Processamento Digital de Imagens

LAB 03 - Análise de filtros

Bianca Xavier // Luana Silva

Professor Fernando Sales

Descrição do problema

Implementar alguns filtros para remover ruídos em diferentes graus de uma certa imagem de referência. Avaliar o desempenho de cada um através de métricas de desempenho. Fazer a comparação entre os filtros através dessas métricas com tabelas e gráficos.

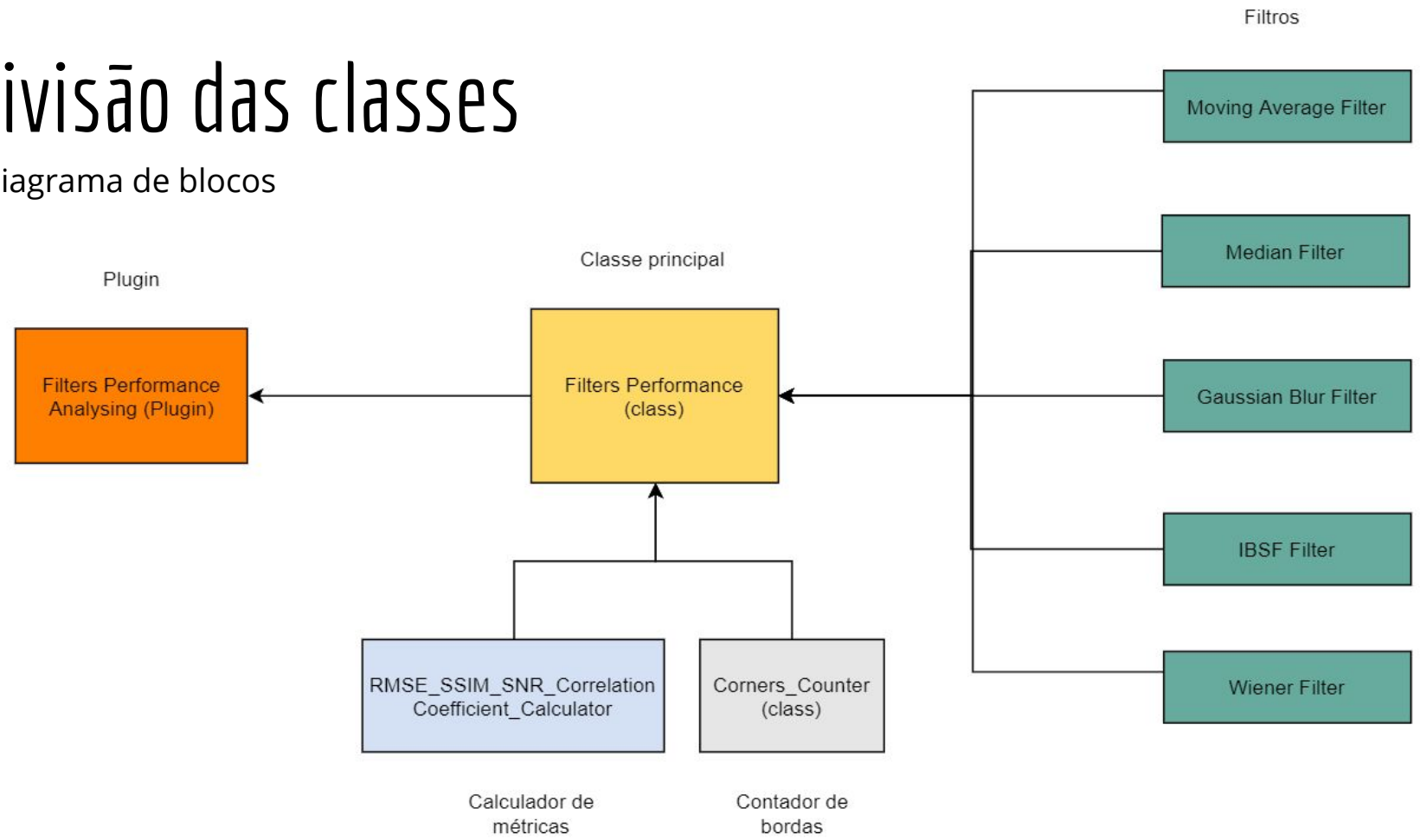
Breve descrição do algoritmo implementado

Classes implementadas:

- Filtros: recebem uma imagem ruidosa, filtram e retornam a imagem filtrada.
- Métricas: pegam dois vetores, e calculam o RMSE, SSIM, SNR, R entre elas;
- Corners: Calcula o número de pontos de junção de uma imagem.
- Classe principal: Executa os filtros e as métricas para n imagens ruidosas dadas, salva as imagens filtradas e gera as tabelas de métricas de cada filtro.
- Plugin: Chama a classe principal, pede para o usuário definir o tamanho das janelas dos filtros.

Divisão das classes

Diagrama de blocos



Breve descrição do algoritmo implementado

Classe Principal:

1. Abre a imagem de referência;
2. Inicia duas tabelas: tabela 1 e 2;
3. Cria um laço, com 1 iteração para cada imagem ruidosa:
    a. Abre uma imagem ruidosa;
    b. Aplica o filtro nela;
    c. Recebe e salva a imagem filtrada;
    d. Compara a imagem filtrada com a original ruidosa através das métricas de comparação;
    e. Adiciona essas métricas na tabela 1;
    f. Compara a imagem filtrada com a imagem de referência
    g. Adiciona as métricas na tabela 2

Breve descrição do algoritmo implementado

Classe principal:

No fim do laço teremos todas imagens filtradas salvas e as tabelas semiprontas, sem a média e o desvio padrão das medidas.

1. Calcula a média e o desvio padrão para cada coluna da tabela 1;
2. Adiciona-os no final da tabela 1;
3. Salva e mostra a tabela 1;
4. Repete os últimos 3 comandos para a tabela 2;

Breve descrição do algoritmo implementado

Filtros implementados:

- Median Filter;
- Gaussian Filter;
- Moving Average Filter;
- IBSF Filter;
- Wiener Filter.

Breve descrição do algoritmo implementado

Filtros: Median Filter

1. Recebe uma imagem e um tamanho da janela;
2. Pega os pixels da imagem e armazena em um vetor;
3. Cria um vetor vazio para receber os novos pixels;
4. Itera:
    a. Para cada pixel, pega uma janela de vizinhança MxN;
    b. Em cada janela, calcula a mediana desta e coloca o valor no vetor de novos pixels na posição relativa ao pixel central;
5. Configura todos os novos pixels na imagem original, que se torna a filtrada;
6. Retorna a filtrada.

Obs: Nos vizinhos de borda, considera-se uma janela menor. Os vizinhos que caem fora da imagem são desconsiderados.

Breve descrição do algoritmo implementado

Filtros: Gaussian Filter

1. Recebe uma imagem e um tamanho MxN de janela;
2. Pega o ImageProcessor da imagem;
3. Gera um kernel gaussiano 1D de tamanho M em um outro de tamanho N;
4. Convolve o ImageProcessor horizontalmente com o kernel M e depois verticalmente com o kernel N, assim tem-se a imagem filtrada;
5. Retorna a filtrada.

Breve descrição do algoritmo implementado

Filtros: Moving Average Filter

1. Recebe uma imagem e um tamanho da janela;
2. Pega os pixels da imagem e armazena em um vetor;
3. Cria um vetor vazio para receber os novos pixels;
4. Itera:
    a. Para cada pixel, pega uma janela de vizinhança MxN
    b. Em cada janela, calcula a média desta e coloca o valor no vetor de novos pixels na posição relativa ao pixel central
5. Configura todos os novos pixels na imagem original, que se torna a filtrada
6. Retorna a filtrada

Breve descrição do algoritmo implementado

Filtros: IBSF Filter

1. Recebe uma imagem e um tamanho da janela;
2. Usa o filtro mediano na imagem e recebe a imagem filtrada mediana;
3. Aplica a função de máxima entre a imagem ruidosa original e a filtrada mediana e obtêm-se um nova imagem, intermediária;
4. Aplica o filtro mediano agora na imagem intermediária com uma janela de 5x5 e recebe a imagem filtrada final;
5. Retorna a imagem filtrada final.

