SEL 0449 - Processamento Digital de Imagens Médicas

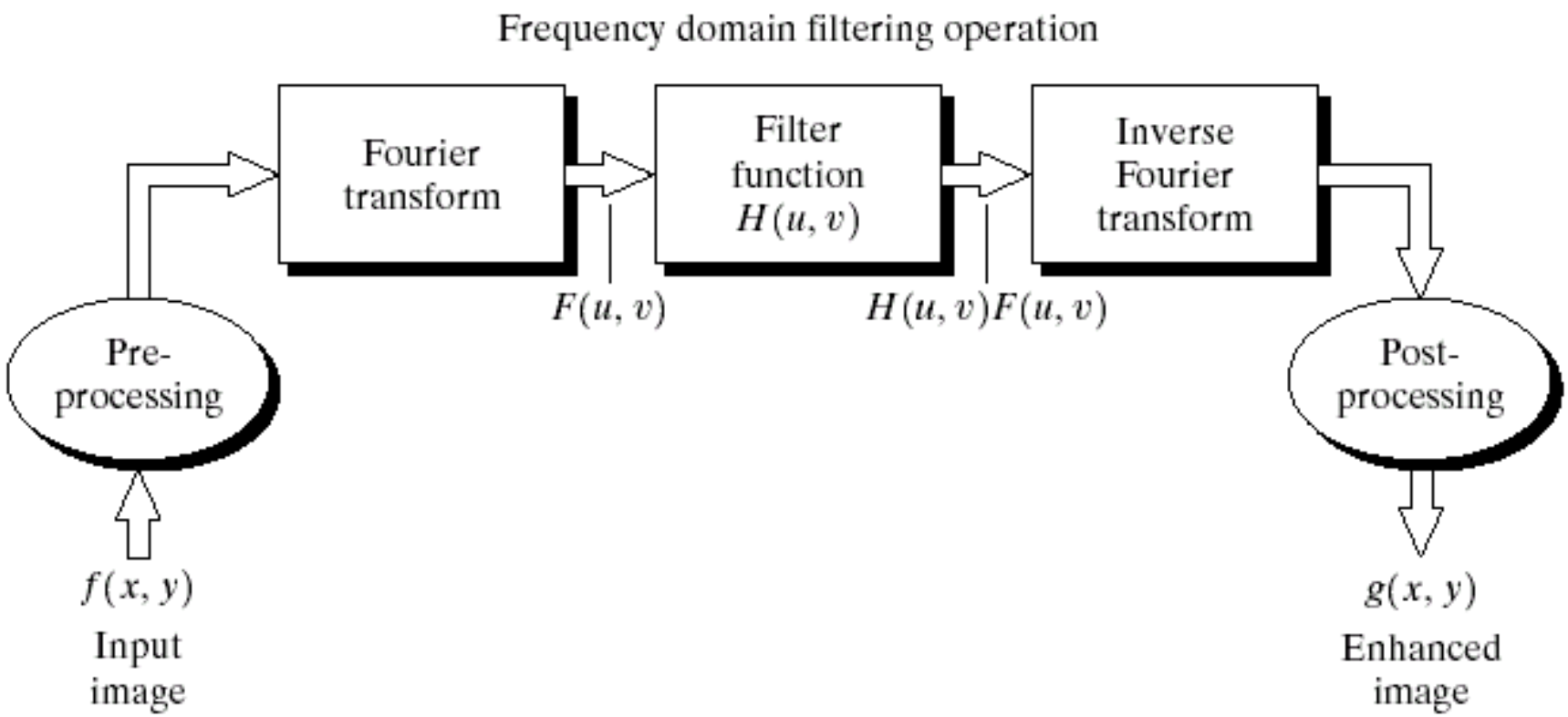
SEL 5895 – Introdução ao Processamento Digital de Imagens

Aula 7 – Processamento no Domínio da Frequência

Prof. Dr. Marcelo Andrade da Costa Vieira

mvieira@sc.usp.br

Processamento no Domínio da Frequência



Filtragem sem alteração a fase

- *Zero-Phase Shift Filters*

- O filtro *H(u,v)* deve multiplicar a matriz complexa *F(u,v)* para garantir que a fase não seja alterada no processo de filtragem:

$$F(u,v) = R(u,v) + jI(u,v) \qquad g(x,y) = \mathcal{F}^{-1}[H(u,v)F(u,v)]$$

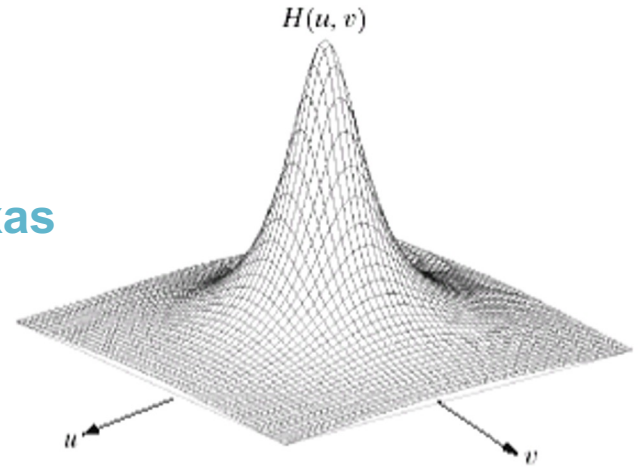
$$g(x,y) = \mathcal{F}^{-1}[H(u,v)R(u,v) + jH(u,v)I(u,v)]$$

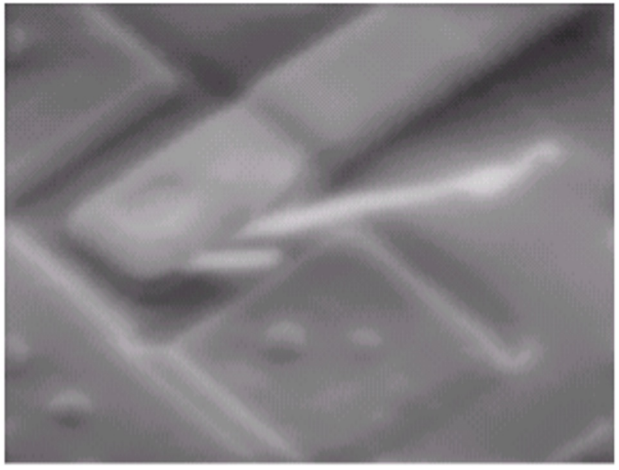
- Note que a fase não é alterada:

$$\varphi(u,v) = \tan^{-1}\left[\frac{H(u,v)I(u,v)}{H(u,v)R(u,v)}\right] = \tan^{-1}\left[\frac{I(u,v)}{R(u,v)}\right]$$

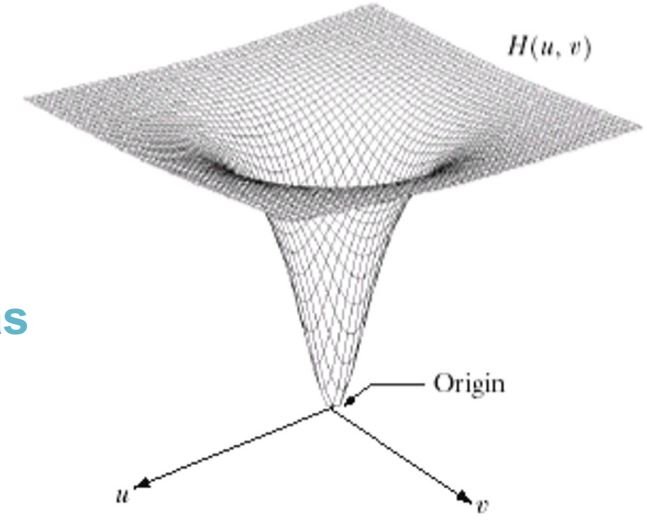
Filtros no Domínio da Frequência

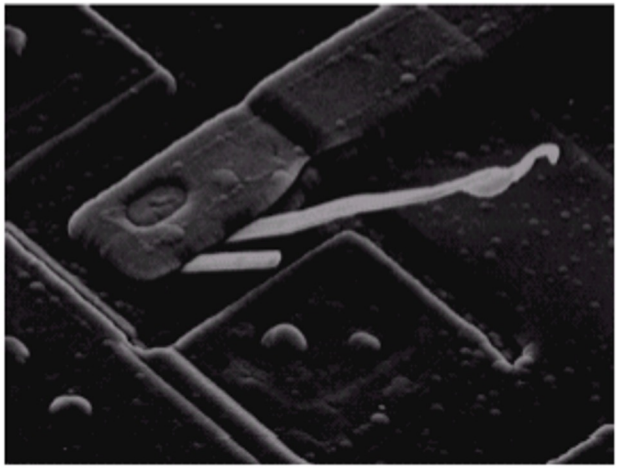
Passa-Baixas



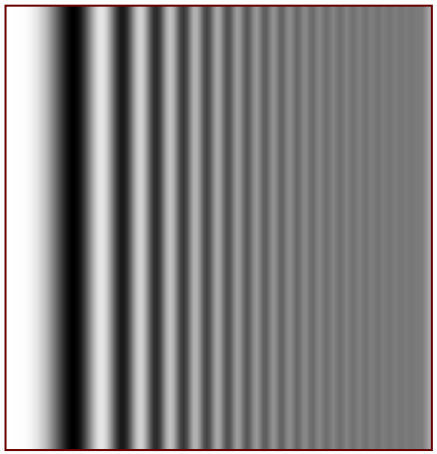


Passa-Altas

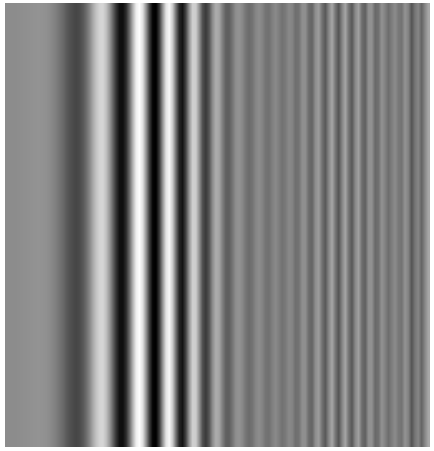




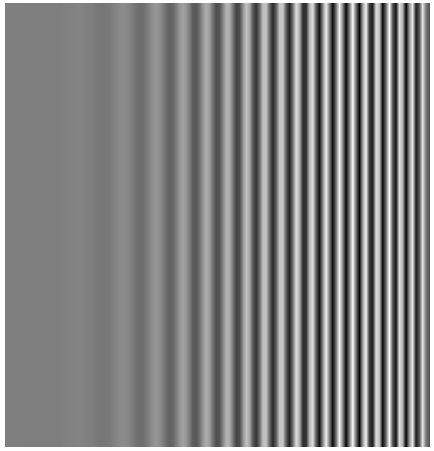
Efeito dos filtros



Filtro
passa-baixa

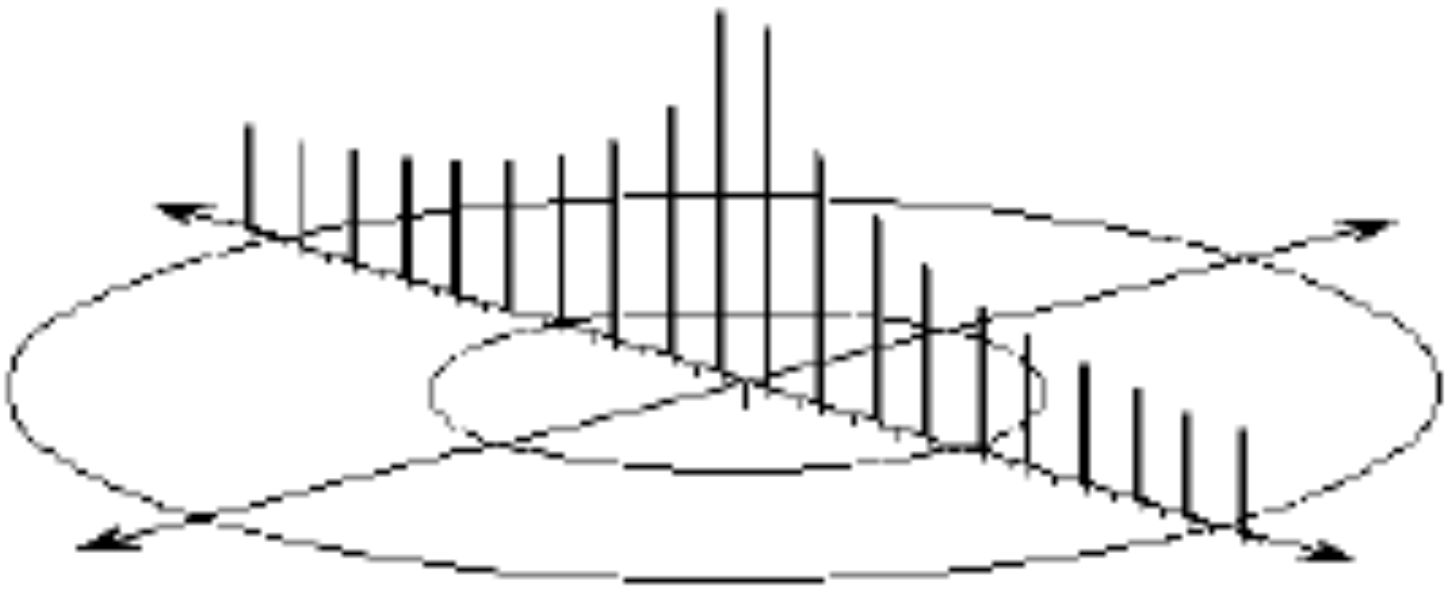


Filtro
passa-banda



Filtro
passa-alta

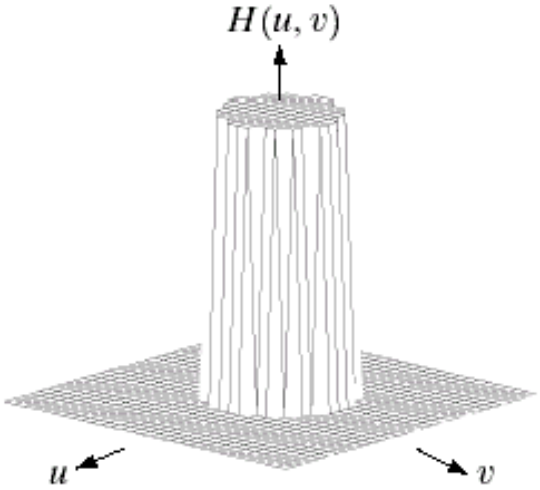
Os Filtros devem ser circulares e concêntricos

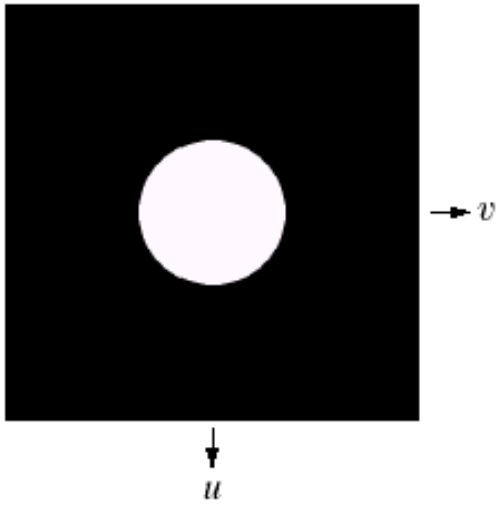


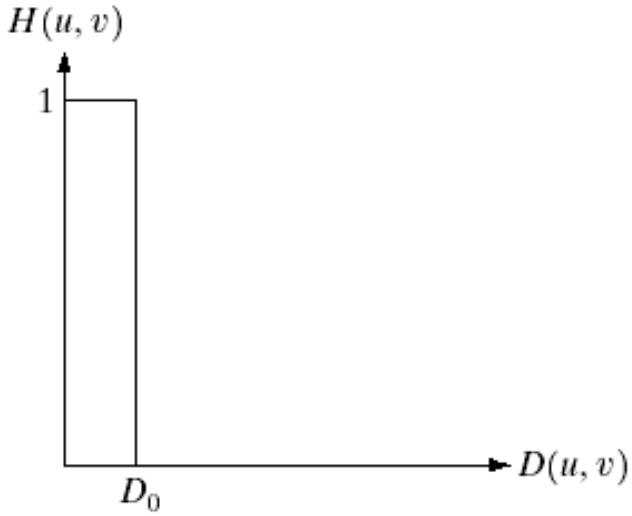
Filtros Passa-Baixa

- Retira (ou atenua) as ondas senoidais de alta-frequência espacial, ou seja, que estão acima da frequência de corte (D_0) definida na construção do filtro;
- Mantém apenas as ondas senoidais de baixa-frequência espacial, ou seja, que estão abaixo da frequência de corte (D_0);
- Não há aumento na amplitude de nenhuma onda senoidal do espectro de Fourier da imagem;
- Podem ser de vários tipos. Os mais comuns são: Ideal, Butterworth e Gaussiano.

Filtro Passa-Baixa Ideal





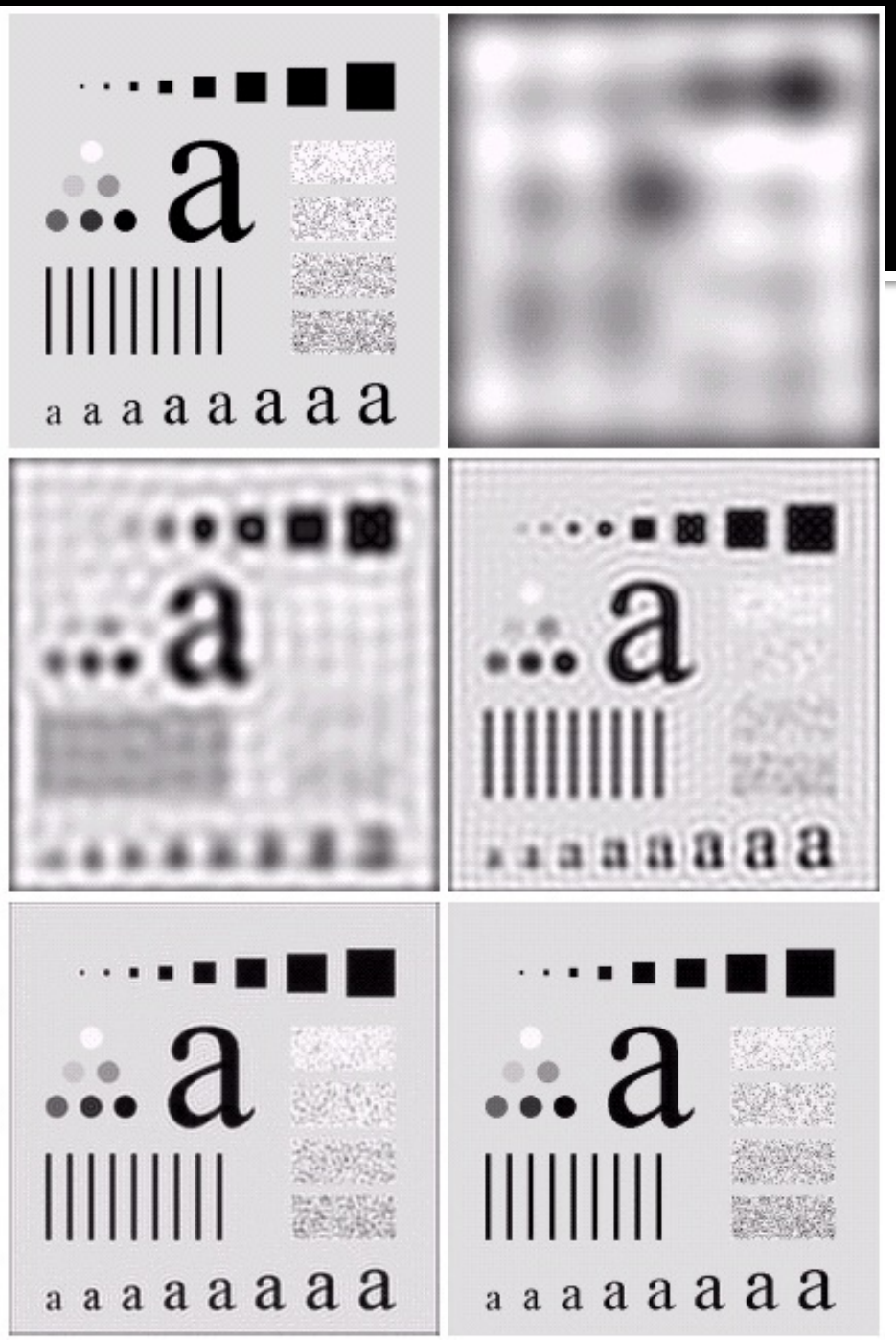


$$H(u,v) = \begin{cases} 1 \text{ , se } D(u,v) \leq D_0 \\ 0 \text{ , se } D(u,v) > D_0 \end{cases}$$

- Todas as ondas senoidais de frequência acima da frequência de corte (D_0) são retiradas da imagem;
- As ondas de frequências mais baixas que D_0 não são alteradas.

