

Instituto de Computação

UNIVERSIDADE ESTADUAL DE CAMPINAS

Relatório de Aplicação da Transformada Rápida de Fourier para Filtragem no Domínio de Frequência e Compressão de Imagens

Relatório de aplicação desenvolvido para disciplina MO443 Introdução ao Processamento de Imagem Digital.

Professor: Hélio Pedrini

Aluna: Juliana Midlej do Espírito Santo

RA: 200208

Campinas - São Paulo
26 de Abril de 2024

1. Introdução	3
2. Materiais e Métodos	3
2.1 Requerimentos	3
2.2 Organização de Arquivos	3
2.3 Criação de Máscaras	4
2.4 Transformada Rápida de Fourier	4
2.5 Magnitude do Espectro de Fourier	4
2.6 Visualização das Imagens	4
2.7 Filtro Passa-Baixa	5
2.8 Filtro Passa-Alta	5
2.9 Filtro Passa-Faixa	5
2.10 Filtro Rejeita-Faixa	5
2.11 Compressão	5
3. Resultados e Discussão	6
3.1 Aplicação do Filtro Passa-Baixa	6
3.2 Aplicação do Filtro Passa-Alta	7
3.3 Aplicação do Filtro Passa-Faixa	8
3.4 Aplicação do Filtro Rejeita-Faixa	9
3.5 Aplicação da Compressão	10
4. Conclusão	11

1. Introdução

As transformadas de imagens são operações que alteram o espaço de representação de uma imagem para outro domínio, de forma que a informação presente na imagem seja preservada no domínio da transformada e a transformada seja reversível.

A transformada rápida de Fourier (do inglês, *Fast Fourier Transform* - FFT) possibilita representar uma imagem no domínio de frequência. O objetivo deste trabalho é aplicar a FFT em imagens digitais, explorando a filtragem e compressão de imagens no domínio de frequência.

O projeto que contém a aplicação da FFT para filtrar e comprimir imagens está disponível no GitHub no link: https://github.com/annwith/digital_image_processing.

2. Materiais e Métodos

Esta seção abordará os requerimentos do projeto, a organização dos arquivos e as escolhas de implementação para filtragem e compressão de imagens. É importante destacar que os códigos devem ser executados dentro da pasta *fast_fourier_transform* – pasta que contém todos os arquivos do projeto – devido ao modo como é feita a importação do arquivo *utils.py*.

2.1 Requerimentos

Para implementar o algoritmo foram utilizados a linguagem de programação Python em sua versão 3.8.10 e os pacotes listados na Tabela 1.

Pacote	Versão
OpenCV (opencv-python)	4.9.0.80
NumPy (numpy)	1.24.4
Matplotlib (matplotlib)	3.7.5

Tabela 1: Pacotes

2.2 Organização de Arquivos

Para melhor organização e clareza, o projeto que contém o código foi dividido nas seguintes pastas e arquivos:

- *images*: Pasta de imagens;
- *compressed_images*: Pasta de imagens comprimidas;
- *low_pass.py*: Arquivo de aplicação de filtros passa-baixa no domínio de frequência;
- *high_pass.py*: Arquivo de aplicação de filtros passa-alta no domínio de frequência;
- *band_pass.py*: Arquivo de aplicação de filtros passa-faixa no domínio de frequência;
- *band_stop.py*: Arquivo de aplicação de filtros rejeita-faixa no domínio de frequência;
- *compression.py*: Arquivo para compressão de uma imagem no domínio de frequência;
- *utils.py*: Arquivo para funções auxiliares;

- ***README.md***: Arquivo que contém informações sobre o projeto e como executar cada arquivo *python*;
- ***requirements.txt***: Arquivo que contém os pacotes necessários para execução dos demais arquivos *python*.

2.3 Criação de Máscaras

A criação das máscaras para filtrar as imagens foi feita com a função *create_mask* do arquivo *utils.py*. Essa função recebe: a altura da imagem; a largura da imagem; o raio desejado para o filtro e se o filtro será passa-alta ou passa-baixa. A função retorna um *numpy n-dimensional array* de mesma altura e largura da imagem contendo em cada posição (x, y) *True* ou *False*.

