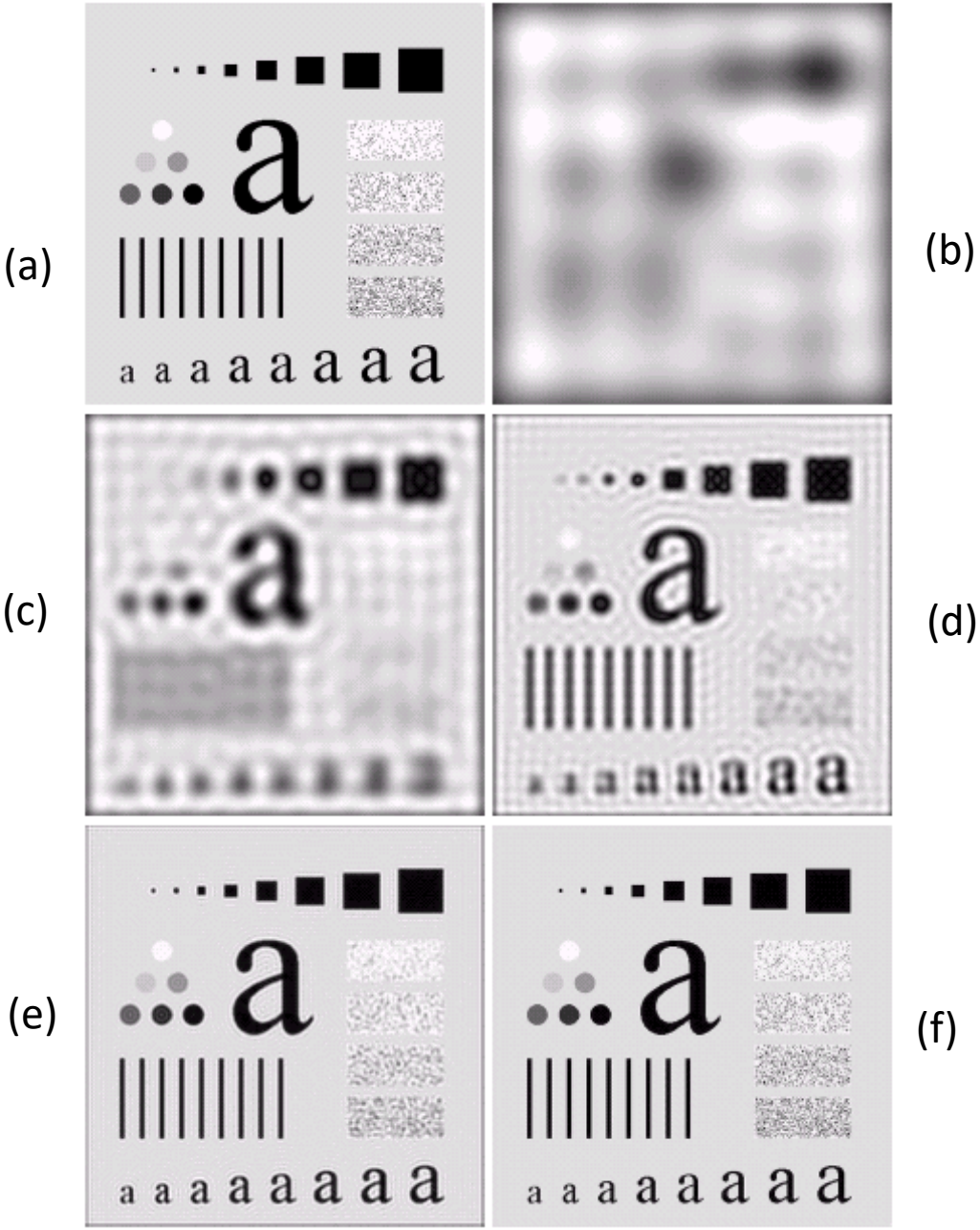
- Menimbulkan efek bergetar (*ringing*) pada citra *blur*, sebagai karakteristik ILPF

- Ini akibat dari diskontinuitas pada fungsi penapis



a b

FIGURE 4.11 (a) An image of size 500 × 500 pixels and (b) its Fourier spectrum. The superimposed circles have radii values of 5, 15, 30, 80, and 230, which enclose 92.0, 94.6, 96.4, 98.0, and 99.5% of the image power, respectively.



(a) Original image, (b)-(f) Results of ideal lowpass filtering with cuttoff frequencies set radii of 5, 15, 30, 80, and 230

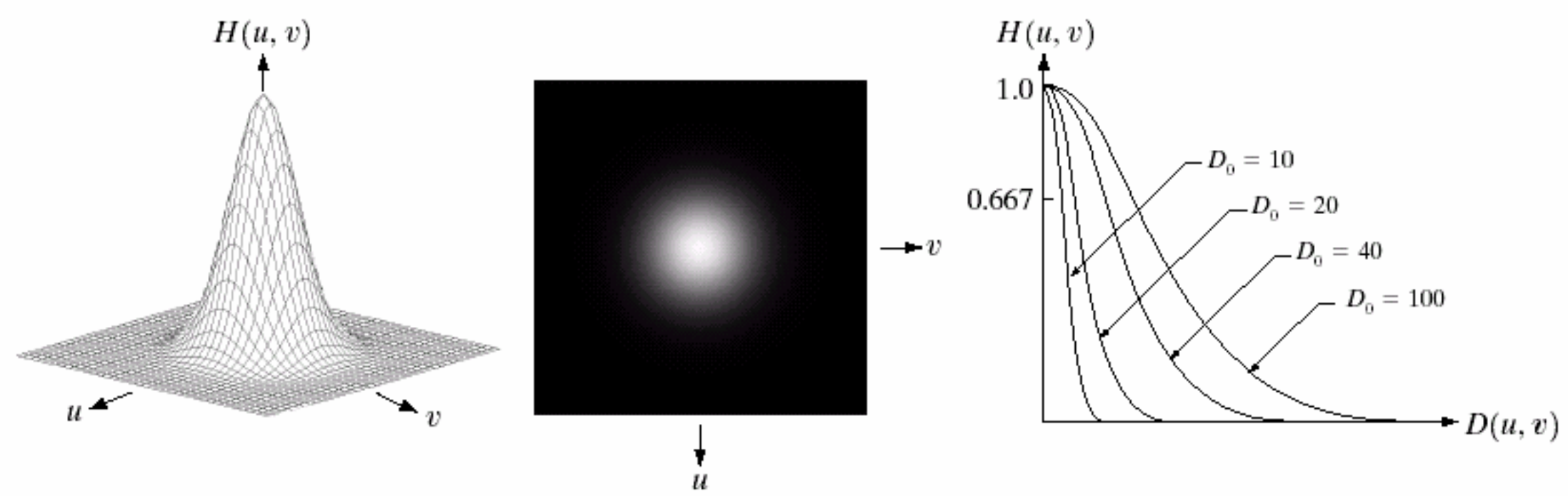
- Bagaimana cara mengatasi efek *ringing*? Gunakan GLPF atau BLPF.

- Respons frekuensi pada GLPF dan BLPF tidak memiliki transisi yang tajam
  - lebih sesuai untuk penghalusan gambar
  - tidak menimbulkan efek *ringing*.

Gaussian Low Pass Filter (GLPF)

Parameter D0 mengukur penyebaran kurva Gaussian.
Semakin besar nilai D0, semakin besar frekuensi cuttoff.

$$H(u,v) = e^{-D^2(u,v)/2D_0^2}$$



a b c

FIGURE 4.17 (a) Perspective plot of a GLPF transfer function. (b) Filter displayed as an image. (c) Filter radial cross sections for various values of D_0.

Catatan:

- Untuk GLPF, maka kode di dalam program ILPF:

```
H = double(D <=D0);
```

diganti menjadi

```
H = exp(-(D.^2)./(2*(D0^2)));
```

```
% Penapisan dalam ranah frekuensi dengan Gaussian Lowpass Filter (GLPF)

f = imread('cameraman.bmp');

% Ukuran citra
[M,N] = size(f);

% Konversi ke double
f = im2double(f);

% ========================================================
% Step 1: Zero padding menjadi 2M x 2N
% ========================================================

P = 2*M;
Q = 2*N;

% ========================================================
% Step 2: Membentuk citra padding
% ========================================================

fp = zeros(P,Q);
fp(1:M,1:N) = f;

figure; imshow(f); title('Original Image');

figure; imshow(fp); title('Padded Image');
```

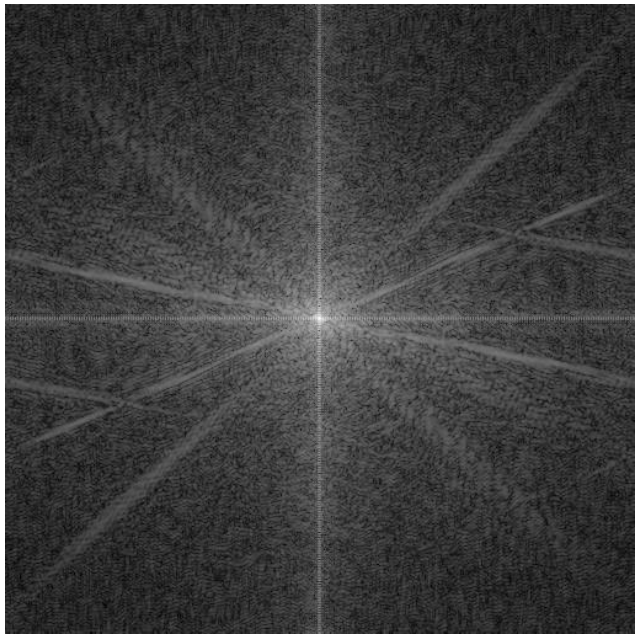
```
% ========================================================
% Step 3: Fourier Transform
% ========================================================

F = fft2(fp);
F = fftshift(F);

S2 = log(1 + abs(F));

figure;
imshow(S2,[]);
title('Fourier Spectrum');

% ========================================================
% Step 4: Membentuk Gaussian Low-Pass Filter
% ========================================================

D0 = 50;

% Koordinat frekuensi
u = -P/2:(P/2-1);
v = -Q/2:(Q/2-1);

[V,U] = meshgrid(v,u);

% Jarak dari pusat frekuensi
D = sqrt(U.^2 + V.^2);

% Gaussian LPF
H = exp(-(D.^2)/(2*D0^2));
```

```
% Tampilkan mask
figure; imshow(H,[]); title('Gaussian LPF Mask');

figure; mesh(H); title('Gaussian LPF Surface');

% ========================================================
% Step 5: Kalikan spektrum dengan filter
% ========================================================

LPF_f = H .* F;

% ========================================================
% Step 6: Inverse Fourier Transform
% ========================================================

LPF_f1 = ifftshift(LPF_f);

LPF_f2 = real(ifft2(LPF_f1));

figure; imshow(LPF_f2,[]);
title('Output Image after Inverse 2D DFT');

% ========================================================
% Step 7: Buang zero padding
% ========================================================

LPF_f2 = LPF_f2(1:M,1:N);

figure; imshow(LPF_f2,[]); title('Output Image');
```

