# VISÃO COMPUTACIONAL

## AULA 4B

## RUÍDO EM IMAGENS

## Filtragem de Ruídos

# RUÍDO EM IMAGENS

## FILTRAGEM DE RUÍDO

- **RUÍDO EM IMAGENS**

Em Visão Computacional, ruído se refere a qualquer entidade em imagens, dados ou resultados intermediários, que não são interessantes para os propósitos do cálculo principal.

Ruído pode ser

  - Flutuações espúrias de valores de píxeis introduzidos pelo sistema de aquisição de imagens, afetando a detecção de linhas ou contornos em algoritmos de processamento de imagens,

  - Flutuações aleatórias ou imprecisões em dados de entrada, precisão numérica, arredondamentos etc...

  - Contornos que não pertençam a nenhum objeto, em algoritmos para agrupamento de linhas.

Matematicamente,

$$\hat{I}(i,j) = I(i,j) + n(i,j)$$

$n(i, j)$ = ruído aditivo e aleatório

$I(i, j)$ = imagem pura

$\hat{I}(i, j)$ = imagem com ruído

- Quantificação do ruído

$$\mathrm{SNR} = \frac{\mu_s}{\sigma_n}$$ (Signal-to-Noise Ratio – imagens)

$\mu_s$ = média do sinal ( $I(i, j)$ )

$\sigma_n$ = desvio padrão do ruído ( $n(i, j)$ )

$$\mathrm{SNR}_{db} = 10.\log_{10}\frac{\mu_s}{\sigma_n}$$

ou

$$\mathrm{SNR}_{db} = 20.\log_{10}\frac{\mu_s}{\sigma_n}$$

Em alguns casos, o ruído pode ser multiplicativo

$$\hat{I} = n.I$$

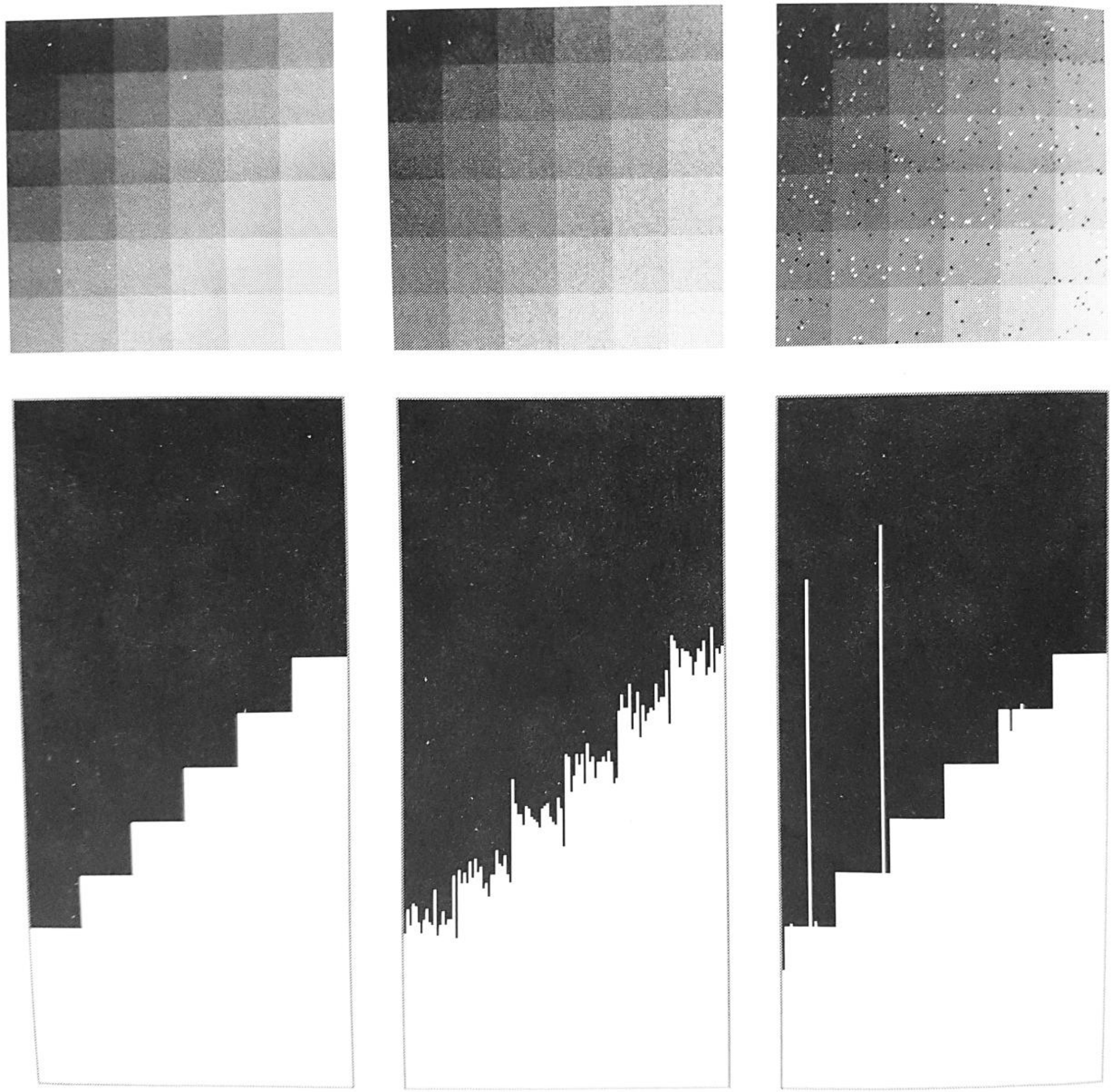
- Degradação de imagens em linhas de TV
- Fotografias (tamanho de grão)
- Variações na sensibilidade do sensor
- Iluminação não uniforme
- Interferência eletromagnética
- Erros de quantização

- **RUÍDO GAUSSIANO**

  - n(i, j) modelado como ruído branco, gaussiano, processo estocástico de média nula.

  - modelo apropriado para sistema de aquisição de qualidade, com baixos níveis de ruído.

  - Mais fácil de modelar estatisticamente.



a) Imagem sintética de tabuleiro de xadrez e valor de cinza ao longo de uma linha; b) após adição de ruído Gaussiano de média nula; c) após adição de ruído sal-e-pimenta.

- **RUÍDO IMPULSO**
  - Pontos brilhantes ou escuros
  - Causados por erros de transmissão, elementos defeituosos no CCD/CMOS, ou ruído externo corrompendo a conversão A-D.

O ruído sal-e-pimenta (salt-and-pepper) pode ser adotado para modelar o ruído impulso como

$$I_{sp}(h,k) = \begin{cases} I(h,k) & x < \varepsilon \\ i_{min} + y.(i_{max} - i_{min}) & x \geq \varepsilon \end{cases}$$

I = imagem real

[x,y] $\in$ [0,1]= variáveis aleatórias uniformemente distribuídas

ε = parâmetro de controle da distribuição qualitativa

$i_{min}$ e $i_{max}$ = intensidade do ruído

O ruído se torna saturado se

$$y = 0 \;\; \text{ou} \;\; y = 1 \qquad \text{e} \quad i_{min} = 0 \; ; \; i_{max} = 255$$

Na figura $\varepsilon = 0{,}99$ ; $i_{min} = 0$ ; $i_{max} = 255$

- **FILTRAGEM DE RUÍDOS**

  É necessária, pois

  - Várias operações são distorcidas pelo efeito de ruídos

    - Derivadas de imagens (base para muitos algoritmos)

  Pode ser

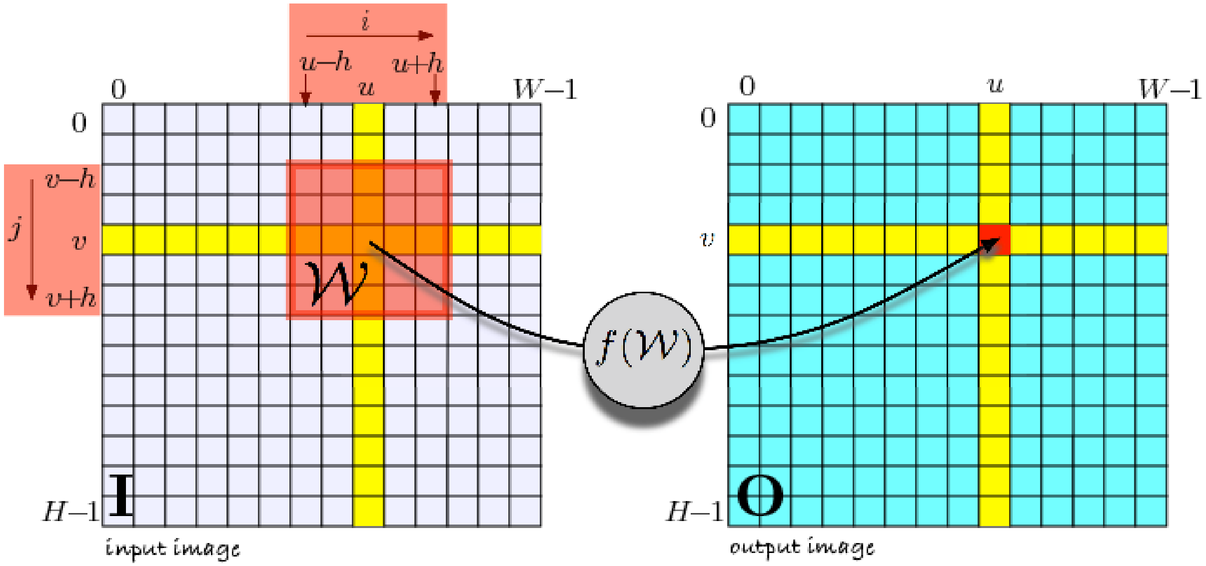
  - Linear

  - Não-linear

- **FILTRAGEM LINEAR**

  - Convolução da imagem com uma matriz constante

    - Máscara ou Kernel




| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 45 | 60 | 98 | 127 | 132 | 133 | 137 | 133 |
| 46 | 65 | 98 | 123 | 126 | 128 | 131 | 133 |
| 47 | 65 | 96 | 115 | 119 | 123 | 135 | 137 |
| 47 | 63 | 91 | 107 | 113 | 122 | 138 | 134 |
| 50 | 59 | 80 | 97 | 110 | 123 | 133 | 134 |
| 49 | 53 | 68 | 83 | 97 | 113 | 128 | 133 |
| 50 | 50 | 58 | 70 | 84 | 102 | 116 | 126 |
| 50 | 50 | 52 | 58 | 69 | 86 | 101 | 120 |