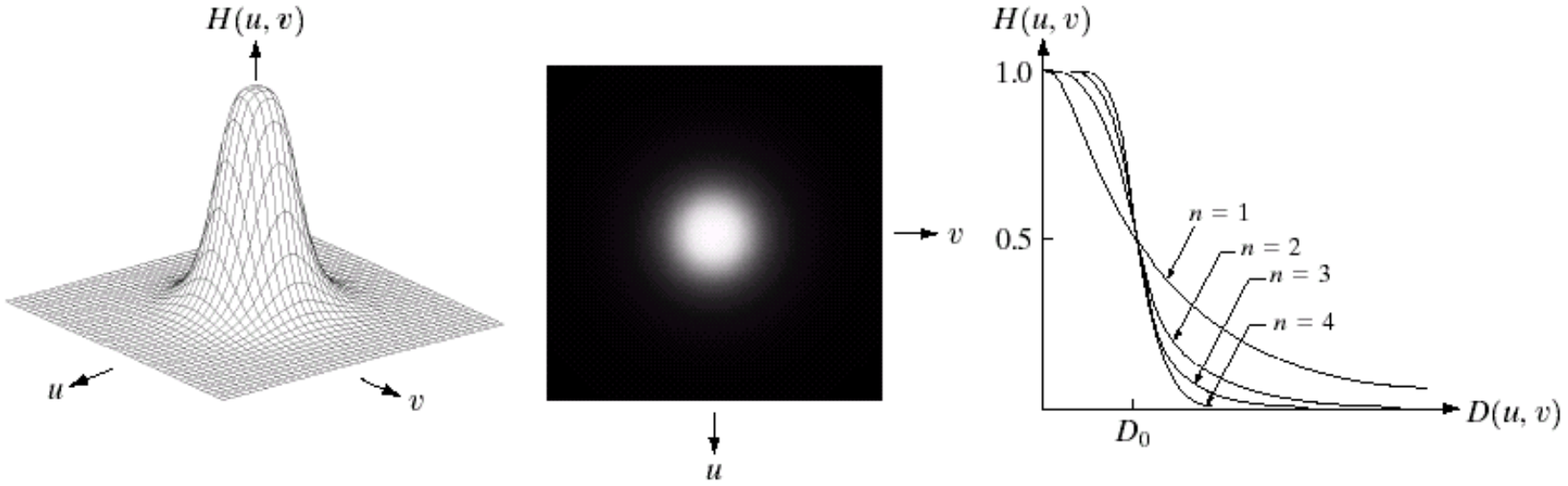
Filtro Passa-Baixas **Ideal**:

Resultados obtidos à medida que se aumenta a frequência de corte



Efeito indesejado de *ringing*

Filtro Passa-Baixa **Butterworth**

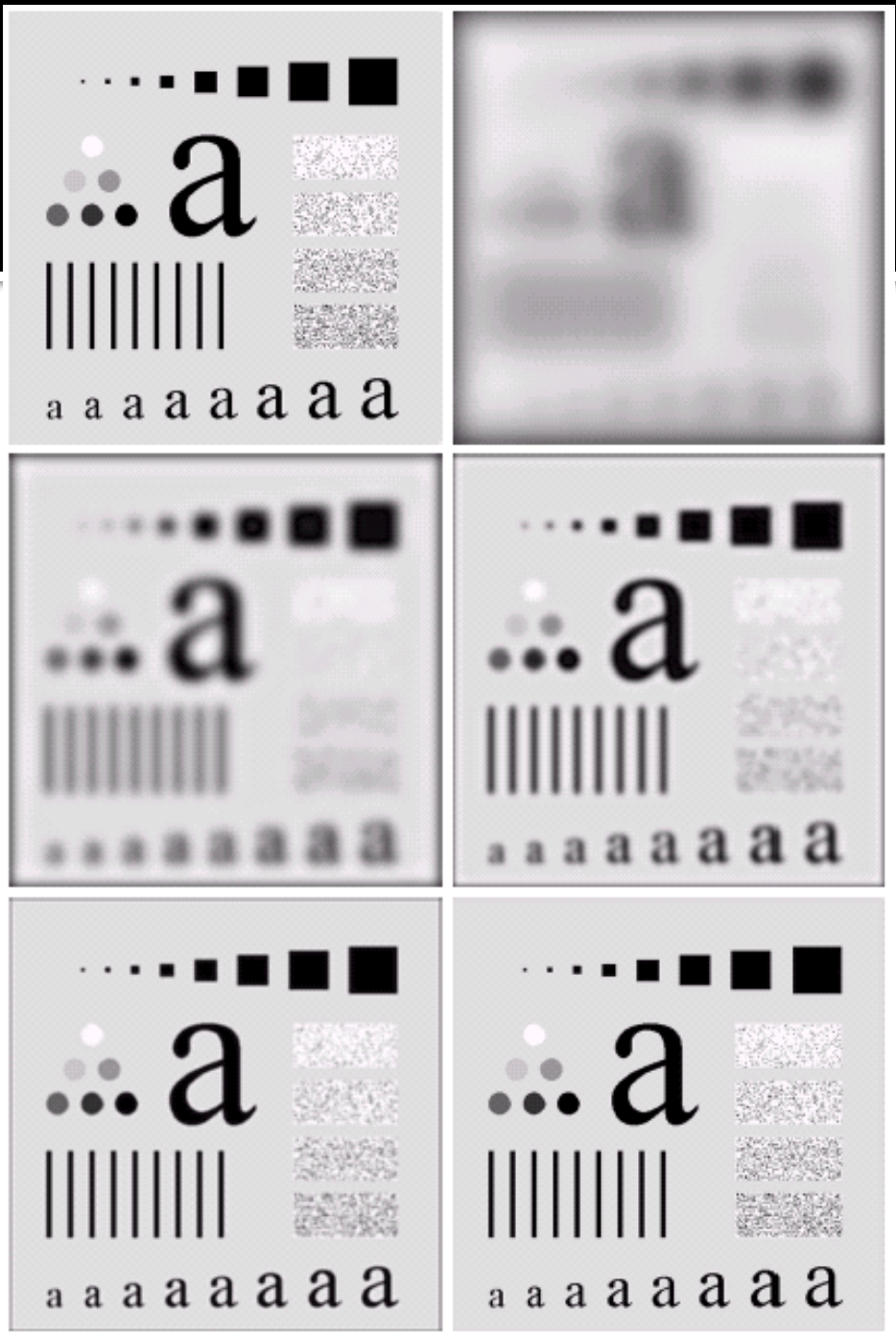


$$H(u,v) = \frac{1}{1+\left[\frac{D(u,v)}{D_0}\right]^{2n}}$$

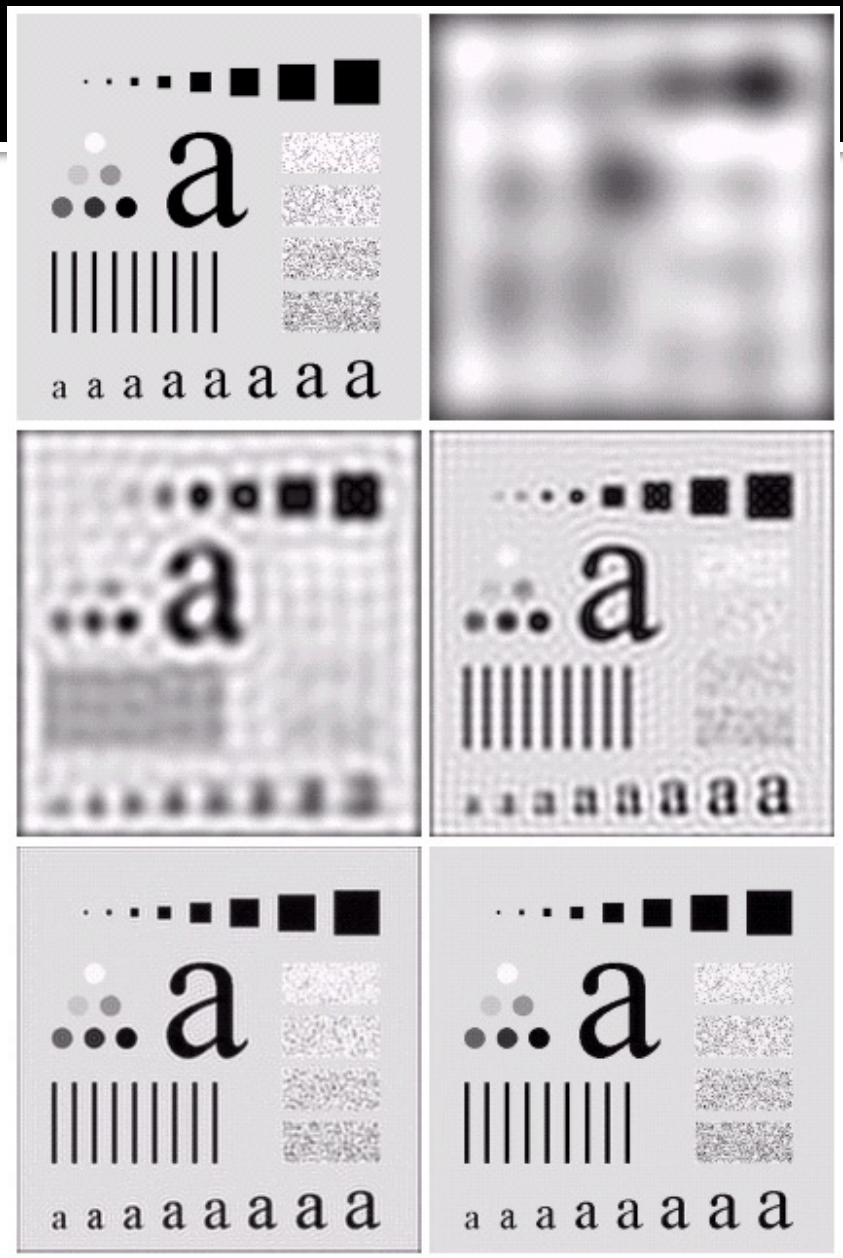
- A frequência de corte (D_0) define o valor onde a amplitude da onda é reduzida em 50%;
- As ondas de alta-frequência são cada vez mais atenuadas na imagem a medida que são maiores que D_0, ou seja, o filtro possui transição mais suave que o filtro ideal;
- O valor de ***n*** (ordem do filtro) determina a "suavidade" do filtro.

Filtro Passa-Baixa **Butterworth** (n=2)

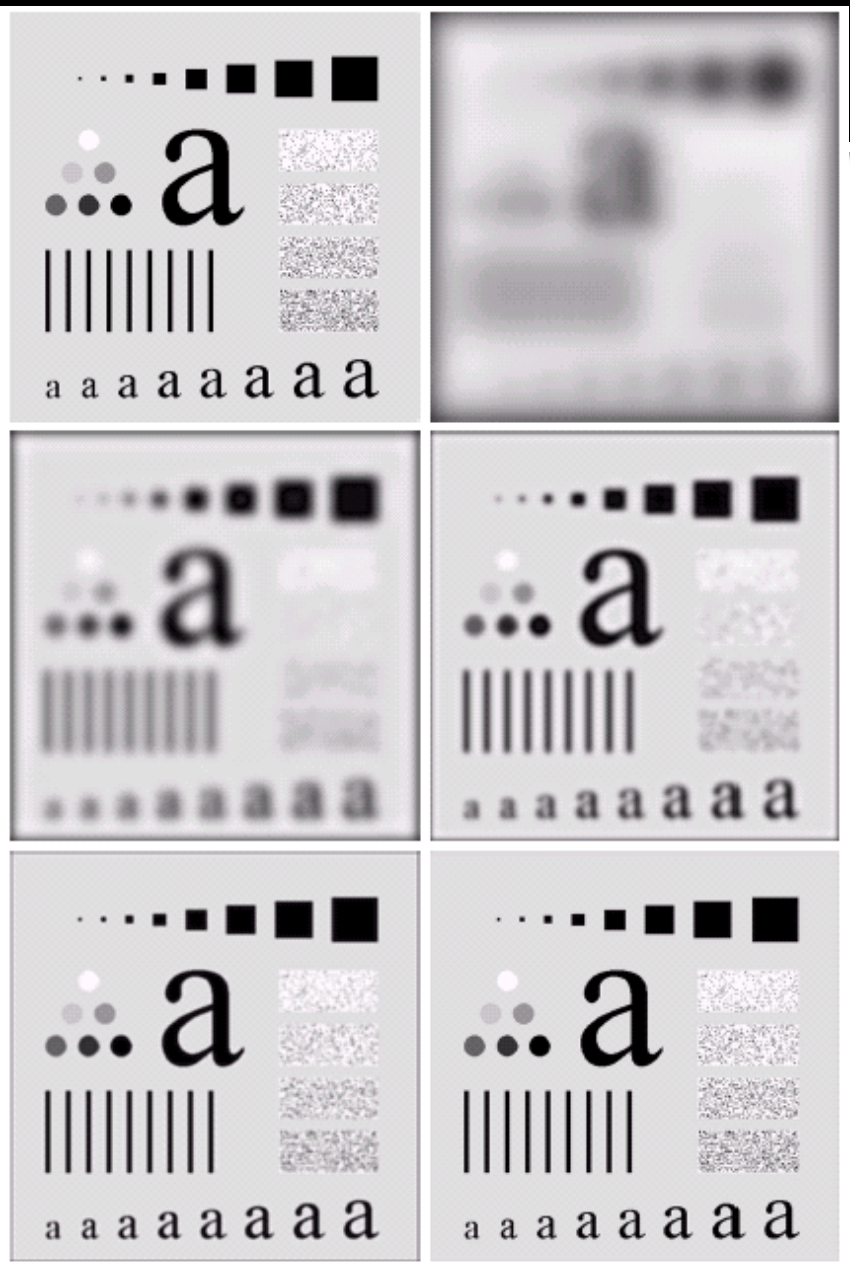
Resultados obtidos à medida que se aumenta a frequência de corte



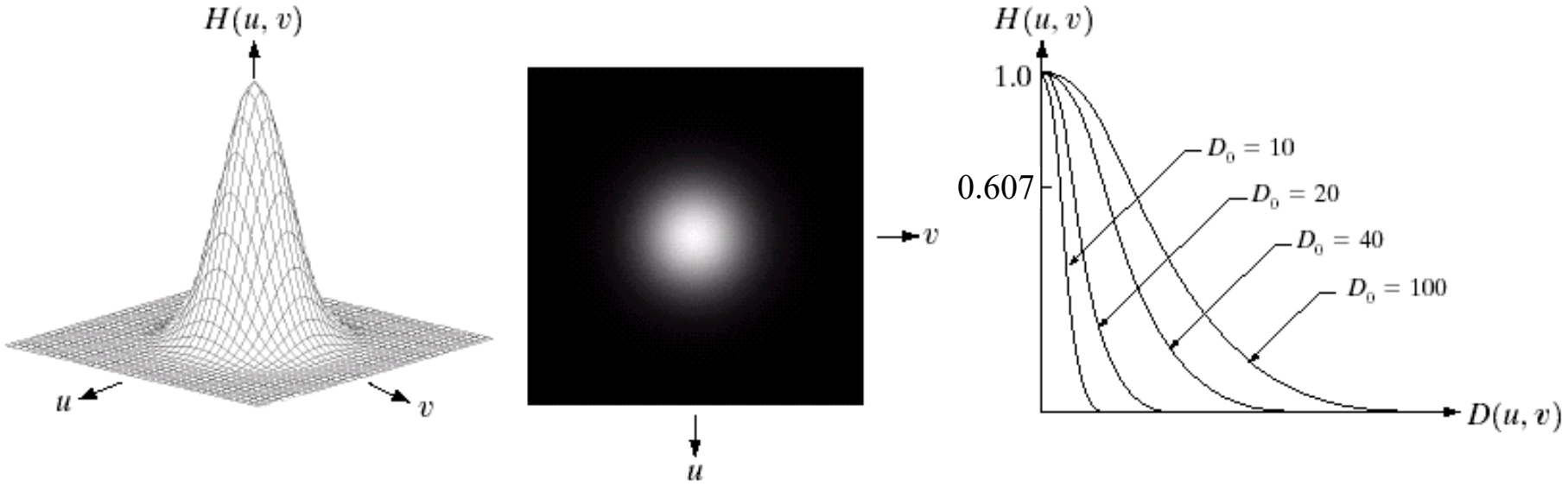
Filtro PB Ideal



Filtro PB Butterworth



Filtro Passa-Baixa **Gaussiano**

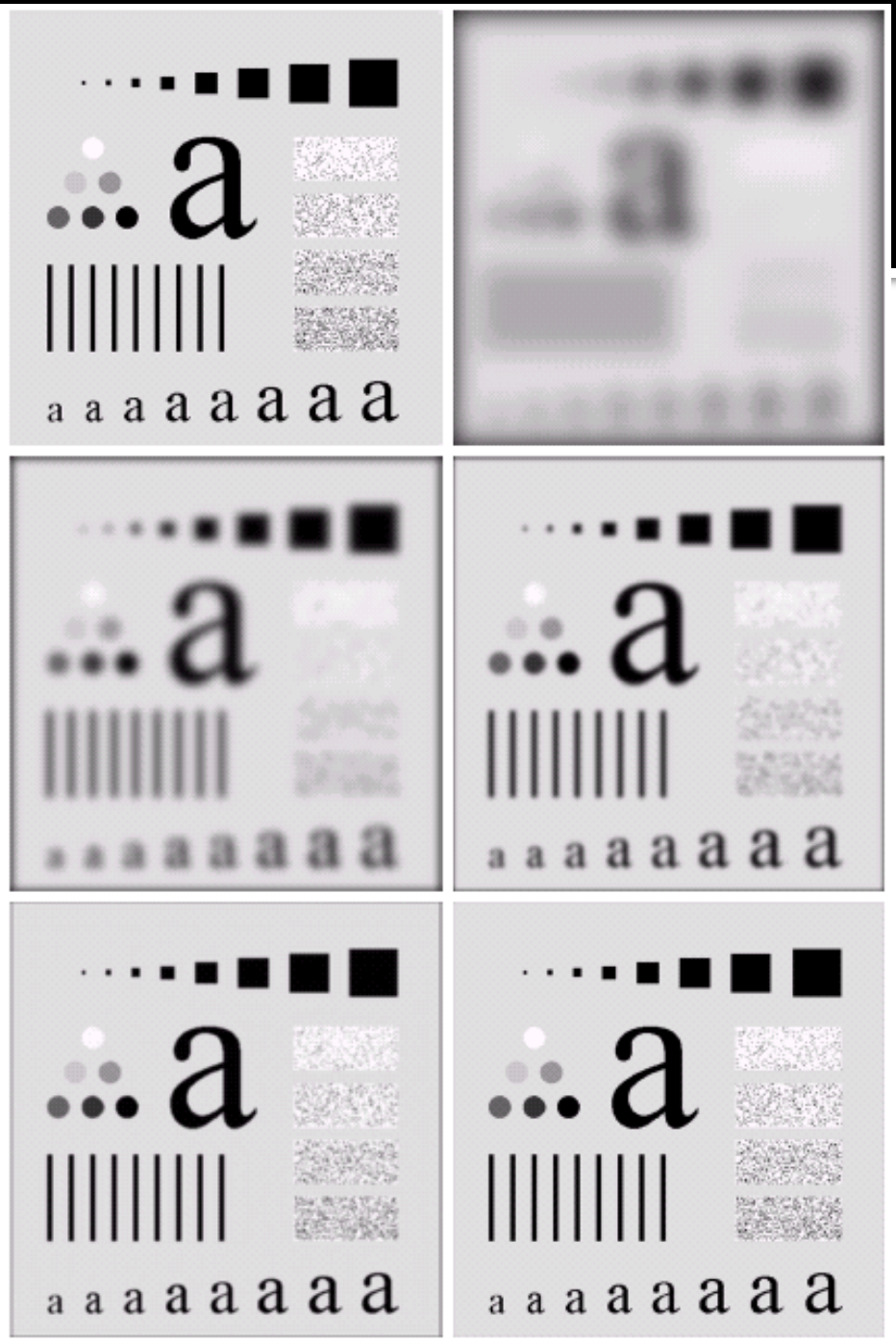


$$H(u,v) = e^{\frac{-[D(u,v)^2]}{2D_0^2}}$$

- A frequência de corte (D_0) define o valor onde a amplitude da onda é reduzida em 60,7%;
- Ondas de alta-frequência são cada vez mais atenuadas na imagem a medida que são maiores que D_0, ou seja, o filtro possui transição mais suave que o filtro ideal;
- O filtro Gaussiano pode ser bem mais suave que o filtro Butterworth.