Para um filtro passa-baixa, todas as posições (x, y) da imagem com distância para o centro da imagem menores ou iguais ao raio são *True*, o restante é *False*. Para um filtro passa-alta, distâncias maiores que o raio são *True*, o restante é *False*.

O cálculo da distância entre cada ponto e o centro da imagem foi feito da seguinte forma: criam-se dois vetores de coordenadas x: [0, 1, ... width - 1] e y: [0, 1, ... height - 1]; criam-se duas matrizes de coordenadas xx e yy representando as coordenadas x e y de cada ponto na imagem usando *np.meshgrid()*; calculam-se as distâncias de todos os pontos da imagem até o centro utilizando a fórmula da distância euclidiana de forma vetorizada com *np.sqrt((xx - centro_x)^2 + (yy - centro_y)^2)*.

2.4 Transformada Rápida de Fourier

Para aplicar a FFT e sua inversa, foram utilizadas as funções *numpy.fft.fft2* e *numpy.fft.ifft2*, respectivamente. A função *numpy.fft.fft2* calcula a transformada discreta bidimensional de Fourier. A função *numpy.fft.ifft2* calcula a inversa da transformada discreta bidimensional de Fourier.

2.5 Magnitude do Espectro de Fourier

A magnitude do espectro de Fourier é uma medida da amplitude das diferentes frequências que compõem a imagem no domínio da frequência. É uma representação gráfica ou numérica da contribuição de cada frequência para a imagem original.

O cálculo da magnitude do espectro de Fourier para uma imagem pode ser feito utilizando a função *numpy.abs*. Para melhorar a visualização do gráfico da magnitude do espectro nos arquivos de filtragem, foi calculado o log da magnitude e multiplicado por 20.

2.6 Visualização das Imagens

A visualização das imagens será feita com a biblioteca *matplotlib*. A função *imshow* normaliza automaticamente a imagem de forma linear, sendo que o menor valor corresponderá a 0 e o maior a 1. Por esse motivo, algumas imagens – como as imagens de magnitude do espectro de Fourier com e sem aplicação de uma máscara – podem parecer mais claras ou mais escuras em alguns gráficos.

A imagem utilizada neste projeto foi a imagem *baboon.png*.

2.7 Filtro Passa-Baixa

A aplicação do filtro passa-baixa em uma imagem no domínio de frequência está no arquivo *low_pass.py*. Esse arquivo recebe como parâmetro de linha de comando uma imagem. A execução desse arquivo abre uma janela interativa em que é possível escolher o tamanho do raio do filtro, assim como visualizar a imagem original, a imagem filtrada, o filtro utilizado e a magnitude do espectro de frequência. O tamanho do raio pode ir de 0 até a metade da altura da imagem (escolha arbitrária).

Para executar o comando necessário é: *python3 low_pass.py -i image.png*. Para fechar a janela é preciso apertar a tecla ‘Q’.

2.8 Filtro Passa-Alta

A aplicação do filtro passa-alta em uma imagem no domínio de frequência está no arquivo *high_pass.py*. O funcionamento é semelhante ao do arquivo *low_pass.py*, explicado na seção anterior.

Para executar o comando necessário é: *python3 high_pass.py -i image.png*. Para fechar a janela é preciso apertar a tecla ‘Q’.

2.9 Filtro Passa-Faixa

A aplicação do filtro passa-faixa em uma imagem no domínio de frequência está no arquivo *band_pass.py*. O funcionamento é semelhante ao do arquivo *low_pass.py*, explicado na seção anterior. Adicionalmente, podemos escolher o tamanho dos dois raios de forma interativa.

Para executar o comando necessário é: *python3 band_pass.py -i image.png*. Para fechar a janela é preciso apertar a tecla ‘Q’.

A criação de uma máscara passa-faixa é feita aplicando a operação lógica AND em duas máscaras, uma passa-baixa de raio maior e uma passa-alta de raio menor.

2.10 Filtro Rejeita-Faixa

A aplicação do filtro rejeita-faixa em uma imagem no domínio de frequência está no arquivo *band_stop.py*. O funcionamento é semelhante ao do arquivo *band_pass.py*, explicado na seção anterior.

Para executar o comando necessário é: *python3 band_stop.py -i image.png*. Para fechar a janela é preciso apertar a tecla ‘Q’.