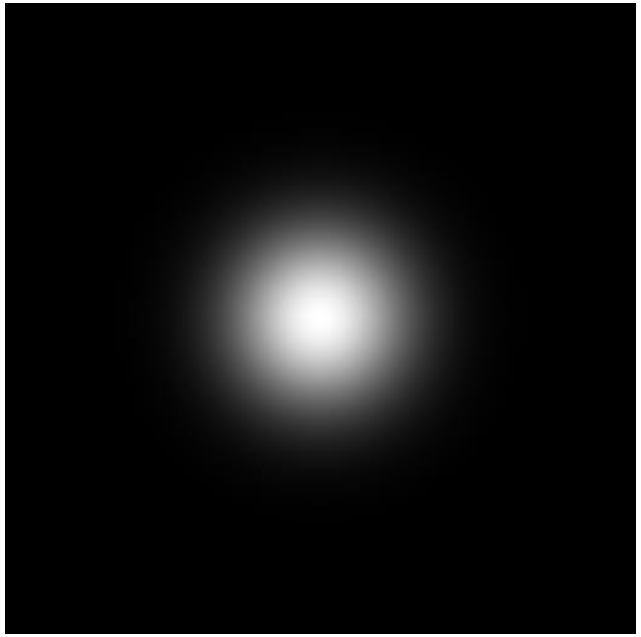
original image

padded image

Fourier spectrum



Gaussian Low Pas Filter

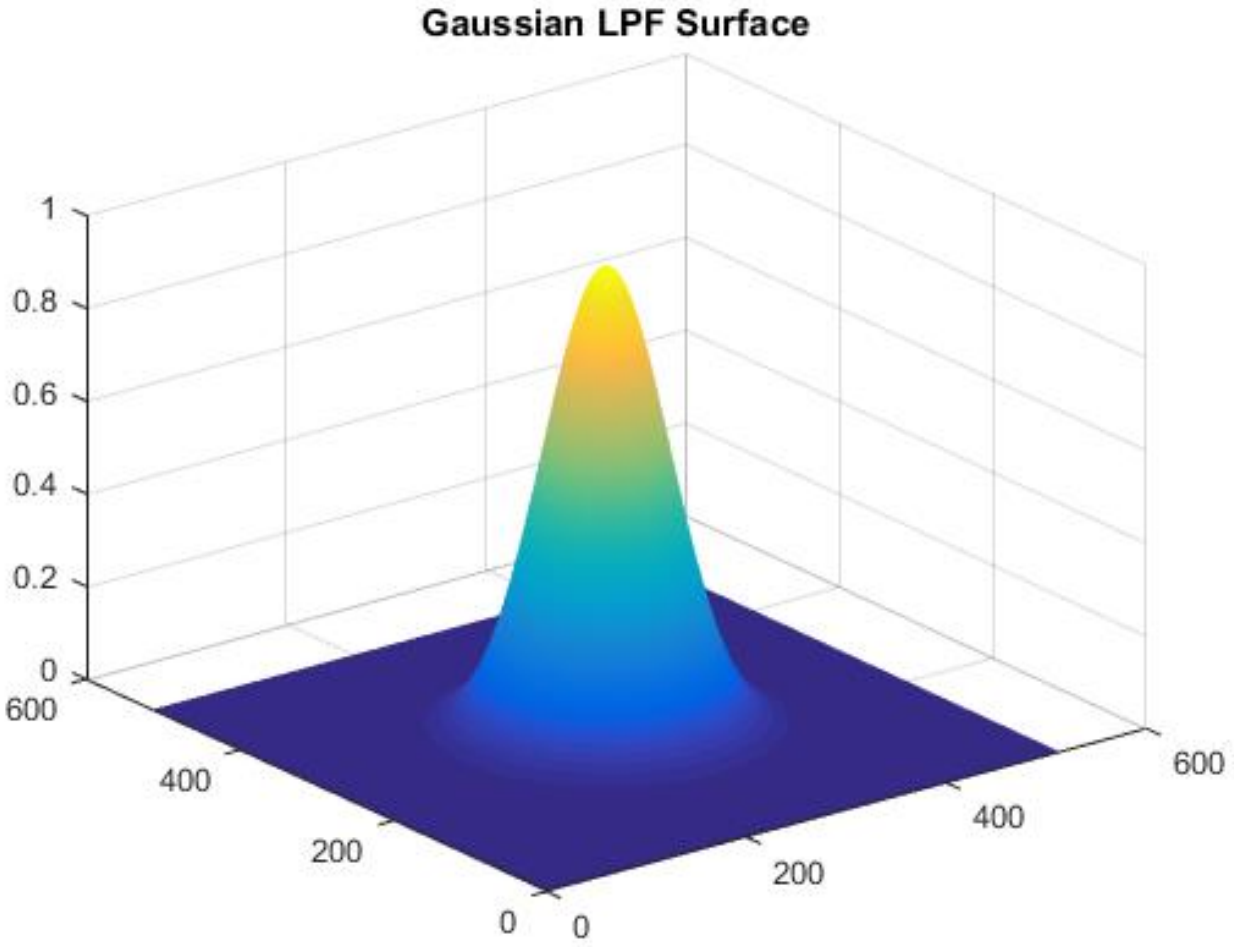


output image after inverse 2D DFT

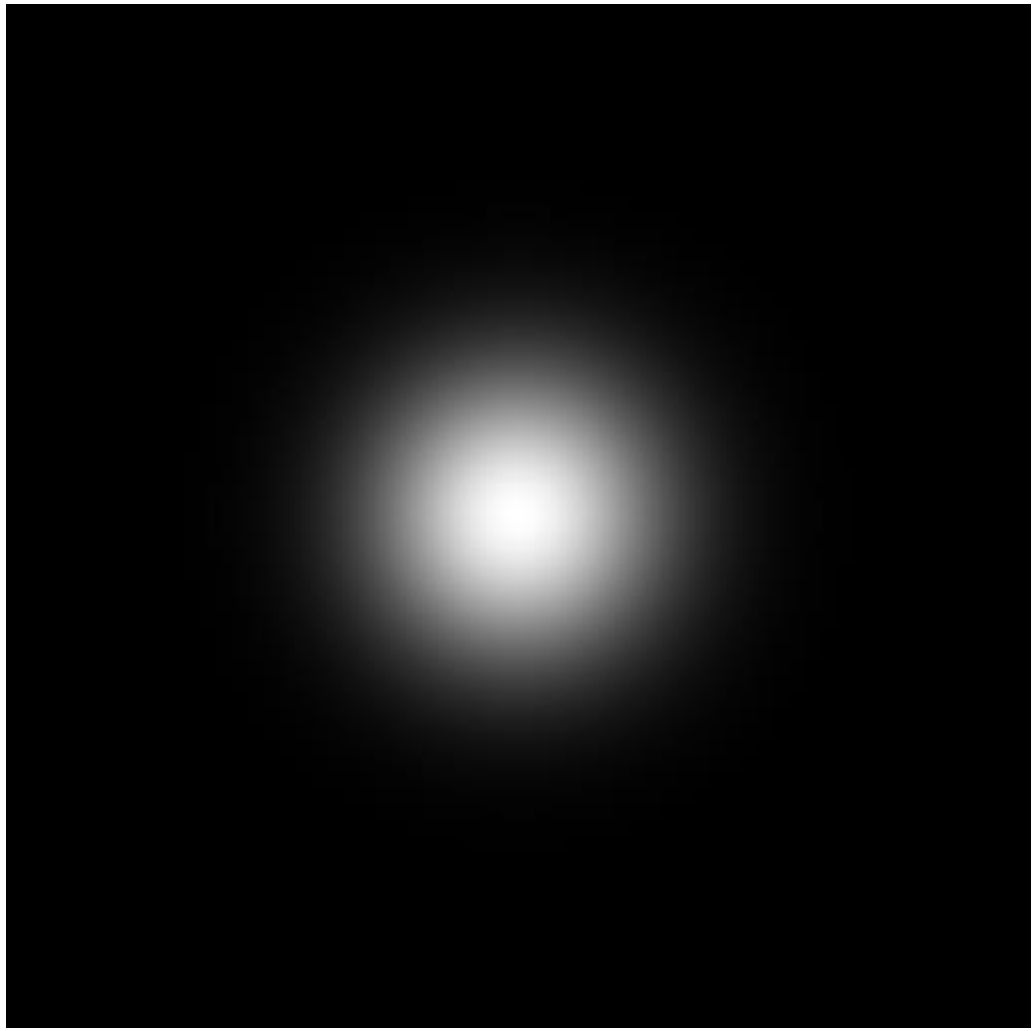
output image

Hasil run program dengan GLPF, D0 = 50

Output Image



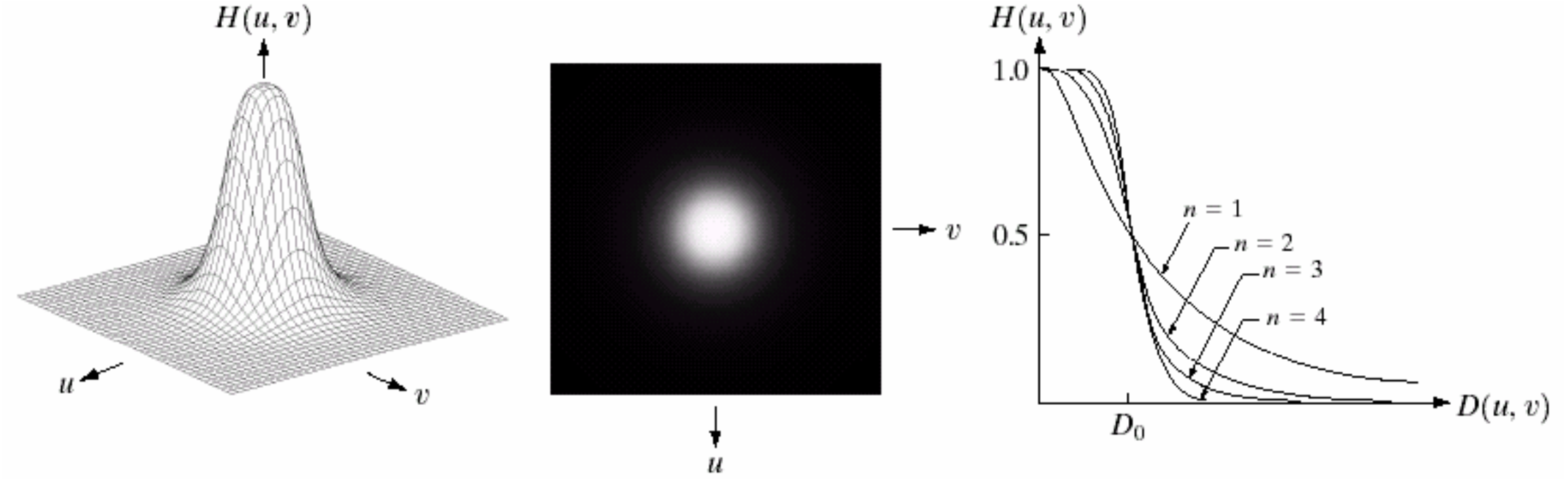
Gaussian LPF Mask



Butterworth Pass Filter (BLPF)

$$H(u,v) = \frac{1}{1+\left[D(u,v)/D_o\right]^{2n}}$$

n : filter order
D_0 : cutoff frequency



a b c

FIGURE 4.14 (a) Perspective plot of a Butterworth lowpass filter transfer function. (b) Filter displayed as an image. (c) Filter radial cross sections of orders 1 through 4.

Catatan:

- Untuk BLPF orde n, maka kode di dalam program ILPF:

```
H = double(D <=D0);
```

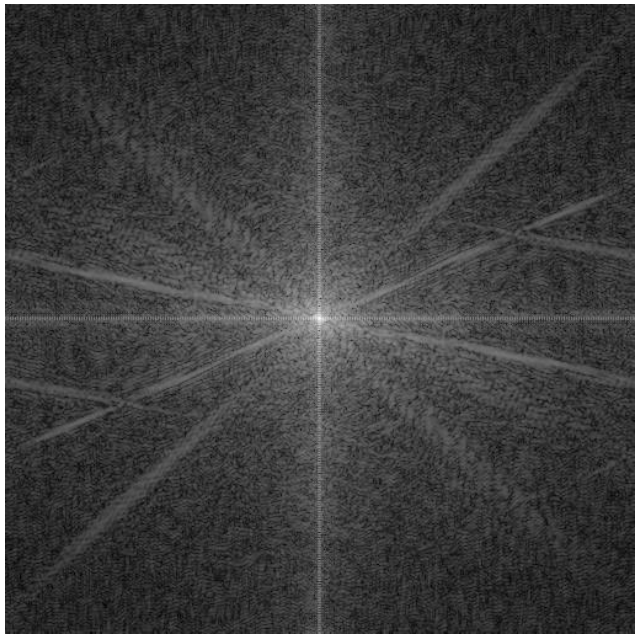
diganti menjadi

```
n = 1;    % default
H = 1./(1 + (D./D0).^(2*n));
```