Filtro Passa-Baixa **Gaussiano**

Resultados obtidos à medida que se aumenta a frequência de corte



Filtro PB Butterworth (n=2)

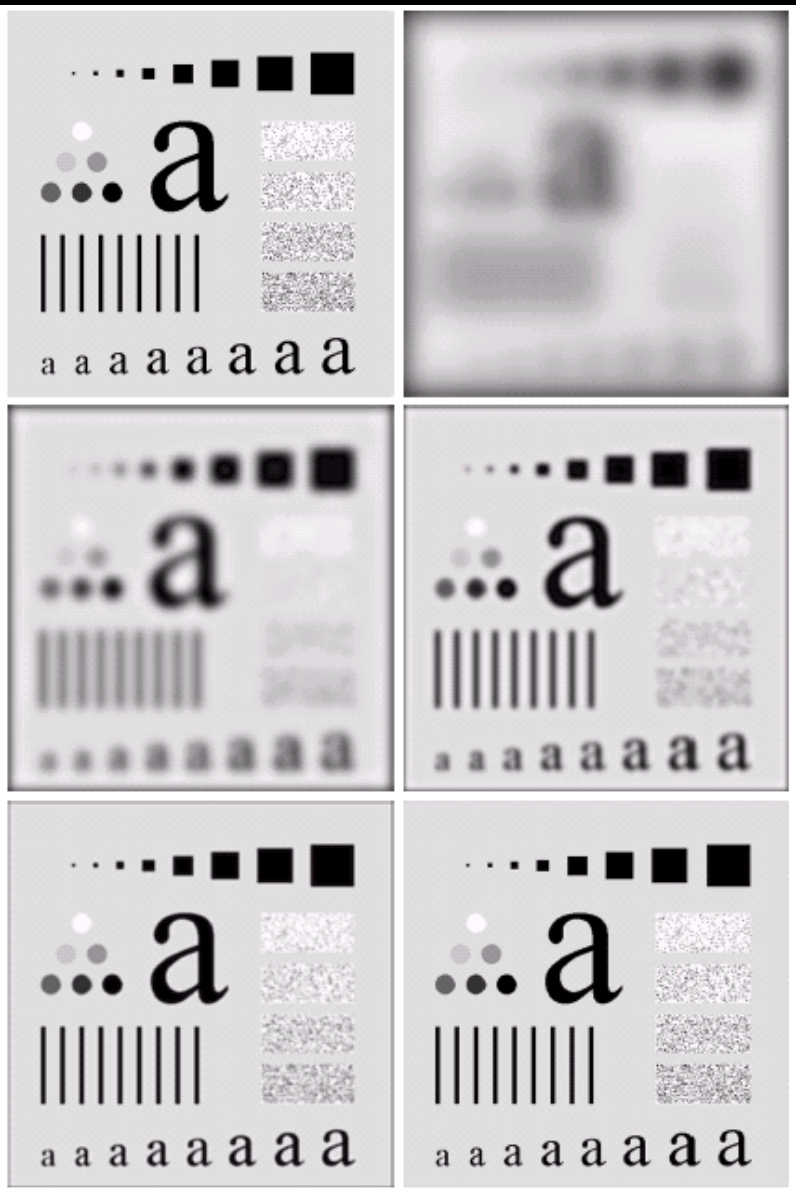


FIGURE 4.15 (a) Original image. (b)–(f) Results of filtering with BLPFs of order 2, with cutoff frequencies at radii of 5, 15, 30, 80, and 230, as shown in Fig. 4.11(b). Compare with Fig. 4.12.

Filtro PB Gaussiano

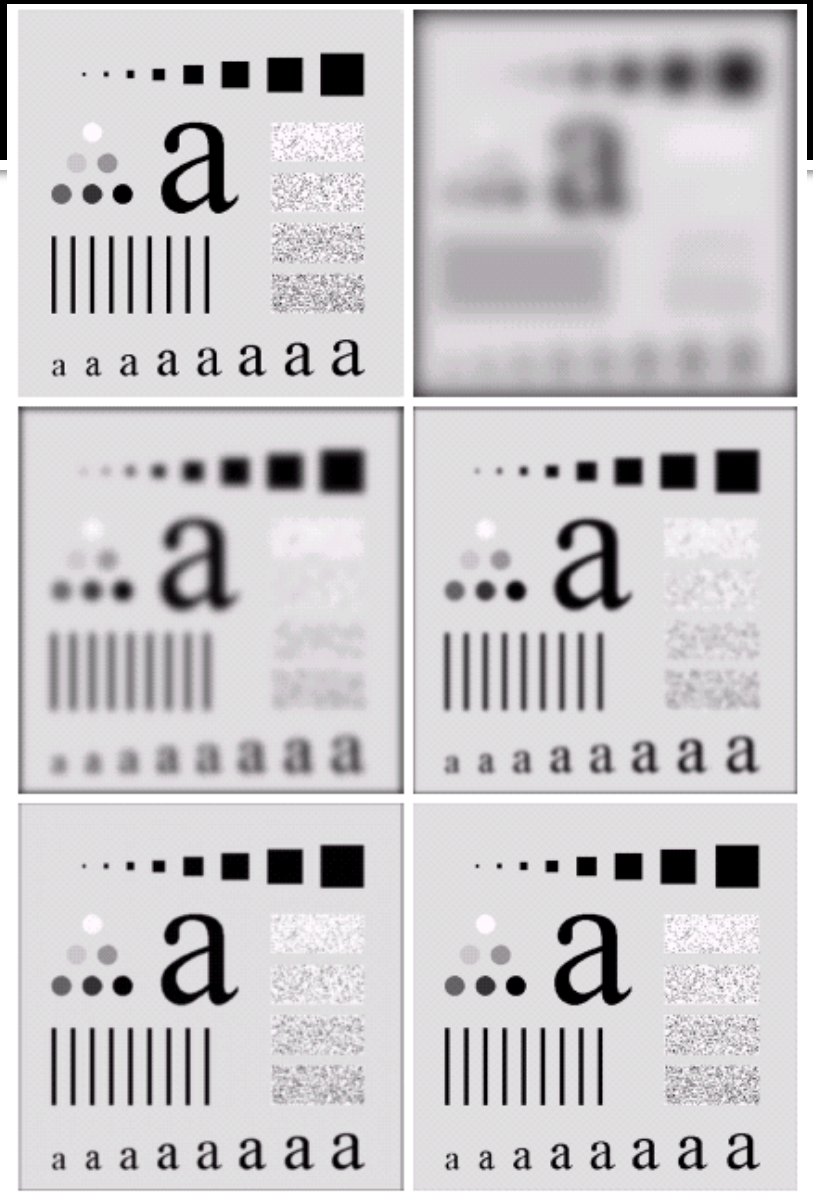
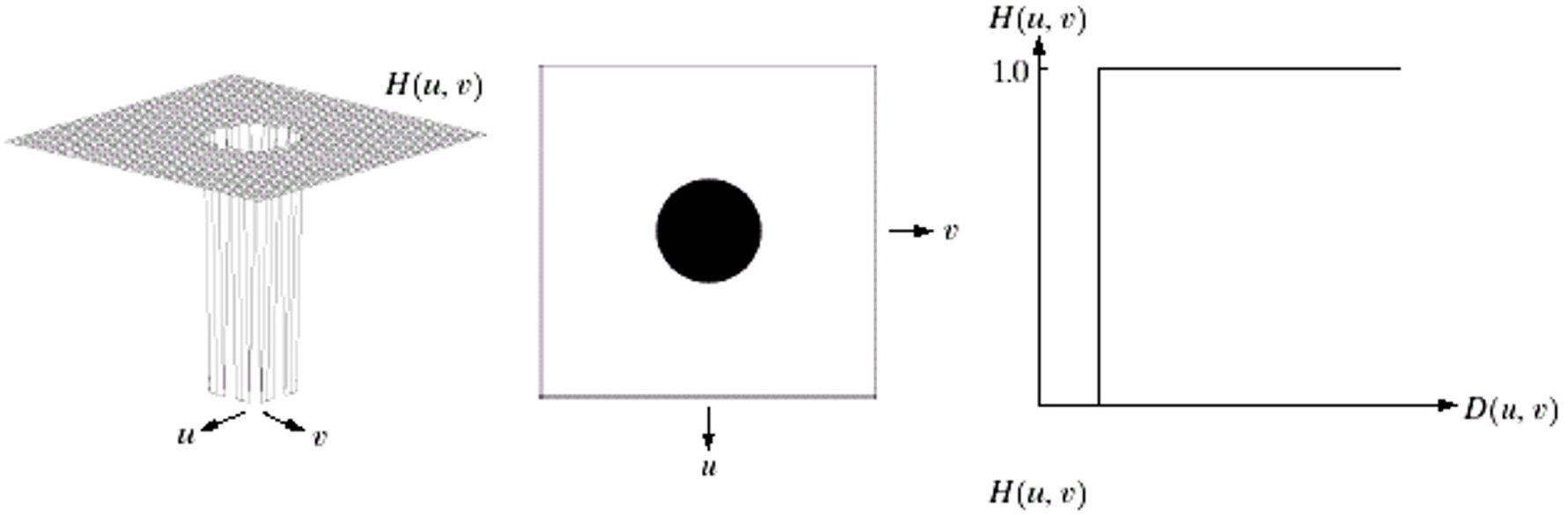


FIGURE 4.18 (a) Original image. (b)–(f) Results of filtering with Gaussian lowpass filters with cutoff frequencies set at radii values of 5, 15, 30, 80, and 230, as shown in Fig. 4.11(b). Compare with Figs. 4.12 and 4.15.

Filtros Passa-Alta

- Retira (ou atenua) as ondas senoidais de baixa-frequência espacial, ou seja, que estão abaixo da frequência de corte (D_0) definida na construção do filtro;
- Mantém apenas as ondas senoidais de alta-frequência espacial, ou seja, que estão acima da frequência de corte (D_0);
- Não há aumento na amplitude de nenhuma onda senoidal do espectro de Fourier da imagem;
- Podem ser de vários tipos. Os mais comuns são: Ideal, Butterworth e Gaussiano.

Filtros Passa-Alta Ideal

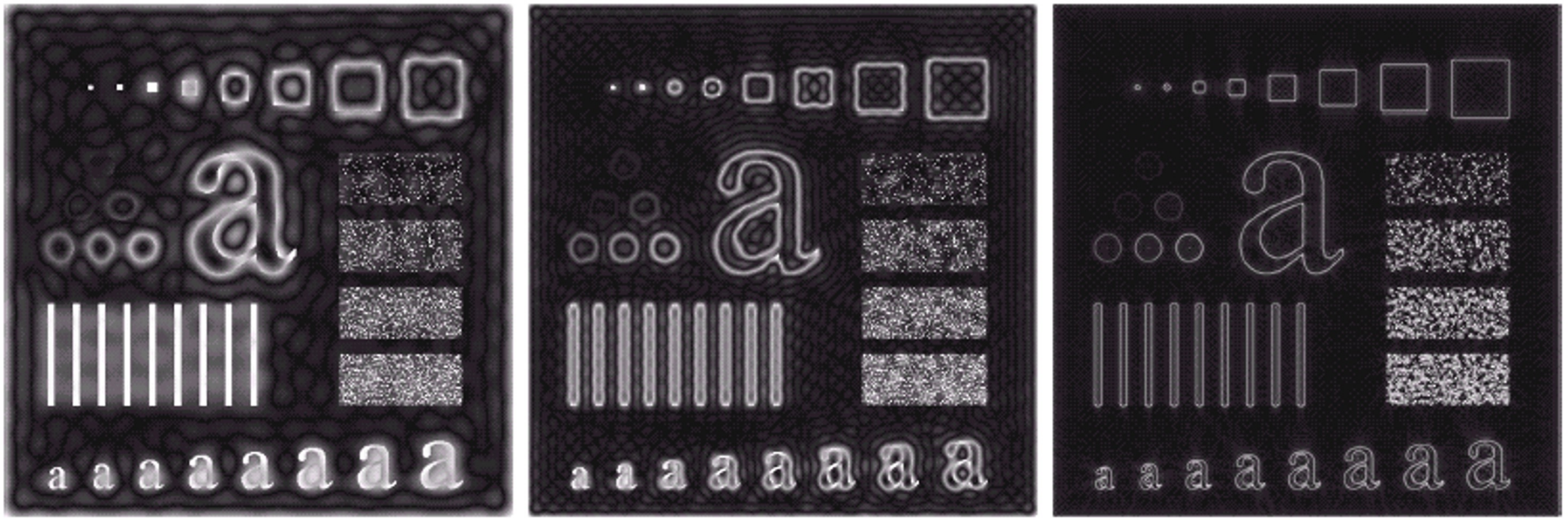


$$H(u,v) = \begin{cases} 0 \text{ , se } D(u,v) \leq D_0 \\ 1 \text{ , se } D(u,v) > D_0 \end{cases}$$

- Todas as ondas senoidais de frequência abaixo da frequência de corte (D_0) são retiradas da imagem;
- As ondas de frequência mais altas que D_0 não são alteradas.

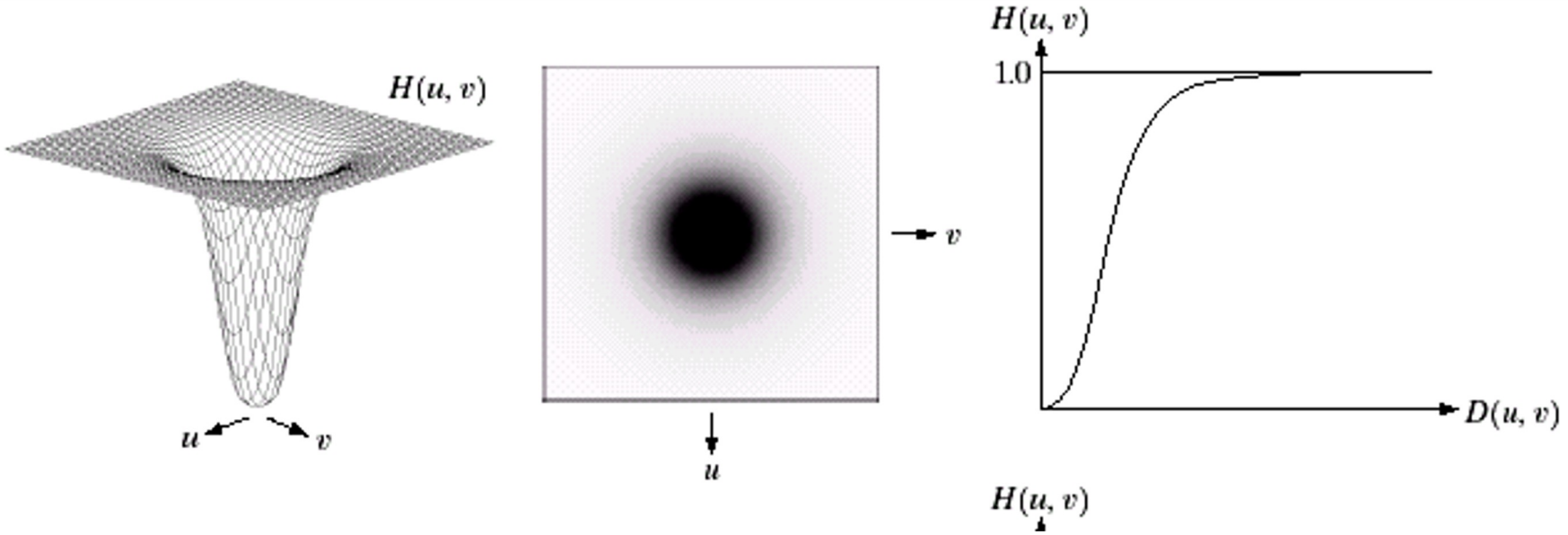
Filtro Passa-Alta **Ideal**

Resultados obtidos à medida que se aumenta a frequência de corte



Efeito indesejado de *ringing*

Filtro Passa-Alta **Butterworth**

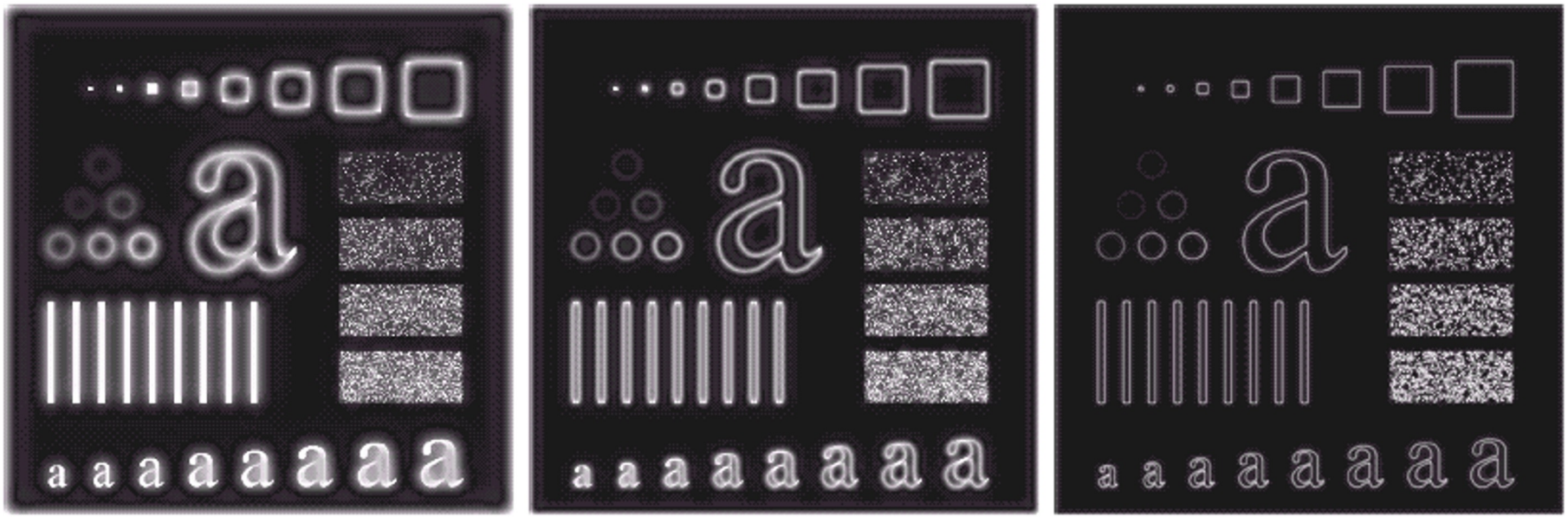


$$H(u,v) = \frac{1}{1 + \left[\frac{D_0}{D(u,v)}\right]^{2n}}$$

- A frequência de corte (D_0) define o valor onde a amplitude da onda senoidal é reduzida em 50%;
- Ondas de baixa-frequência são cada vez mais atenuadas na imagem a medida que são menores que D_0, ou seja, o filtro possui transição mais suave que o filtro ideal;
- O valor de ***n*** (ordem do filtro) determina a "suavidade" do filtro.