A criação de uma máscara rejeita-faixa é feita aplicando a operação lógica OR em duas máscaras, uma passa-baixa de raio menor e uma passa-alta de raio maior.

2.11 Compressão

A compressão da imagem é feita no arquivo *compression.py*. Esse arquivo recebe como parâmetros de linha de comando uma imagem e um limiar ou uma porcentagem. Caso sejam passados o limiar e a porcentagem, apenas o limiar será considerado para compressão. Para executar o comando necessário é: *python3 compression.py -i image.png -t 10000 ou -p 10*.

`python3 compression.py -i image.png -p 60`. Se nenhum limiar for passado, o seguinte erro ocorrerá: “*ValueError: Please provide either a threshold value or a percentage value.*”.

A compressão é feita eliminando coeficientes cuja magnitude sejam menores que um limiar. Se a porcentagem é passada como parâmetro, o limiar é calculado de forma que o número de coeficientes eliminados corresponda a porcentagem que foi passada como parâmetro.

Com a execução desse arquivo, são mostradas a imagem original e seu histograma, o espectro de magnitude e seu histograma e a imagem comprimida e seu histograma. Na imagem do histograma da magnitude do espectro de Fourier é mostrado também o limiar escolhido. A imagem comprimida é salva na pasta *compressed_images* e possui mesmo nome da imagem original, acrescido do limiar utilizado para compressão, por exemplo: *baboon_4490.png*.

3. Resultados e Discussão

3.1 Aplicação do Filtro Passa-Baixa

O filtro passa-baixa é usado para "passar" componentes de frequência mais baixa enquanto atenua ou "rejeita" componentes de frequência mais alta. Em imagens, um filtro passa-baixa é comumente usado para suavizar ou borrar as imagens. Ele remove detalhes finos e ruídos de alta frequência.

A aplicação do filtro passa-baixa na imagem *baboon.png* é mostrada na Figura 1. Na Figura 2 podemos ver que reduzir o tamanho do raio aumenta a intensidade do filtro, borrando mais a imagem.