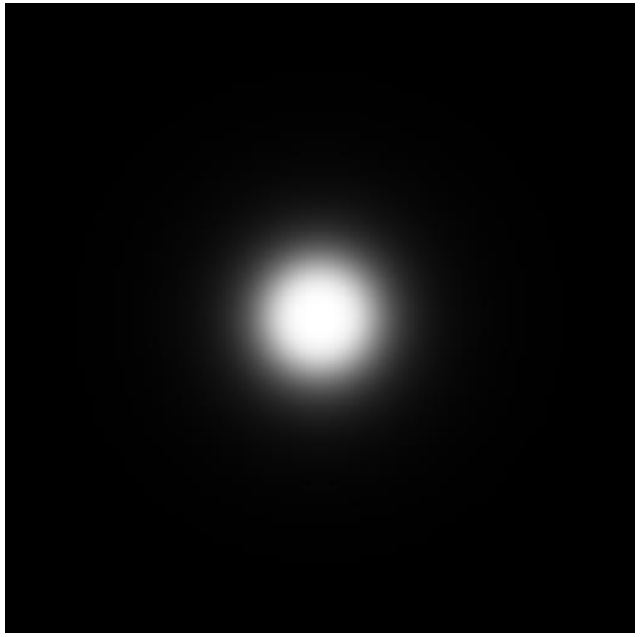
original image

padded image

Fourier spectrum



Butterworth Low Pas Filter

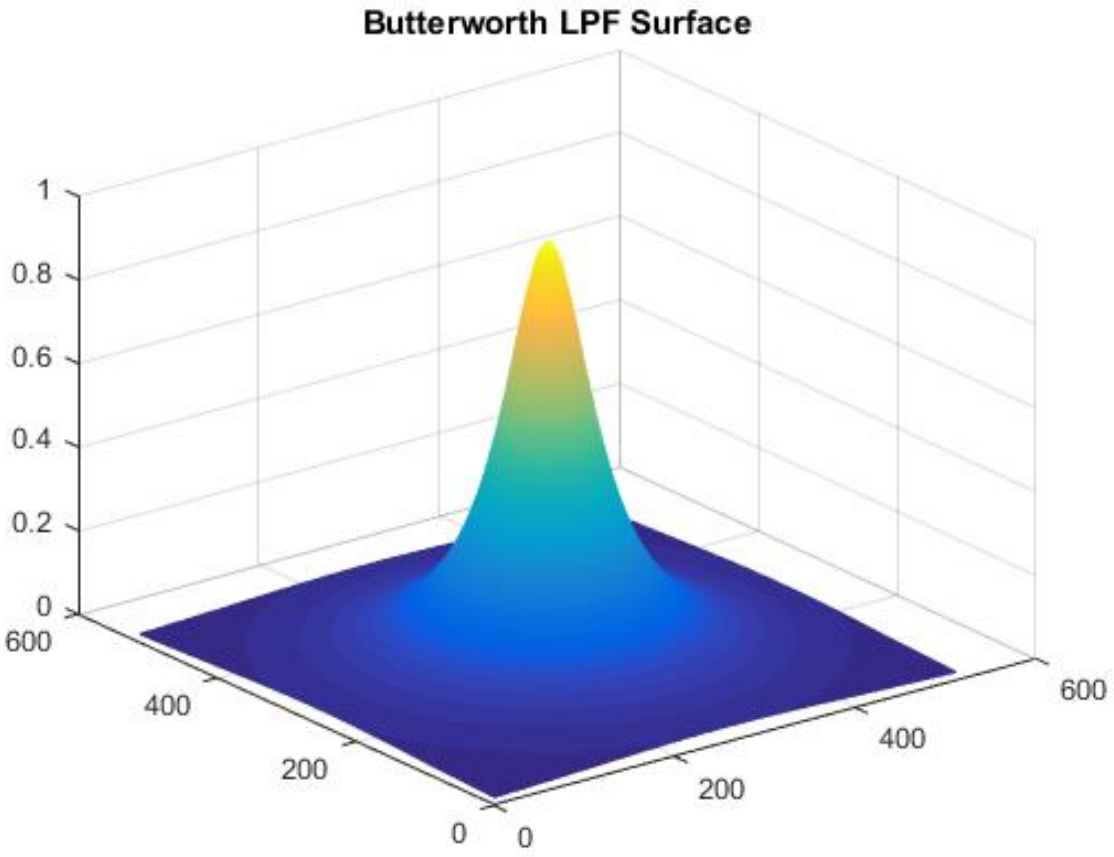


output image after inverse 2D DFT

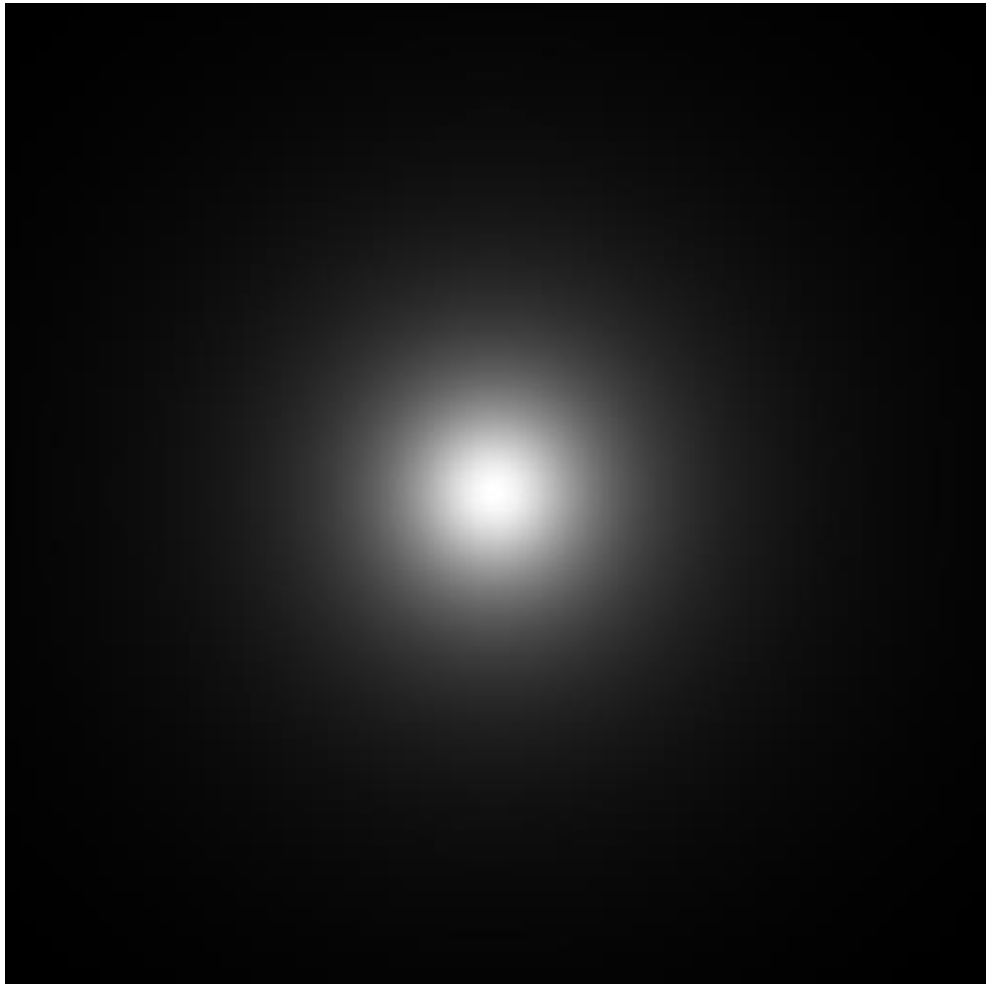
output image

Hasil run program dengan BLPF, n = 2, D0 = 50

Output Image



Butterworth LPF Mask



- Ideal, Gaussian, dan Butterworth low pass filter ketiga mempunyai tujuan dasar yang sama: melewatkan frekuensi rendah dan meredam frekuensi tinggi.
- Yang berbeda adalah seberapa tajam transisinya.
- Ideal LPF transisinya sangat tajam:

  frekuensi ≤ D0 → lolos

  frekuensi > D0 → ditolak

  Akibatnya dapat muncul ringing (Gibbs phenomenon) di sekitar tepi objek.

- Gaussian LPF transisinya sangat halus. Karena itu hasil *filtering* biasanya lebih natural dan *ringing* sangat kecil/tidak ada.
- Butterworth LPF berada di antara keduanya. Tingkat ketajaman transisinya dapat diatur menggunakan orde n.

- Perbandingan hasil

ILPF

GLPF

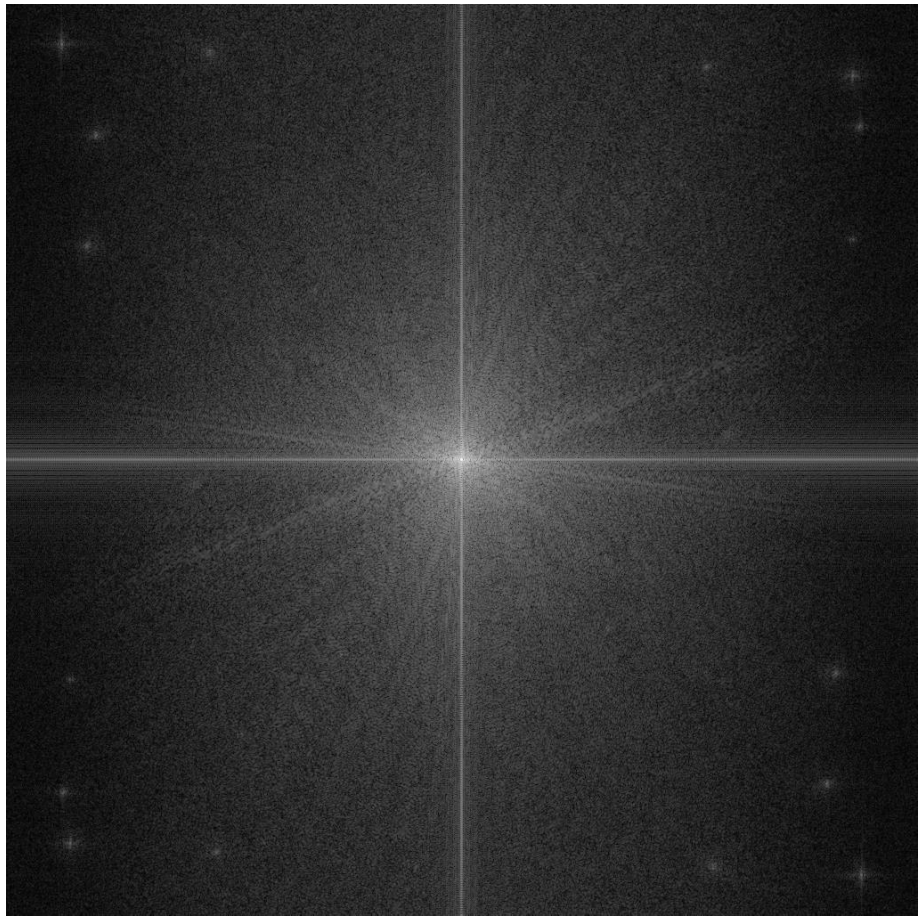
BLPF

Contoh hasil run pada citra *pepper* dengan Gaussian Low Pass Filter:

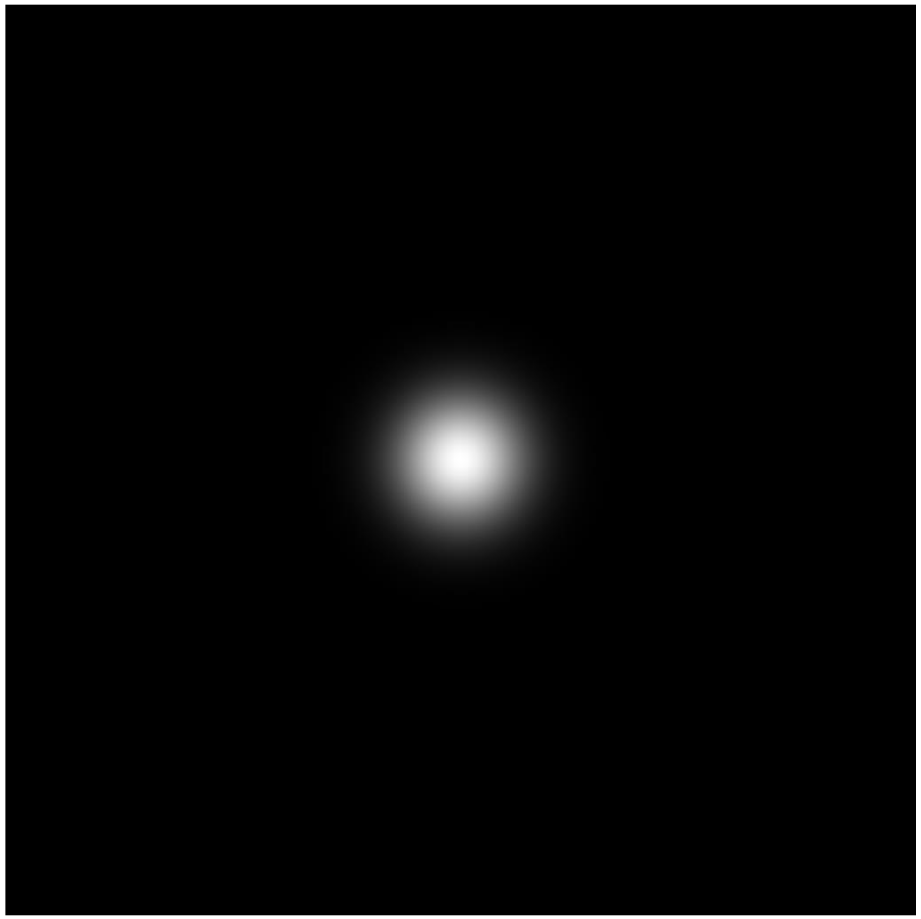
original image

padded image

Fourier spectrum



Gaussian Low Pas Filter



output image after inverse 2D DFT

output image

Contoh hasil ILPF

Contoh hasil GLPF

Contoh hasil BLPF

a b c d e f

FIGURE 4.15 (a) Original image. (b)–(f) Results of filtering with BLPFs with cutoff frequencies at radii of 5, 15, 30, 80, and 230, as shown in F Compare with Fig. 4.12.

a b c d e f

FIGURE 4.15 (a) Original image. (b)–(f) Results of filtering with BLPFs of order 2, with cutoff frequencies at radii of 5, 15, 30, 80, and 230, as shown in Fig. 4.11(b). Compare with Fig. 4.12.