Breve descrição do algoritmo implementado

Filtros: Wiener Filter

1. Recebe uma imagem e um tamanho da janela;
2. Pega os pixels da imagem e armazena em um vetor;
3. Cria um vetor vazio para receber os novos pixels;
4. Pega uma roi de uma região homogênea da imagem ruidosa (pede ao usuário);
5. Calcula o coeficiente de variação qH da roi, que é uma constante;
6. Itera:
    a. Para cada pixel da imagem, pega uma janela MxN;
    b. Para cada janela, obtém o desvio padrão, a média, o coeficiente de variação e o alpha de cada janela.
    c. Verifica se alfa está no intervalo de 0 e 1.

Breve descrição do algoritmo implementado

Filtros: Wiener Filter

- Se alfa estiver ok:
  - Com os valores da janela, calcula-se o valor do novo pixel de acordo com a relação de Lee e atribui o valor ao vetor de novos pixels;
- Se alfa não estiver ok:
  - Pega o pixel original e atribui ao vetor de novos pixels (isto é, não altera o pixel da imagem).

5. Após varrer todos os pixels da ruidosa, configura a imagem original com os novos pixels, que se torna a filtrada;
6. Retorna a imagem filtrada.

Breve descrição do algoritmo implementado

Filtros: Wiener Filter

- Relação de Lee:

$$g(x,y) = \alpha \cdot f(x,y) + (1-\alpha)\bar{f}(x,y)$$

- Alfa:

$$\alpha = 1 - \frac{q_H^2}{q(x,y)^2} \qquad q(x,y) = \frac{\sigma(x,y)}{\mu(x,y)}$$

- Coeficiente de correlação:

$$q_H = \frac{\sigma_H}{\mu_H} \qquad \bar{f}(x,y) = \frac{1}{n(W(x,y))} \sum_{(i,j)\in W(x,y)} f(x,y)$$

Breve descrição do algoritmo implementado

Métricas implementadas:

1. RMSE;
2. SNR;
3. SSIM;
4. Coeficiente de Correlação;
5. Corners;

$$\sigma_{xy} = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \mu_x)(y_i - \mu_y).$$

$$\sigma_x = \left(\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \mu_x)^2\right)^{\frac{1}{2}}.$$

Breve descrição do algoritmo implementado

Métricas: RMSE

1. Recebe os pixels de duas imagens na forma de vetores unidimensionais, imagens A e B;
2. Pega o número total de pixel N, que é igual para as duas imagens.
3. Itera-se nos índices dos vetores de pixels:
   a. Para cada índice, pega o pixel da imagem A e o pixel da imagem B de mesmo índice:
   b. Pega a diferença entre eles, eleva ao quadrado e acrescenta na soma dos quadrados da diferença.
4. Após a iteração, divide a soma dos quadrados da diferença por N e isso é o RMSE.
5. Retorna o RMSE

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2}$$

Breve descrição do algoritmo implementado

Métricas: SSIM

1. Recebe os pixels de duas imagens na forma de vetores unidimensionais, imagens A e B;
2. Pega o número total de pixel N, que é igual para as duas imagens;
3. Itera-se nos índices dos vetores de pixels para pegar a soma dos quadrados e soma dos elementos de cada imagem e o produto das duas imagens;
4. Com as medidas anteriores, pode-se calcular desvio padrão, média, luminância, covariância, coeficiente de correlação e contraste. Com isso, obtém-se o SSIM.

Breve descrição do algoritmo implementado

Métricas: SSIM

- Luminância:

$$l(\mathbf{x},\mathbf{y}) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1}$$

- Coeficiente de correlação:

$$s(\mathbf{x},\mathbf{y}) = \frac{\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3}.$$

- Contraste:

$$c(\mathbf{x},\mathbf{y}) = \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2}$$

$$\mathrm{SSIM}(\mathbf{x},\mathbf{y}) = [l(\mathbf{x},\mathbf{y})]^{\alpha} \cdot [c(\mathbf{x},\mathbf{y})]^{\beta} \cdot [s(\mathbf{x},\mathbf{y})]^{\gamma}$$

Breve descrição do algoritmo implementado

Métricas: SNR

1. Recebe os pixels de duas imagens na forma de vetores unidimensionais, imagens A e B;
2. Pega o número total de pixel N, que é igual para as duas imagens;
3. Itera-se nos índices dos vetores de pixels para pegar a soma dos quadrados da imagem original (que é a potência do sinal) e soma dos quadrados das diferenças entre imagem original e ruídos (que é a potência de ruído);
4. Aplica os dois valores da fórmula do SNR;

$$P_{sinal} = \sum_{x=1}^{w} \sum_{y=1}^{h} [I_{original}(x,y)]^2$$

$$P_{ruído} = \sum_{x=1}^{w} \sum_{y=1}^{h} [I_{original}(x,y) - I_{filtrada}(x,y)]^2$$

$$SNR = 10 \log\left(\frac{P_{sinal}}{P_{ruído}}\right)$$

Breve descrição do algoritmo implementado

Métricas: Coeficiente de correlação (r)

1. Recebe os pixels de duas imagens na forma de vetores unidimensionais, imagens A e B;
2. Pega o número total de pixel N, que é igual para as duas imagens;
3. Itera-se nos índices dos vetores de pixels para pegar a soma dos quadrados e soma dos elementos de cada imagem e o produto das duas imagens;
4. Com esses valores, obtém-se média, desvio padrão, e assim, covariância e o r.

$$s(\mathbf{x}, \mathbf{y}) = \frac{\sigma_{xy} + C_3}{\sigma_x \sigma_y + C_3}.$$

Breve descrição do algoritmo implementado

Métricas: Corners

1. Pega duas cópias da imagem, imagem X e imagem Y;
2. Convolve X horizontalmente com um pré-filtro e após com a filtro de derivada, obtendo a derivada em X pré-filtrada;
3. Convolve Y verticalmente com um pré-filtro e após com a filtro de derivada, obtendo a derivada em Y pré-filtrada;
4. Pega o quadrado de X (A), o quadrado de Y (B), e pega X*Y (C);
5. Convolve A, B e C com o filtro gaussiano para obter a matriz estrutural com A’, B’, C’;
6. Pega os pixels de A’, B’, C’;
7. Itera nos pixels de A’, B’, e C’ para obter a função resposta Q;
8. itera nos elementos de Q, para identificar quais são candidatos a borda, avaliando o limiar e o máximo local, adicionando os candidatos em um conjunto de corners;

Breve descrição do algoritmo implementado

Métricas: Corners

1. Itera no conjunto de candidatos para verificar a proximidade deles, elementos muito próximos são descartados;
2. Pega o tamanho do conjunto de corners. E esse tamanho é a nossa métrica.
3. Retorna-o.

$$\boldsymbol{M} = \begin{pmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{pmatrix} = \begin{pmatrix} A & C \\ C & B \end{pmatrix}. \qquad \bar{\boldsymbol{M}} = \begin{pmatrix} A * H^{G,\sigma} & C * H^{G,\sigma} \\ C * H^{G,\sigma} & B * H^{G,\sigma} \end{pmatrix} = \begin{pmatrix} \bar{A} & \bar{C} \\ \bar{C} & \bar{B} \end{pmatrix}$$

Matriz estrutural

$$\begin{aligned} Q(u,v) &= \det(\bar{\boldsymbol{M}}) - \alpha \cdot \left(\operatorname{trace}(\bar{\boldsymbol{M}})\right)^2 \\ &= (\bar{A}\bar{B} - \bar{C}^2) - \alpha \cdot (\bar{A} + \bar{B})^2 \end{aligned}$$

Função resposta (CRF)

Problemas encontrados durante a execução

- Erros básicos de programação;
- Dificuldades:
  - Implementação dos Corners;
  - Filtro Wiener;
  - Gaussian Blur;