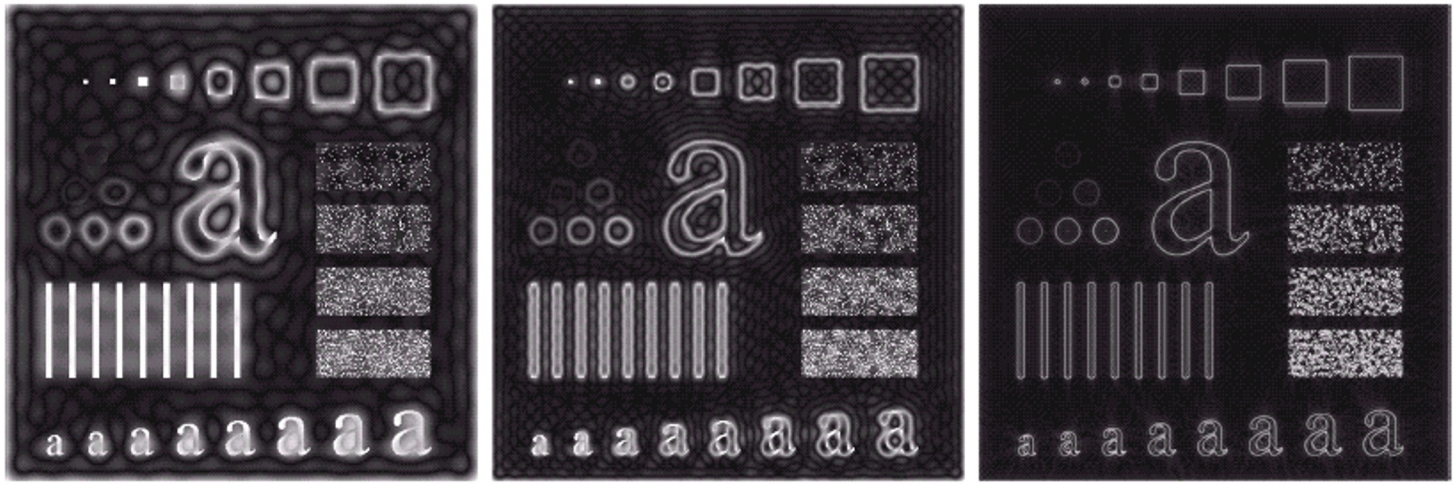
Filtro Passa-Alta **Butterworth** (n=2)

Resultados obtidos à medida que se aumenta a frequência de corte

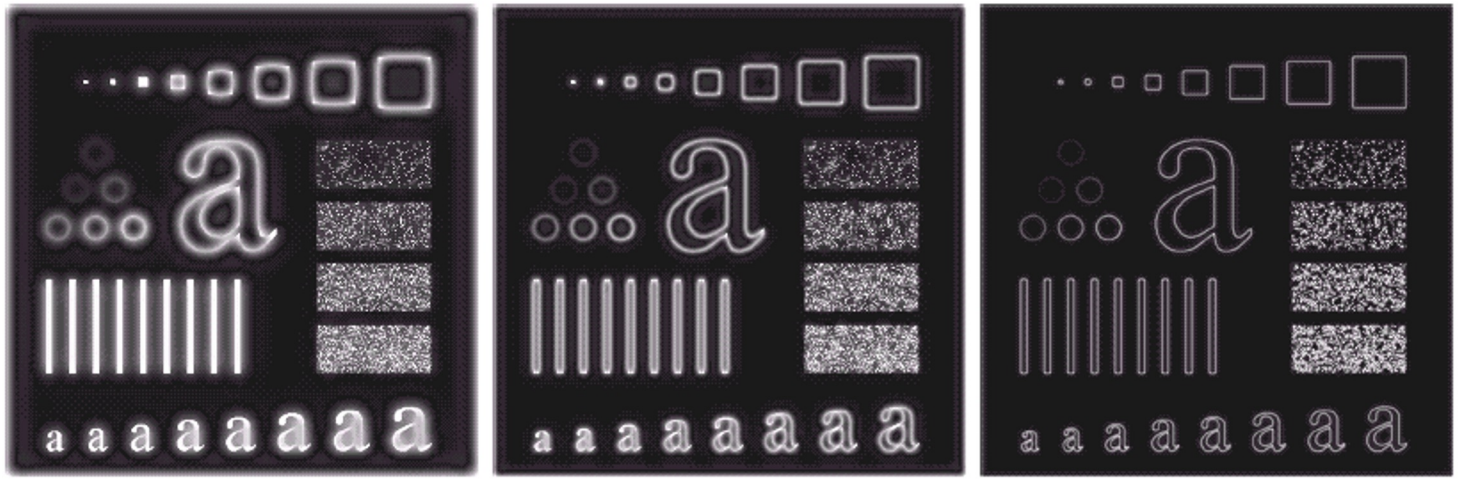


Filtros Passa-Alta

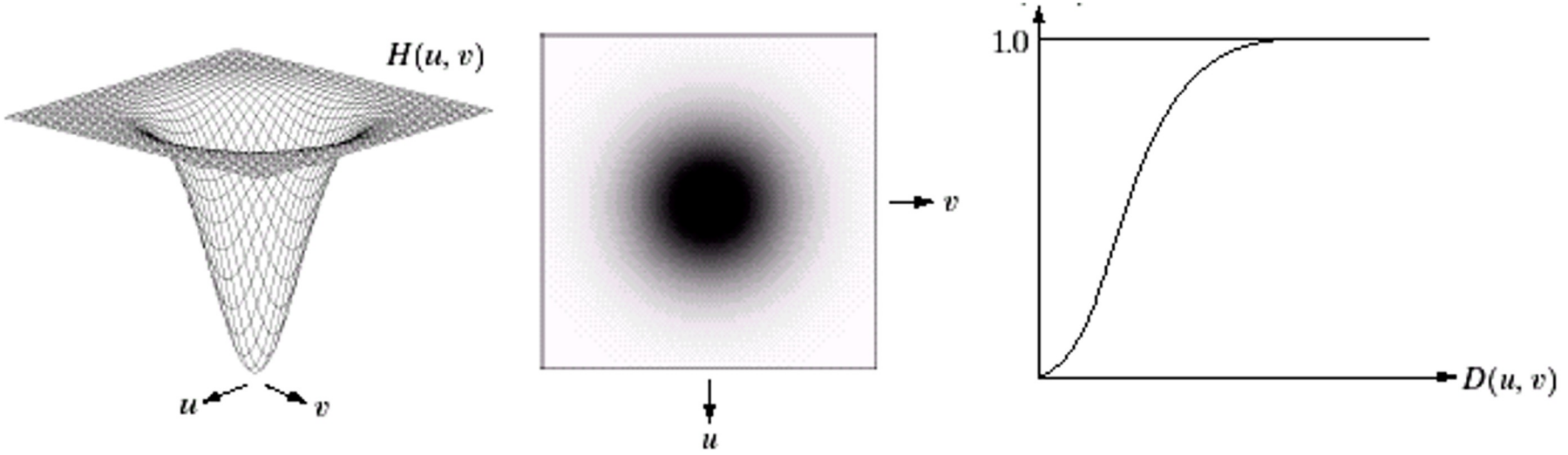
Ideal



Butterworth



Filtro Passa-Alta **Gaussiano**

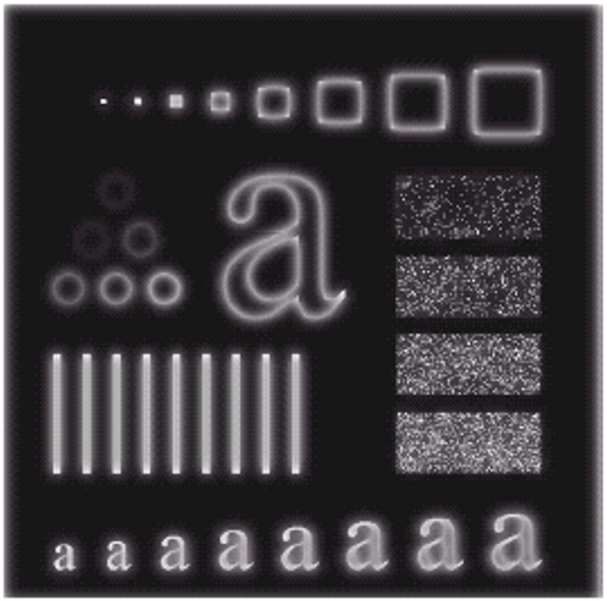


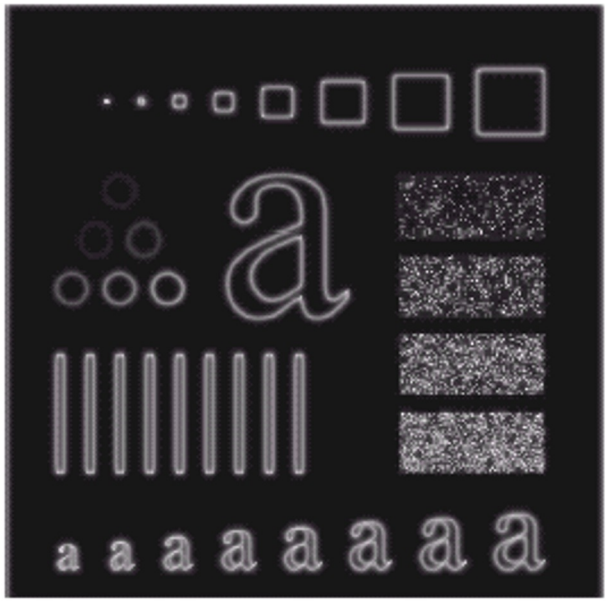
$$H(u,v) = 1 - e^{\frac{-[D(u,v)^2]}{2D_0{}^2}}$$

- A frequência de corte (D_0) define o valor onde a amplitude da onda senoindal é reduzida em 60,7%;
- Ondas de baixa-frequência são cada vez mais atenuadas na imagem a medida que são menores que D_0, ou seja, o filtro possui transição mais suave que o filtro ideal;
- O filtro Gaussiano pode ser bem mais suave que o filtro Butterworth.

Filtro Passa-Alta **Gaussiano**

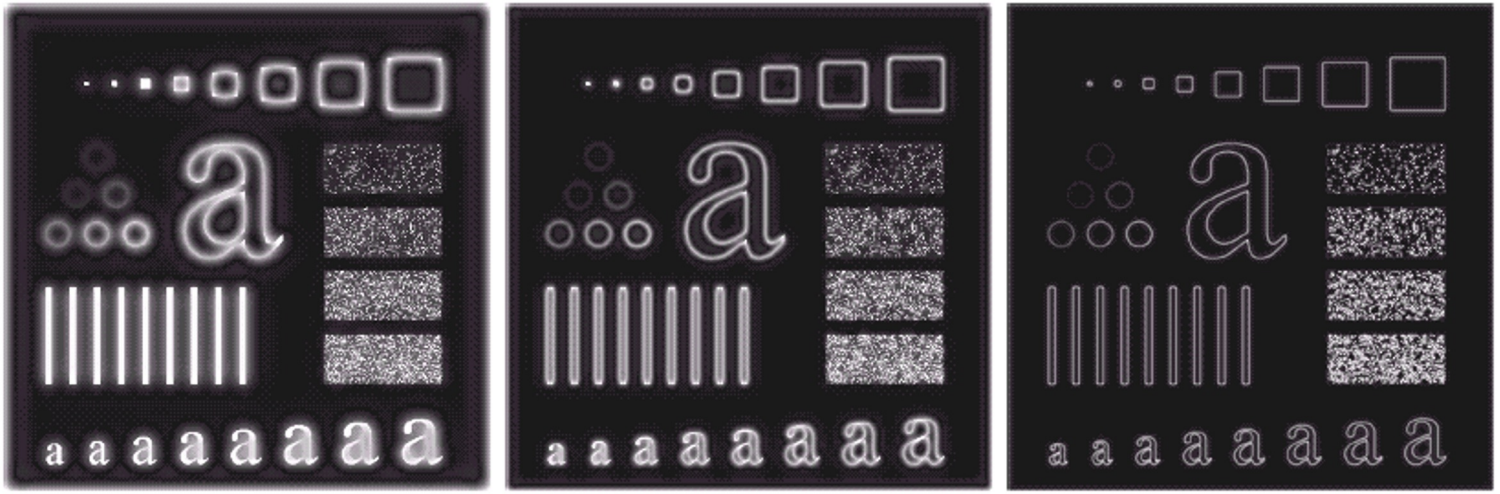
Resultados obtidos à medida que se aumenta a frequência de corte





Filtros Passa-Alta

Butterworth →



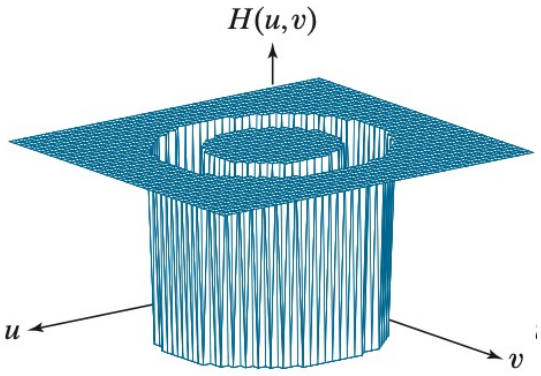
Gaussiano →

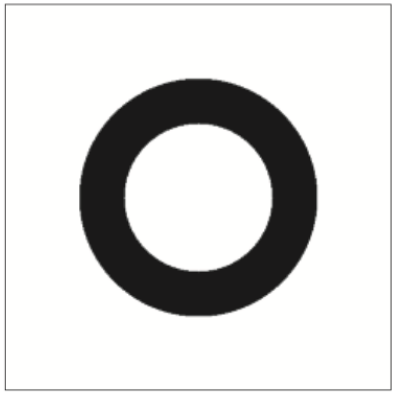
Filtros Rejeita-Banda e Passa-Banda

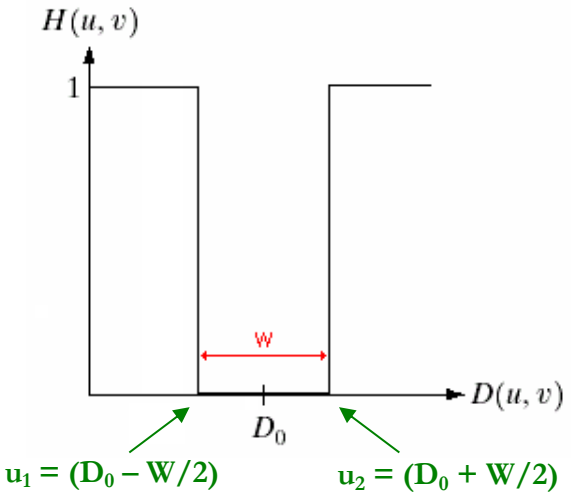
Filtros Rejeita-Banda

- Retira (ou atenua) as ondas senoidais cujas frequências espaciais estão dentro de uma faixa (banda) definida na construção do filtro;
- Mantém apenas as ondas senoidais cujas frequências espaciais estão fora da banda definida;
- Não há aumento na amplitude de nenhuma onda senoidal do espectro de Fourier da imagem;
- São feitos a partir da combinação de filtros passa-baixa e passa-alta;
- Podem ser de vários tipos. Os mais comuns são: Ideal, Butterworth e Gaussiano.

Filtro Rejeita-Banda **Ideal**



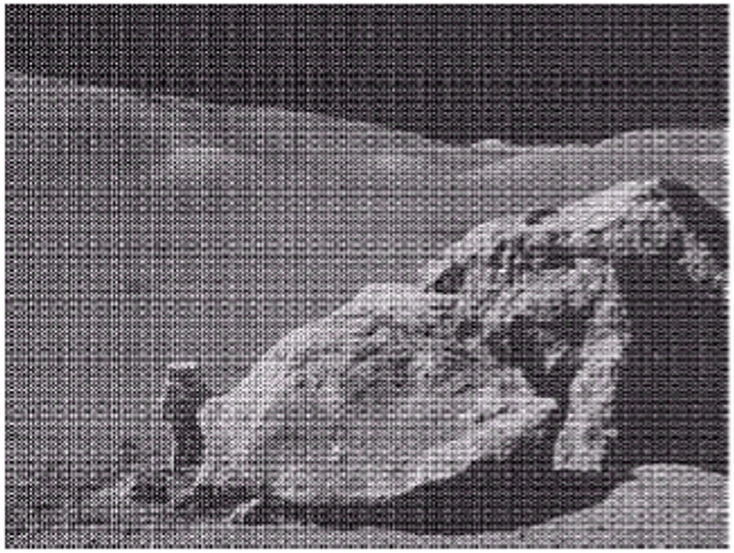


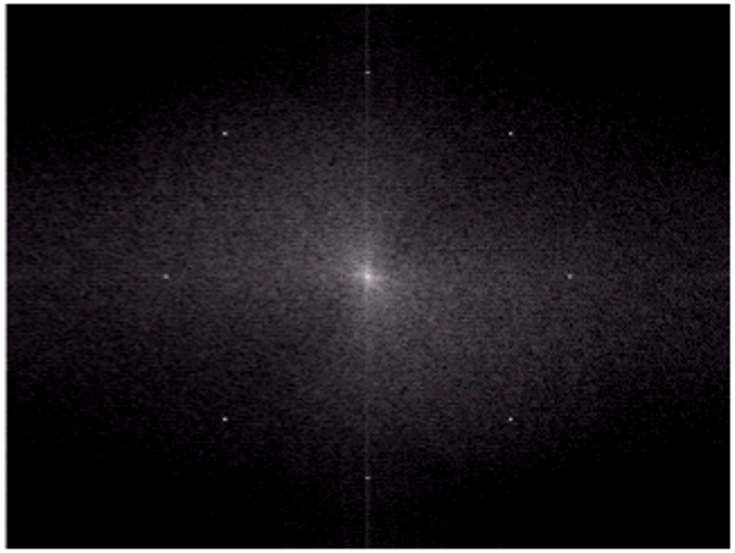


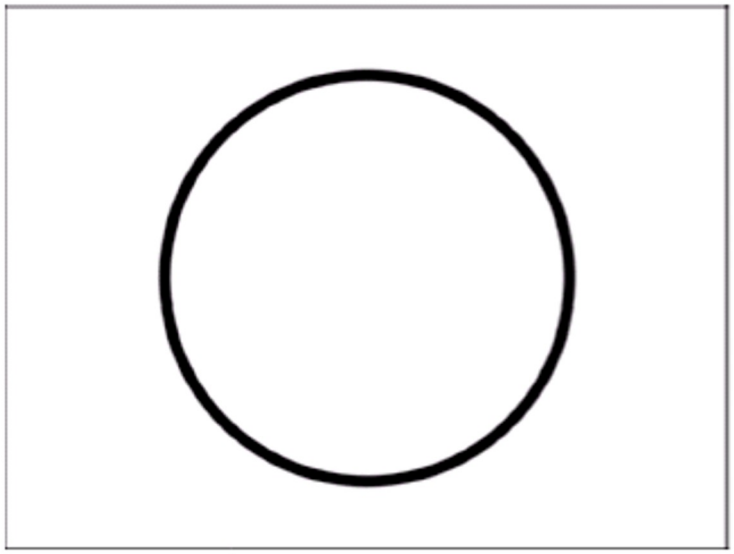
$$H(u,v) = \begin{cases} 1 \text{ , se } D(u,v) < (D_0 - W/2) \\ 0 \text{ , se } (D_0 - W/2) \leq D(u,v) \leq (D_0 + W/2) \\ 1 \text{ , se } D(u,v) > (D_0 + W/2) \end{cases}$$

- As ondas senoidais cuja frequência espacial pertence à faixa definida por W (banda) são retiradas da imagem. As ondas cujas frequências são externas à banda W não são alteradas;
- D_0 corresponde ao centro da banda W, onde o valor do filtro deve ser zero;
- u_1 e u_2 são as frequências de corte do filtro.