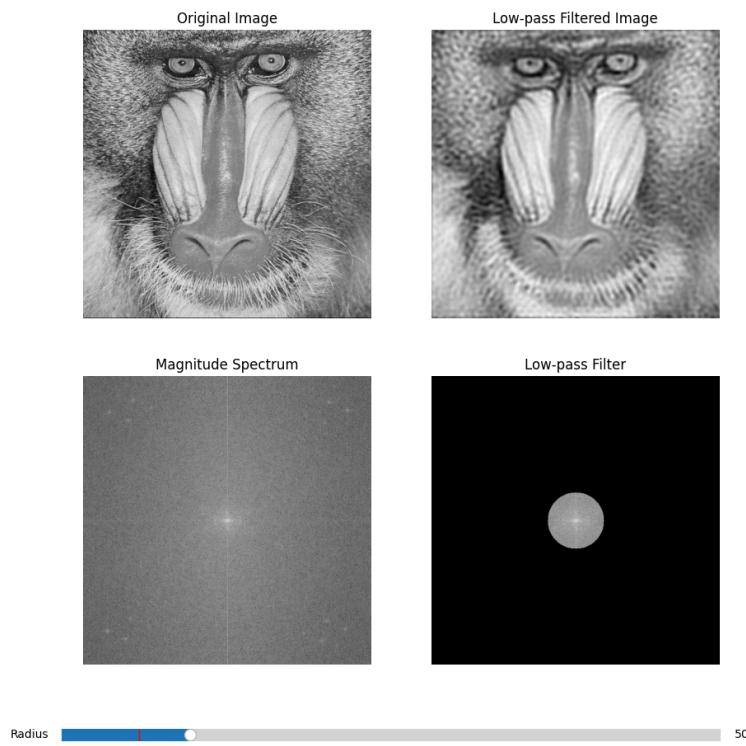


Figura 1: Aplicação do Filtro Passa-Baixa

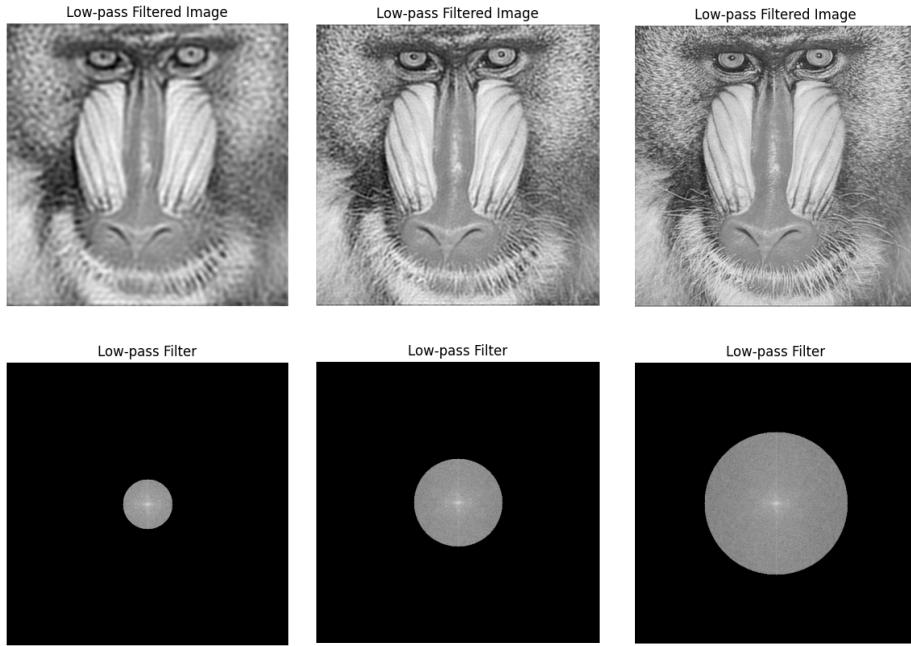


Figura 2: Aplicação do Filtro com Alteração no Raio

3.2 Aplicação do Filtro Passa-Alta

Um filtro passa-alta permite que componentes de alta frequência passem e atenua componentes de baixa frequência. Portanto, um filtro passa-alta é comumente usado para realçar bordas e detalhes de alta frequência.

A aplicação do filtro passa-alta na imagem *baboon.png* é mostrada na Figura 3. Na Figura 4 podemos ver que aumentar o tamanho do raio aumenta a intensidade do filtro. Além disso, é possível perceber que raios pequenos já fazem uma grande diferença na imagem, com raios maiores podemos acabar perdendo muita informação.