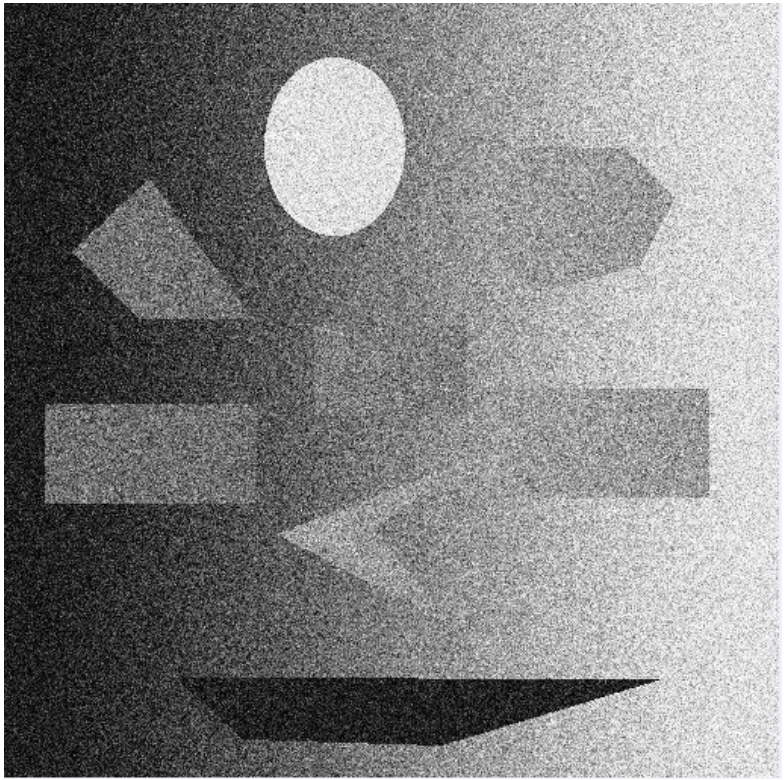
Limitações do algoritmo

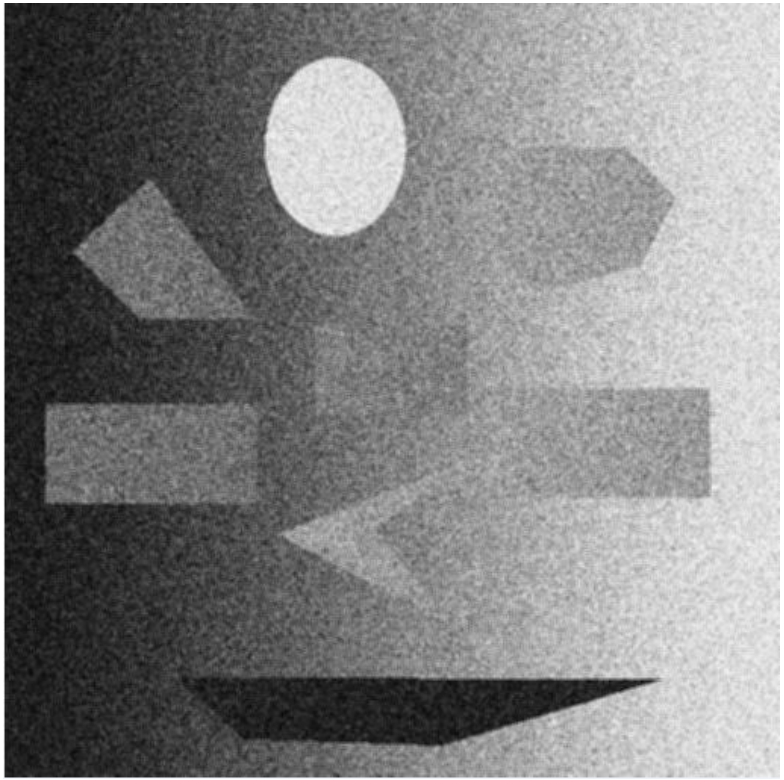
Pouca interação com o usuário, isto é:

- O algoritmo não possibilita que:
  - escolha os filtros que vão ser analisados;
  - escolha as imagens ruidosas.

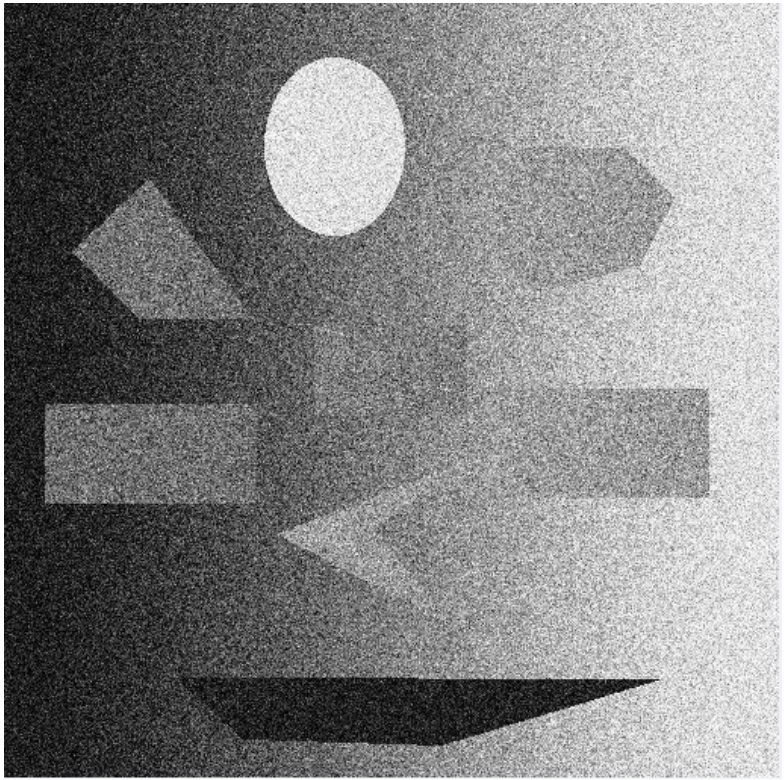
Isso é uma limitação porque a aplicação fica restrita a um conjunto de imagens, mas pode ser modificado para ter esse recurso. Não foi feito isto por questão de tempo, mas é uma sugestão de melhoria.