$f(x,y)$

*

| | | |
|---|---|---|
| 0.1 | 0.1 | 0.1 |
| 0.1 | 0.2 | 0.1 |
| 0.1 | 0.1 | 0.1 |

$h(x,y)$

=

| | | | | | |
|---|---|---|---|---|---|
| 69 | 95 | 116 | 125 | 129 | 132 |
| 68 | 92 | 110 | 120 | 126 | 132 |
| 66 | 86 | 104 | 114 | 124 | 132 |
| 62 | 78 | 94 | 108 | 120 | 129 |
| 57 | 69 | 83 | 98 | 112 | 124 |
| 53 | 60 | 71 | 85 | 100 | 114 |

$g(x,y)$

## Algoritmo LINEAR_FILTER

$I$ = matriz N X M

A = máscara do filtro linear (m x m)

m = número ímpar menor do que N e M

$I_A$ = imagem filtrada

$$I_A(i,j) = I * A = \sum_{h=-m/2}^{m/2} \sum_{k=-m/2}^{m/2} A(h,k).I(i-h, j-k)$$

m/2 = divisão inteira ( p. ex. , 3/2 = 1)

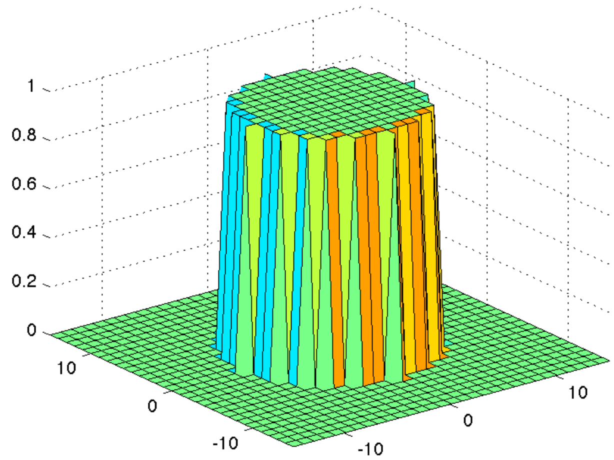
- Substituir I (i , j) por soma ponderada da vizinhança de (i , j).
- Pesos são os elementos da máscara A .
- $\Im\{I * A\} = \Im\{I\}.\Im\{A\}$

- **ATENUAÇÃO PELA MÉDIA**

Quando todos os elementos de A são não negativos

- Filtro Média (mean filter)

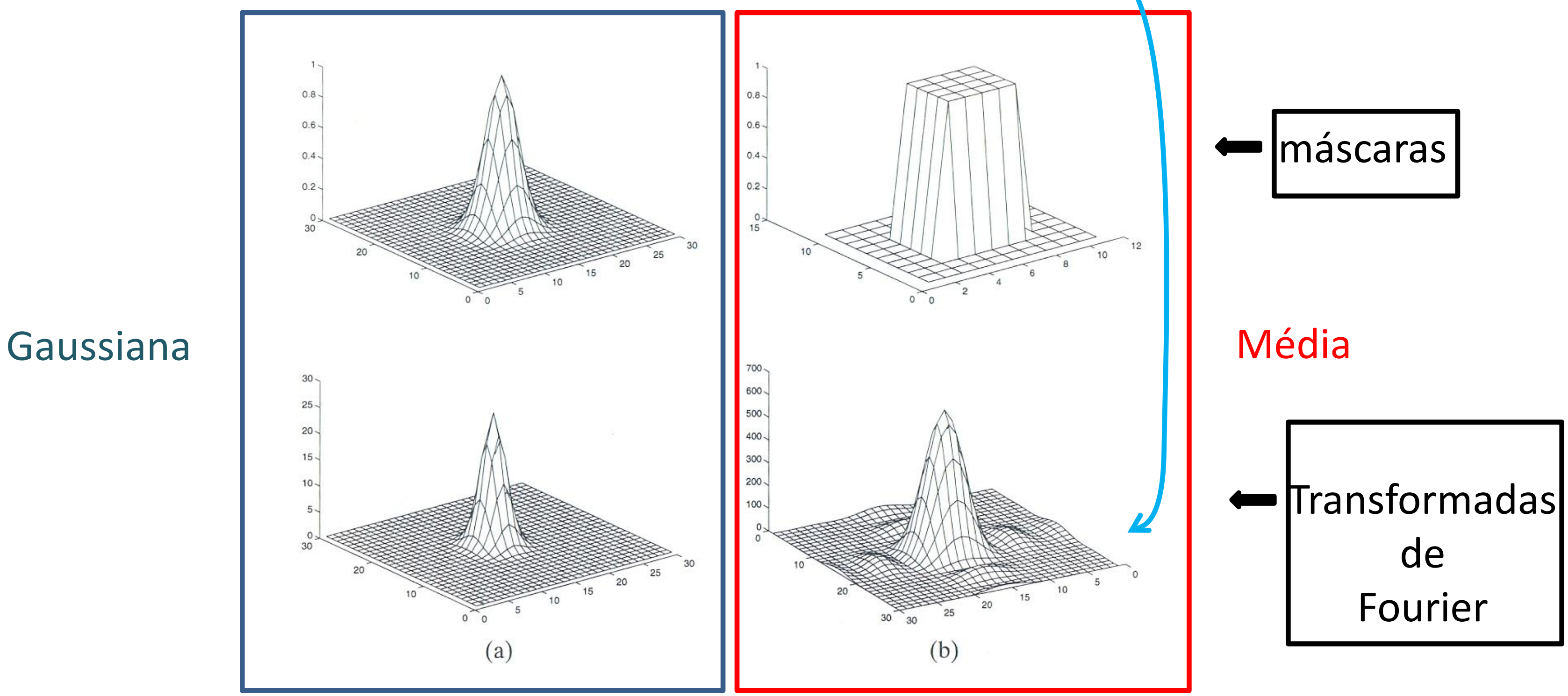
$$A_{avg} = \frac{1}{9} \cdot \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$



Tomando a média de $m^2$ valores em torno de (i, j), o desvio padrão do ruído é dividido por $\sqrt{m^2} = m$.

- A resposta de frequência de um filtro média 1-D de largura 2W é

$$\sin c(\omega) = \frac{2.\mathrm{sen}(\omega.W)}{\omega}$$




a) Gráfico de uma máscara Gaussiana de largura 5 (acima) e sua transformada de Fourier (abaixo);
b) O mesmo para uma máscara do filtro-média.

- O Filtro Média é um filtro passa-baixa

- Limitações da Filtragem por Média

1. Freqüências próximas às do ruído são perdidas ➔ imagem perde contorno.

2. Ruído de Impulso é atenuado, mas não eliminado.

3. Os picos secundários da T.F. do filtro média deixam ruído na imagem filtrada.

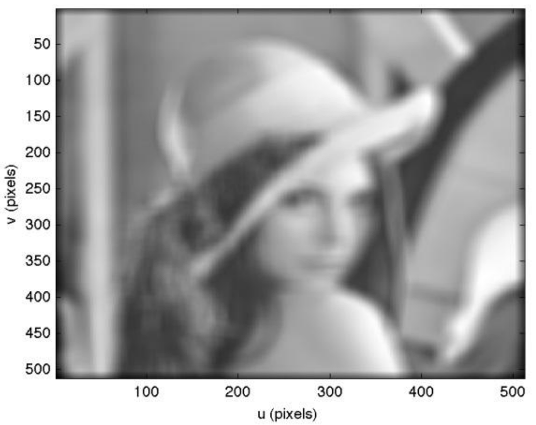




Imagem original (512x512)

Filtro média (21x21)

- **ATENUAÇÃO GAUSSIANA**

Imagem corrompida com ruído gaussiano e depois foi aplicada máscara gaussiana

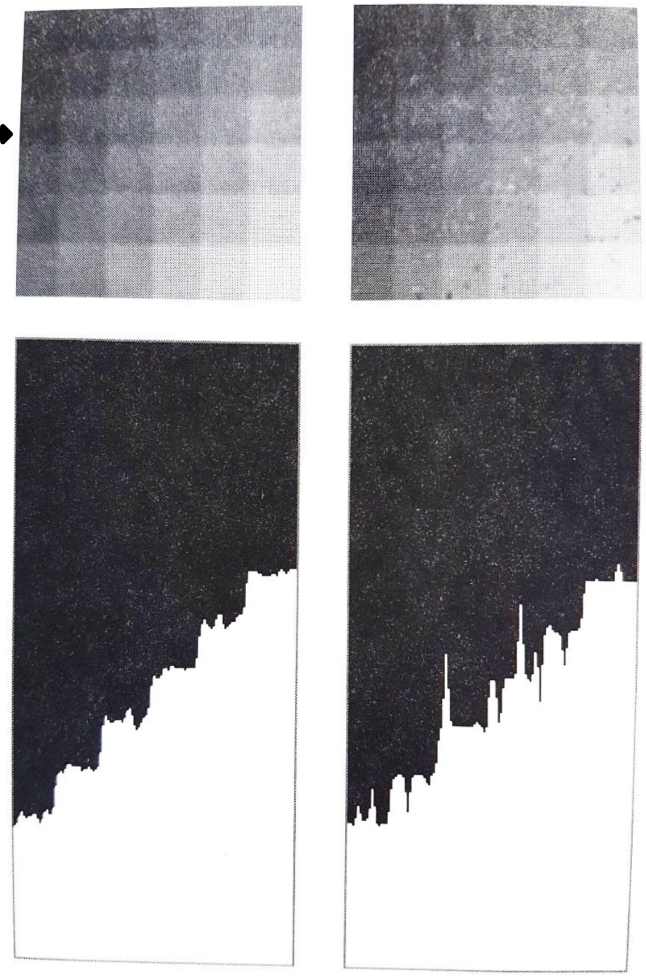
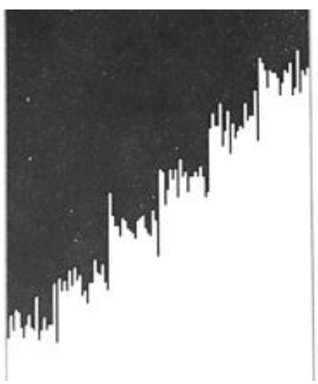