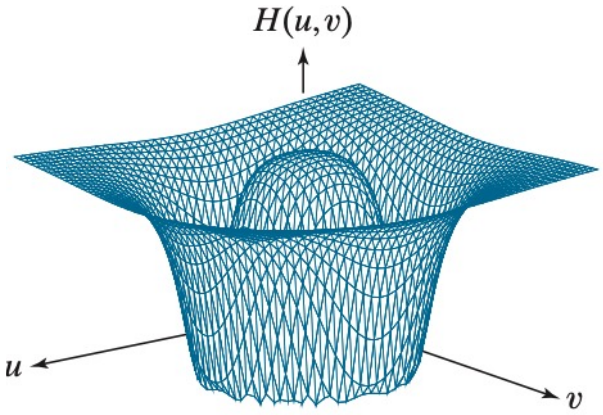
Filtro Rejeita-Banda **Ideal**

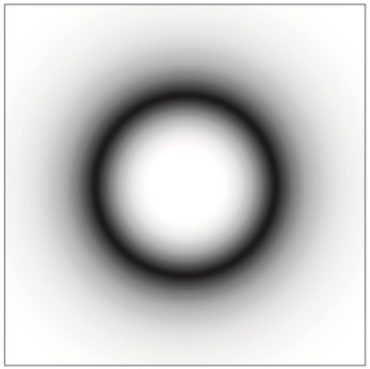


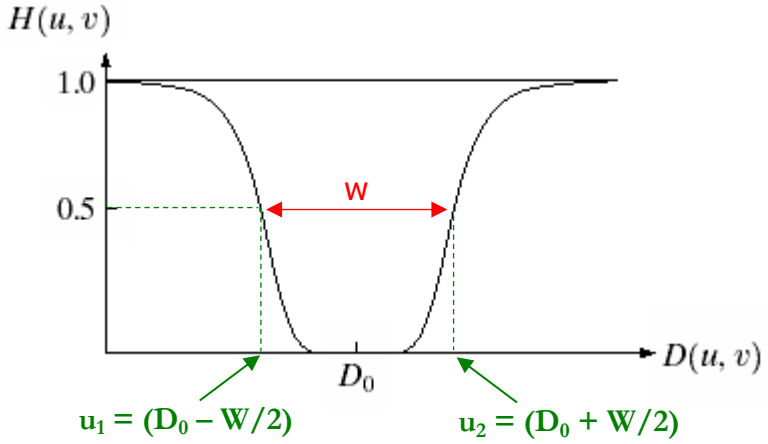




Filtro Rejeita-Banda **Butterworth**



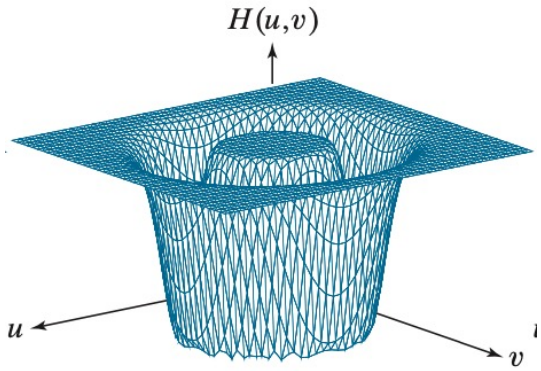


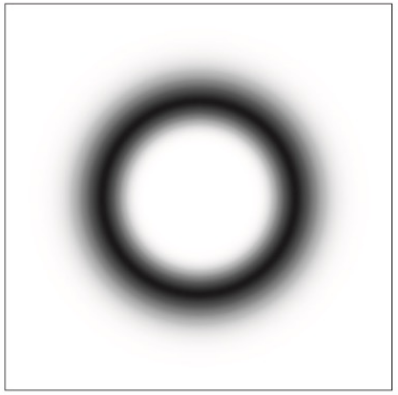


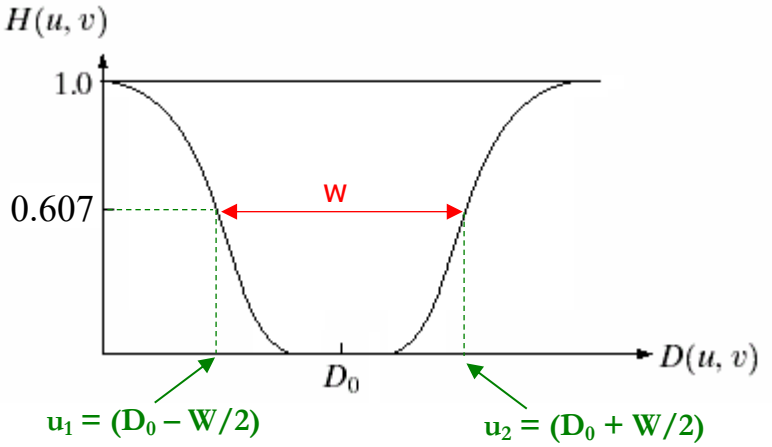
$$H(u,v) = \frac{1}{1+\left[\dfrac{D(u,v)\cdot W}{D(u,v)^2 - D_0{}^2}\right]^{2n}}$$

- O centro da banda (D_0) define o valor onde a amplitude do filtro é zero;
- As frequências de corte u_1 e u_2 definem os valores onde a amplitude da onda senoidal é reduzida em 50%;
- As ondas senoidais de frequência espacial dentro da faixa definida são cada vez mais atenuadas na imagem a medida que se aproximam de D_0, ou seja, o filtro possui transição mais suave que o filtro ideal;
- O valor de ***n*** (ordem do filtro) determina a "suavidade" do filtro.

Filtro Rejeita-Banda **Gaussiano**







$$H(u,v) = 1 - e^{-\left[\frac{D(u,v)^2 - D_0^2}{D(u,v)\cdot W}\right]^2}$$

- O centro da banda (D_0) define o valor onde a amplitude do filtro é zero;
- As frequências de corte u_1 e u_2 definem os valores onde a amplitude da onda senoidal é reduzida em 60,7%;
- As ondas senoidais de frequência espacial dentro da faixa definida são cada vez mais atenuadas na imagem a medida que se aproximam de D_0, ou seja, o filtro possui transição mais suave que o filtro ideal;
- O filtro Gaussiano pode ser bem mais suave que o filtro Butterworth.

Filtros Passa-Banda

- Retira (ou atenua) as ondas senoidais cujas frequências espaciais estão fora de uma faixa (banda) definida na construção do filtro;
- Mantém apenas as ondas senoidais cujas frequências espaciais estão dentro da banda definida;
- Não há aumento na amplitude de nenhuma onda senoidal do espectro de Fourier da imagem;
- São feitos a partir da combinação de filtros passa-baixa e passa-alta;
- Podem ser de vários tipos. Os mais comuns são: Ideal, Butterworth e Gaussiano.

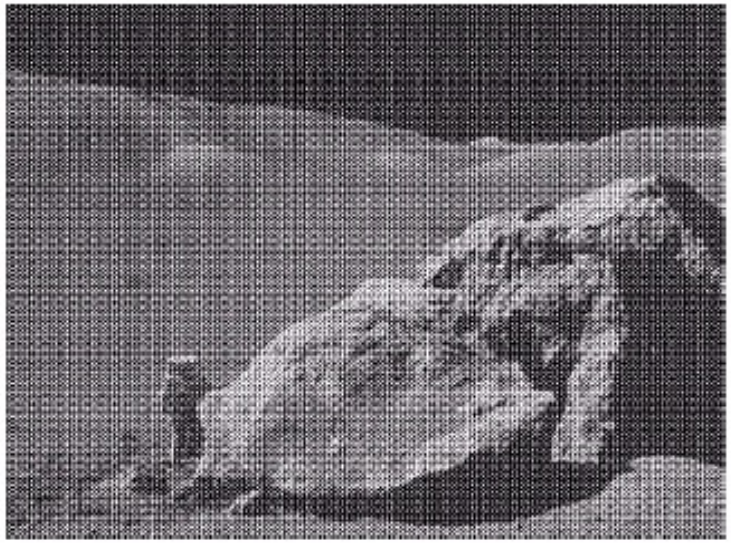
Filtros Passa-Banda

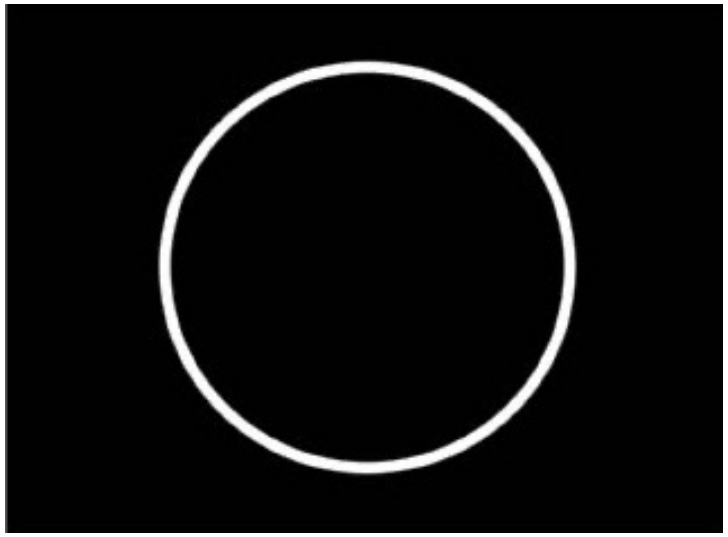
As equações dos filtros Passa-Banda podem ser obtidos a partir das equações dos filtros Rejeita-Banda:

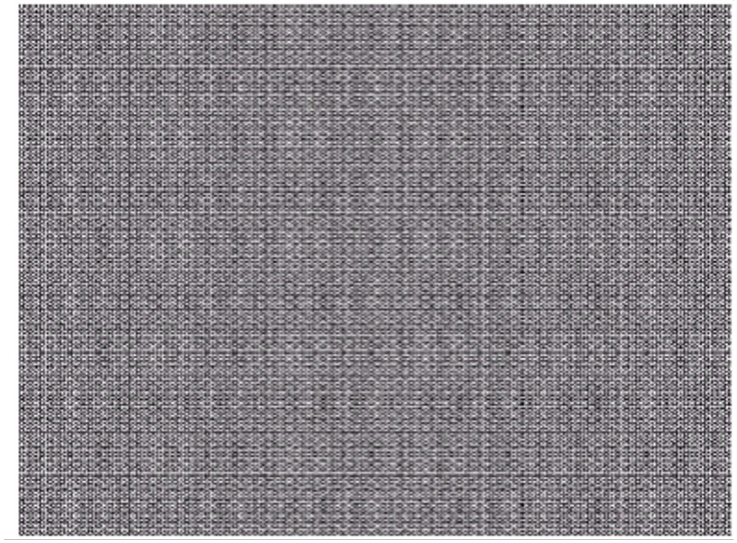
$$H(u,v)_{PB} = 1 - H(u,v)_{RB}$$

O centro da banda (D_0) define o valor onde a amplitude do filtro é 1.

Filtro Passa-Banda **Ideal**

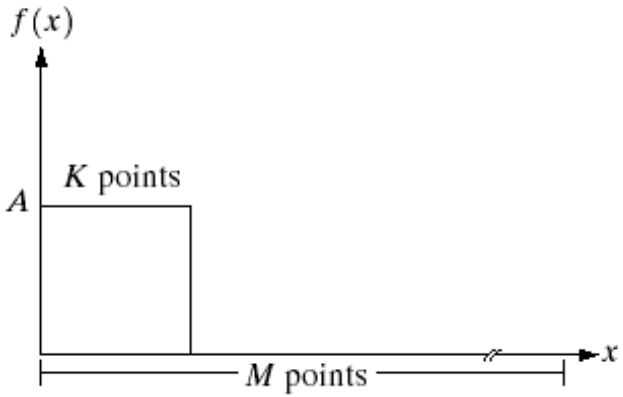


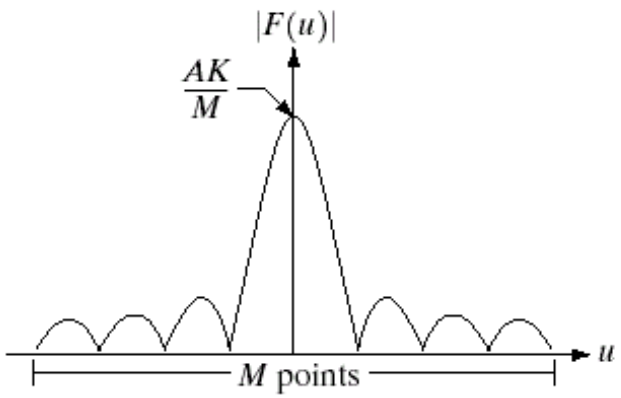


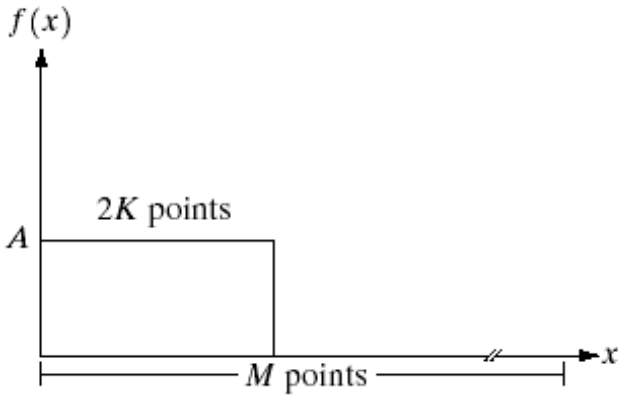


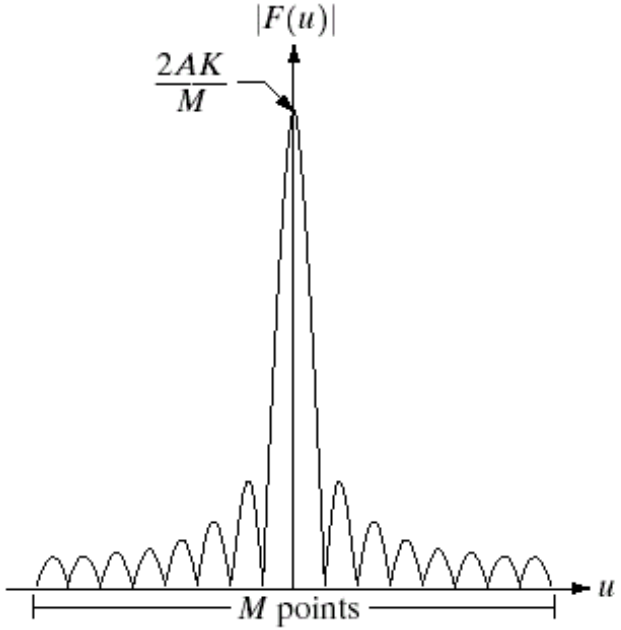
Relação entre a posição do pixel e a frequência espacial

DFT 2-D


f(x)
A
K points
M points
x


|F(u)|
AK/M
M points
u


f(x)
A
2K points
M points
x


|F(u)|
2AK/M
M points
u

Intervalo de frequência a partir do centro (passo)

Imagem M x N pixels

$$\Delta u = \frac{1}{M \cdot \Delta x} \qquad \Delta v = \frac{1}{N \cdot \Delta y}$$

Em uma imagem digital, qual o valor da frequência máxima representada no espectro de Fourier?

Intervalo de frequência a partir do centro (passo) =

$$\Delta u = \frac{1}{M \cdot \Delta x}$$

Frequência Máxima (Teorema de Nyquist) =

$$u_{máx} = \frac{1}{2\Delta x}$$

Image Padding

Preenchimento da imagem para resolver o problema com as bordas

Solução para os pixels das bordas na Convolução:

Podem ser usadas quatro soluções:

1. Preenchimento da imagem com qualquer valor (geralmente zeros) antes do cálculo da imagem final (*padding*) (*constant**);
2. Espelhamento dos pixels das bordas (*reflect**);
3. Replicação dos pixels das bordas (*replicate**);
4. Convolução periódica ou circular (*wrap**);

** Função usada pelo Python*

Multiplicação e Convolução

$$f(x,y) * g(x,y) \Leftrightarrow F(u,v)G(u,v)$$

$$f(x,y)g(x,y) \Leftrightarrow F(u,v) * G(u,v)$$

Convolução *no domínio do tempo/espaço*

Multiplicação *no domínio da frequência*

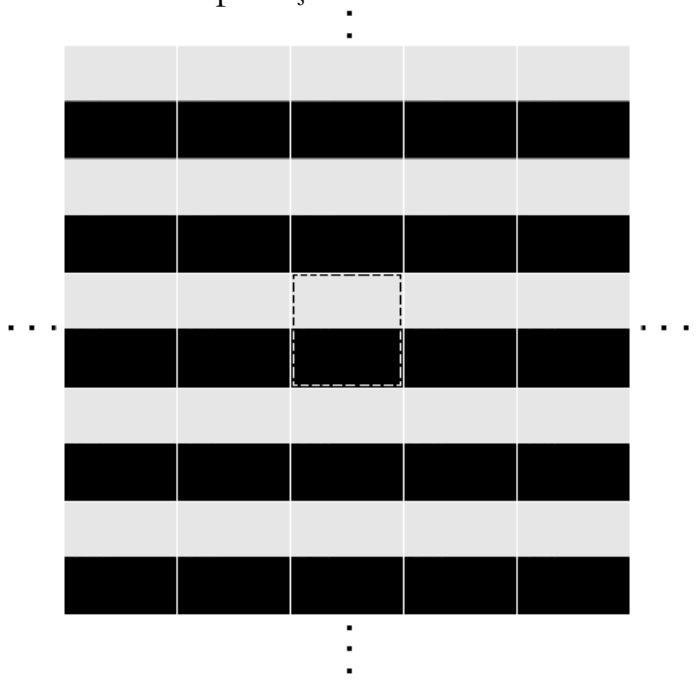
Multiplicação *no domínio do tempo/espaço*

Convolução *no domínio da frequência*

O que acontece com as bordas da imagem para filtragem no domínio da frequência?