Untuk apa sebenarnya ILPF, GLPF, dan BLPF?

Ketiga LPF tersebut pada dasarnya adalah alat untuk mengendalikan atau menghilangkan komponen frekuensi tinggi pada citra sembari memertahankan komponen frekuensi rendah. *Blurring* hanyalah salah satu akibatnya.

Kegunaannya untuk:

1. Mengurangi derau

Banyak derau pada citra mempunyai komponen frekuensi tinggi. Karena LPF melemahkan frekuensi tinggi, LPF dapat digunakan untuk mengurangi derau acak

2. *Preprocessing* sebelum segmentasi

LPF sering digunakan sebelum melakukan *image segmentation*.Misalnya citra mempunyai derau kecil-kecil yang menyebabkan *thresholding* menghasilkan banyak objek kecil yang tidak diinginkan. LPF membantu membuat objek lebih homogen sehingga proses segmentasi menjadi lebih stabil.

Contoh pengurangan derau dengan GLPF:

```
% Membaca citra
f = imread('cameraman.bmp');

% Tambahkan Gaussian noise
g = imnoise(f, 'gaussian', 0, 0.01);

% Ukuran citra
[M,N] = size(f);

% Transformasi Fourier
F = fft2(double(g));

% Membuat koordinat frekuensi
u = 0:(M-1);
v = 0:(N-1);

idx = find(u > M/2);
u(idx) = u(idx) - M;

idy = find(v > N/2);
v(idy) = v(idy) - N;

[V,U] = meshgrid(v,u);
```

```
% Jarak dari pusat frekuensi
D = sqrt(U.^2 + V.^2);

% Gaussian Low-Pass Filter
D0 = 30;
H = exp(-(D.^2)/(2*D0^2));

% Karena F memiliki frekuensi nol di pojok,
% sedangkan H juga dibuat dengan frekuensi nol di pojok,
% keduanya dapat langsung dikalikan.
G = H .* F;

% Transformasi Fourier balik
g_filtered = real(ifft2(G));

% Menampilkan hasil
figure;

subplot(1,3,1);
imshow(f);
title('Citra Asli');

subplot(1,3,2);
imshow(g);
title('Citra + Gaussian Noise');

subplot(1,3,3);
imshow(uint8(g_filtered));
title('Hasil Gaussian LPF');
```

Hasil run program:

Citra Asli

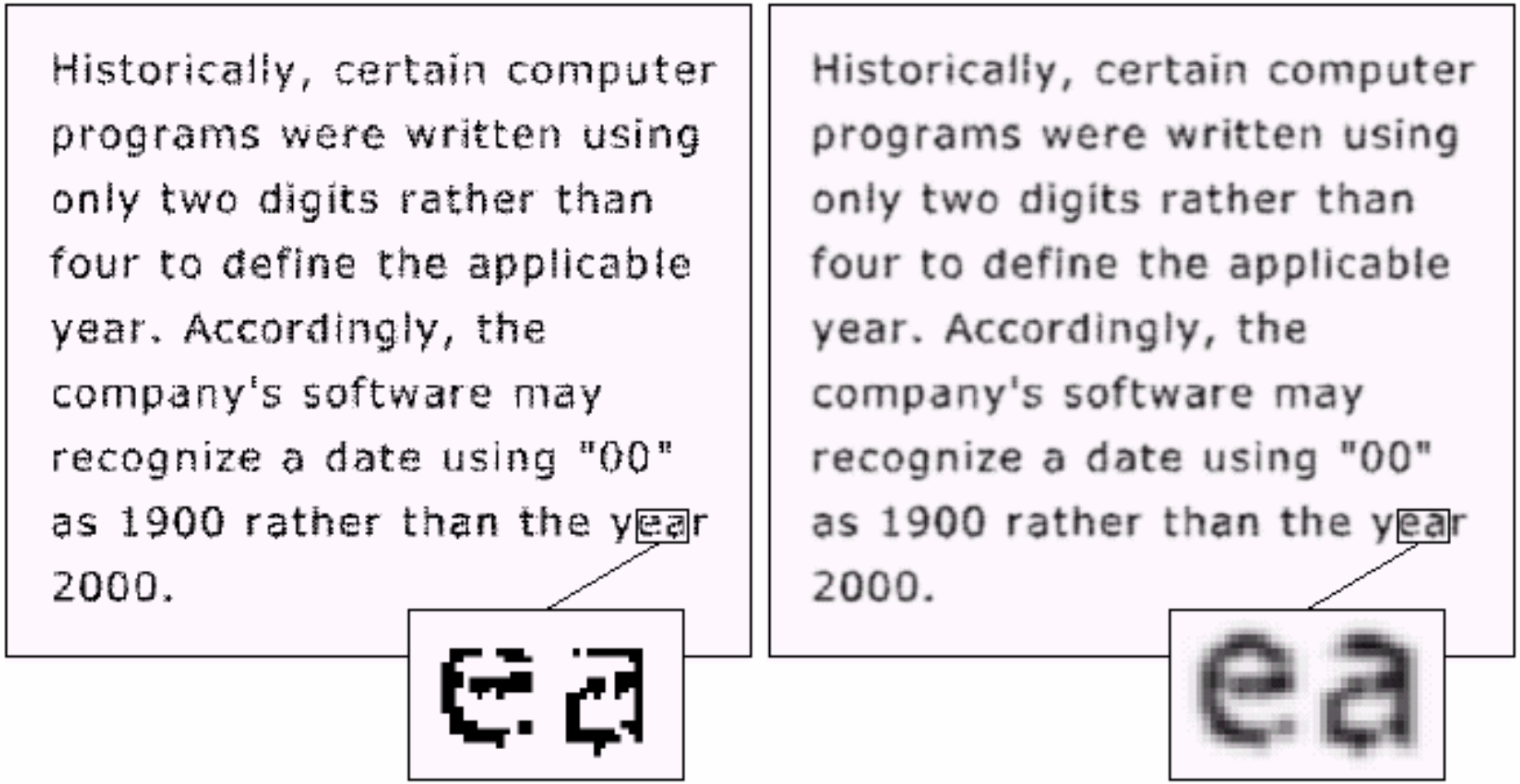
Citra + Gaussian Noise

Hasil Gaussian LPF

Efek *blurring* pada GLPF dapat menyambung bagian-bagian yang putus pada citra font karakter:

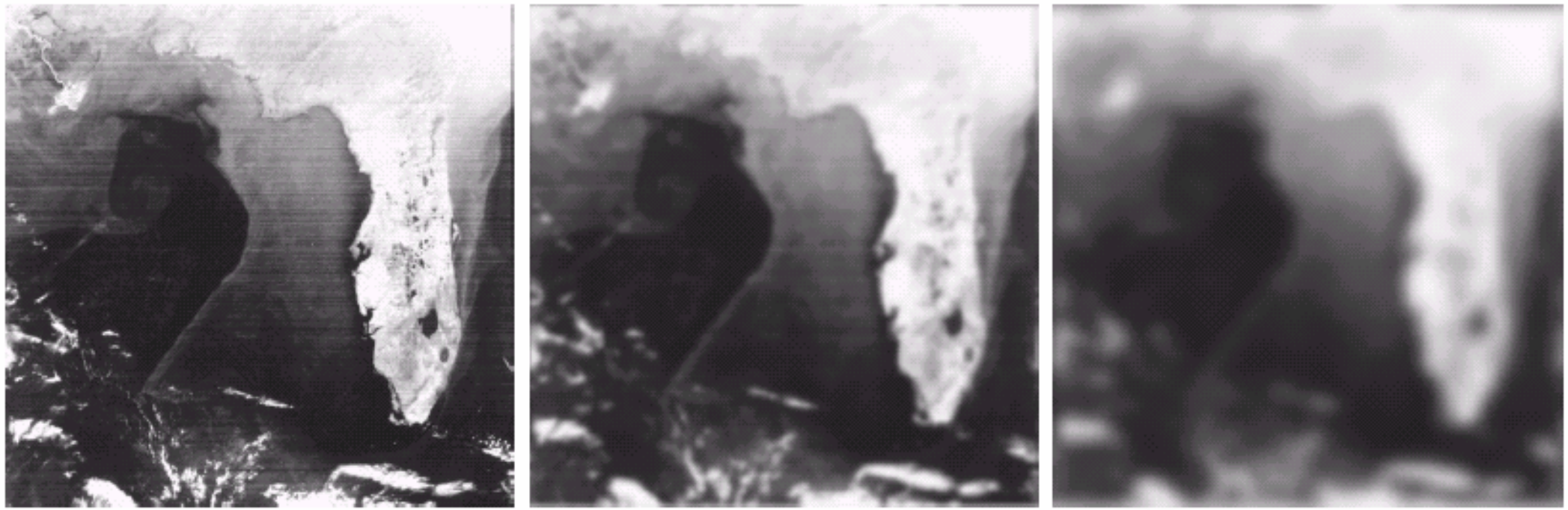
a b

FIGURE 4.19
(a) Sample text of poor resolution (note broken characters in magnified view). (b) Result of filtering with a GLPF (broken character segments were joined).



a b c

FIGURE 4.20 (a) Original image (1028 × 732 pixels). (b) Result of filtering with a GLPF with $D_0 = 100$. (c) Result of filtering with a GLPF with $D_0 = 80$. Note reduction in skin fine lines in the magnified sections of (b) and (c).



a b c

FIGURE 4.21 (a) Image showing prominent scan lines. (b) Result of using a GLPF with $D_0 = 30$. (c) Result of using a GLPF with $D_0 = 10$. (Original image courtesy of NOAA.)