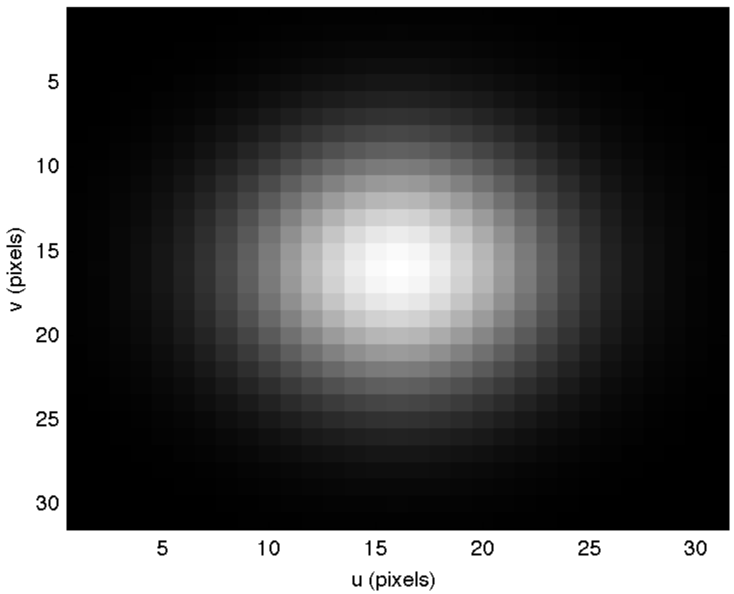
Imagem corrompida com ruído sal-e-pimenta e depois foi aplicada máscara gaussiana

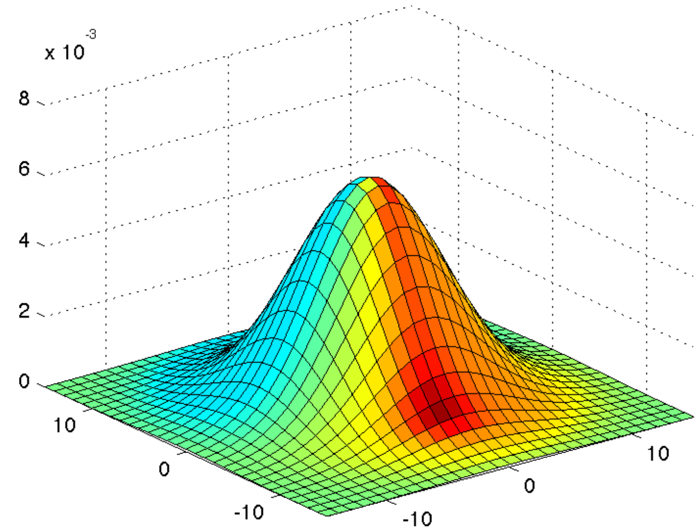


Perfil de Imagem sintética com ruído gaussiano original

Perfil de Imagem sintética com ruído impulsivo original

a) Resultados da aplicação de filtragem Gaussiana (máscara com 5 píxeis, $\sigma = 1$) à imagem do tabuleiro de xadrez corrompida por ruído Gaussiano e perfil de valor de cinza ao longo de uma linha; b) mesmo para a imagem do tabuleiro de xadrez corrompida por ruído sal-e-pimenta.




Máscara gaussiana

Comportamento da máscara gaussiana no domínio da frequência:

- A FT permanece gaussiana, sem picos secundários
- Funciona melhor como filtro passa-baixa do que o filtro média (veja fig. anterior)

- Separabilidade da máscara Gaussiana

$$I_G = I * G =$$

$$= \sum_{h=-m/2}^{m/2} \sum_{k=-m/2}^{m/2} G(h,k).I(i-h, j-k)$$

$$= \sum_{h=-m/2}^{m/2} \sum_{k=-m/2}^{m/2} e^{-\frac{h^2+k^2}{2.\sigma^2}}.I(i-h, j-k) =$$

$$= \sum_{h=-m/2}^{m/2} e^{-\frac{h^2}{2.\sigma^2}} \sum_{k=-m/2}^{m/2} e^{-\frac{k^2}{2.\sigma^2}}.I(i-h, j-k) =$$

Convolução de Imagem I com Máscara Gaussiana 2-D

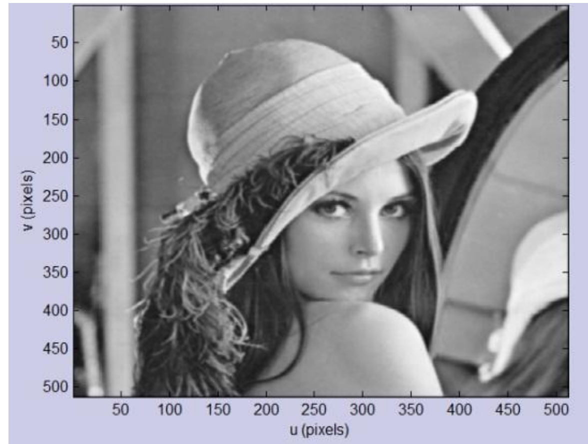
=

Convolução de todas as linhas de I com máscara Gaussiana 1-D

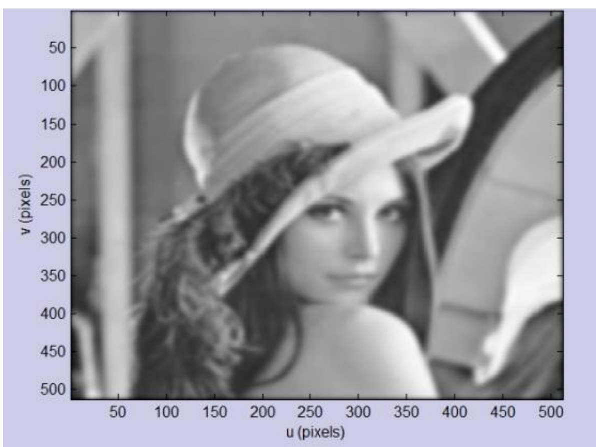
+

Convolução de todas as colunas de I com máscara Gaussiana 1-D

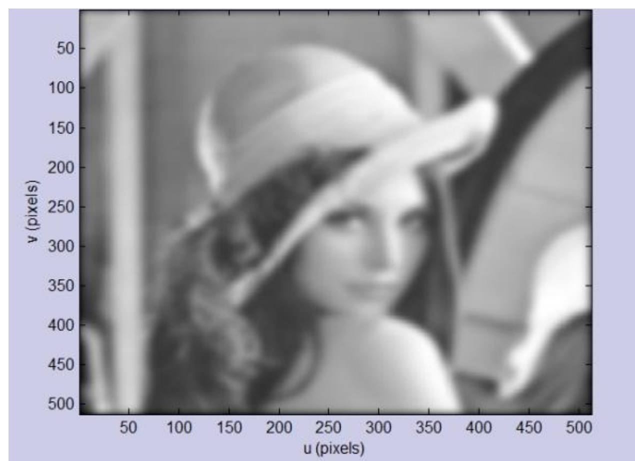

Imagem original




Filtragem gaussiana (σ = 1)




Filtragem gaussiana (σ = 3)




Filtragem gaussiana (σ = 5)

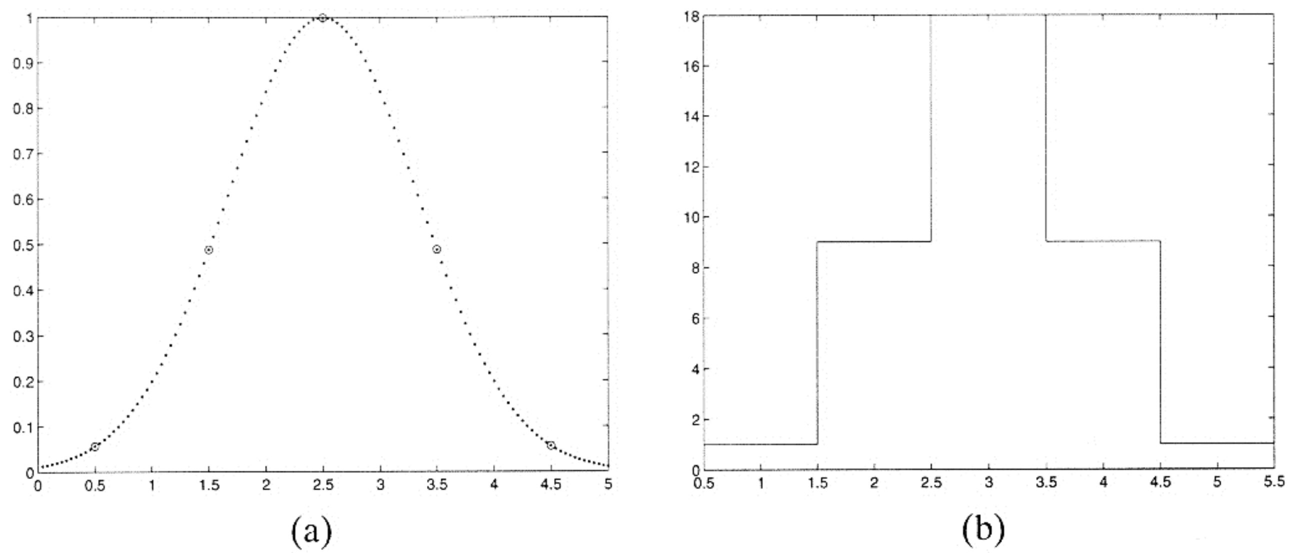
## ALGORITMO SEPAR_FILTER

Convolução de I com Máscara Gaussiana m x m (2-D), com $\sigma = \sigma_G$

1. Construa uma máscara Gaussiana, g, 1-D, de largura *m*, com $\sigma_g = \sigma_G$
2. Convolva cada linha de I com g , produzindo nova imagem $I_r$.
3. Convolva cada coluna de $I_r$ com g.

A máscara gaussiana pode ser convertida em apenas números inteiros

- Normalize a máscara real, fazendo seu menor valor = 1
- Arredonde os resultados e divida pela soma dos valores




a) Amostras Gaussianas (1-D) e reais (círculos) para máscara 5x5; b) esboço de máscara inteira correspondente.