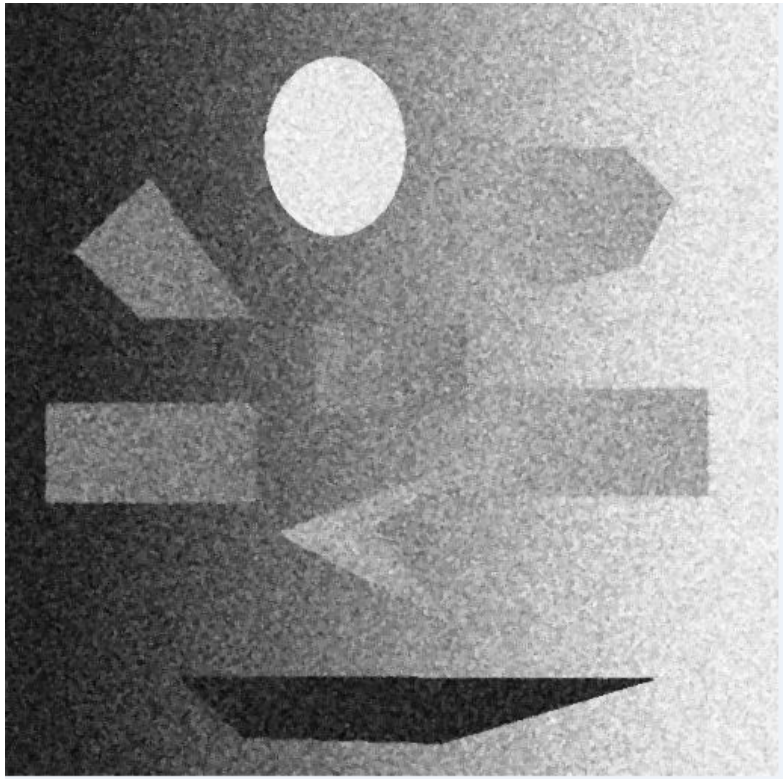
Moving Average 3x3



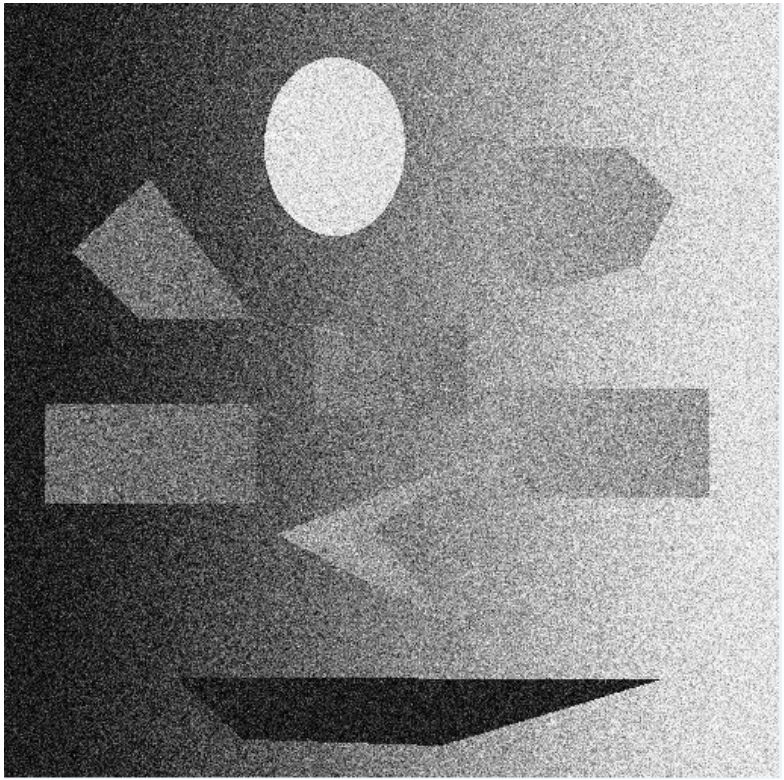


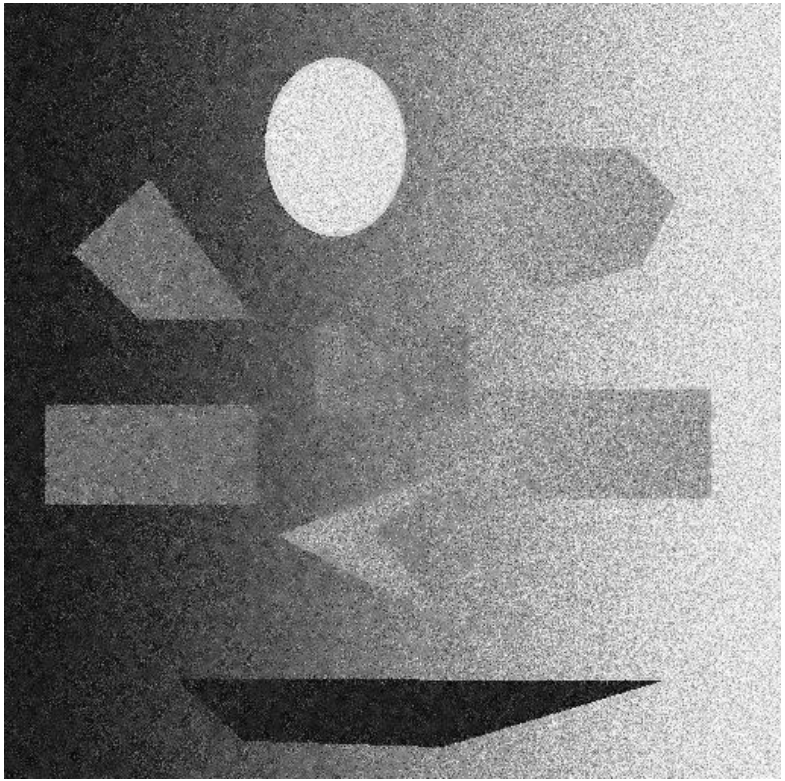
Median 5x5



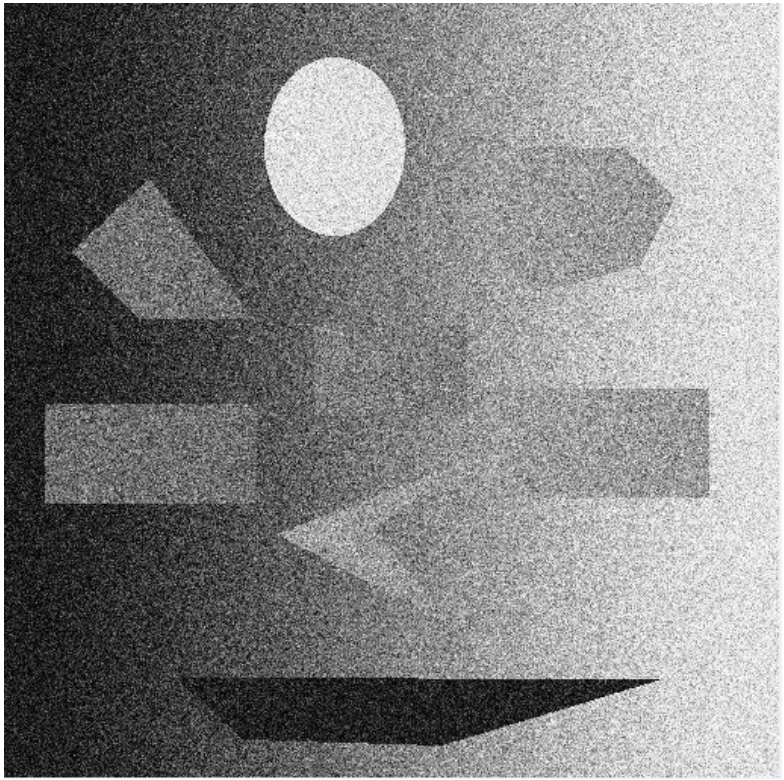


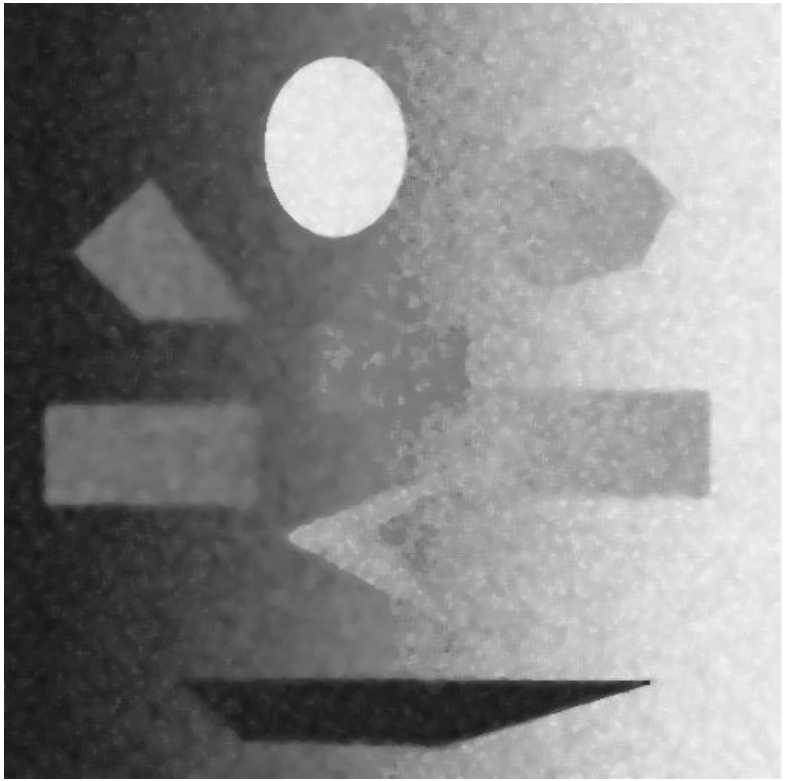
Wiener 7x7



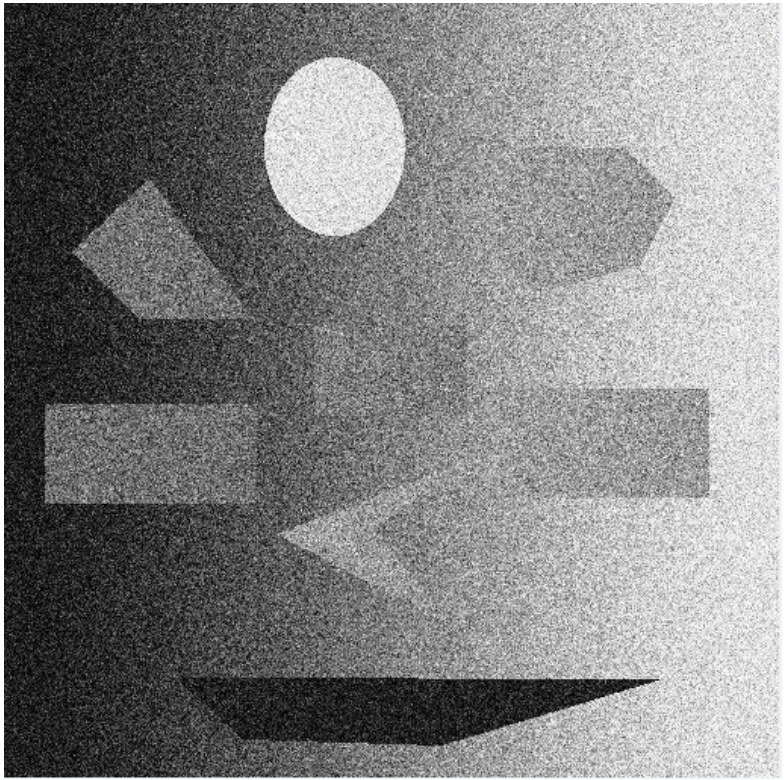


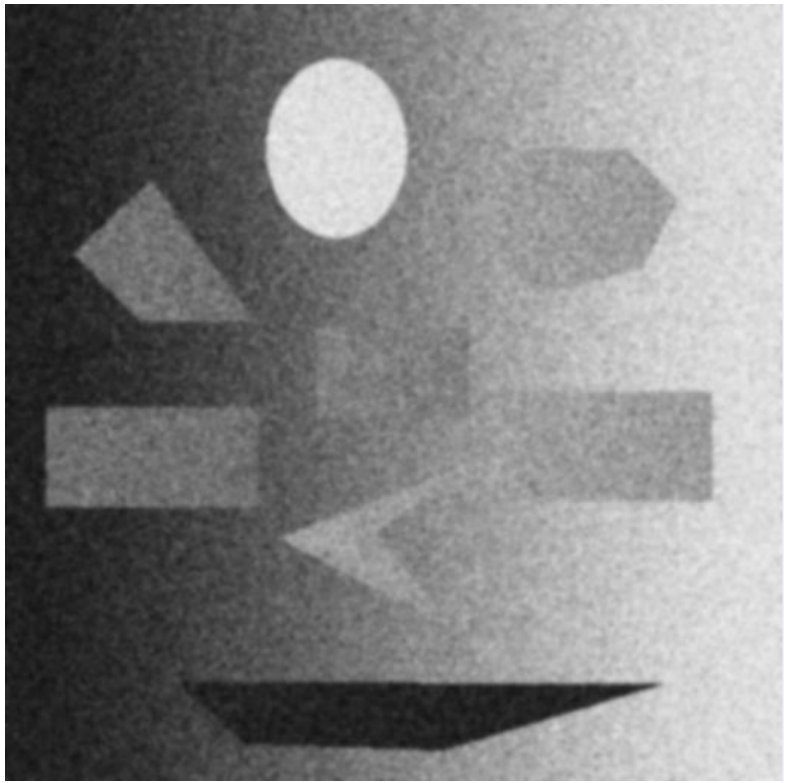
IBSF 9x9





Gaussian Blur 11x11





Moving Average

Metrics of Moving-Average-3x3 with original image

File Edit Font

Label	RMSE	SSIM	r	SNR	Corners
Noise_0	35.14	0.89	0.89	6.63	14.00
Noise_1	46.92	0.79	0.81	4.12	16.00
Noise_2	60.03	0.68	0.68	1.97	43.00
Noise_3	70.69	0.54	0.57	0.53	100.00
Noise_4	78.67	0.33	0.47	-0.39	115.00
Noise_5	84.75	0.28	0.41	-1.00	121.00
Noise_6	90.61	0.17	0.37	-1.51	132.00
Noise_7	97.43	0.15	0.32	-2.05	125.00
Noise_8	104.64	-0.00	0.27	-2.59	129.00
Noise_9	112.14	-0.00	0.23	-3.13	123.00
Média	78.10	0.38	0.50	0.26	91.80
Desvio Padrão	23.74	0.31	0.22	2.97	45.51

Metrics of Moving-Average-3x3 with target image

File Edit Font

Label	RMSE	SSIM	r	SNR	Corners
Noise_0	32.63	0.78	0.91	7.28	14.00
Noise_1	36.17	0.76	0.89	6.38	16.00
Noise_2	40.20	0.74	0.86	5.46	43.00
Noise_3	46.83	0.70	0.81	4.14	100.00
Noise_4	53.71	0.71	0.75	2.95	115.00
Noise_5	60.89	0.66	0.70	1.86	121.00
Noise_6	68.97	0.54	0.63	0.78	132.00
Noise_7	77.46	0.48	0.57	-0.23	125.00
Noise_8	86.96	0.35	0.49	-1.24	129.00
Noise_9	96.93	0.21	0.42	-2.18	123.00
Média	60.07	0.59	0.70	2.52	91.80
Desvio Padrão	20.97	0.18	0.16	3.10	45.51

Median

Metrics of Median-5x5 with original image

File Edit Font

Label	RMSE	SSIM	r	SNR	Corners
Noise_0	35.91	0.82	0.89	6.44	26.00
Noise_1	47.31	0.80	0.80	4.04	25.00
Noise_2	60.56	0.68	0.68	1.89	24.00
Noise_3	71.24	0.49	0.56	0.46	42.00
Noise_4	78.86	0.32	0.46	-0.41	77.00
Noise_5	84.39	0.24	0.40	-0.97	104.00
Noise_6	89.38	0.10	0.36	-1.39	116.00
Noise_7	94.89	0.09	0.32	-1.82	126.00
Noise_8	101.50	-0.00	0.27	-2.33	121.00
Noise_9	108.59	-0.00	0.23	-2.85	124.00
Média	77.26	0.35	0.50	0.31	78.50
Desvio Padrão	22.37	0.31	0.22	2.84	42.56

Metrics of Median-5x5 with target image

File Edit Font