Imagem original

Imagem periódica considerada na aplicação da DFT



Como a DFT é periódica. A multiplicação no domínio da frequência equivale a uma convolução periódica (ou circular) no domínio do espaço.

Esse fenômeno é chamado de *Wrap-around Error*

O que acontece com as bordas da imagem para filtragem no domínio da frequência?

Assim, se nenhum *padding* for realizado, a filtragem no domínio da frequência vai apresentar o mesmo resultado que uma convolução periódica no domínio do espaço:

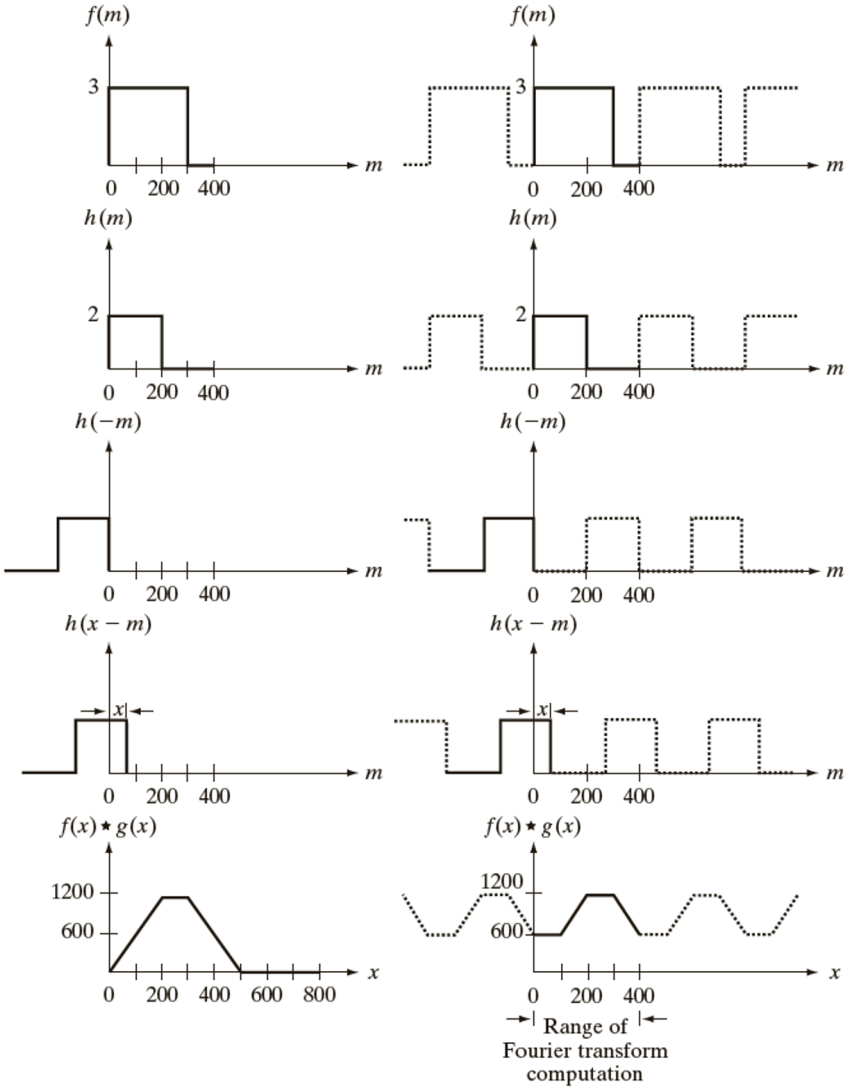
Imagem original

Imagem filtrada por um filtro passa baixa no domínio da frequência

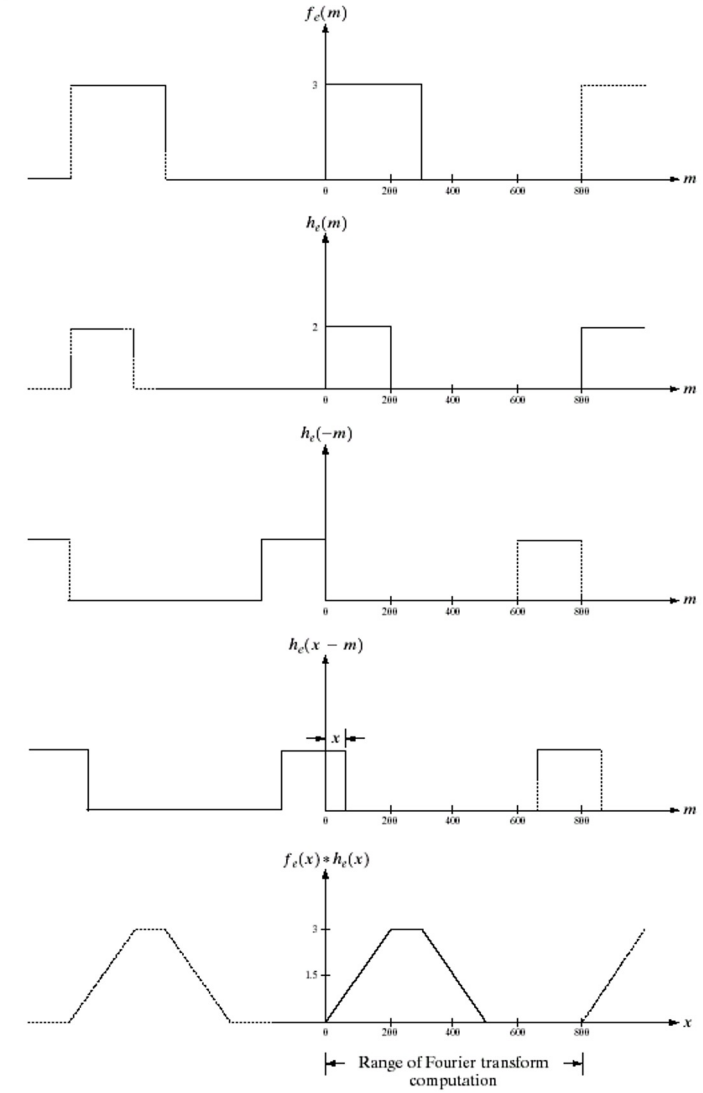
Image Padding

- Se a convolução periódica não é desejável, deve-se realizar outro tipo de *padding* para resolver o problema do *wrap-around error;*
- O *padding* deve ser realizado levando-se em conta que o filtro é do mesmo tamanho da imagem, pois a filtragem é realizada no domínio da frequência;
- Assim, a dimensão da imagem, após o *padding*, deve ser de pelo menos o dobro do tamanho da imagem original.

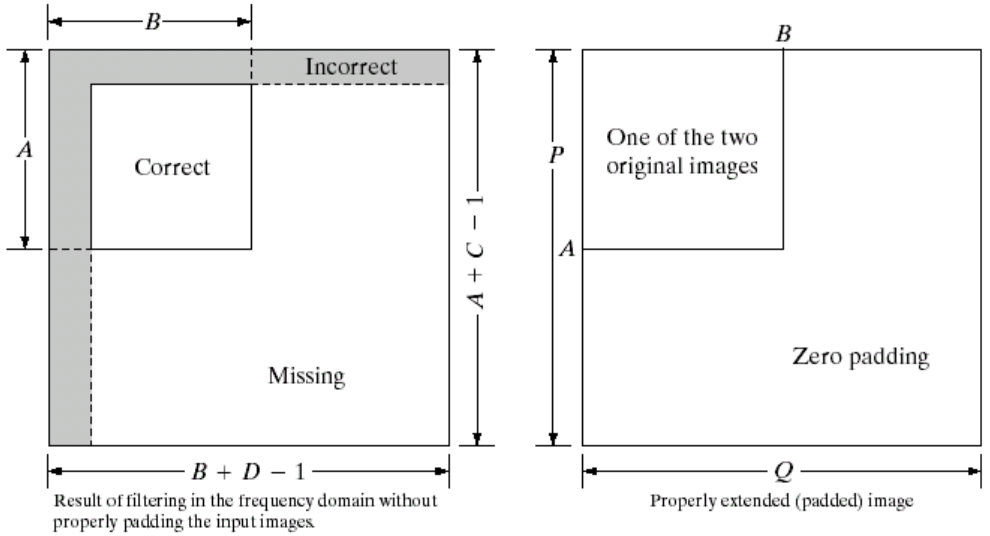
Image Padding – 2D

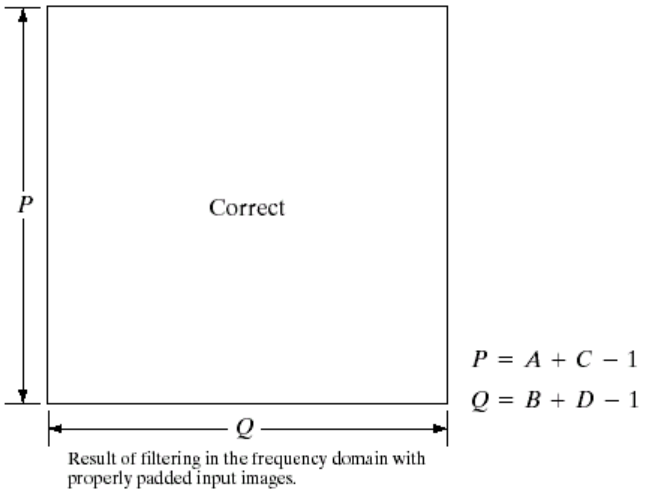


Zero Padding



Zero Padding

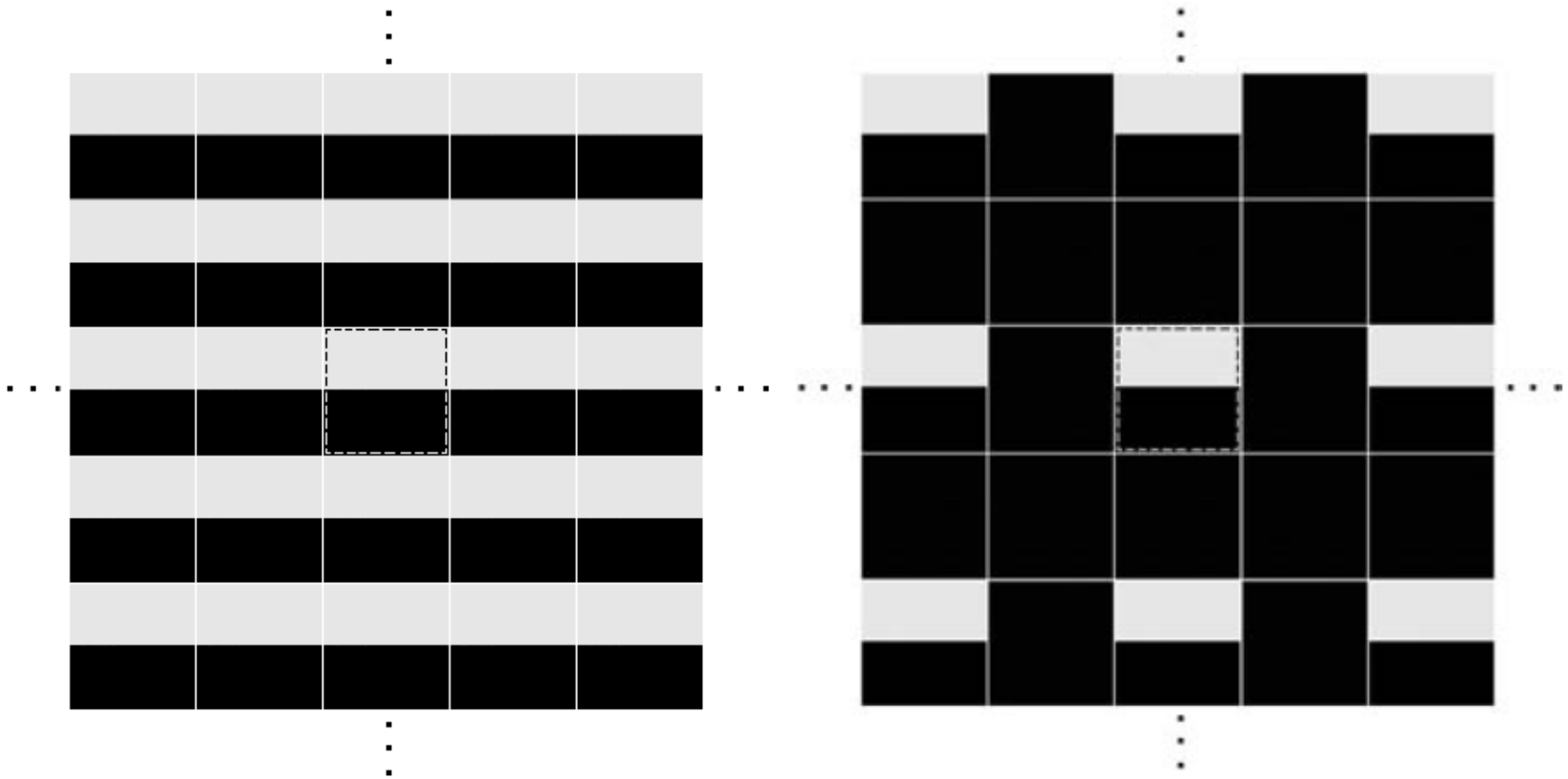




a b
c

FIGURE 4.38
Illustration of the need for function padding. (a) Result of performing 2-D convolution without padding. (b) Proper function padding. (c) Correct convolution result.