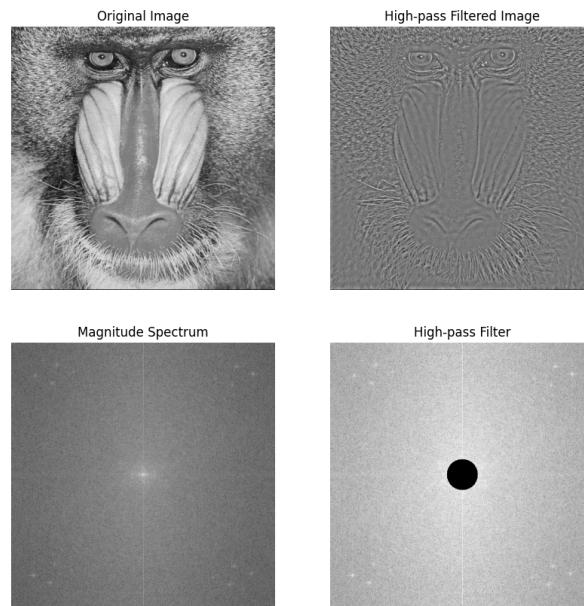


Figura 3: Aplicação do Filtro Passa Alta

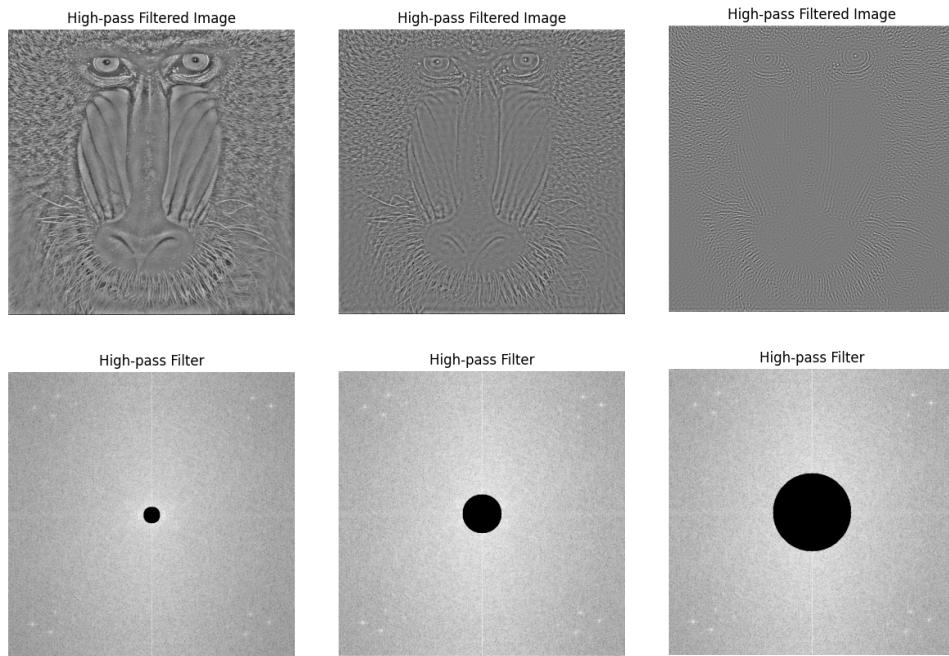


Figura 4: Aplicação do Filtro com Alteração no Raio

3.3 Aplicação do Filtro Passa-Faixa

Um filtro passa-faixa é um tipo de filtro que permite que um intervalo específico de frequências passe enquanto atenua ou elimina frequências fora desse intervalo. O filtro passa-faixa é útil quando se deseja selecionar ou realçar apenas as frequências que estão dentro de uma faixa específica de interesse, enquanto rejeita as frequências que estão abaixo ou acima dessa faixa.

A aplicação do filtro passa-faixa na imagem *baboon.png* é mostrada na Figura 5. Na Figura 6 podemos ver como diferentes escolhas de raios alteram a imagem. Nos raios testados, parece que estamos encontrando bordas e depois suavizando elas.