Label	RMSE	SSIM	r	SNR	Corners
Noise_0	24.03	0.94	0.95	9.93	26.00
Noise_1	29.44	0.87	0.92	8.17	25.00
Noise_2	34.11	0.77	0.90	6.89	24.00
Noise_3	39.83	0.81	0.86	5.54	42.00
Noise_4	47.64	0.76	0.80	3.99	77.00
Noise_5	54.01	0.74	0.75	2.90	104.00
Noise_6	60.77	0.70	0.70	1.88	116.00
Noise_7	67.62	0.64	0.65	0.95	126.00
Noise_8	75.34	0.58	0.59	0.01	121.00
Noise_9	84.89	0.48	0.52	-1.03	124.00
Média	51.77	0.73	0.76	3.92	78.50
Desvio Padrão	19.31	0.13	0.14	3.46	42.56

Wiener

Metrics of Wiener-7x7 with original image

File Edit Font

Label	RMSE	SSIM	r	SNR	Corners
Noise_0	16.79	0.98	0.98	13.04	29.00
Noise_1	31.80	0.89	0.91	7.49	78.00
Noise_2	46.12	0.81	0.81	4.26	120.00
Noise_3	51.00	0.47	0.78	3.37	127.00
Noise_4	54.48	0.60	0.75	2.80	128.00
Noise_5	55.10	-0.52	0.75	2.73	130.00
Noise_6	59.51	-0.23	0.72	2.14	126.00
Noise_7	64.08	-0.14	0.70	1.59	128.00
Noise_8	68.86	0.00	0.67	1.04	124.00
Noise_9	73.47	0.00	0.64	0.55	124.00
Média	52.12	0.29	0.77	3.90	111.40
Desvio Padrão	16.27	0.50	0.10	3.57	31.07

Metrics of Wiener-7x7 with target image

File Edit Font

Label	RMSE	SSIM	r	SNR	Corners
Noise_0	36.13	0.76	0.89	6.39	29.00
Noise_1	39.74	0.74	0.86	5.56	78.00
Noise_2	46.66	0.70	0.81	4.17	120.00
Noise_3	58.74	0.20	0.71	2.17	127.00
Noise_4	68.52	0.17	0.61	0.83	128.00
Noise_5	77.88	-0.48	0.51	-0.28	130.00
Noise_6	84.31	-0.43	0.44	-0.97	126.00
Noise_7	89.49	-0.37	0.40	-1.49	128.00
Noise_8	95.12	-0.30	0.35	-2.02	124.00
Noise_9	100.61	-0.24	0.30	-2.50	124.00
Média	69.72	0.08	0.59	1.19	111.40
Desvio Padrão	22.25	0.48	0.21	3.07	31.07

IBSF

Metrics of IBSF-9x9 with original image

File Edit Font

Label	RMSE	SSIM	r	SNR	Corners
Noise_0	36.04	0.85	0.89	6.40	27.00
Noise_1	47.75	0.31	0.80	3.96	25.00
Noise_2	61.07	-0.00	0.67	1.82	15.00
Noise_3	71.38	-0.51	0.55	0.45	17.00
Noise_4	79.18	-0.37	0.46	-0.45	29.00
Noise_5	84.78	-0.27	0.39	-1.01	45.00
Noise_6	89.41	-0.11	0.35	-1.39	49.00
Noise_7	94.64	-0.09	0.31	-1.80	64.00
Noise_8	101.28	0.00	0.26	-2.31	71.00
Noise_9	108.64	0.00	0.22	-2.85	79.00
Média	77.42	-0.02	0.49	0.28	42.10
Desvio Padrão	22.22	0.36	0.22	2.82	21.88