Zero Padding



Zero Padding

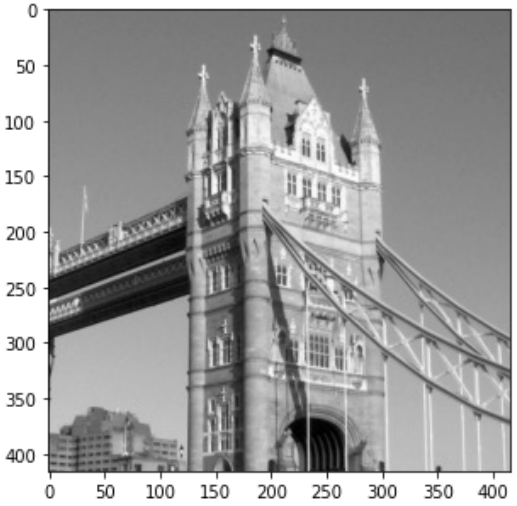


Imagem original
416 x 416

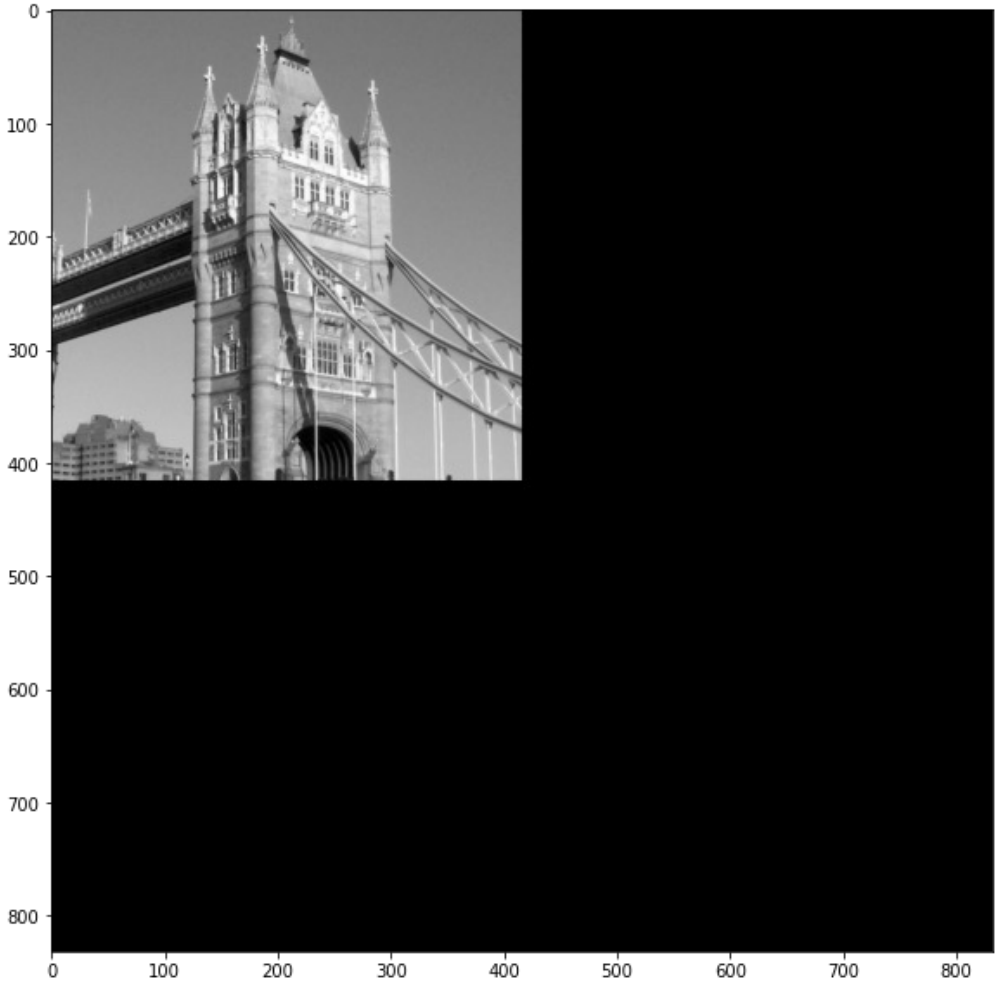


Imagem preenchida com zeros
832 x 832

Zero Padding

Imagem original

Imagem filtrada por um filtro passa baixa no domínio da frequência SEM *padding*

Imagem filtrada por um filtro passa baixa no domínio da frequência COM *padding*

Image Padding

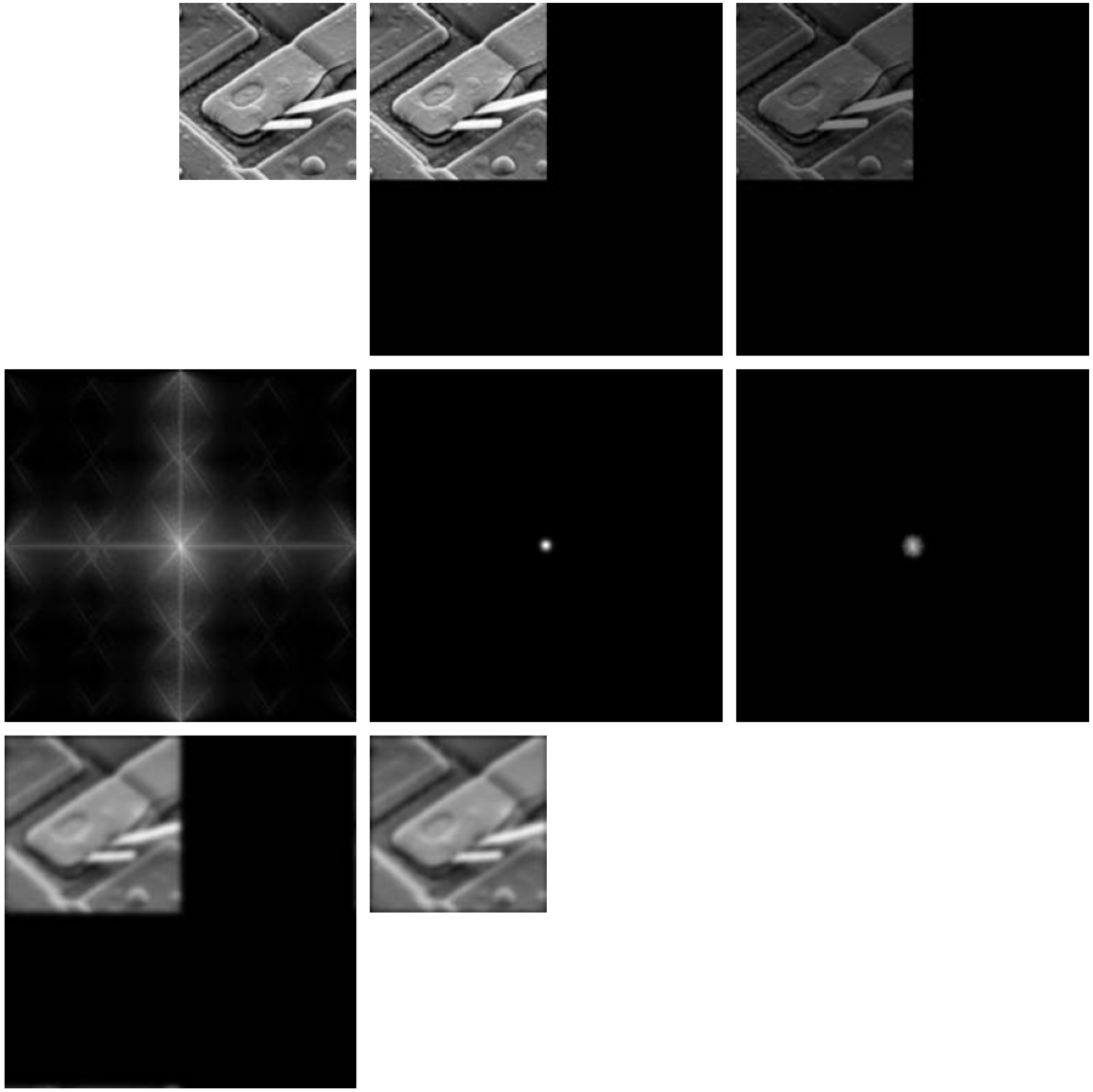
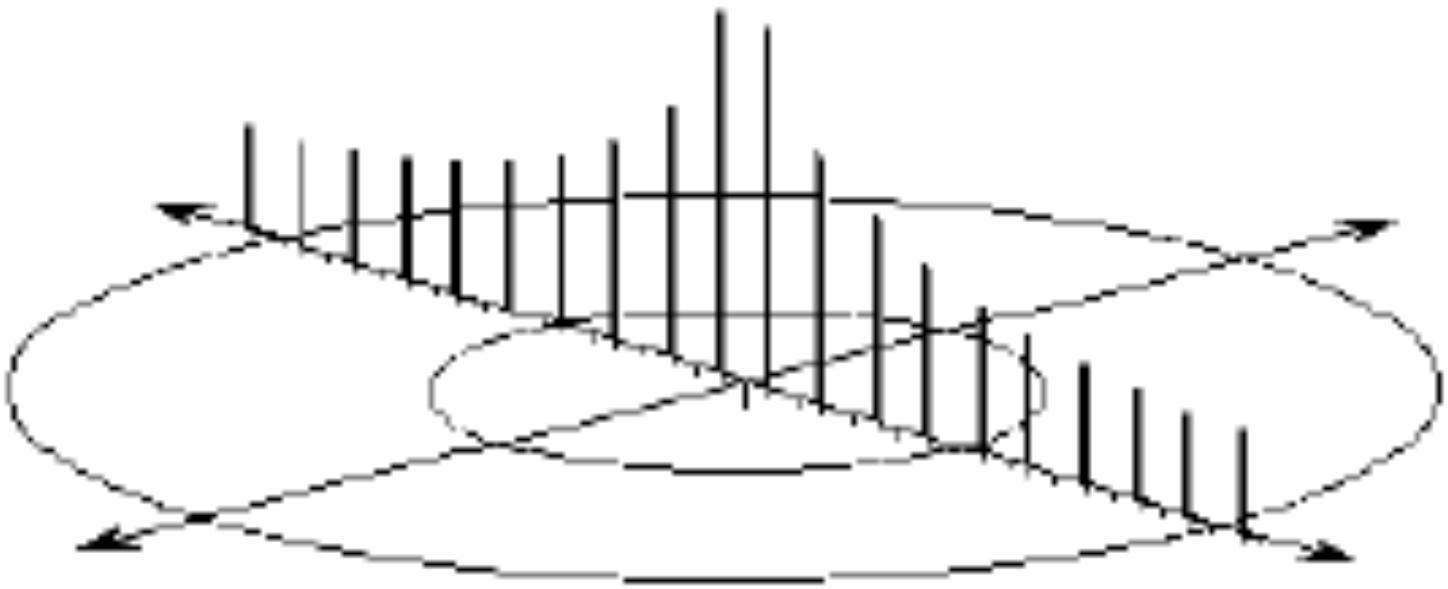


Image Padding

- Vantagens de se usar *Image Padding* para filtragem no domínio da frequência:
  - Evitar o *wrap-around error*, ou seja, evitar o resultado de uma convolução periódica;
  - Aumentar o tamanho da imagem para se obter um número maior de pixels na representação da imagem no domínio da frequência (maior precisão do espectro de Fourier), ou seja, alterar os valores de Δu e Δv, já que eles dependem de M e N, respectivamente;
  - Alterar os valores de M e N (e consequentemente de Δu e Δv) para que eles sejam iguais, de modo a tornar a imagem e o espectro de Fourier quadrados, o que facilita a construção dos filtros;

Os Filtros devem ser circulares e concêntricos



Métricas de Distância

- Os filtros circulares devem ser construídos calculando-se as distâncias dos pixels em relação ao ponto central do espectro (frequência zero).
- À medida que a distância aumenta, a frequência de corte também aumenta.

Distância Euclidiana: $$D_e(p,q) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

Onde fica o centro do espectro de Fourier, ou seja, a frequência espacial ZERO em Python?

Distribuição de Frequências após a Transformada de Fourier em Python

Matriz de Tamanho Ímpar 7x7

Y \ X	0	1	2	3	4	5	6
0	0,30822 - 3,557i	7,4462 + 0,26201i	9,2165 + 1,4641i	**-6,9946 - 4,3351i**	-9,145 - 12,2i	7,7022 + 14,61i	-1,0266 - 3,3356i
1	2,2726 + 0,65998i	19,807 - 9,3154i	-15,659 + 12,658i	**-22,223 + 6,7232i**	15,244 - 2,057i	-3,0456 + 5,3591i	1,7265 + 4,6776i
2	19,809 - 4,2954i	35,918 + 47,943i	36,384 - 8,7932i	**5,2174 - 50,745i**	15,072 - 27,215i	6,5109 + 14,033i	11,957 - 5,9855i
3	**2,4511 - 4,7286i**	**-34,496 - 39,76i**	**-44,456 - 93,287i**	4927	**-44,456 + 93,287i**	**-34,496 + 39,76i**	**2,4511 + 4,7286i**
4	11,957 + 5,9855i	6,5109 - 14,033i	15,072 + 27,215i	**5,2174 + 50,745i**	36,384 + 8,7932i	35,918 - 47,943i	19,809 + 4,2954i
5	1,7265 - 4,6776i	-3,0456 - 5,3591i	15,244 + 2,057i	**-22,223 - 6,7232i**	-15,659 - 12,658i	19,807 + 9,3154i	2,2726 - 0,65998i
6	-1,0266 + 3,3356i	7,7022 - 14,61i	-9,145 + 12,2i	**-6,9946 + 4,3351i**	9,2165 - 1,4641i	7,4462 - 0,26201i	0,30822 + 3,557i

Distribuição de Frequências após a Transformada de Fourier em PYTHON

Matriz de Tamanho Par 6 x 6

Y \ X	0	1	2	3	4	5
0	7	8 - 1,7321i	1 + 13,856i	**-7**	1 - 13,856i	8 + 1,7321i
1	7 + 5,1962i	11,5 + 12,99i	-8 - 8,6603i	**-20 - 6,9282i**	5,5 + 6,0622i	7 - 3,4641i
2	13 + 12,124i	-4 + 22,517i	38,5 + 12,99i	**20 - 27,713i**	7 - 17,321i	24,5 + 12,99i
3	**-11**	**-8 - 39,837i**	**-35 - 65,818i**	**3583**	**-35 + 65,818i**	**-8 + 39,837i**
4	13 - 12,124i	24,5 - 12,99i	7 + 17,321i	**20 + 27,713i**	38,5 - 12,99i	-4 - 22,517i
5	7 - 5,1962i	7 + 3,4641i	5,5 - 6,0622i	**-20 + 6,9282i**	-8 + 8,6603i	11,5 - 12,99i

Posição central do espectro no Python (tanto para matriz ímpar como para par)

Y \ X	0	1	2	3	4	5	6
0	0	0	0	**0**	0	0	0
1	0	0	0	**0**	0	0	0
2	0	0	0	**0**	0	0	0
3	**0**	**0**	**0**	**1**	**0**	**0**	**0**
4	0	0	0	**0**	0	0	0
5	0	0	0	**0**	0	0	0
6	0	0	0	**0**	0	0	0

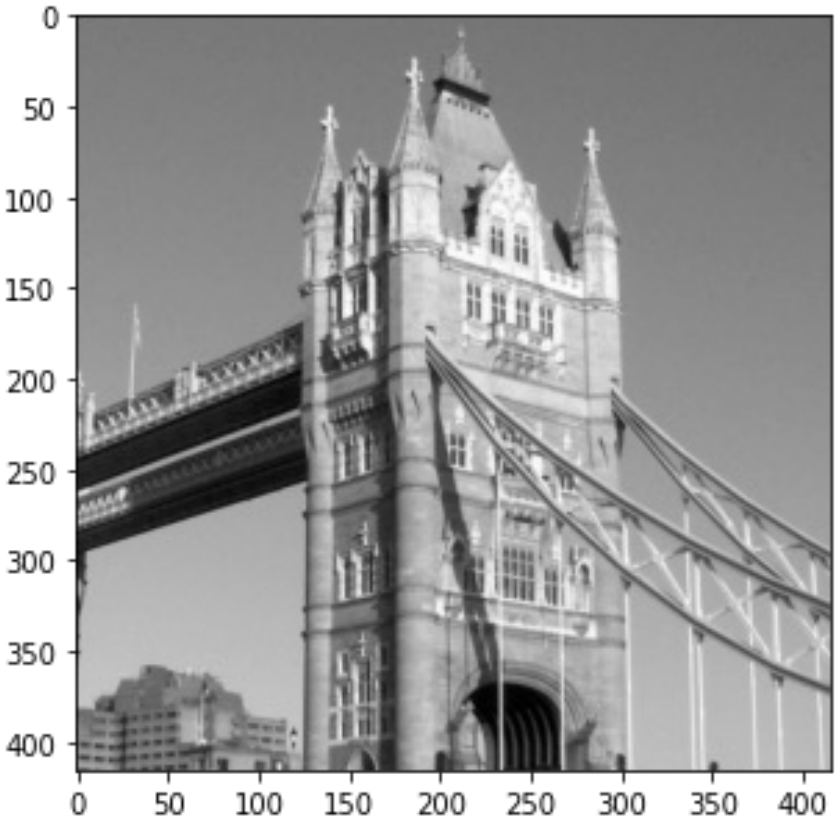
Y \ X	0	1	2	3	4	5
0	0	0	0	**0**	0	0
1	0	0	0	**0**	0	0
2	0	0	0	**0**	0	0
3	**0**	**0**	**0**	**1**	**0**	**0**
4	0	0	0	**0**	0	0
5	0	0	0	**0**	0	0

```
cx = int(M/2)
cy = int(N/2)
```

Parte inteira da divisão por 2

Filtragem no domínio da frequência em Python

Imagem Original

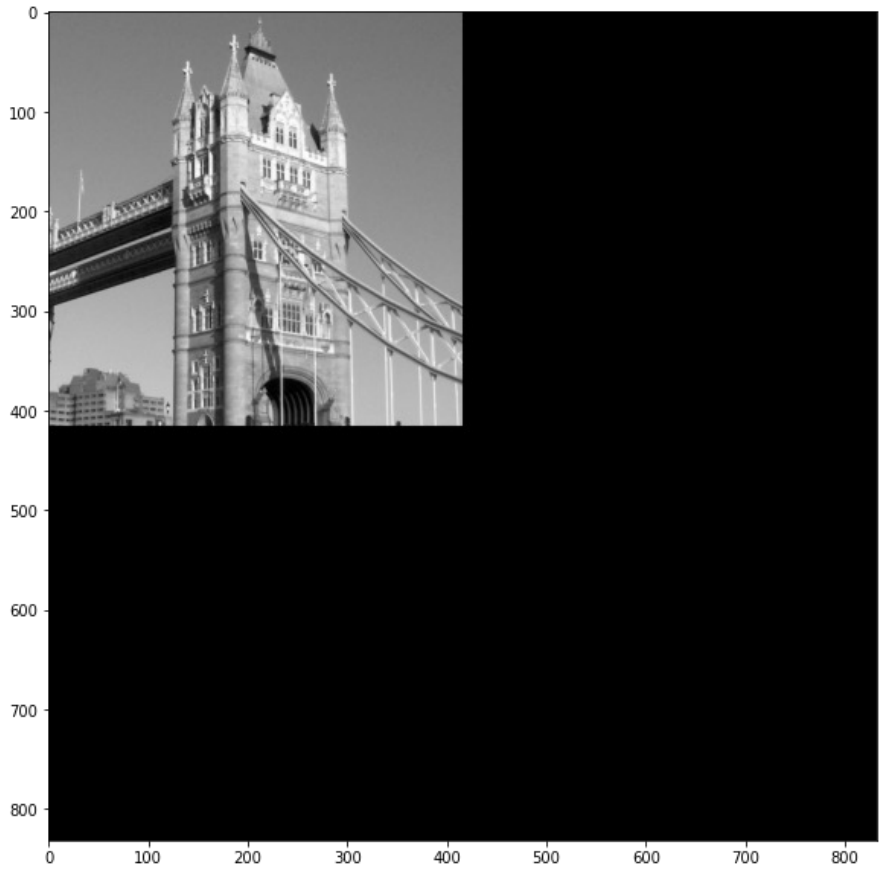


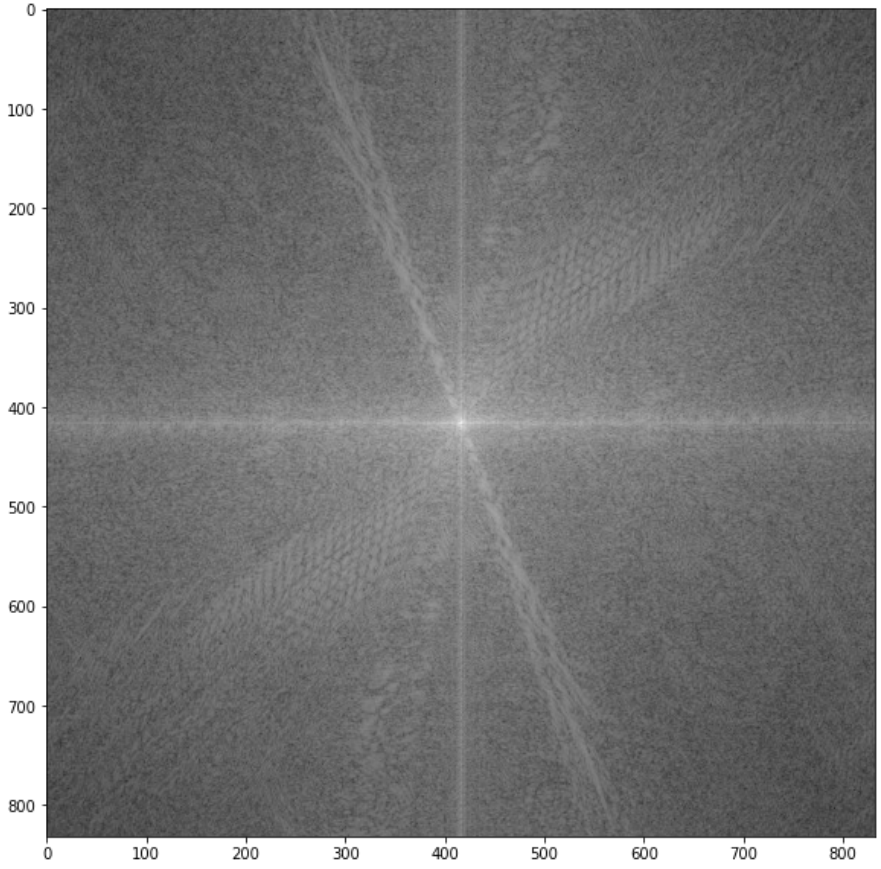
M = 416
N = 416

Visualização do Espectro de Fourier

```
import numpy as np
import matplotlib.pyplot as plt
import cv2 as cv

# Leitura da imagem
a = cv.imread('towerbridge.tif',cv.IMREAD_UNCHANGED)

# Dimensões imagem
M,N = a.shape

# Visualização da imagem
plt.figure()
plt.imshow(a,'gray')
plt.show()

# Dimensões considerando padding
Mf = 2*M;
Nf = 2*N;

# Transformação domínio de frequências
b = np.fft.fft2(a,s=(Mf,Nf)) #s = parâmetro zero padding
b = np.fft.fftshift(b)

# Visualização do espectro de Fourier
c = np.abs(b)
c_plot = 20*np.log(c+1)
plt.figure()
plt.imshow(c_plot,'gray')
plt.show()
```

Espectro de Fourier





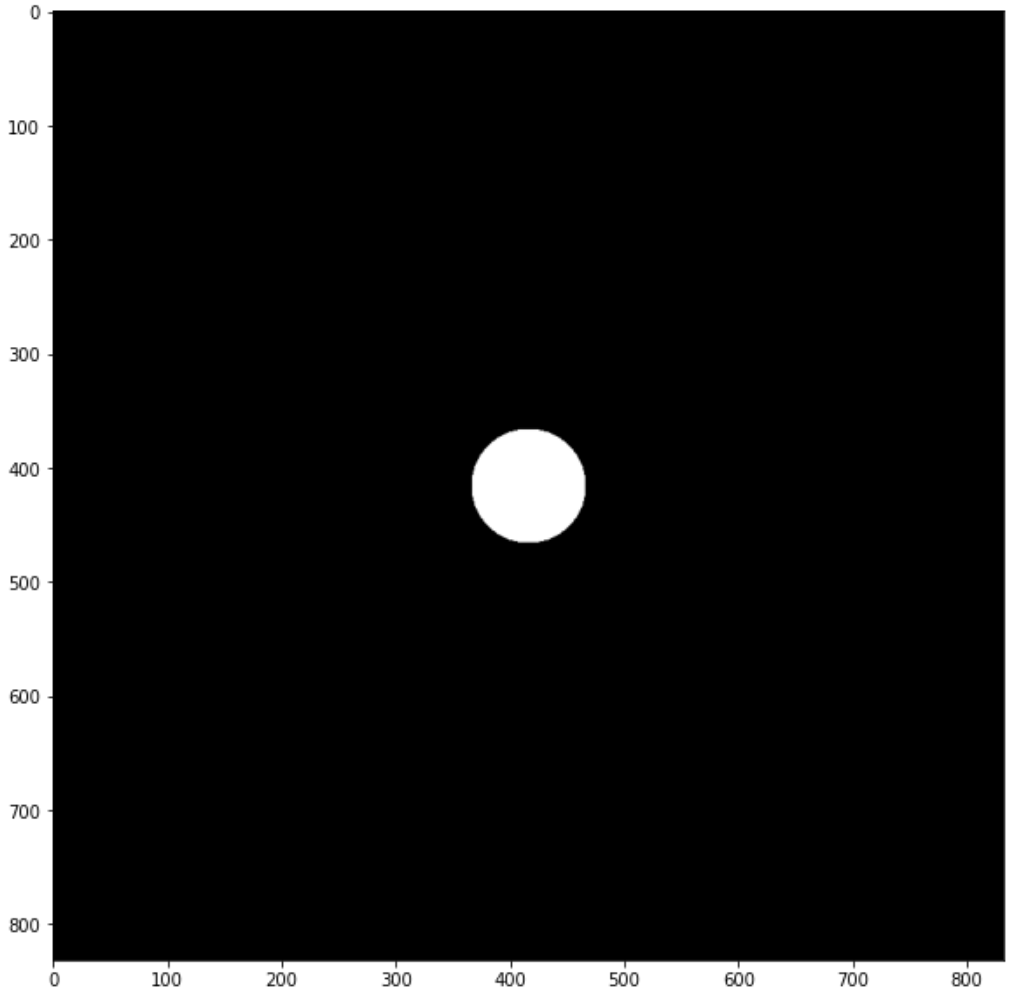
$Mf = 416*2 = 832$

$Nf = 416*2 = 832$

Filtro Passa-Baixa Ideal

```
# Calculando o centro (// = parte inteira da divisão)
cx = Nf//2
cy = Mf//2

# Frequência de corte, em pixels a partir da origem
D0 = 50

filtro = np.zeros((Mf,Nf))
for x in range(0,Mf):
    for y in range (0,Nf):
        D = np.sqrt((x-cx)**2+(y-cy)**2)
        if D < D0:
            filtro [x,y] = 1

plt.figure()
plt.imshow(filtro,'gray')
plt.show()
```