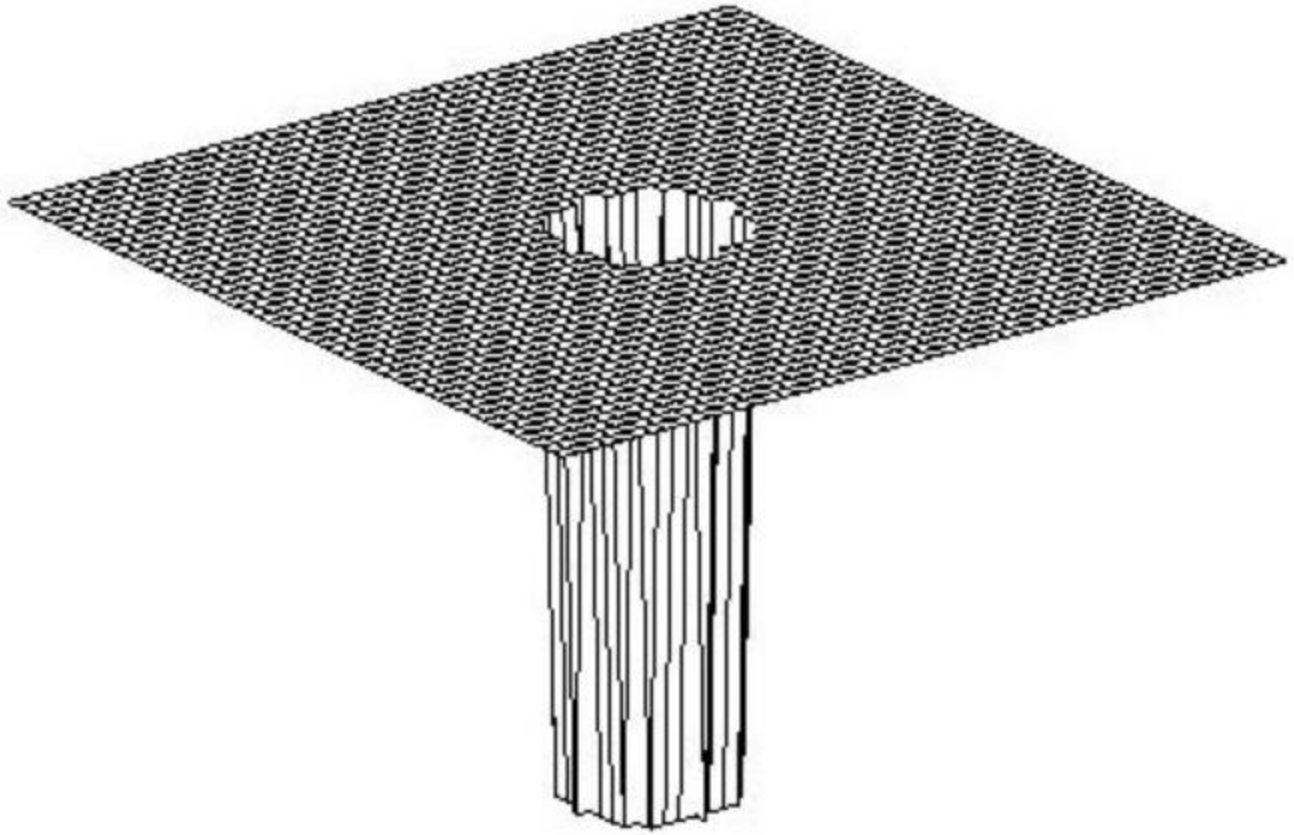
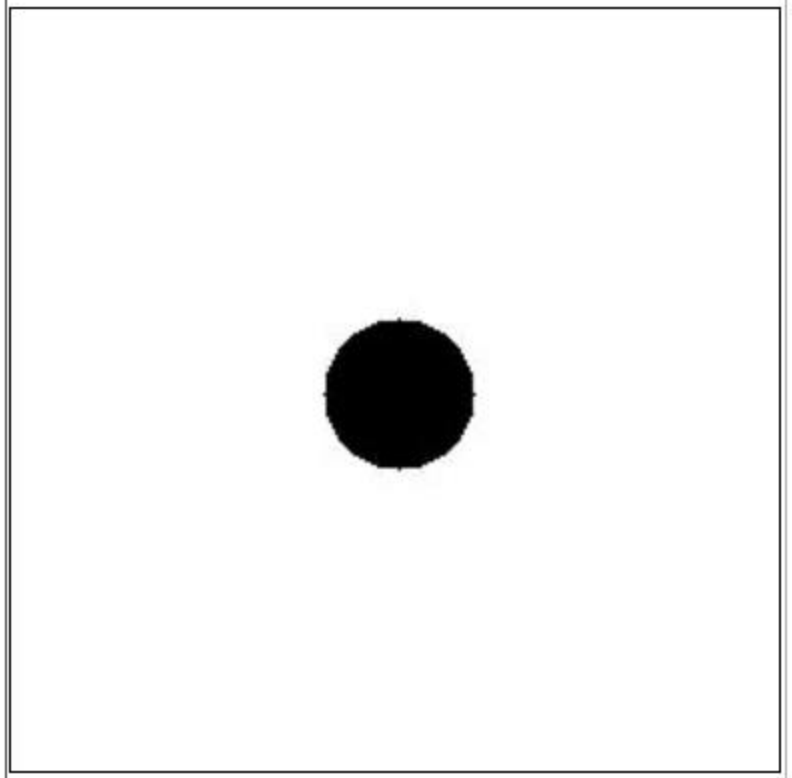
Highpass filter (HPF) dalam ranah frekuensi

- Penapis lolos tinggi (*highpass filter*) bertujuan menekan komponen berfrekuensi rendah dan dan meloloskan (sekaligus memperkuat) komponen berfrekuensi tinggi.

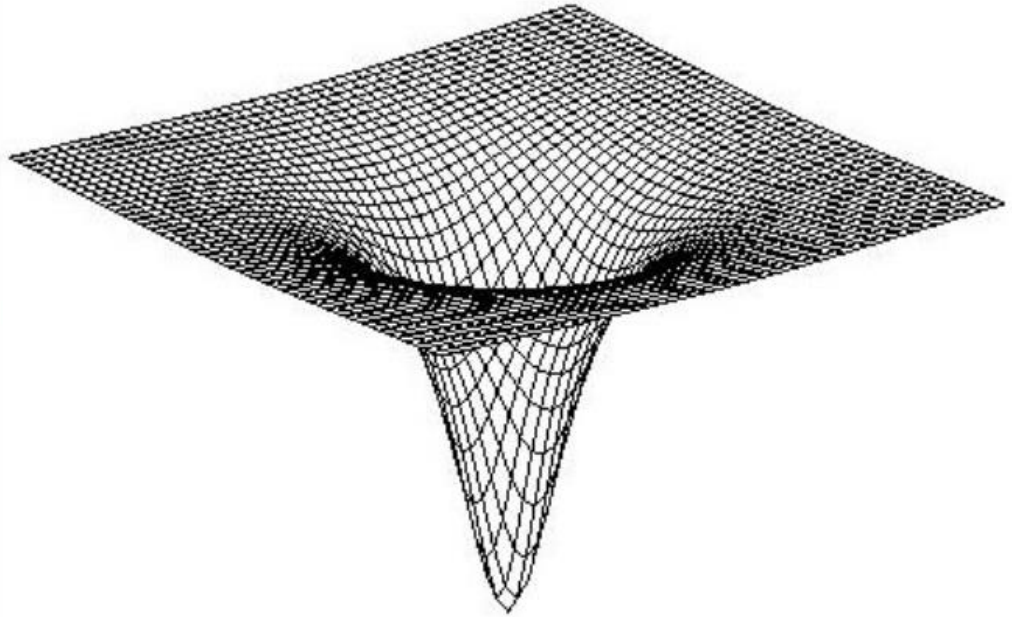
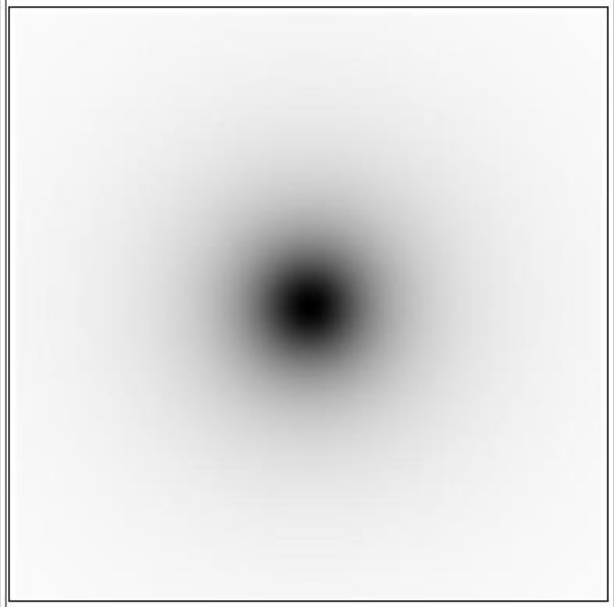
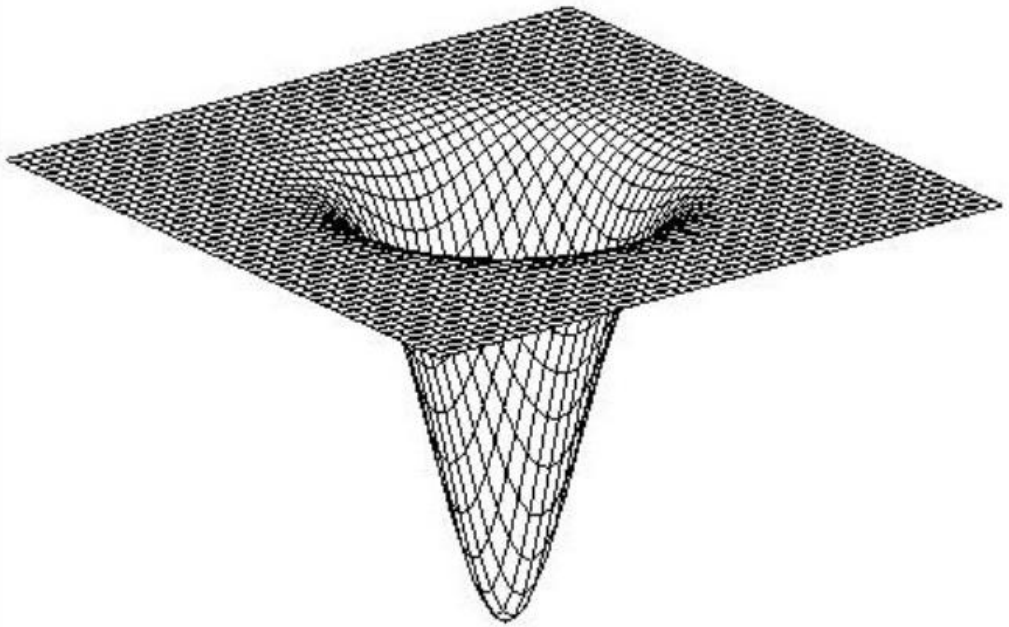
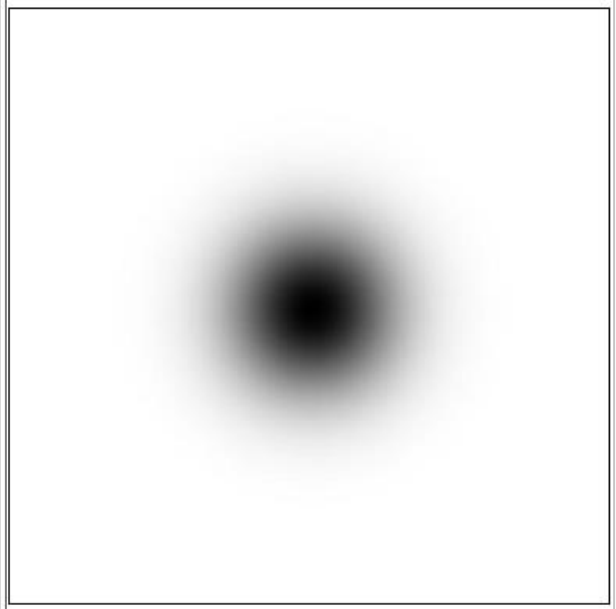
- Menghasilkan efek penajaman pada tepi (edge) citra (atau *sharpened image*)
- Hubungan antara penapis lolos tinggi (H_{hp}) dan penapis lolos rendah (H_{lp})

$$H_{hp}(u,v) = 1 - H_{lp}(u,v)$$

- Tiga buah HPF yang utama:
  1. *Ideal highpass filter* (IHPF)
  2. *Butterworth highpass filter* (BHPF)
  3. *Gaussian highpass filter* (GHPF)

Highpass Filter	Mesh	Image
Ideal		

$$H(u,v) = \begin{cases} 0 \text{ if } D(u,v) \le D_0 \\ 1 \text{ if } D(u,v) > D_0 \end{cases}$$

Butterworth			$H(u,v)=\frac{1}{1+\left[\mathrm{D}_0/D(u,v)\right]^{2n}}$
Gaussian			$H(u,v)=1-e^{-D^2(u,v)/2D_0^2}$