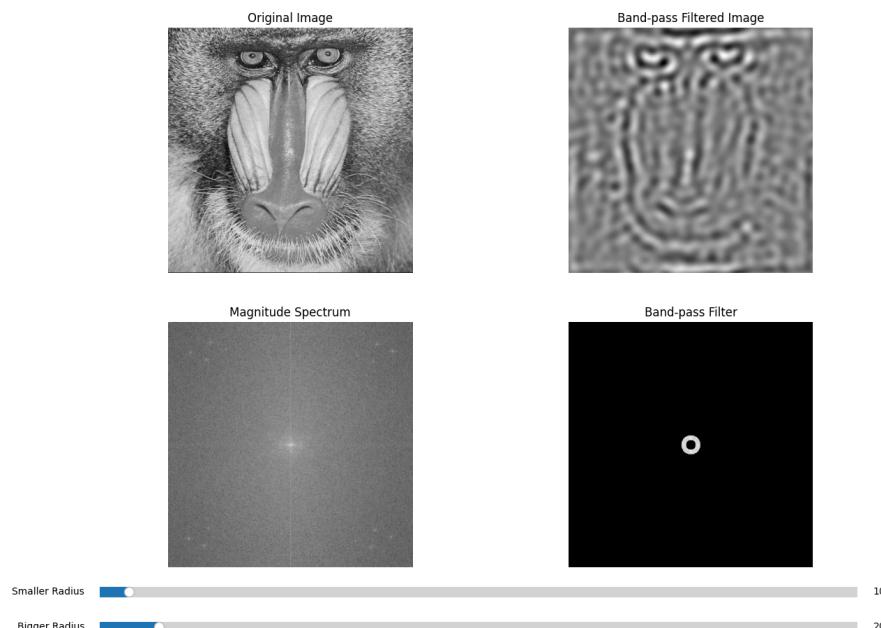


Figura 5: Aplicação do Filtro Passa-Faixa

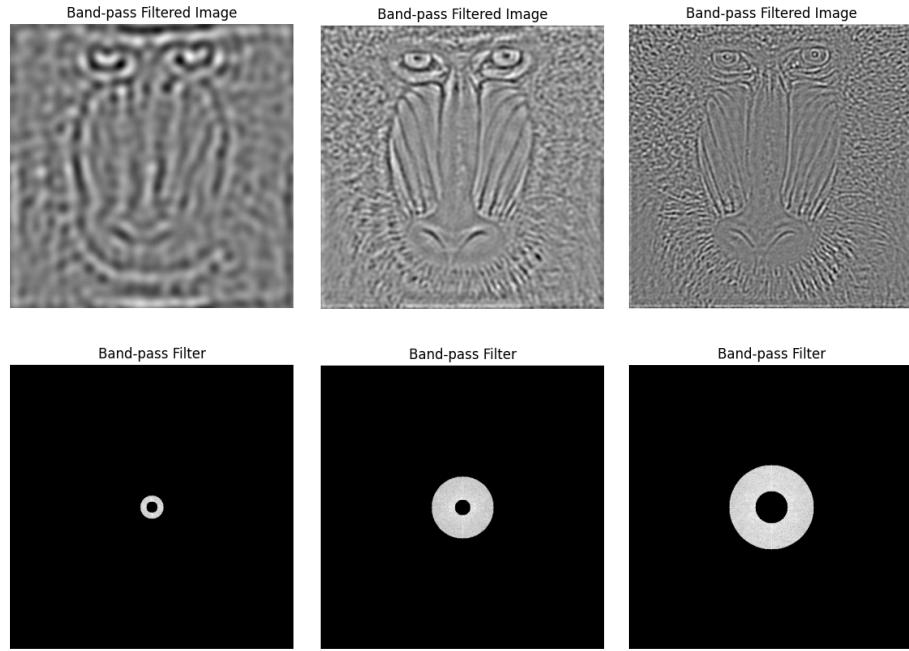


Figura 6: Aplicação do Filtro com Alteração nos Raios

3.4 Aplicação do Filtro Rejeita-Faixa

Um filtro rejeita-faixa é um tipo de filtro que atenua ou elimina uma faixa específica de frequências, enquanto permite a passagem de frequências fora dessa faixa. Este tipo de filtro é útil quando se deseja eliminar seletivamente uma banda de frequências indesejadas, enquanto mantém as frequências que estão fora desta banda.

A aplicação do filtro rejeita-faixa na imagem *baboon.png* é mostrada na Figura 7. Na Figura 8 podemos ver como diferentes escolhas de raios alteram a imagem.