## Algorítmo INT_GAUSS_KER

Máscara inteira $G_i$

1. Compute a máscara G(h, k) do mesmo tamanho de $G_i$ ; faça $g_{min} = G(0,0)$ ser o menor valor de G.

2. Determine o fator de normalização $f = 1/g_{min}$

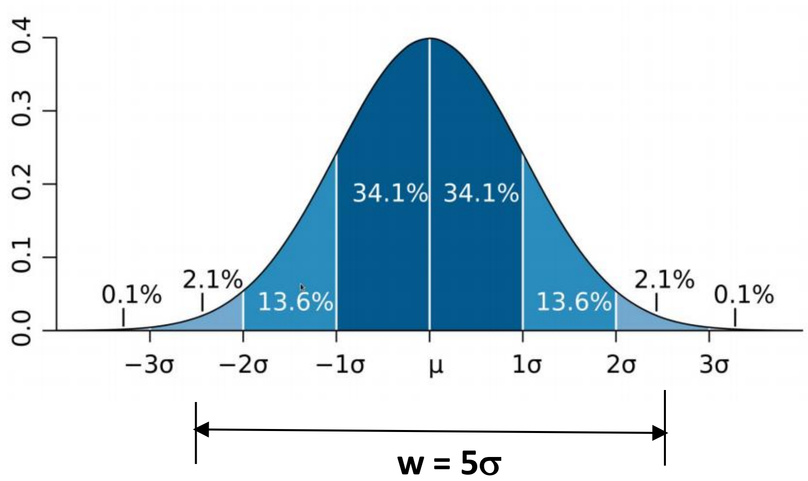
3. Compute os valores do filtro não-normalizado $G_i(h,k) = \text{int}[f.G(h,k)]$, onde *int* indica o inteiro mais próximo.

- Construção de Máscaras Gaussianas

  - Podem ser construídas em 1-D

    Conhecido $\sigma_G$ ➔ largura w?

    Conhecida largura (w) ➔ $\sigma$ ?




Relação aceitável w = 5.σ ➔ 98,76% da área

Na máscara real ➔ p/ máscara de w = 3 ➔ $\sigma_3 = \frac{3}{5} = 0{,}6$ pixel

$w = 5$ ➔ $\sigma_5 = \frac{5}{5} = 1$ pixel

Em geral $\sigma_w = \frac{w}{5}$

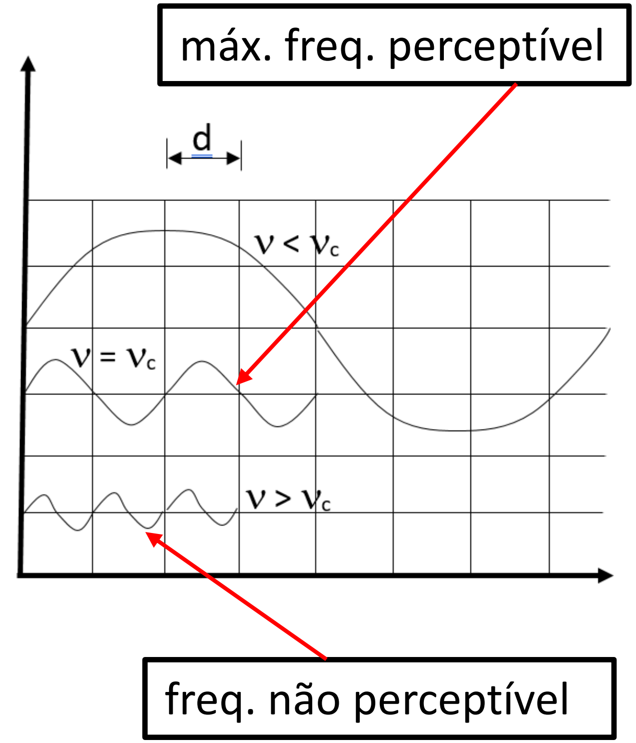
- São as amostras realmente Gaussianas ?

Para frequências de amostragem d = 1 pixel

$\nu$ = componente de frequência da imagem
$\nu_c$ = máximo componente de frequência perceptível

se $$|\nu| > \nu_c = 2.\pi.(\frac{1}{2d}) = \pi$$

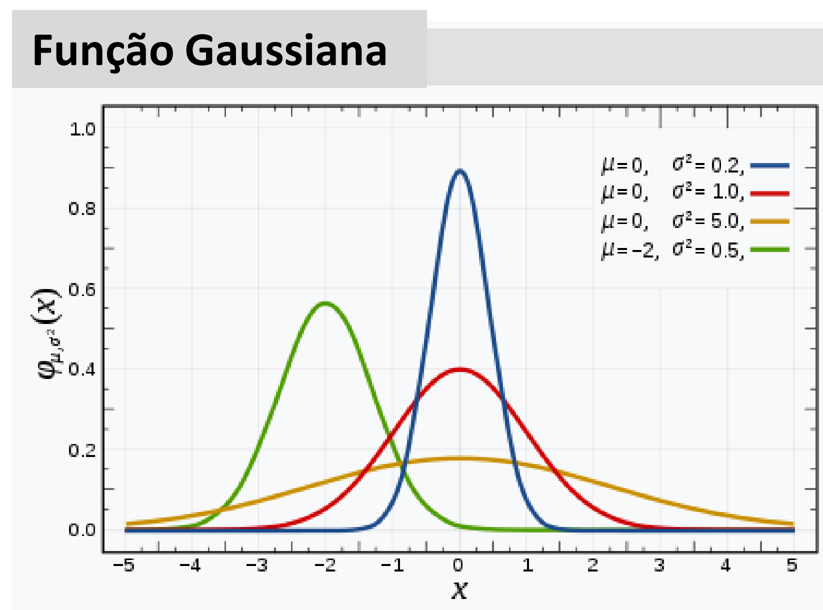
, então as frequências $\nu$ são perdidas.




A T.F. da Gaussiana

$$g(x,\sigma) = e^{-\frac{x^2}{2.\sigma^2}} \xrightarrow{\text{T.F.}} g(x,\sigma')$$

é uma gaussiana g(x, σ´) em que $\sigma' = \frac{1}{\sigma}$

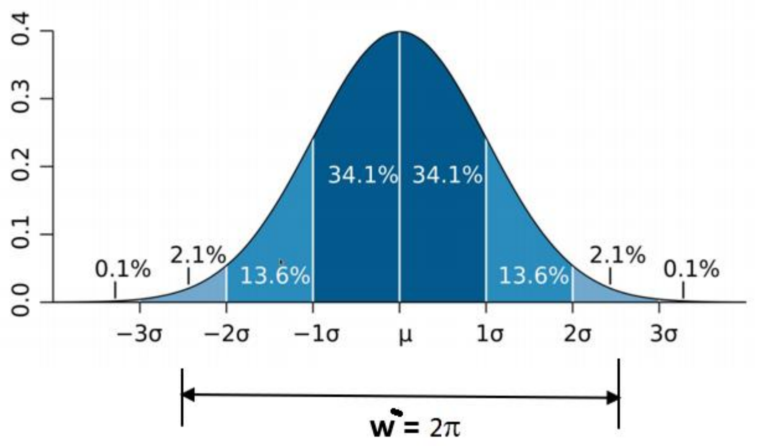


Mantendo 98,86% da energia entre os intervalos [-π, π] *(intervalo de frequências perceptíveis)*

$$w' = 5.\sigma' = \frac{5}{\sigma} \leq 2.\pi = \text{largura no domínio da frequência}$$

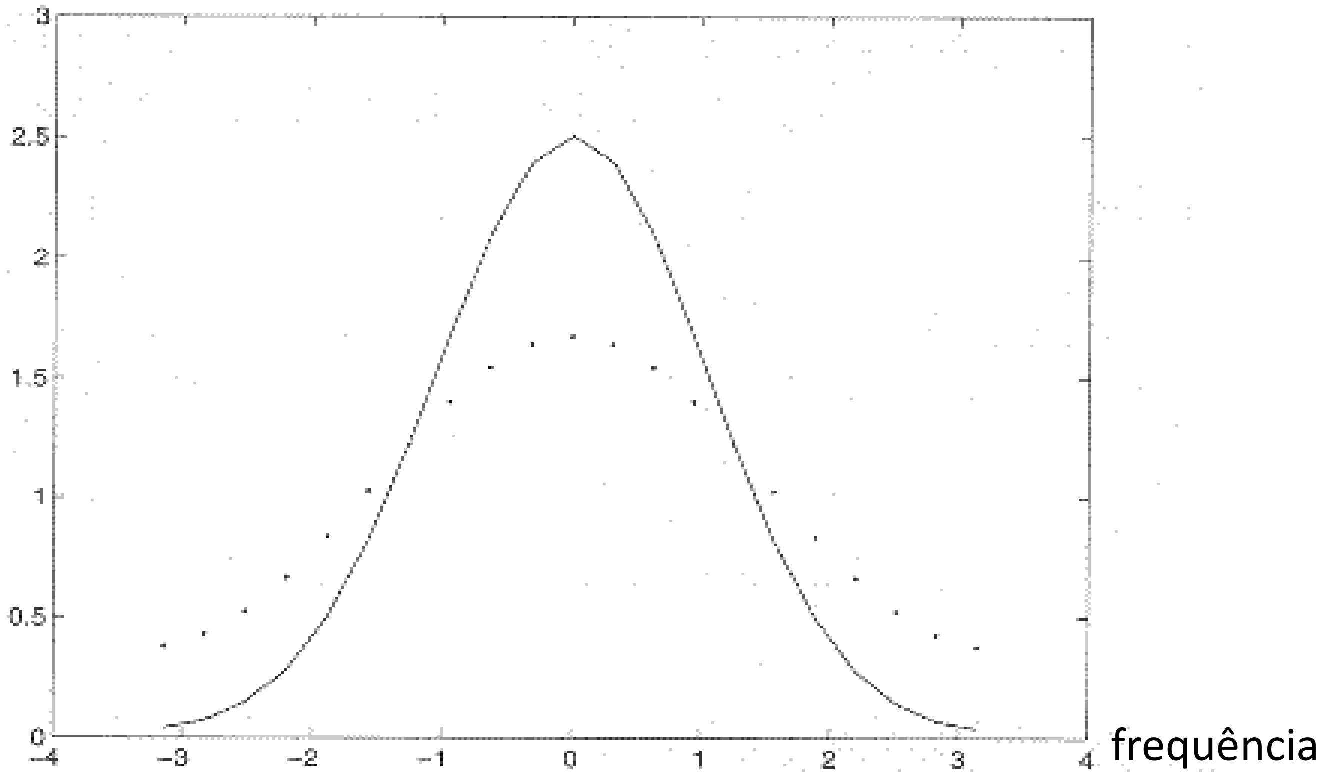
ou

$$\sigma \geq \frac{5}{2.\pi} = 0{,}796 \;\cong\; 0{,}8$$




Se w = 3 ➔ σ = 0,6
e w = 5 ➔ σ = 1

➔ menor largura da máscara gaussiana é 5.