Filtro Passa-Baixa Ideal

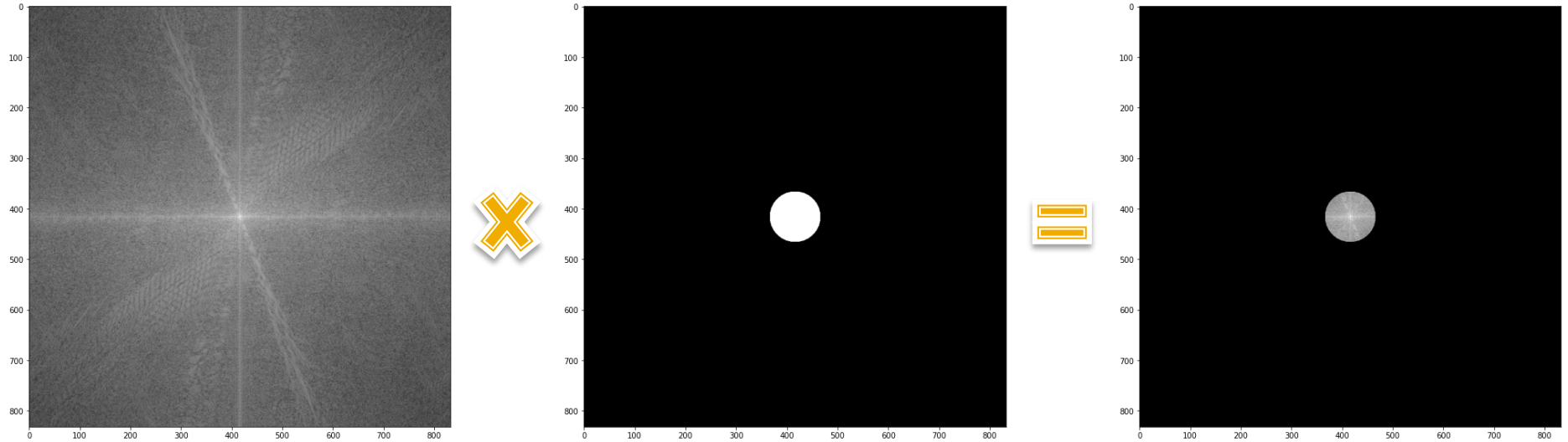


cx = 416
cy = 416

Filtragem no Domínio da Frequência

```
#Processamento no domínio da frequência
b = b * filtro

# Visualização do espectro de Fourier após filtragem
c = np.abs(b)
c_plot = 20*np.log(c+1)
plt.figure()
plt.imshow(c_plot,'gray')
plt.show()
```

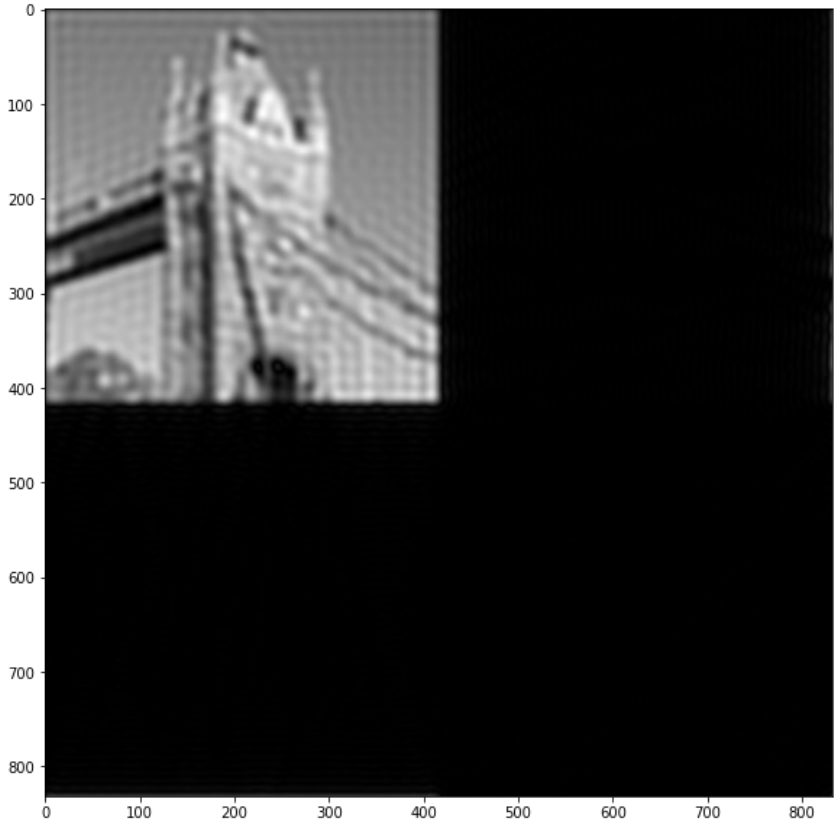
Filtragem no Domínio da Frequência

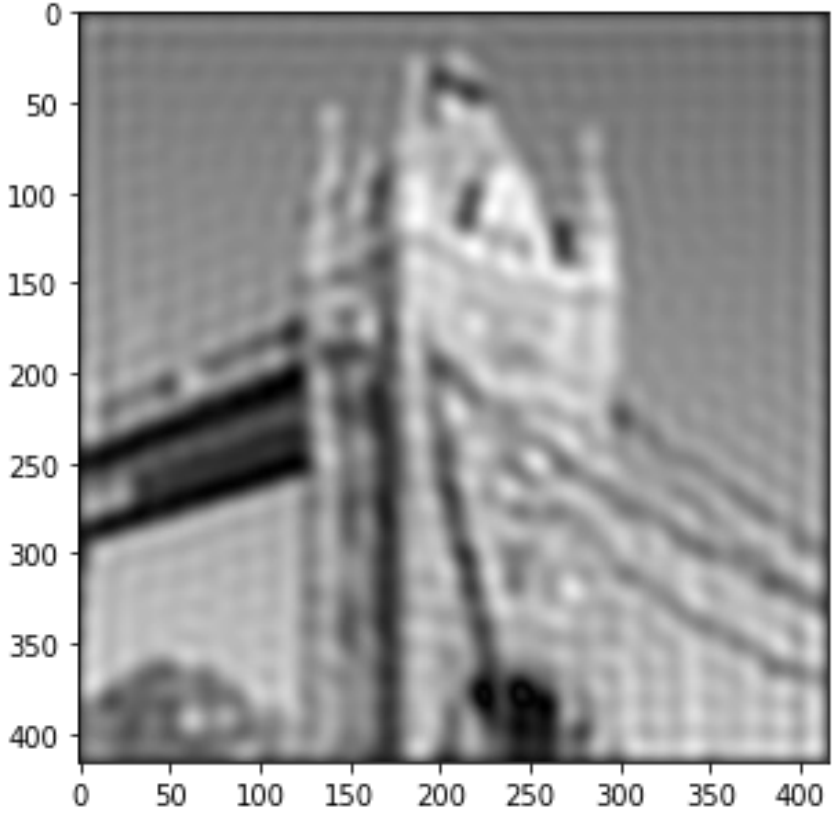


Transformada Inversa e Visualização da Imagem Filtrada

```
# Transformação inversa
i1 = np.fft.ifftshift(b)
i1 = np.fft.ifft2(i1)
i1 = np.abs(i1)

# Visualização da imagem filtrada
plt.figure()
plt.imshow(i1,'gray')
plt.show()

# Recorte tamanho original
i1 = i1[0:M,0:N]

# Visualização da imagem filtrada
plt.figure()
plt.imshow(i1,'gray')
plt.show()
```

Transformada Inversa e Visualização da Imagem Filtrada





Padding em PYTHON

Padding em Python (*numpy*)

- Pelo `numpy.fft.fft2`, só é possível fazer diretamente o *padding* com zeros (utilizando o parâmetro *s*);

- Para fazer outros *paddings* (*symmetric*, *edge*, *wrap*, etc.), deve-se utilizar, primeiramente, a função `numpy.pad` para realizar o *padding* antes de utilizar a função `numpy.fft.fft2`.

Função *numpy.pad*

Podem ser usadas diversas soluções para o *padding*.
As mais comuns são:

1. Preenchimento da imagem com qualquer valor (geralmente zeros) antes do cálculo da imagem final (*padding*) (*constant**);
2. Espelhamento dos pixels das bordas (*symmetric**);
3. Replicação dos pixels das bordas (*edge**);
4. Convolução periódica ou circular (*wrap**);

** Função usada pelo numpy.pad*

Função *numpy.pad*

Sintaxe:

numpy.pad(imagem,((antes_x,depois_x),(antes_y,depois_y)),'tipo')

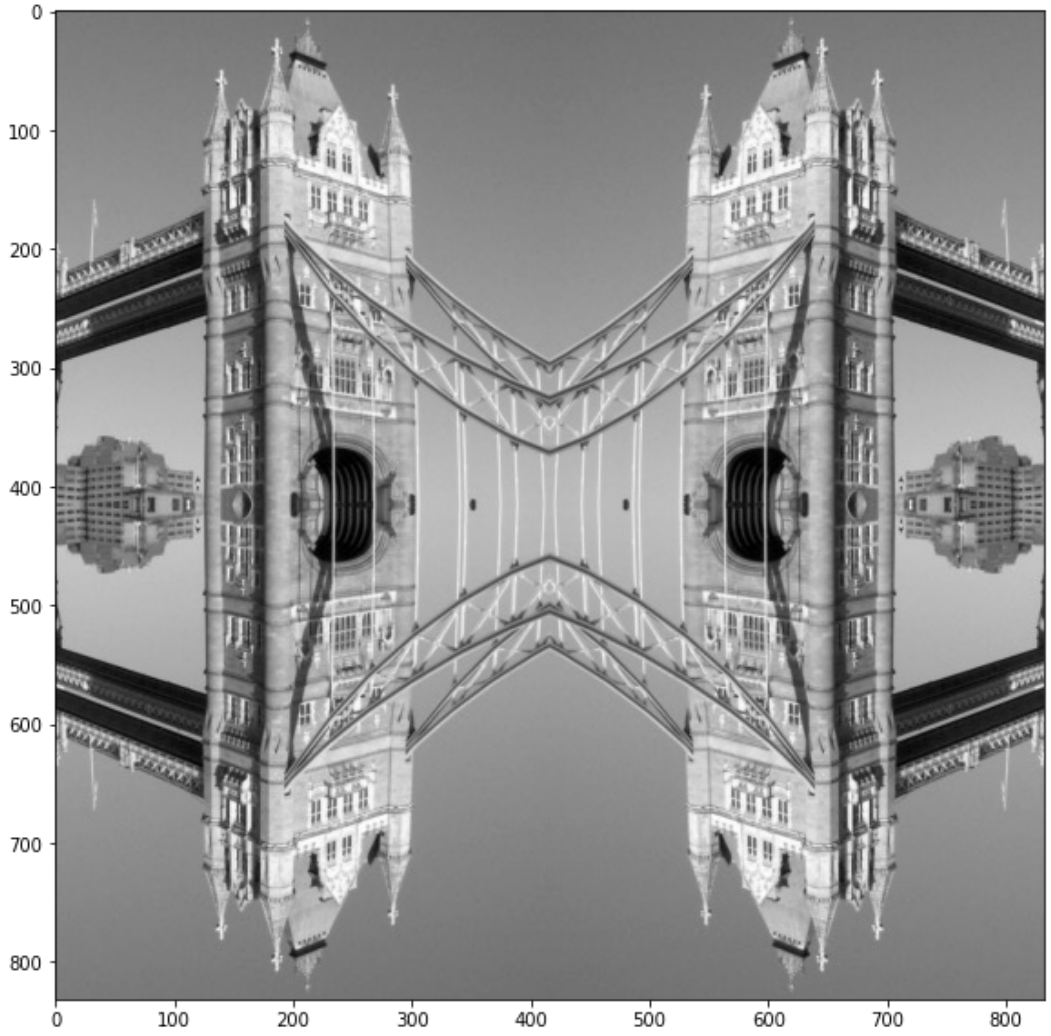
```
a = [[1,2],[3,4]]
b = np.pad(a,((0,2),(0,3)),'symmetric')
print(b)
```

```
[[1 2 2 1 1]
 [3 4 4 3 3]
 [3 4 4 3 3]
 [1 2 2 1 1]]
```

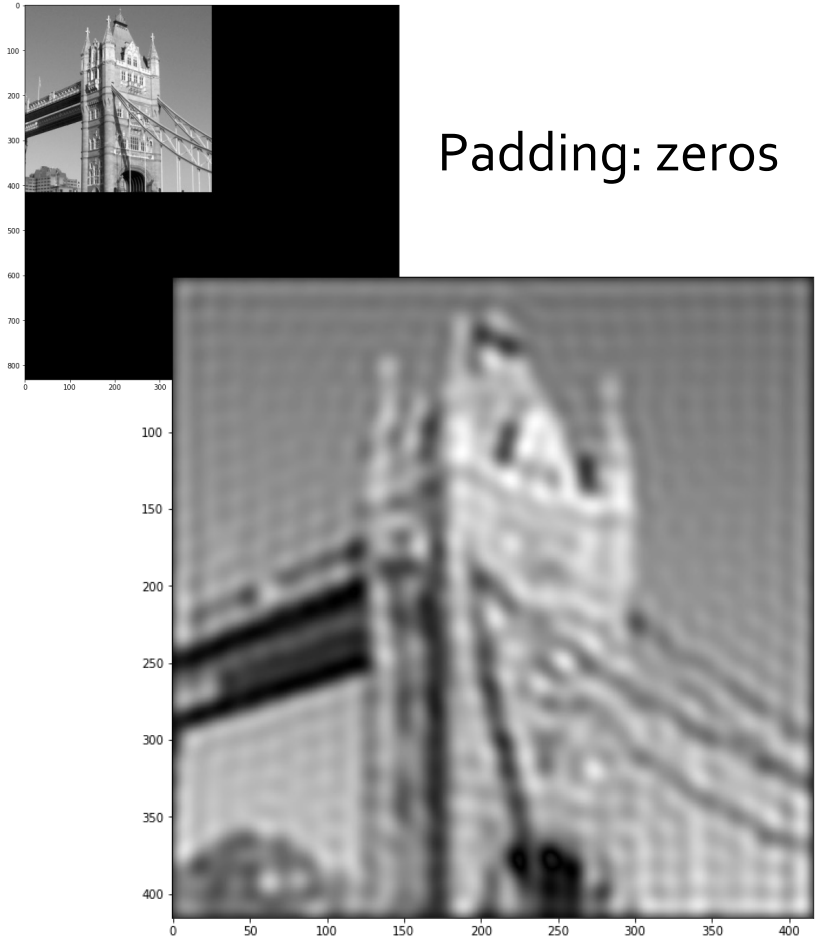
Padding Symmetric

```
import numpy as np
import matplotlib.pyplot as plt
import cv2 as cv

# Leitura da imagem
a = cv.imread('towerbridge.tif',cv.IMREAD_UNCHANGED)

# Dimensões imagem linhas,colunas
M,N = a.shape

# Visualização da imagem
plt.figure()
plt.imshow(a,'gray')
plt.show()

a_padding_sym = np.pad(a,((0,M),(0,N)),"symmetric")

# Transformação domínio de frequências
b = np.fft.fft2(a_padding_sym)
b = np.fft.fftshift(b)

# Visualização do espectro de Fourier
c = np.abs(b)
c_plot = 20*np.log(c+1)
plt.figure()
plt.imshow(c_plot,'gray')
plt.show()
```

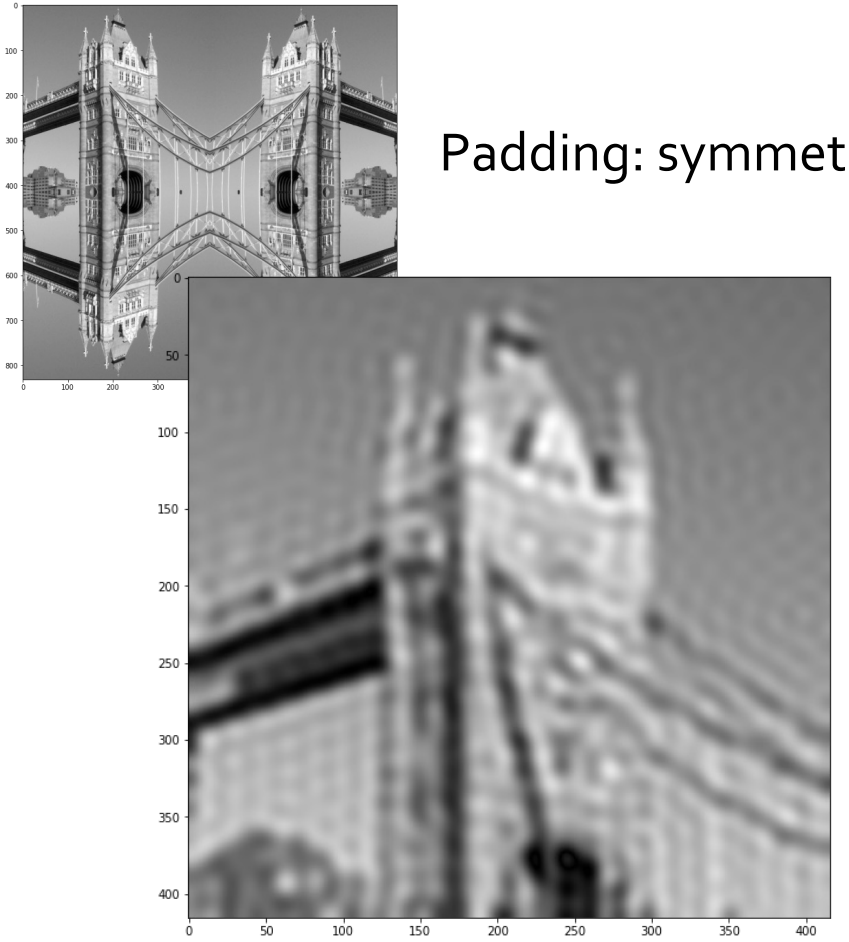
Padding Symmetric



Resultados com a filtragem



Padding: zeros



Padding: symmetric

FIM