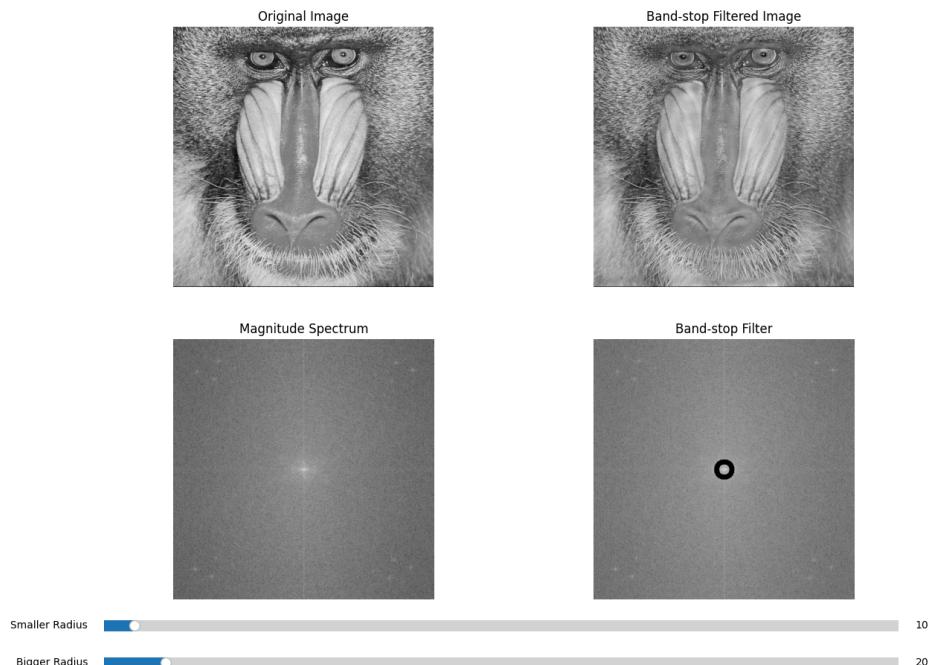


Figura 7: Aplicação do Filtro Rejeita-Faixa

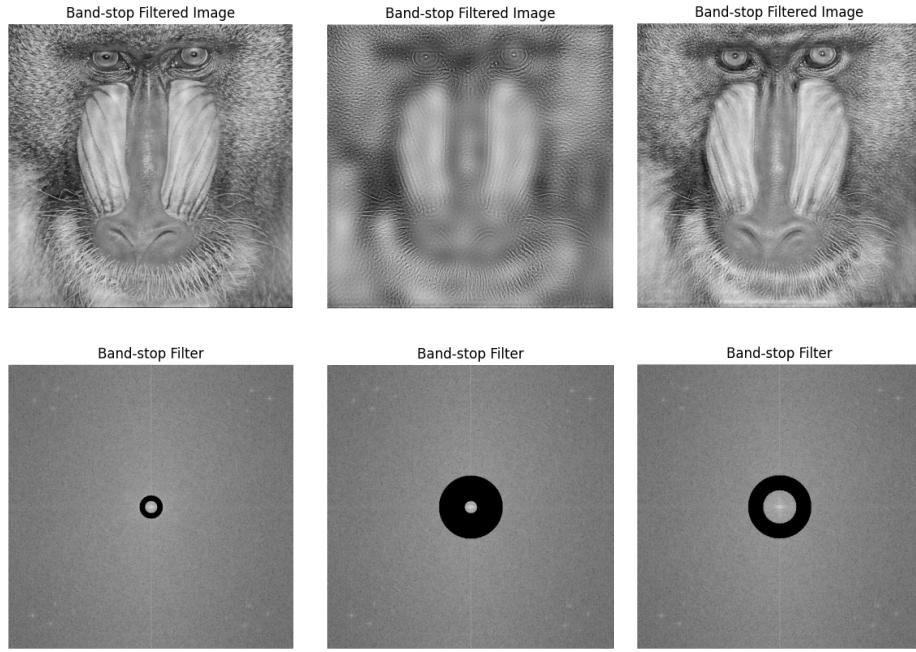


Figura 8: Aplicação do Filtro com Alteração nos Raios

3.5 Aplicação da Compressão

A compressão foi aplicada na imagem do *baboon.png* e os resultados obtidos são mostrados nas Figuras 9 e 10. Foi percebido que é possível remover grande parte dos coeficientes de menor valor sem que ocorram alterações perceptíveis na imagem. Parece que a maior parte da informação da imagem está nos coeficientes de magnitude maior.

Além disso, a imagem de saída é um pouco menor do que a imagem de entrada, lembrando que o formato das imagens é PNG, então não há perdas na compressão realizada pelo *cv.imwrite*. A imagem de entrada possui aproximadamente 238 Kb, enquanto as imagens de saída possuem 204 e 182 Kb para as compressões com porcentagem de 60 e 90 respectivamente.