Transformadas de Fourier de duas amostras Gaussianas, para w = 3 ($\sigma$ = 0,6, linha pontilhada) e w = 5 ($\sigma$ = 1, linha sólida). Uma parte menor da transformada correspondente a $\sigma$ = 1 é perdida entre $-\pi$ e $\pi$.

- Filtragem Gaussiana por Repetição de Filtragem por Média

Pelo Teorema do Limite Central

Convolver uma máscara de média 3 x 3 *n* vezes

=

Convolução de máscara Gaussiana de $\sigma = \sqrt{\frac{n}{3}}$

e $w = 3.(n+1) - n = 2.n + 3$

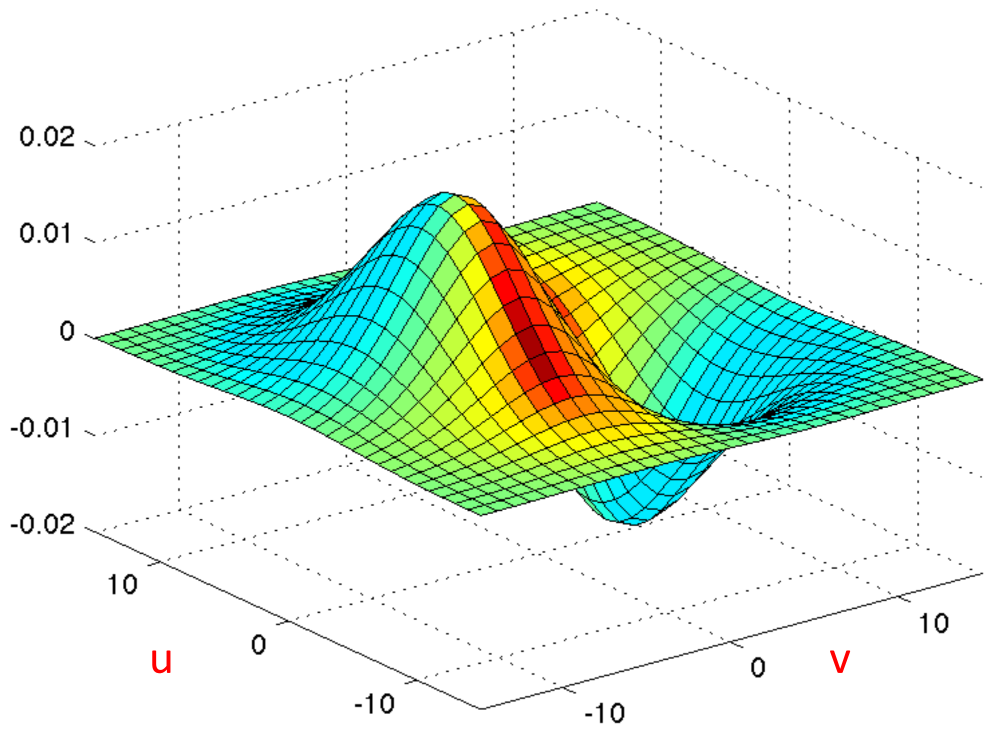
Ex: Máscara gaussiana de $\sigma=1$ e largura $w=9$

=

Máscara Média 3x3 aplicada 3 vezes.

**OUTROS FILTROS LINEARES**

Derivada da Gaussiana (DoG)

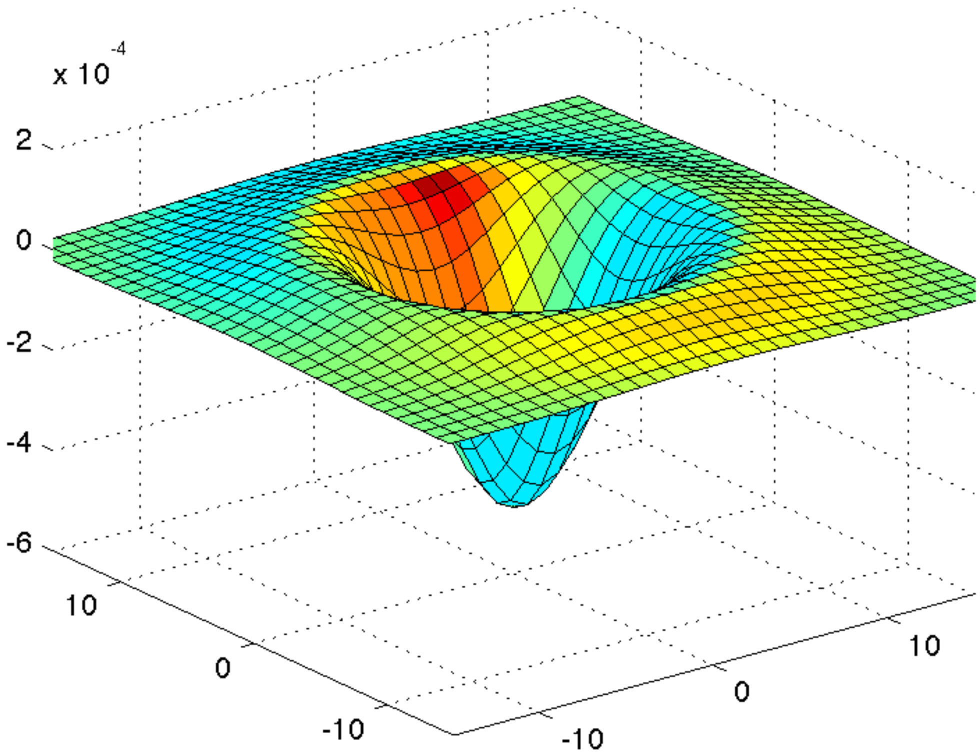



$$\nabla \mathbf{I} = \mathbf{D} \otimes \left(\mathbf{G}(\sigma) \otimes I\right) = \underbrace{\left(\mathbf{D} \otimes \mathbf{G}(\sigma)\right)}_{\text{DoG}} \otimes \mathbf{I}$$

$$\mathbf{G}_u(u, v) = -\frac{u}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}}$$

Utilizado para detecção de contornos

Laplaciano do Gaussiano (LoG)




$$\nabla^2 \mathbf{I} = \mathbf{L} \otimes \left(\mathbf{G}(\sigma) \otimes \mathbf{I}\right) = \underbrace{\left(\mathbf{L} \otimes \mathbf{G}(\sigma)\right)}_{\text{LoG}} \otimes \mathbf{I}$$

$$\mathbf{LoG}(u, v) = \frac{\partial^2 \mathbf{G}}{\partial u^2} + \frac{\partial^2 \mathbf{G}}{\partial v^2}$$

$$= \frac{1}{\pi \sigma^4} \left( \frac{u^2 + v^2}{2\sigma^2} - 1 \right) e^{-\frac{u^2 + v^2}{2\sigma^2}}$$

Utilizado para detecção de máximos em contornos

- **FILTRAGEM NÃO-LINEAR**

Problemas com filtros lineares:

1. Imagem borrada
2. Localização de características comprometida
3. Picos secundários no domínio da frequência
4. Supressão incompleta de ruídos de impulso

- Filtros gaussianos resolvem o 3º item
- Filtros não-lineares resolvem os outros itens
- Filtros não-lineares não podem ser modelados por convolução

- Filtros de Mediana

---

Algorítmo MEDIAN-FILTER

I = imagem de entrada ; Im = imagem filtrada ; n = valor ímpar

1. Compute o valor mediano m(i,j) dos valores em uma vizinhança n x n de (i, j)

$$I(i+h, j+k) \quad , \quad h,k \in \left[-\frac{n}{2}, \frac{n}{2}\right]$$

n/2 = divisão inteira

2. Atribua Im (i,j) = m(i,j)

Imagem corrompida com ruído **gaussiano** antes e depois

Imagem corrompida com ruído **impulso** antes e depois

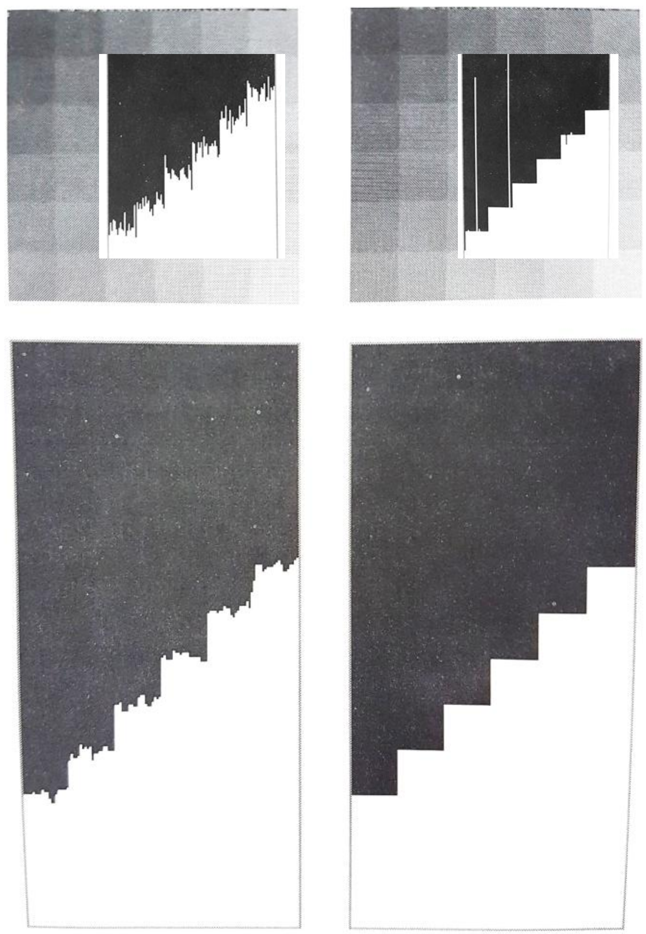
Imagem corrompida com ruído **gaussiano** antes e depois