Metrics of IBSF-9x9 with target image

File Edit Font

Label	RMSE	SSIM	r	SNR	Corners
Noise_0	29.69	0.67	0.92	8.10	27.00
Noise_1	34.95	0.25	0.90	6.68	25.00
Noise_2	38.31	-0.00	0.87	5.88	15.00
Noise_3	42.43	-0.45	0.85	4.99	17.00
Noise_4	47.54	-0.70	0.81	4.01	29.00
Noise_5	51.19	-0.75	0.79	3.37	45.00
Noise_6	55.99	-0.74	0.75	2.59	49.00
Noise_7	60.99	-0.72	0.72	1.84	64.00
Noise_8	67.80	-0.64	0.68	0.92	71.00
Noise_9	76.50	-0.52	0.63	-0.12	79.00
Média	50.54	-0.36	0.79	3.83	42.10
Desvio Padrão	14.19	0.47	0.09	2.48	21.88

Gaussian Blur

Metrics of Gaussian-11x11 with original image

File Edit Font

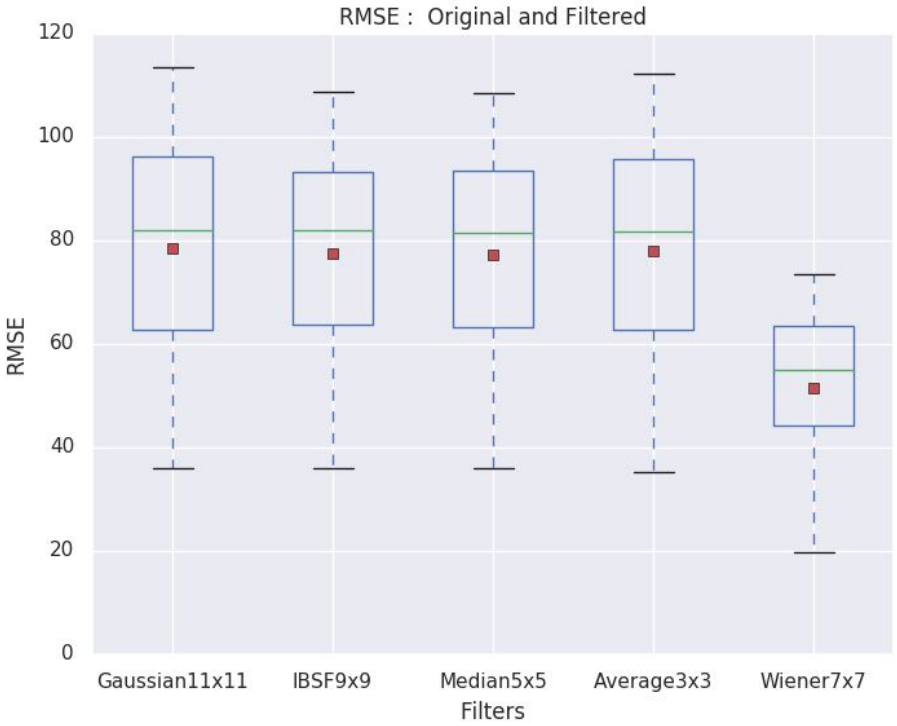
Label	RMSE	SSIM	r	SNR	Corners
Noise_0	36.05	0.86	0.89	6.40	9.00
Noise_1	47.21	0.80	0.80	4.06	9.00
Noise_2	60.16	0.68	0.68	1.95	8.00
Noise_3	70.93	0.54	0.56	0.50	12.00
Noise_4	78.77	0.33	0.47	-0.40	22.00
Noise_5	85.18	0.28	0.41	-1.05	49.00
Noise_6	91.04	0.14	0.37	-1.55	73.00
Noise_7	98.00	0.12	0.32	-2.10	96.00
Noise_8	105.38	-0.00	0.28	-2.65	104.00
Noise_9	113.42	-0.00	0.23	-3.23	112.00
Média	78.61	0.38	0.50	0.19	49.40
Desvio Padrão	23.87	0.31	0.21	2.95	40.93

Metrics of Gaussian-11x11 with target image

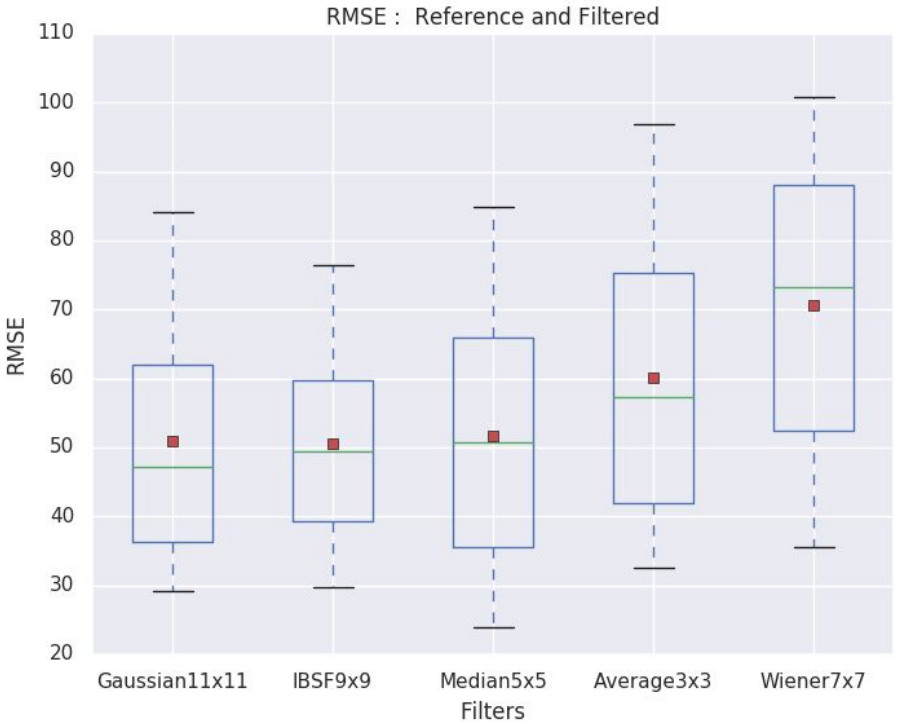
File Edit Font

Label	RMSE	SSIM	r	SNR	Corners
Noise_0	29.15	0.88	0.93	8.26	9.00
Noise_1	33.03	0.86	0.90	7.17	9.00
Noise_2	35.31	0.77	0.89	6.59	8.00
Noise_3	39.63	0.74	0.86	5.59	12.00
Noise_4	44.61	0.79	0.83	4.56	22.00
Noise_5	49.84	0.75	0.80	3.60	49.00
Noise_6	56.49	0.71	0.76	2.51	73.00
Noise_7	63.93	0.67	0.72	1.43	96.00
Noise_8	73.29	0.55	0.66	0.25	104.00
Noise_9	84.09	0.48	0.59	-0.95	112.00
Média	50.94	0.72	0.79	3.90	49.40
Desvio Padrão	17.34	0.12	0.11	2.92	40.93