Ideal highpass filter

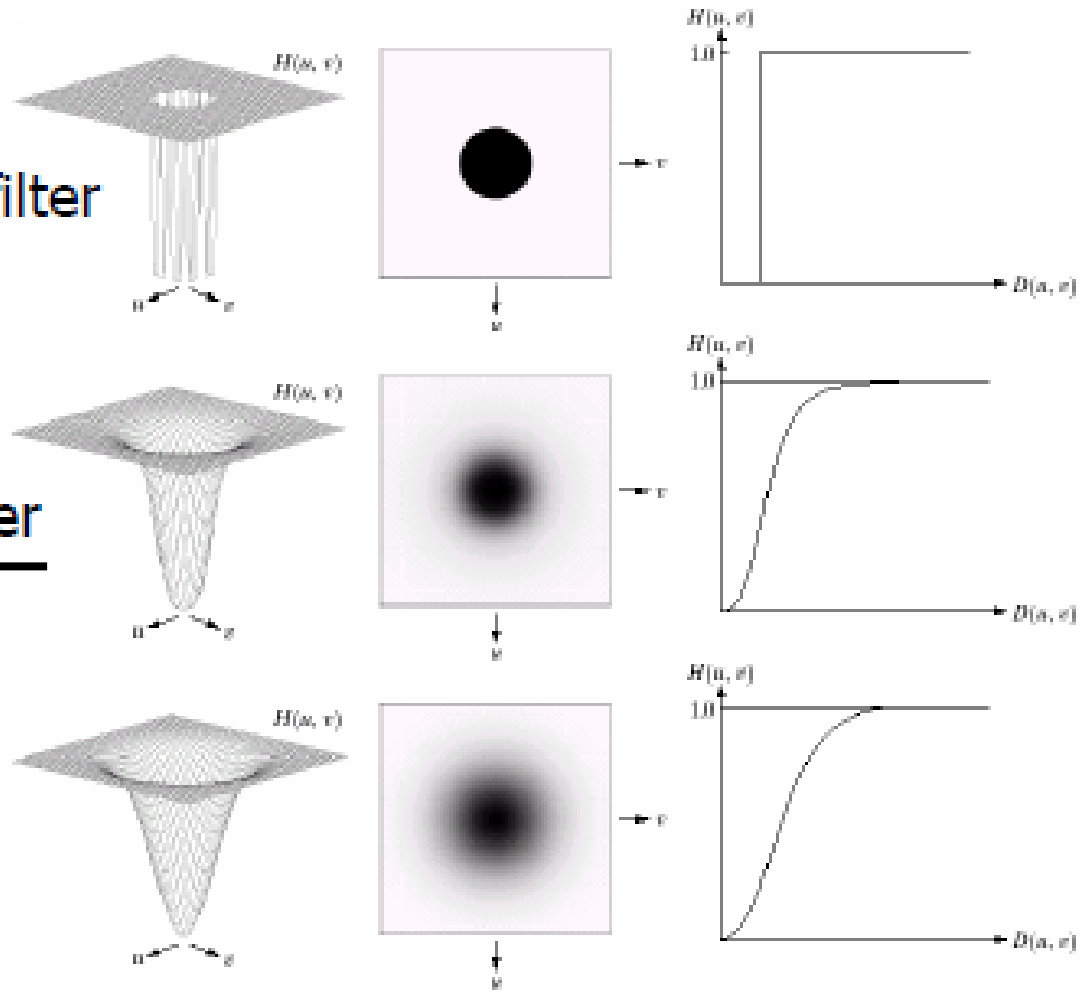
$$H(u,v)=\begin{cases}0 \text{ if } D(u,v)\le D_0\\ 1 \text{ if } D(u,v)> D_0\end{cases}$$

Butterworth highpass filter

$$H(u,v)=\frac{1}{1+\left[D_0/D(u,v)\right]^{2n}}$$

Gaussian highpass filter

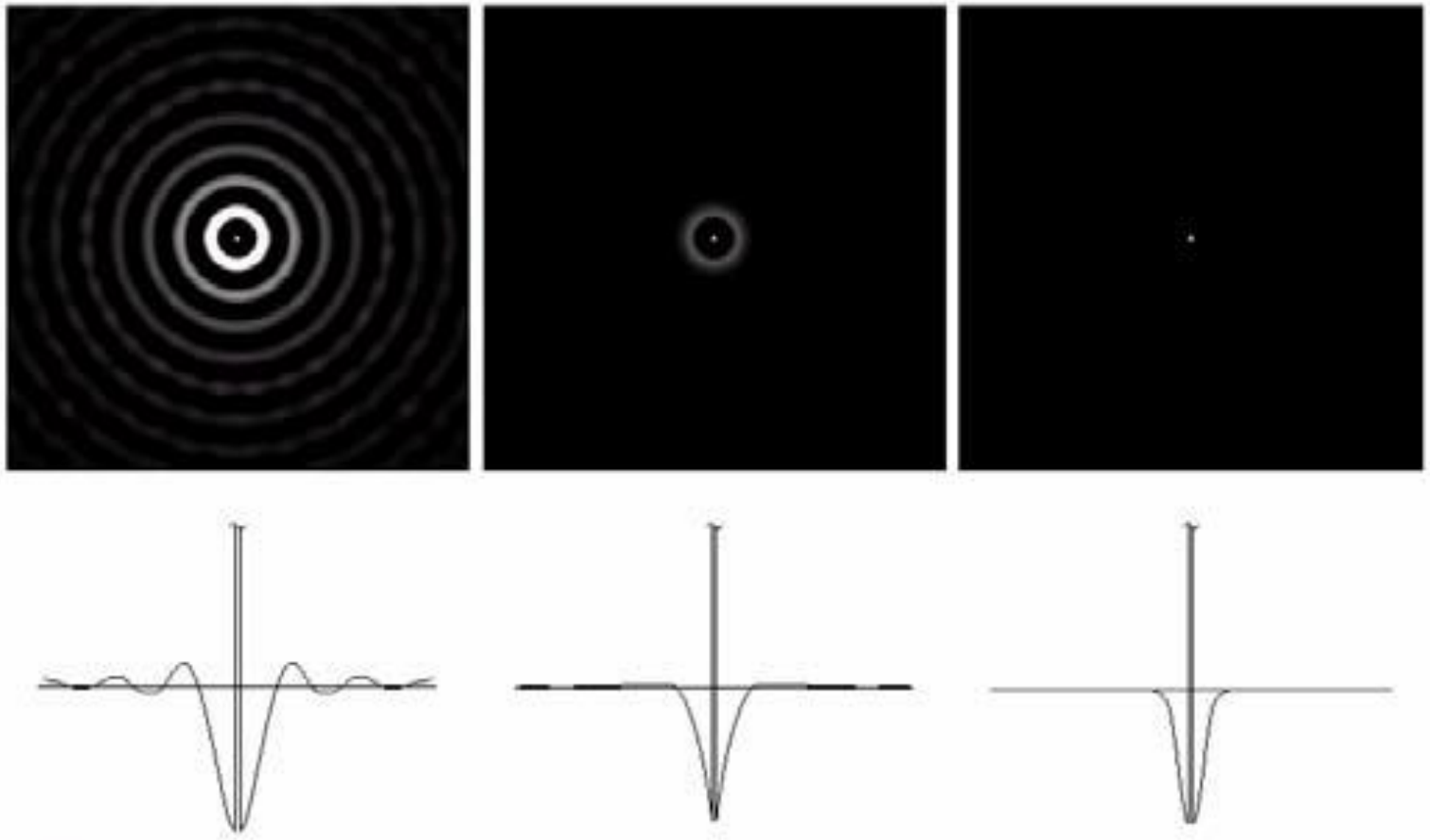
$$H(u,v)=1-e^{-D^2(u,v)/2D_0^2}$$



a b c
d e f
g h i

FIGURE 4.22 Top row: Perspective plot, image representation, and cross section of a typical ideal highpass filter. Middle and bottom rows: The same sequence for typical Butterworth and Gaussian highpass filters.

Representasi Spasial Ideal, Butterworth dan Gaussian Highpass Filter



a b c

FIGURE 4.23 Spatial representations of typical (a) ideal, (b) Butterworth, and (c) Gaussian frequency domain highpass filters, and corresponding gray-level profiles.

```
% ============================================================
% Step 3: Fourier Transform
% ============================================================
F = fft2(fp);
F = fftshift(F);

% Fourier spectrum
S2 = log(1 + abs(F));

figure; imshow(S2,[]); title('Fourier Spectrum');

% ============================================================
% Step 4: Membentuk Gaussian High-Pass Filter
% ============================================================
D0 = 50;

% Koordinat frekuensi
u = -P/2:(P/2-1);
v = -Q/2:(Q/2-1);

[V,U] = meshgrid(v,u);

% Jarak dari pusat frekuensi
D = sqrt(U.^2 + V.^2);

% Gaussian High-Pass Filter
H = 1 - exp(-(D.^2)/(2*D0^2));

% Tampilkan mask
figure; imshow(H,[]); title('Gaussian HPF Mask');
figure; mesh(H); title('Gaussian HPF Surface');
```

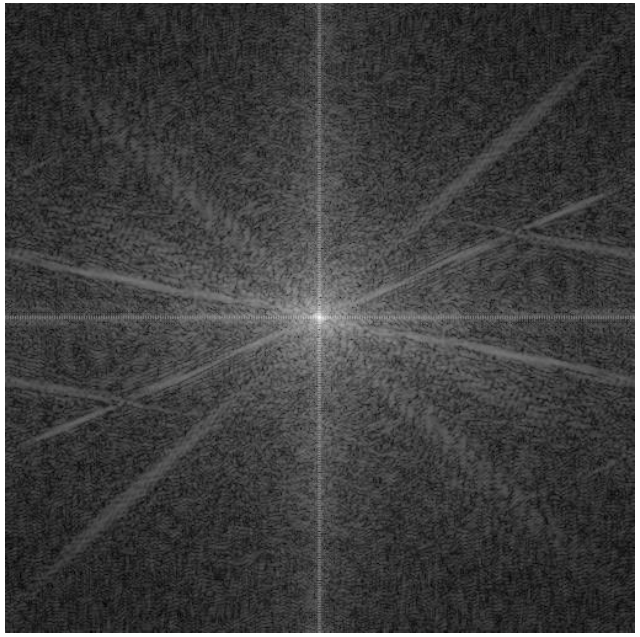
```
% ============================================================
% Step 5: Kalikan spektrum dengan filter
% ============================================================

HPF_f = H .* F;

% ============================================================
% Step 6: Inverse Fourier Transform
% ============================================================

HPF_f1 = ifftshift(HPF_f);

HPF_f2 = real(ifft2(HPF_f1));

figure;
imshow(HPF_f2,[]);
title('Output Image after Inverse 2D DFT');

% ============================================================
% Step 7: Buang zero padding
% ============================================================

HPF_f2 = HPF_f2(1:M,1:N);

figure;
imshow(HPF_f2,[]);
title('Output Image - Gaussian HPF');
```

```
% ================================================================
% PENAPISAN DALAM RANAH FREKUENSI DENGAN GAUSSIAN HIGH-PASS FILTER (GHPF)
% ================================================================

% Membaca citra
f = imread('cameraman.bmp');

% Ukuran citra
[M,N] = size(f);

% Konversi ke double
f = im2double(f);

% ========================================================
% Step 1: Zero padding menjadi 2M x 2N
% ========================================================
P = 2*M;
Q = 2*N;

% ========================================================
% Step 2: Membentuk citra padding
% ========================================================
fp = zeros(P,Q);
fp(1:M,1:N) = f;

figure; imshow(f); title('Original Image');

figure; imshow(fp); title('Padded Image’);
```