Imagem corrompida com ruído **impulso** antes e depois

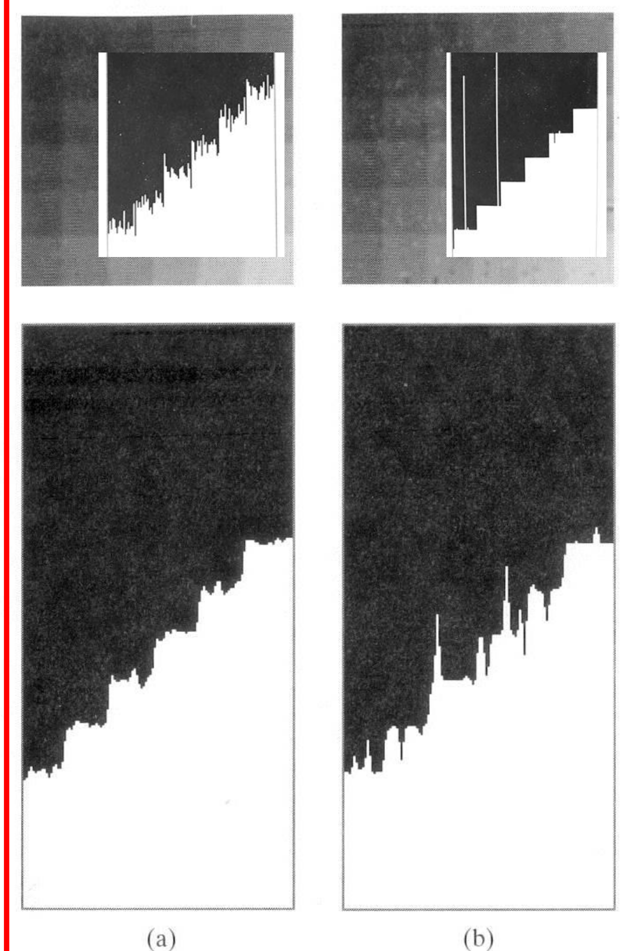
antes

depois

**Aplicação do Filtro Mediana**



Aplicação dos filtros Mediana (w=3) e Gaussiano às imagens do tabuleiro de xadrez corrompidas por ruído gaussiano e por ruído impulsivo.

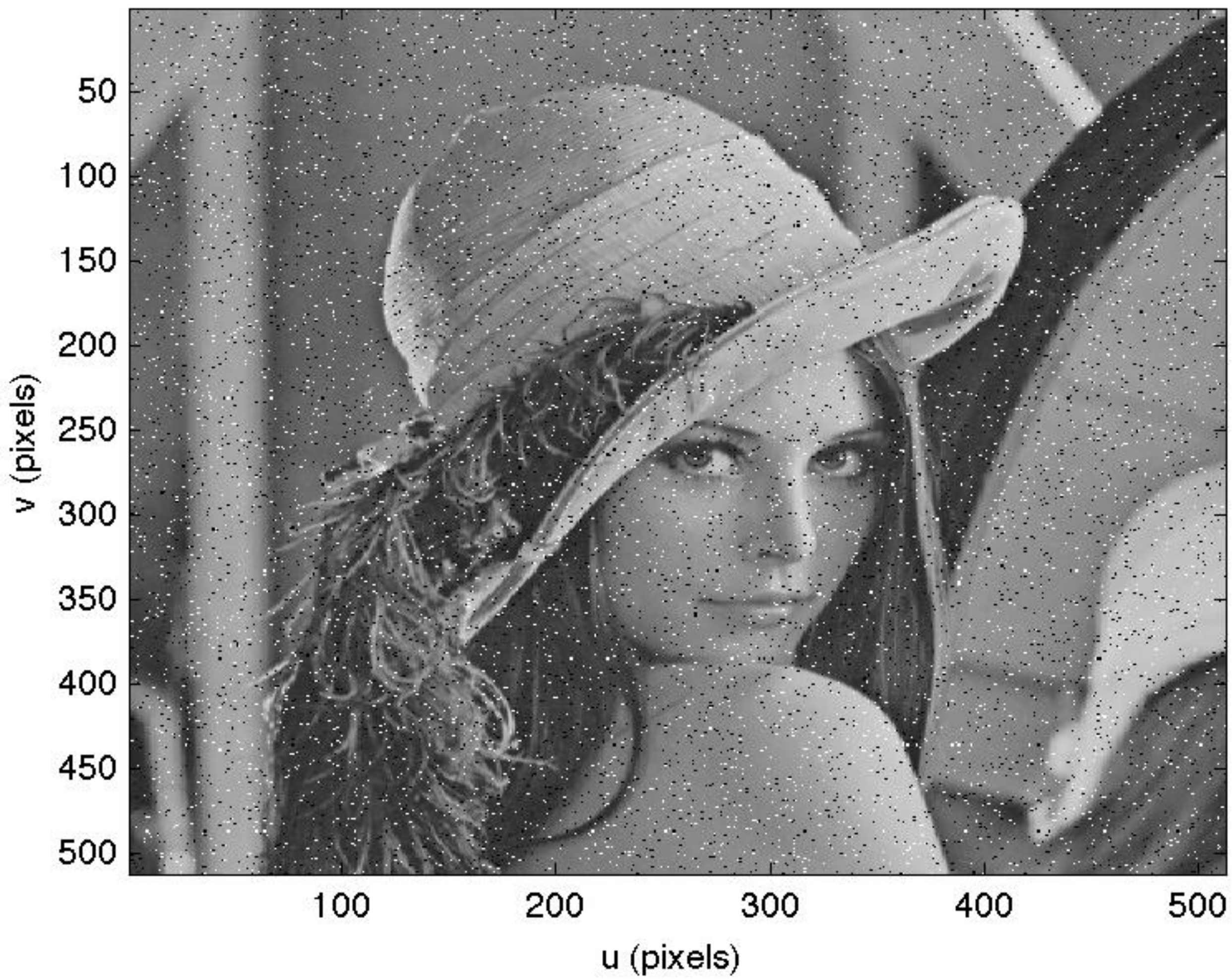



antes

depois

**Aplicação do Filtro Gaussiano**

Filtro Mediana preserva melhor as descontinuidades (p. ex. contornos).




Ruído Não-linear


Filtragem Não-linear

- Outros Filtros Não-Lineares
  - Filtro Mínimo
    - Substitui o pixel central pelo menor valor da vizinhança
      - Escurece a imagem
  - Filtro Máximo
    - Substitui o pixel central pelo maior valor da vizinhança
      - Clareia a imagem
  - Filtro Moda
    - Substitui o pixel central pelo valor mais frequente da vizinhança
      - Uniformiza a imagem

# FILTRAGEM NA FREQUÊNCIA

$$F(u,v) = \sum\sum f(x,y)\,.\,e^{-j2\pi\left(\frac{ux}{M}+\frac{vx}{N}\right)}$$

$F(u,v)$ é a Transformada de Fourier da imagem.
f$(x,y)$ é a imagem no domínio espacial.
$M$ e $N$ são as dimensões da imagem.

$$f(x,y) = \frac{1}{MN}\sum\sum F(u,v).\,e^{j2\pi\left(\frac{ux}{M}+\frac{vx}{N}\right)}$$

A filtragem na frequência é realizada multiplicando-se o espectro da imagem, $F(u,v)$, pela máscara do filtro, $H(u,v)$ , pixel sobre pixel

# FILTRAGEM NA FREQUÊNCIA

Imagem original: $f(x,y)$

FFT 2D

Mesmo número de células e pixels

Espectro da Imagem: $F(u,v)$

$$F(u,v) = \sum \sum f(x,y) \, . \, e^{-j2\pi\left(\frac{ux}{M}+\frac{vx}{N}\right)}$$

$. \; H(u,v)$ ← Máscara mesmo tamanho da imagem

Espectro Filtrado: $G(u,v)$

$$G(u,v) = F(u,v).\,H(u,v)$$

FFT Inversa 2D

Imagem Filtrada: $g(x,y)$

$$g(x,y) = \frac{1}{MN} \sum \sum G(u,v).\, e^{j2\pi\left(\frac{ux}{M}+\frac{vx}{N}\right)}$$