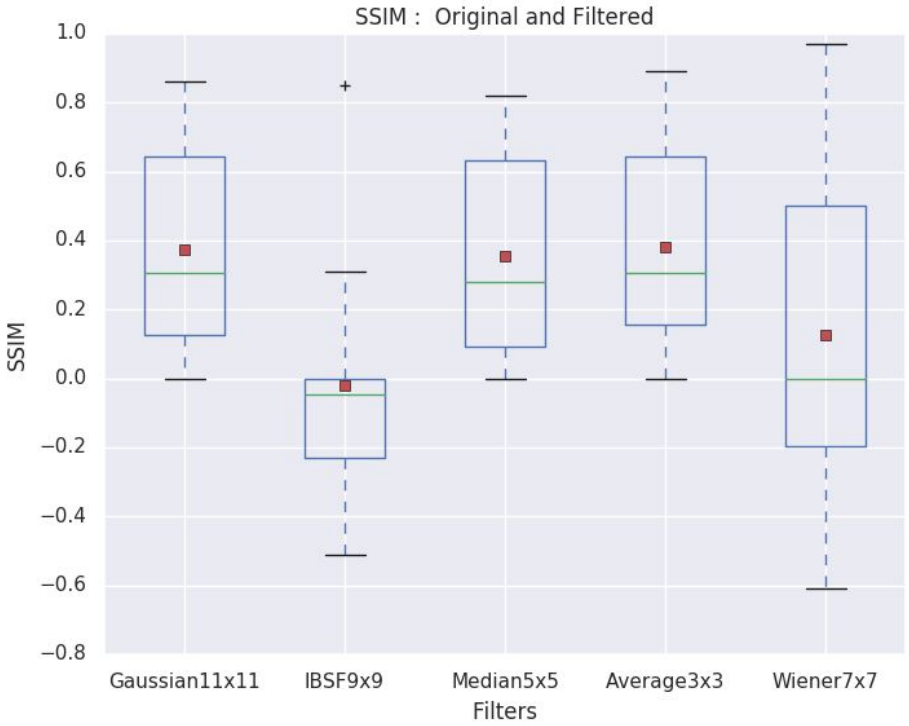
Métrica RMSE



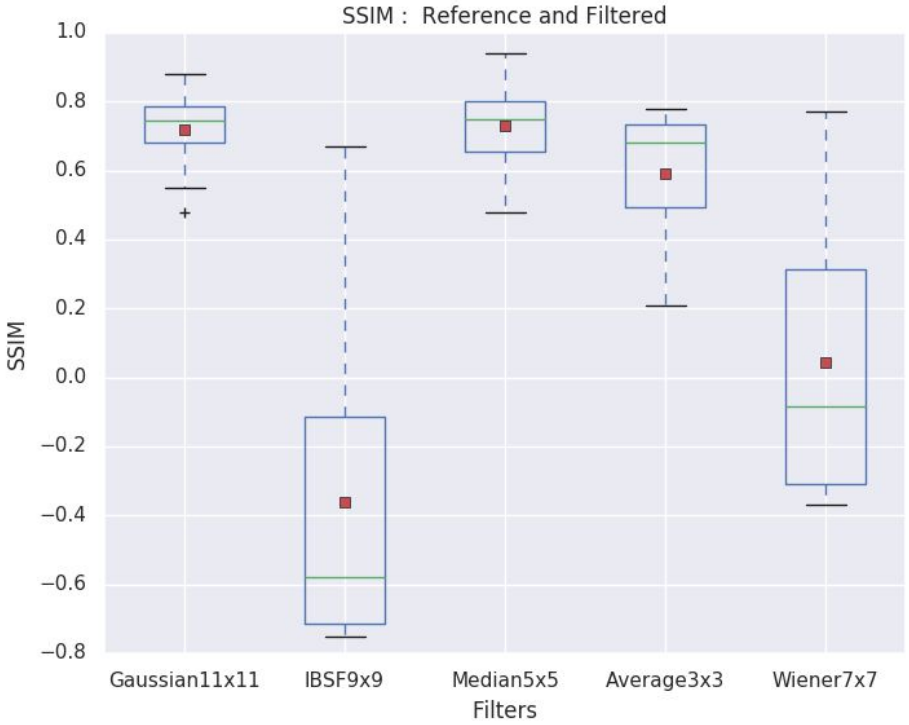
Métrica RMSE



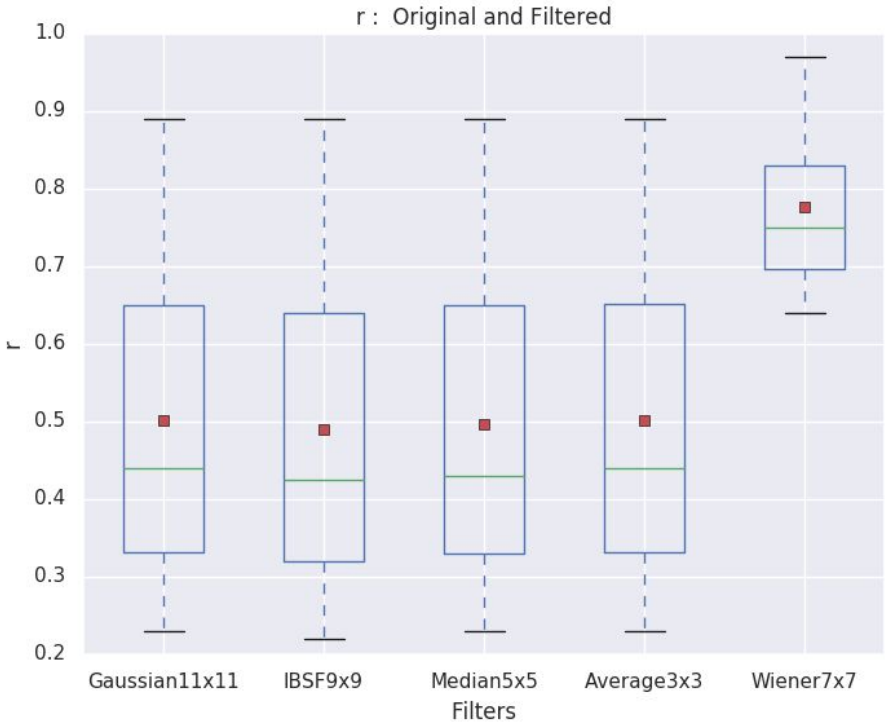
Métrica SSIM



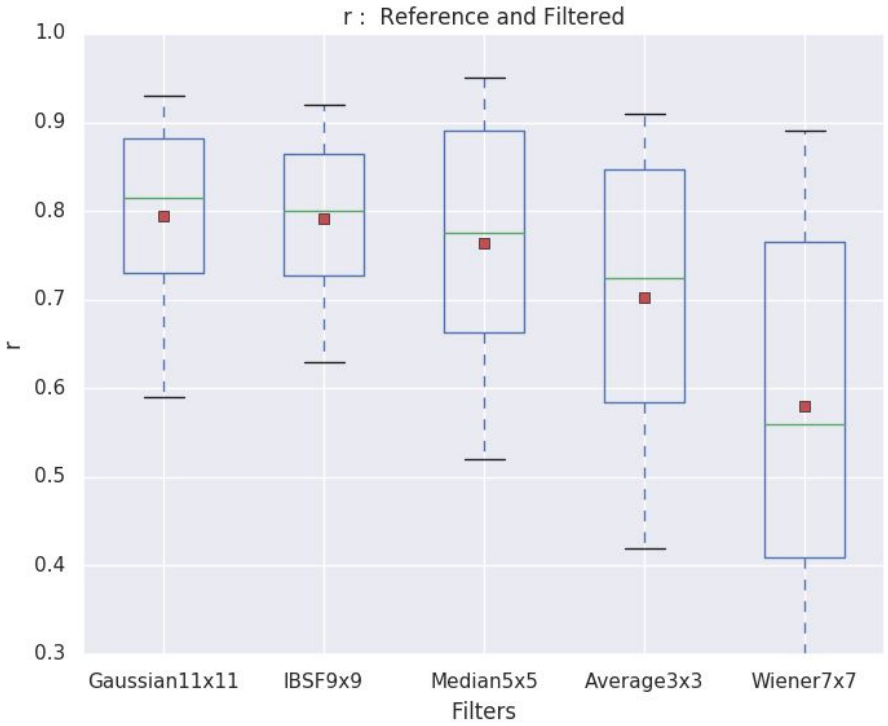
Métrica SSIM



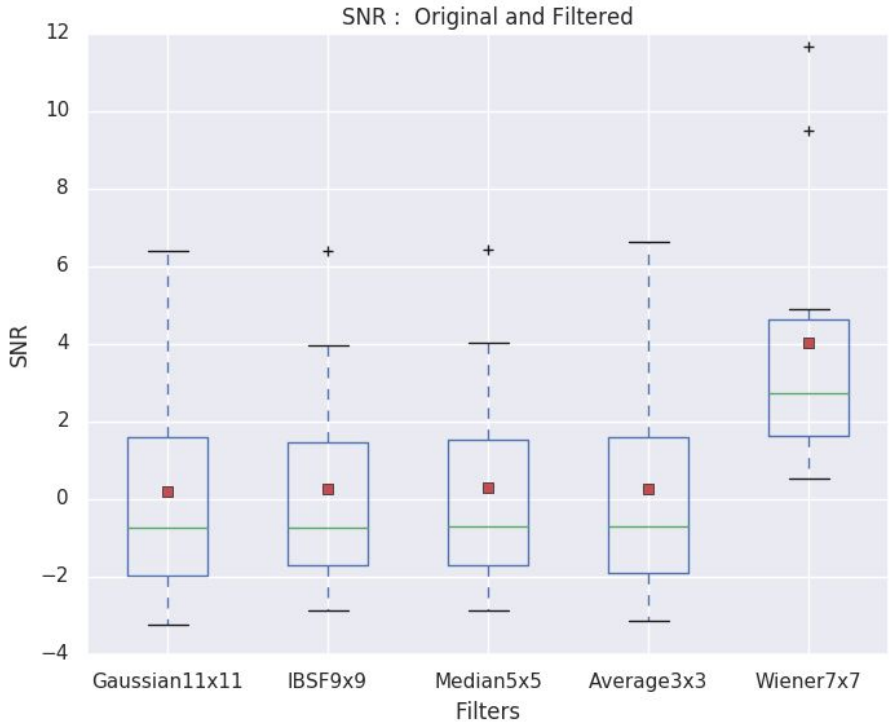
Métrica Coeficiente r



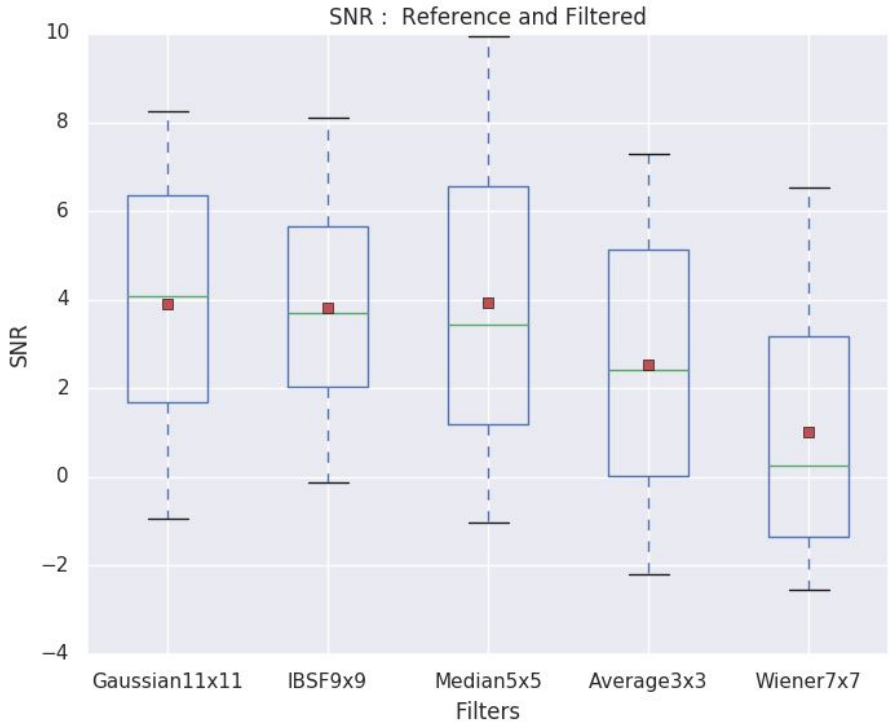
Métrica Coeficiente r



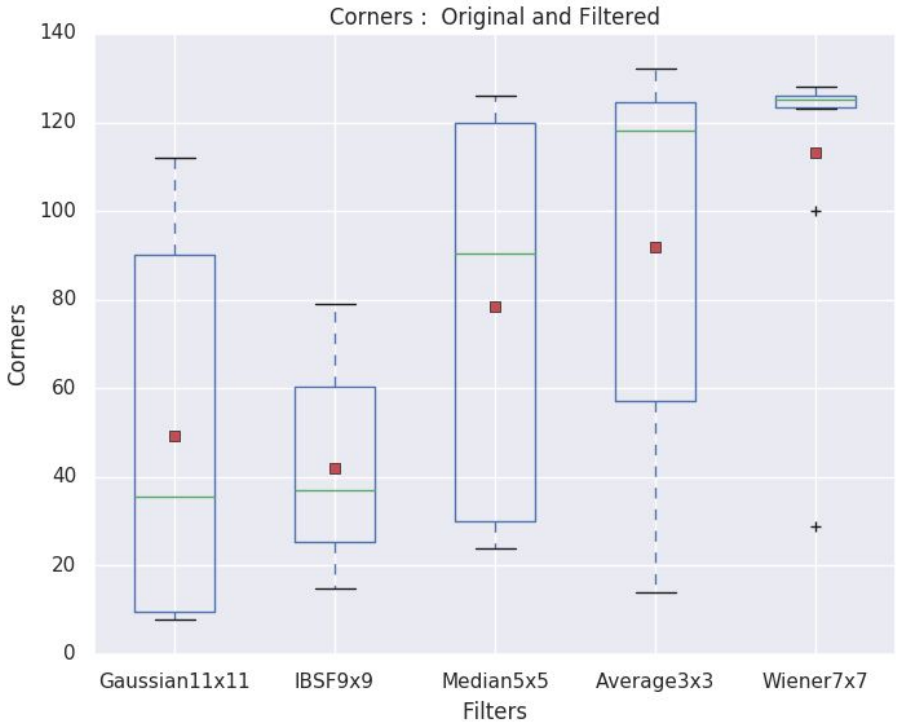
Métrica SNR


SNR : Original and Filtered
SNR
12
10
8
6
4
2
0
−2
−4
Gaussian11x11
IBSF9x9
Median5x5
Average3x3
Wiener7x7
Filters

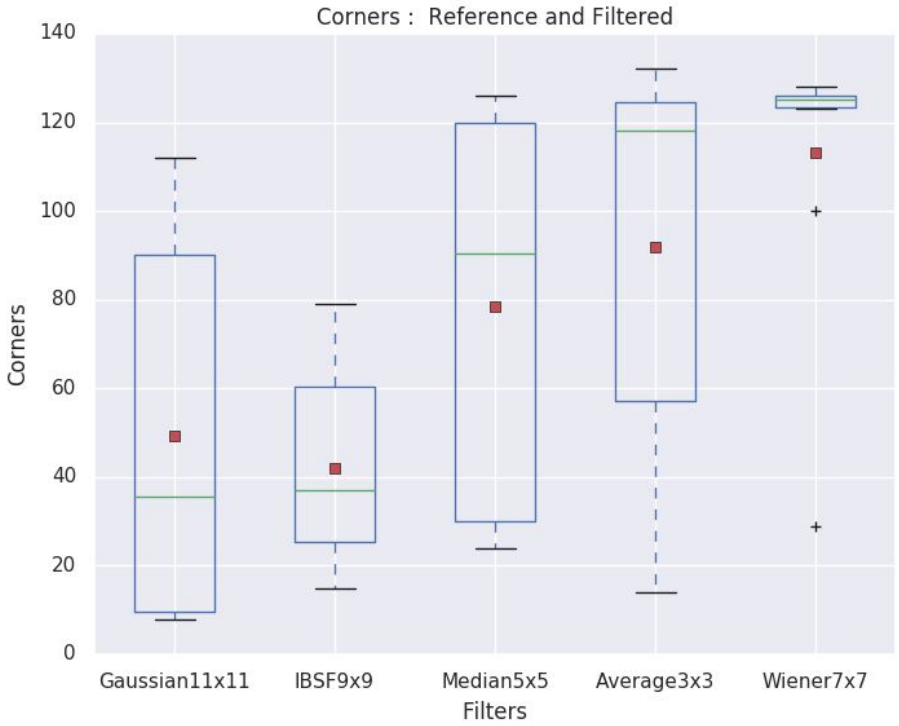
Métrica SNR


SNR : Reference and Filtered
SNR
10
8
6
4
2
0
−2
−4
Gaussian11x11
IBSF9x9
Median5x5
Average3x3
Wiener7x7
Filters

Métrica Corners



Métrica Corners



Conclusões

De acordo com os gráficos boxplots, o filtro Wiener foi o que apresentou menor diferença interquartil. No entanto, pelo RMSE, foi o que menos modificou o ruído.

Visualmente, o IBSF foi o que suavizou mais o ruído, deixando a imagem filtrada menos granulosa, entretanto foi a que mais borrou a imagem.

Referências Bibliográficas

- Principles of Digital Images Processing - Core Algorithm - Wilhelm Burger.
- Principles of Digital Images Processing - Fundamental Techniques - Wilhelm Burger.
- Processamento Digital de Imagens - Rafael C. Gonzalez.
- Artigo Image Quality Assessment: From Error Visibility to Structural Similarity - Zhou Wang.
- Artigo Edge-preserving Speckle Texture Removal by Interference-Based Speckle Filtering Followed by Anisotropic Diffusion - Fernando M. Cardoso.

Obrigada!

Link do projeto: https://github.com/Luana-Leticia/LAB_PDI