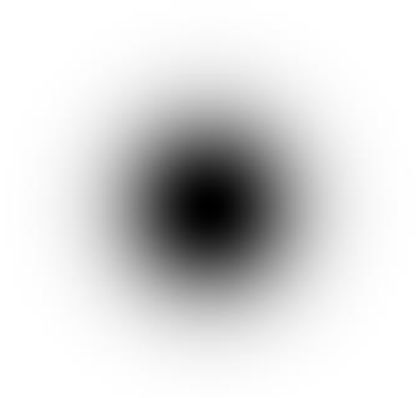
original image

padded image

Fourier spectrum



Gaussian High Pas Filter

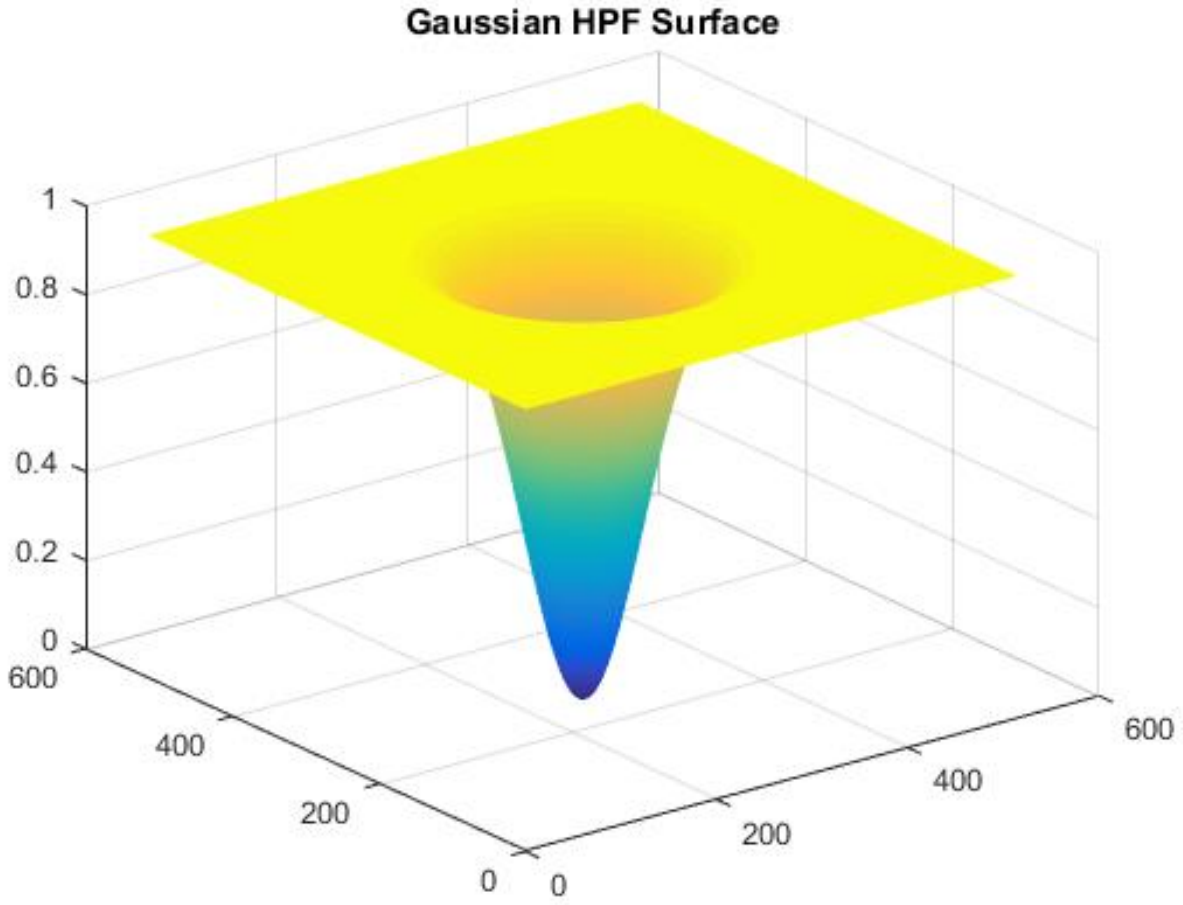


Output Image after Inverse 2D DFT

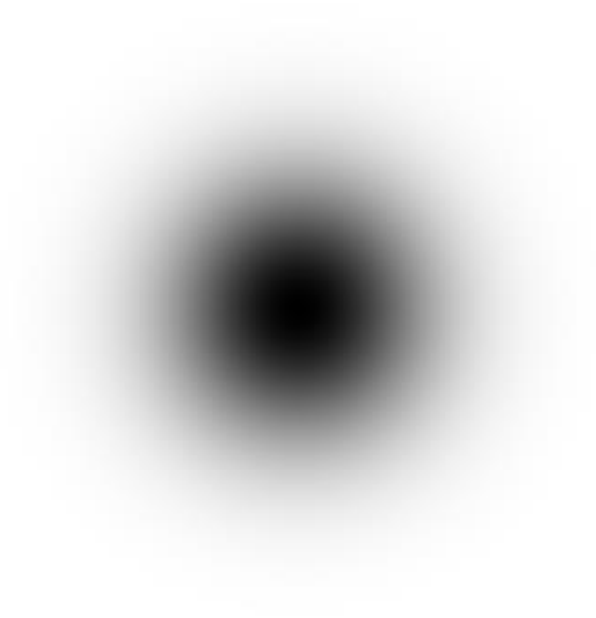
Output Image - Gaussian HPF

Hasil run program dengan GHPF, D0 = 50

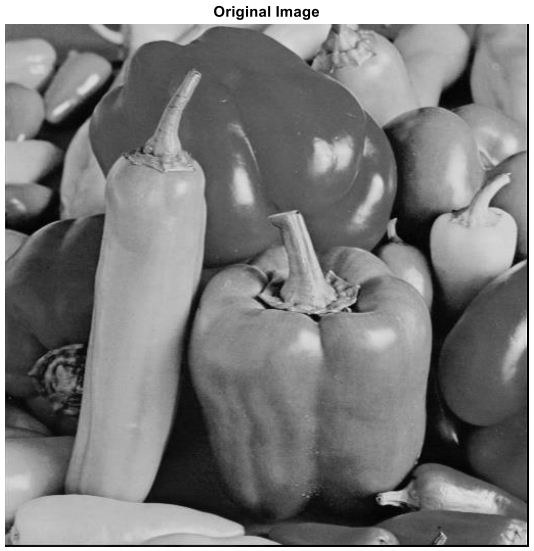
Output Image - Gaussian HPF

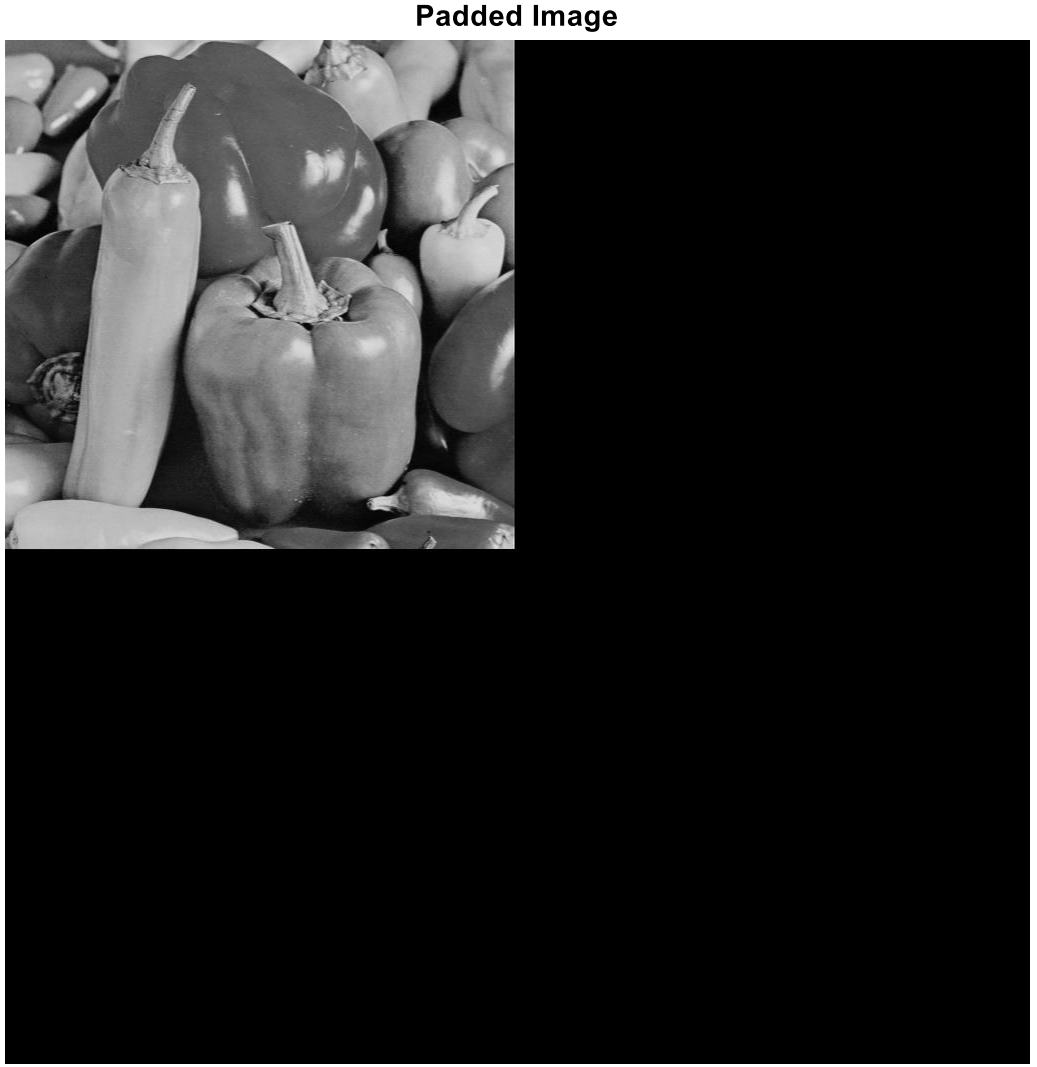


Gaussian HPF Mask

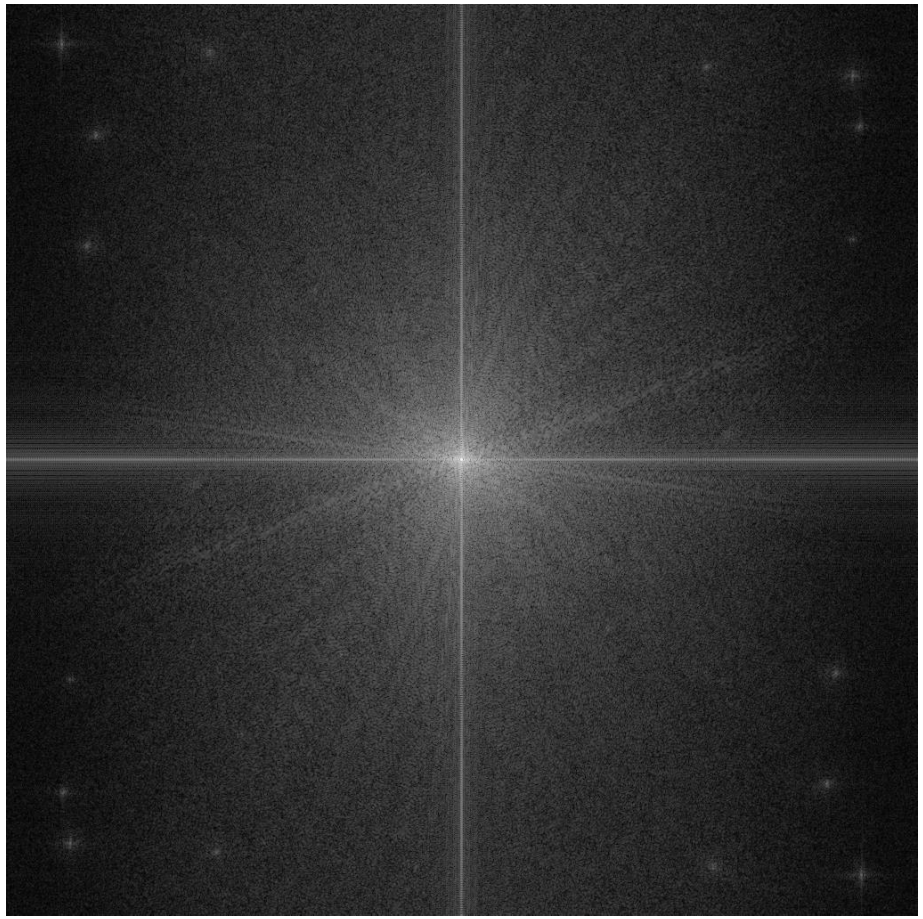


Contoh hasil lain dengan citra *pepper*:

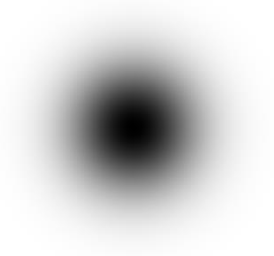




Fourier spectrum



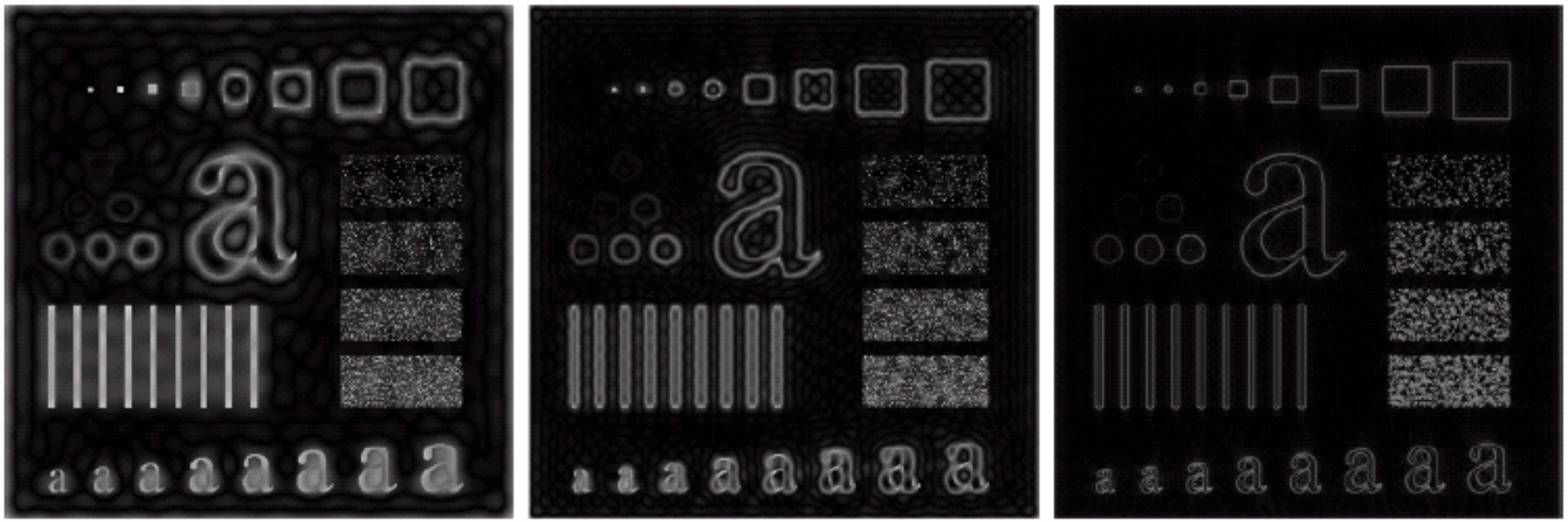
Gaussian High Pas Filter



Output Image after Inverse 2D DFT

Output Image - Gaussian HPF

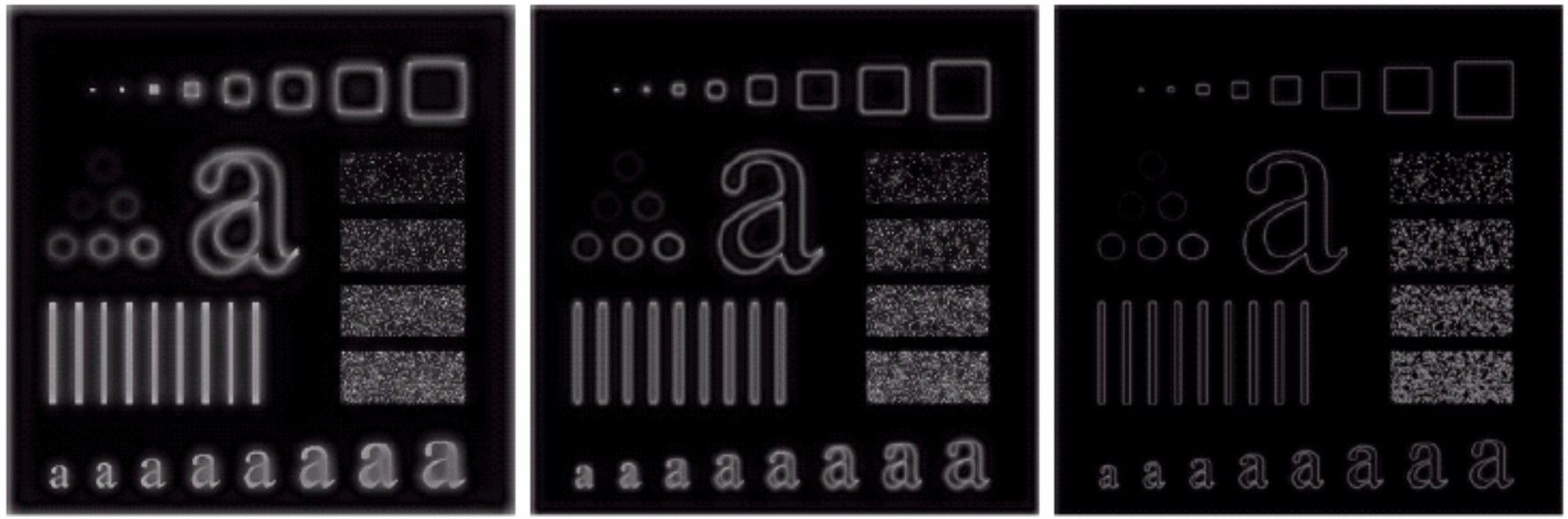
Example: result of IHPF



a b c

FIGURE 4.24 Results of ideal highpass filtering the image in Fig. 4.11(a) with $D_0 = 15$, 30, and 80, respectively. Problems with ringing are quite evident in (a) and (b).

Example: result of BHPF



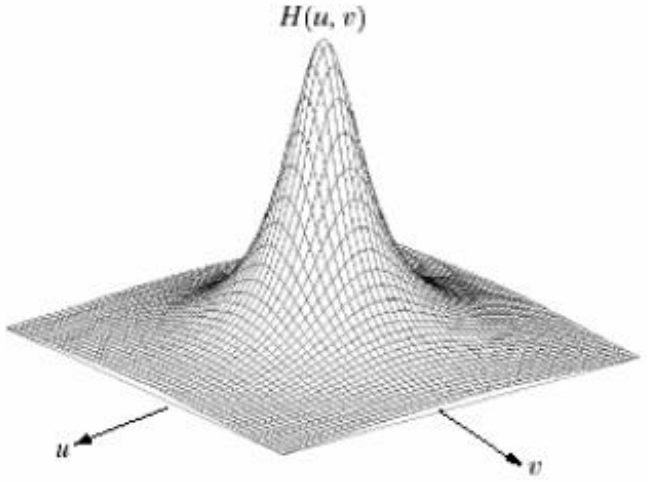
a b c

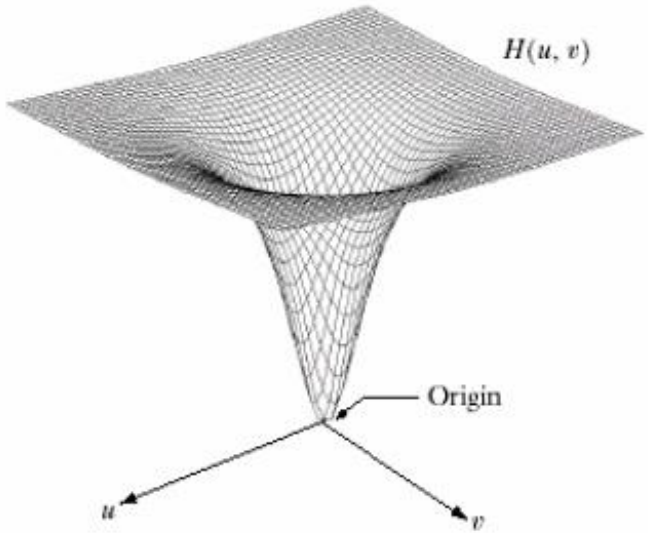
FIGURE 4.25 Results of highpass filtering the image in Fig. 4.11(a) using a BHPF of order 2 with $D_0 = 15$, 30, and 80, respectively. These results are much smoother than those obtained with an ILPF.

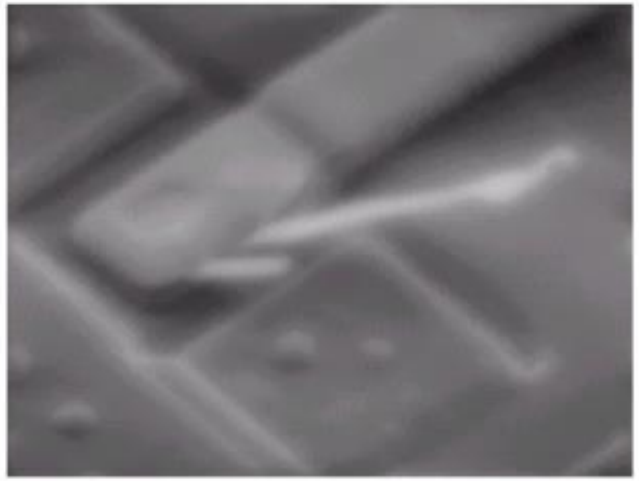
Example: result of GHPF

a b c

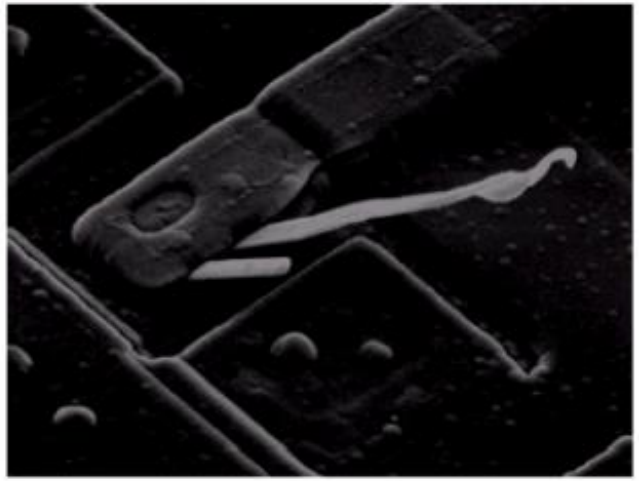
FIGURE 4.26 Results of highpass filtering the image of Fig. 4.11(a) using a GHPF of order 2 with $D_0 = 15$, 30, and 80, respectively. Compare with Figs. 4.24 and 4.25.







Low pass filter



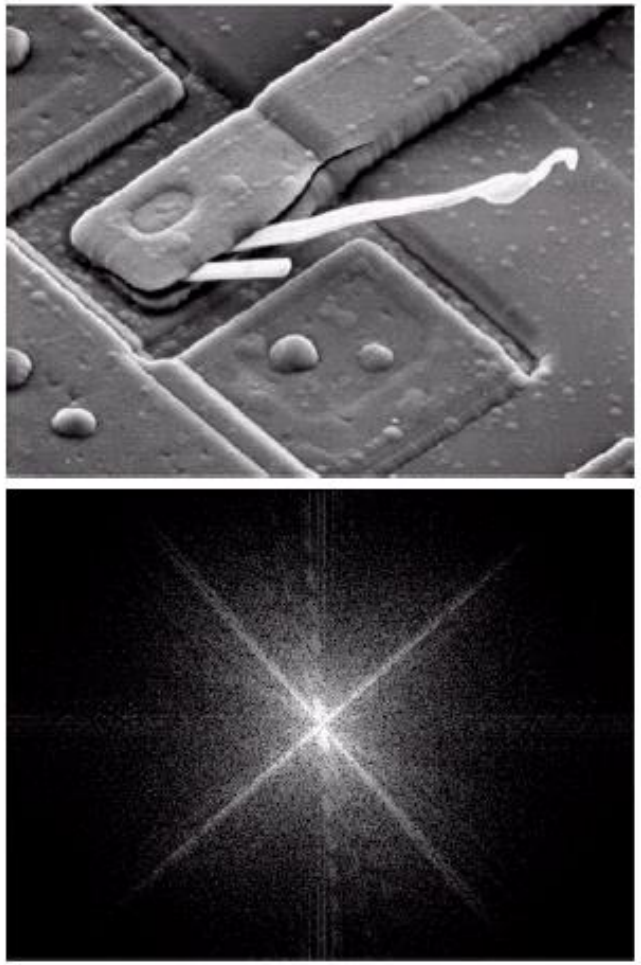
High pass filter

a b
c d

FIGURE 4.7 (a) A two-dimensional lowpass filter function. (b) Result of lowpass filtering the image in Fig. 4.4(a). (c) A two-dimensional highpass filter function. (d) Result of highpass filtering the image in Fig. 4.4(a).

Penapis lubang (*notch filter*)

$$H(u,v) = \begin{cases} 0 & if\ (u,v) = (M/2, N/2) \\ 1 & otherwise \end{cases}$$



a
b

FIGURE 4.4
(a) SEM image of a damaged integrated circuit. (b) Fourier spectrum of (a). (Original image courtesy of Dr. J. M. Hudak, Brockhouse Institute for Materials Research, McMaster University, Hamilton, Ontario, Canada.)

Note: F(0,0) is equal to the average value of f(x,y)

$$F(0,0) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y),$$

F(0,0) adalah pada lokasi u = M/2 dan v = N/2

FIGURE 4.6
Result of filtering the image in Fig. 4.4(a) with a notch filter that set to 0 the $F(0, 0)$ term in the Fourier transform.

Unsharp Masking dan High-Boost Filtering dalam Ranah Frekuensi

- Unsharp Masking: $f_s(x,y) = f(x,y) - f_{lp}(x,y)$

$$F_s(u,v) = F(u,v) - F_{lp}(u,v) = \underbrace{(1 - H_{lp}(u,v))}_{H_{hp}(u,v)} F(u,v)$$

- High-Boost Filtering: $f_{hb}(x,y) = Af(x,y) - f_{lp}(x,y)$ $(A \geq 1)$

$$F_{hb}(u,v) = AF(u,v) - F_{lp}(u,v) =$$

$$= (A - H_{lp}(u,v))F(u,v) = (A - 1 + H_{hp}(u,v))F(u,v)$$