# FILTRAGEM NA FREQUÊNCIA

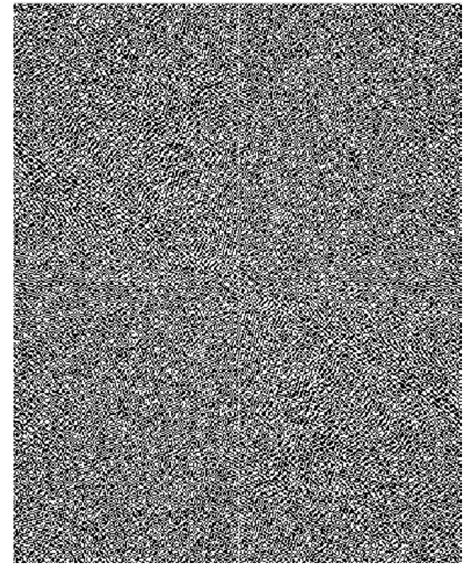
$f(x, y)$

430 x 347

**Filtro passa alta**

FFT 2D

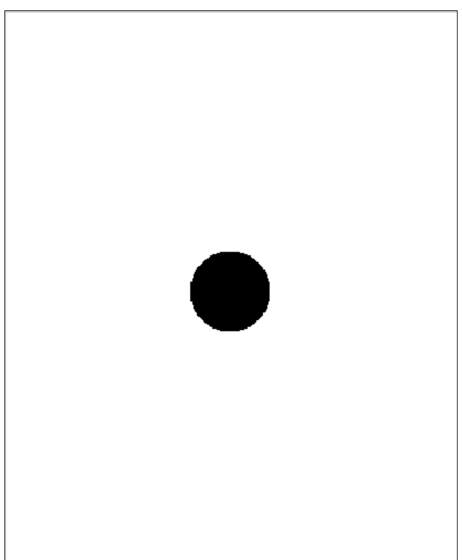
$\longrightarrow$



$F(u, v)$

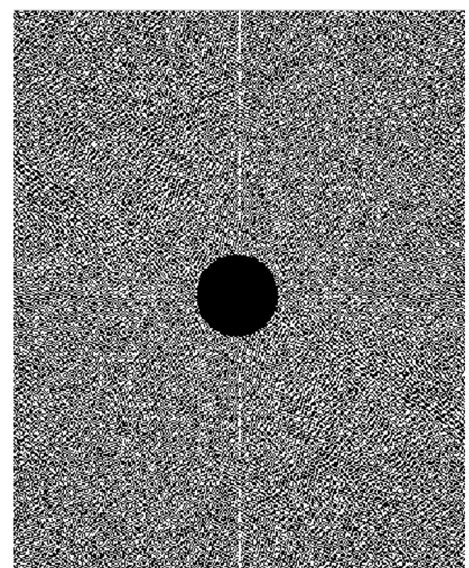
$H(u, v)$

Raio = 30



$F(u, v). H(u, v)$

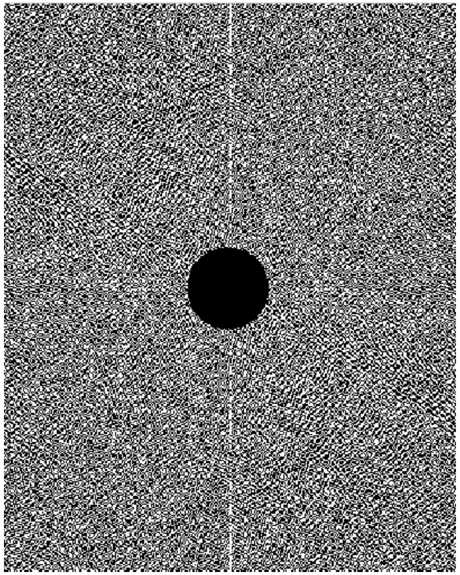
$\longrightarrow$



$G(u, v)$

# FILTRAGEM NA FREQUÊNCIA

$G(u, v)$



**Filtro passa alta**

FFT Inversa2D

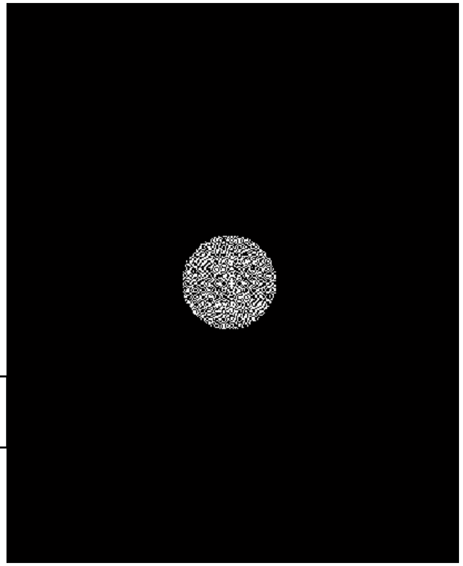
$g(x, y)$

$f(x, y)$

# FILTRAGEM NA FREQUÊNCIA

$G(u, v)$

Raio = 50



**Filtro passa baixa**

FFT Inversa2D

$\Longrightarrow$

$g(x, y)$

$f(x, y)$

# EXEMPLO DE FILTRAGEM UTILIZANDO CNN

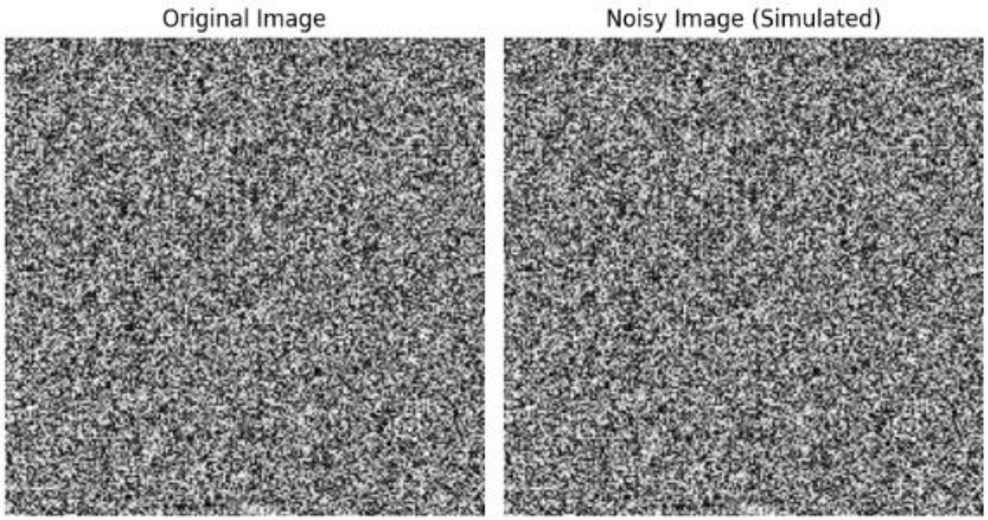



Imagem original + ruído gaussiano kernel (5x5) desvio padrão = 5

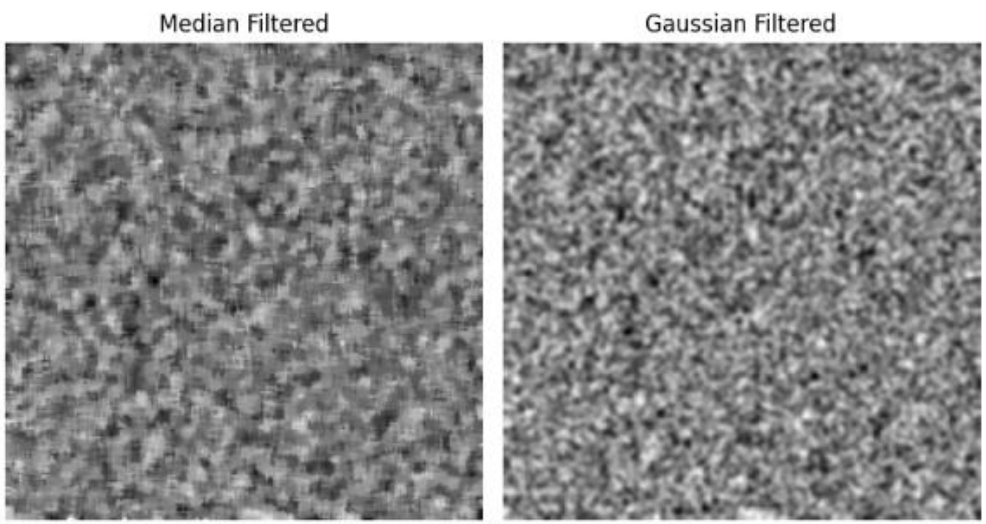



Imagem ruídos com aplicação de filtro mediana (5x5) e filtro gaussiano (5x5)

# EXEMPLO DE FILTRAGEM UTILIZANDO CNN

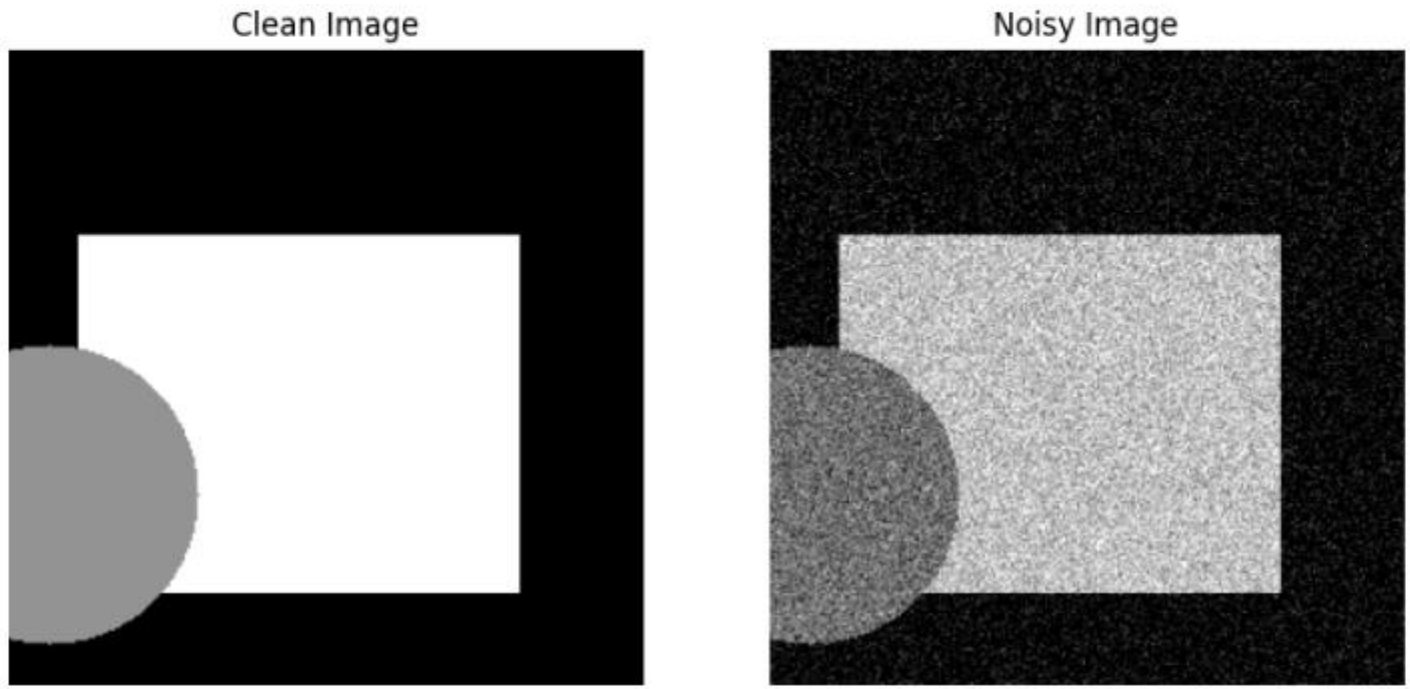
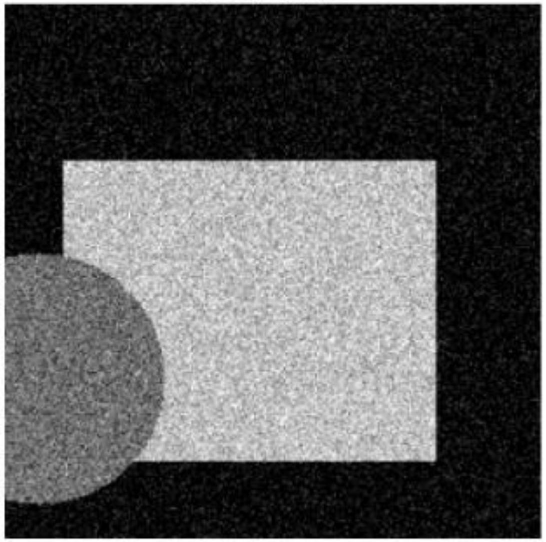