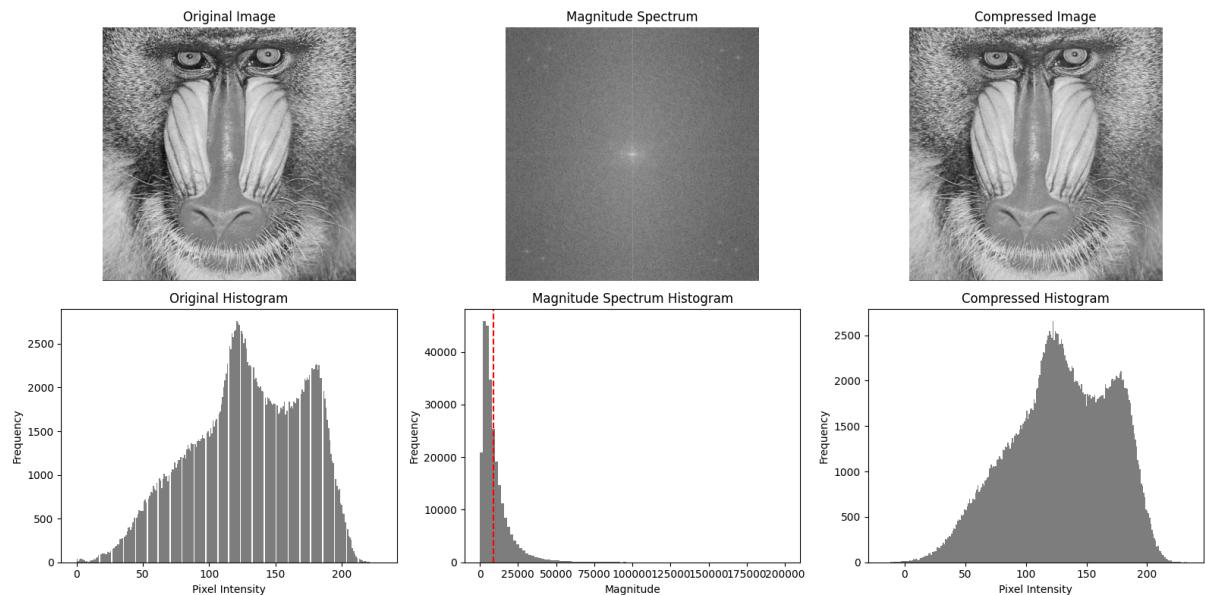


Figura 9: Compressão (Porcentagem = 60)

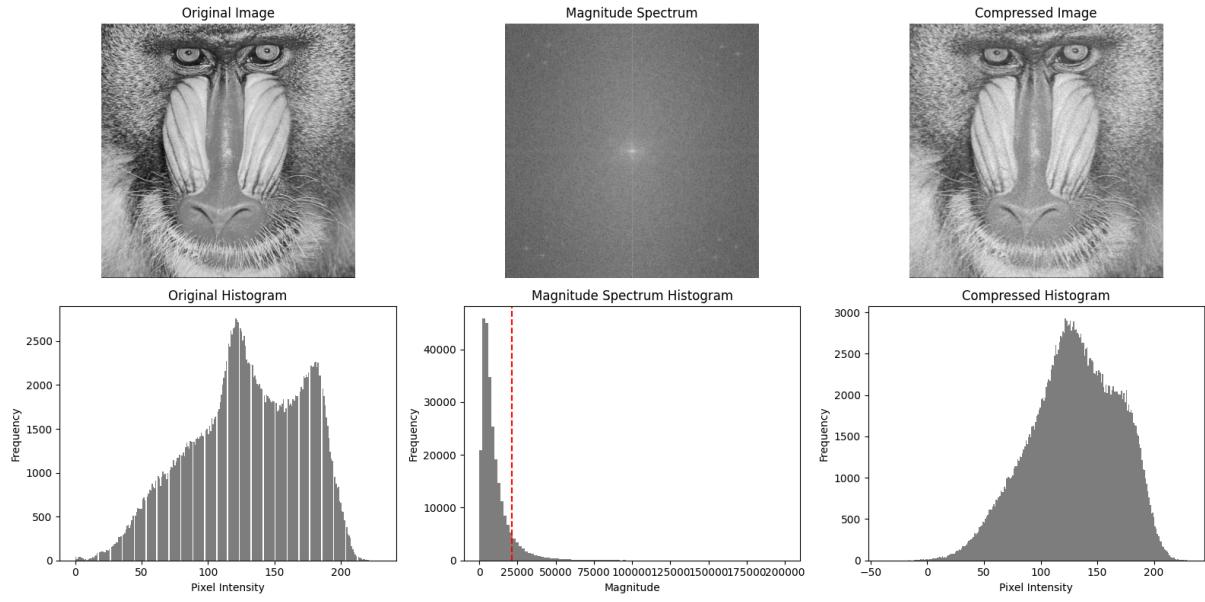


Figura 10: Compressão (Porcentagem = 90)

4. Conclusão

Concluo que é muito mais fácil e menos custoso aplicar filtros no domínio de frequência. Além de ser surpreendente como é possível comprimir a imagem, mais especificamente quantos pixels é possível eliminar sem que haja mudança perceptível.