Imagem sintética e ruidosa com aplicação de ruído gaussiano

# EXEMPLO DE FILTRAGEM UTILIZANDO CNN

Noisy Image

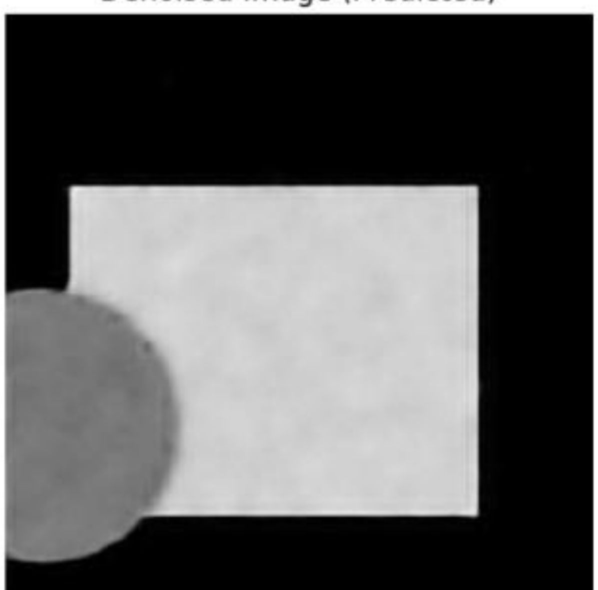


Clean Image

25 épocas

90 segundos

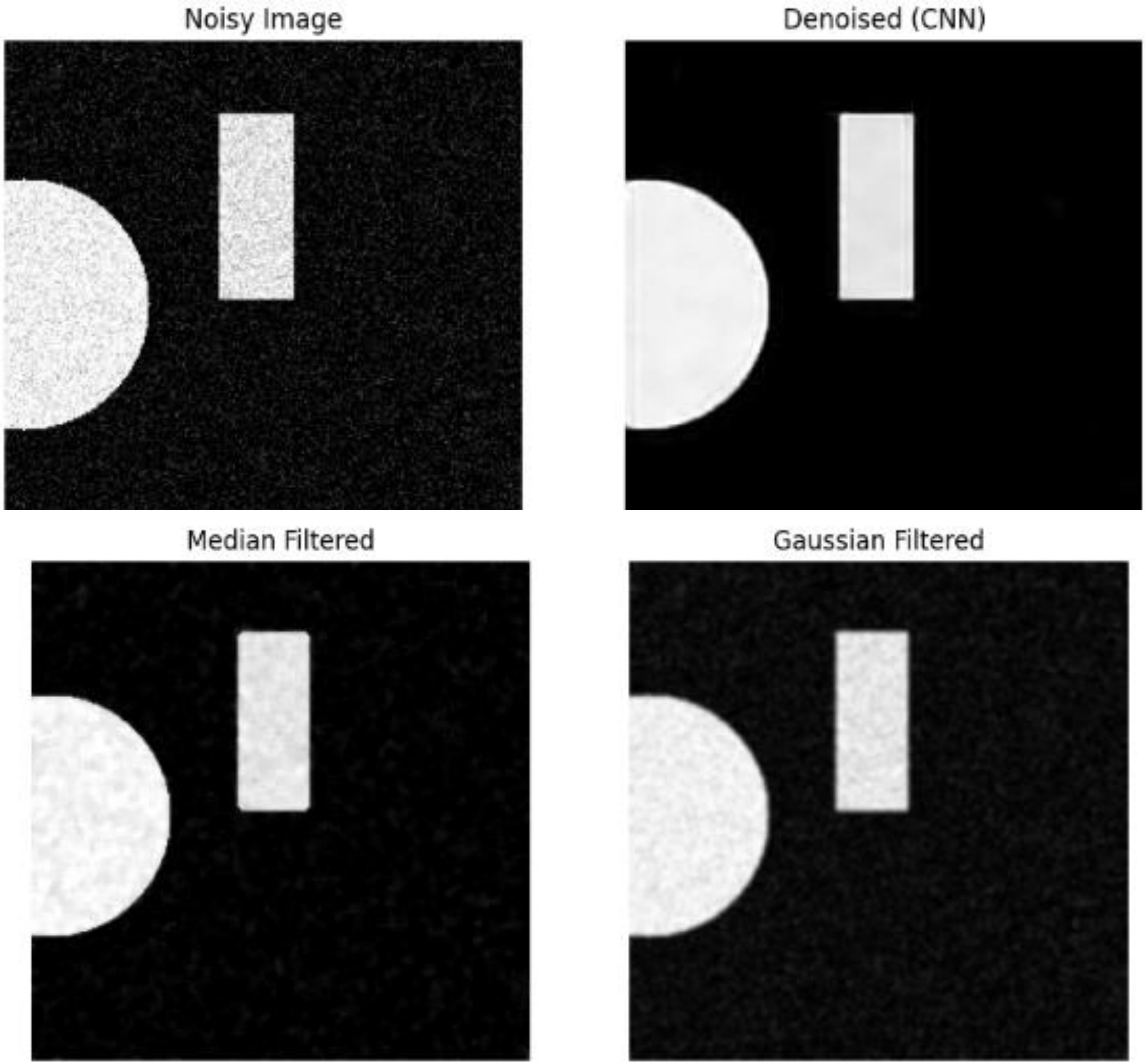
Denoised Image (Predicted)



| Layer (type) | Output Shape | Param # |
|---|---|---|
| input_layer (InputLayer) | (None, 256, 256, 1) | 0 |
| conv2d (Conv2D) | (None, 256, 256, 32) | 320 |
| max_pooling2d (MaxPooling2D) | (None, 128, 128, 32) | 0 |
| conv2d_1 (Conv2D) | (None, 128, 128, 64) | 18,496 |
| max_pooling2d_1 (MaxPooling2D) | (None, 64, 64, 64) | 0 |
| conv2d_2 (Conv2D) | (None, 64, 64, 64) | 36,928 |
| up_sampling2d (UpSampling2D) | (None, 128, 128, 64) | 0 |
| conv2d_3 (Conv2D) | (None, 128, 128, 32) | 18,464 |
| up_sampling2d_1 (UpSampling2D) | (None, 256, 256, 32) | 0 |
| conv2d_4 (Conv2D) | (None, 256, 256, 1) | 289 |

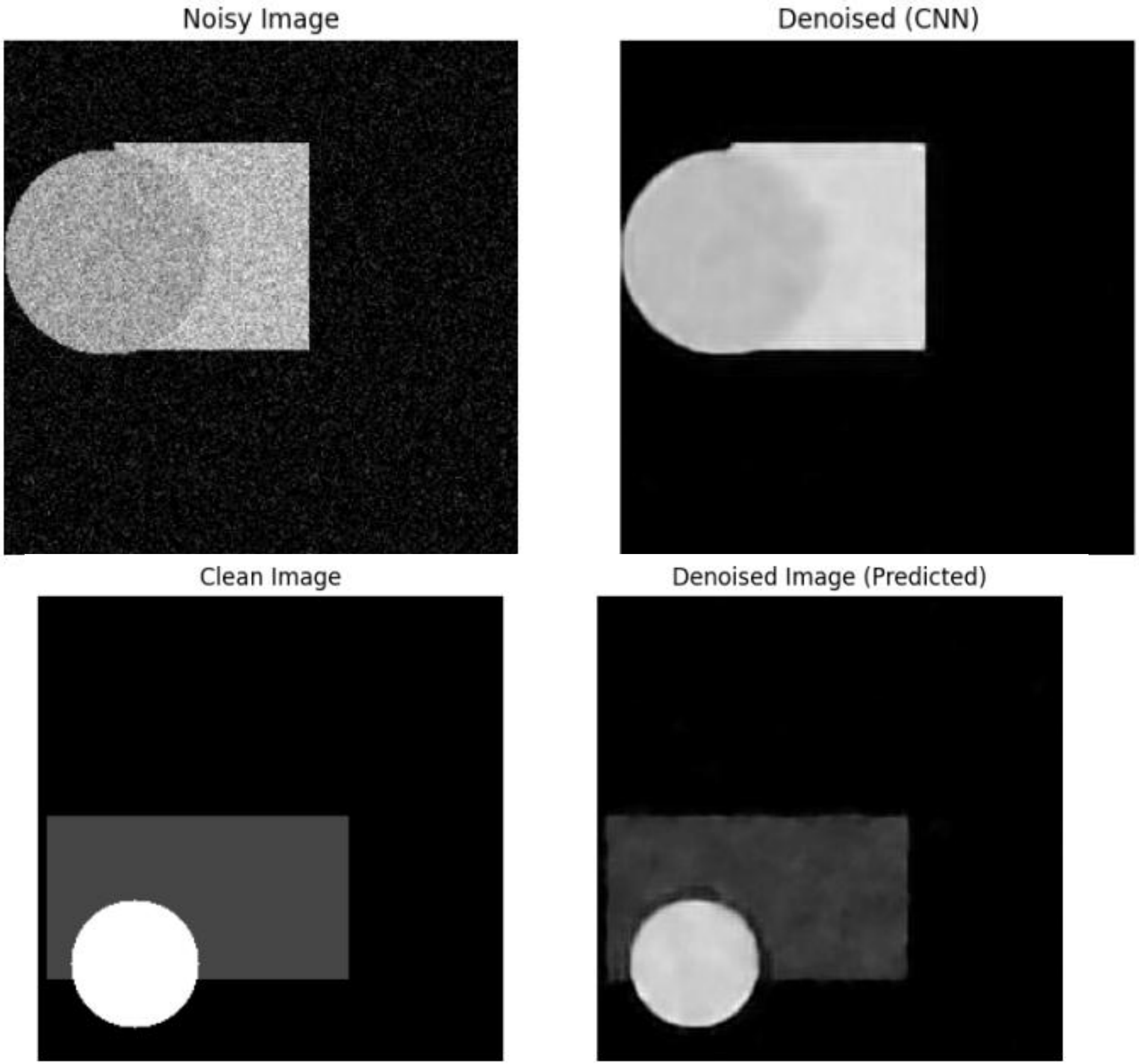
```
Total params: 74,497 (291.00 KB)
Trainable params: 74,497 (291.00 KB)
Non-trainable params: 0 (0.00 B)
```

# EXEMPLO DE FILTRAGEM UTILIZANDO CNN




Mesma rede anterior aplicada em outra imagem sintética

# FILTRAGEM UTILIZANDO CNN




Mesma rede anterior aplicada em outra imagem sintética

Imagem filtrada ainda apresenta ruído, apesar de ser mais uniforme