Transformada de Fourier em Processamento de Imagens Digitais

Bruno Sobreira França (217787)

Abril 2024

1 Introdução

Os estudos de Fourier resultaram em uma série de conceitos matemáticos de extrema importância para o mundo contemporâneo. Entre eles, destaca-se a Transformada de Fourier, que, quando aplicada a um sinal no domínio do tempo ou do espaço, permite obter sua representação no domínio da frequência, também conhecida como espectro de Fourier. Além disso, Fourier propôs a inversa desta transformada, permitindo resolver problemas dessa natureza em outro domínio e, posteriormente, obter seu valor no domínio original. Este conjunto de operações possui uma ampla gama de aplicações em diversos campos do conhecimento. Neste trabalho, o foco reside na análise de como o uso dessas transformadas pode ser aplicado no contexto do processamento de imagens digitais em tons de cinza, buscando entender como a manipulação do espectro de Fourier desses objetos impacta sua visualização posterior.

Dada a natureza discreta das representações das imagens e dos sistemas computacionais utilizados para processá-las, o presente trabalho empregará o uso da Transformada Discreta de Fourier e de sua inversa, por meio do algoritmo Fast Fourier Transform (FFT), para obter o espectro de frequência das imagens e as próprias imagens a partir de seus domínios de frequência. No domínio de frequência das imagens, serão aplicados os filtros passa-baixa, passa-alta, passa-faixa e rejeita-faixa com o objetivo de analisar como a aplicação desses no domínio de frequência impacta as imagens em seu domínio original. Além disso, será analisada uma técnica de compressão de imagens baseada também na manipulação do espectro de Fourier.

Os programas responsáveis por realizar esses processamentos nas imagens foram escritos na linguagem Python 3.10, com o auxílio das bibliotecas NumPy, OpenCV e matplotlib, utilizadas para a manipulação, transformações das imagens e exibição de gráficos. Com esse ferramental, foram desenvolvidos os seguintes scripts/programas, que se encontram no dirtório src/ e com suas intruções de execução no arquivo README:

- `fft_filters.py`: Dado uma imagem, gera uma nova imagem na qual um filtro no domínio de frequência foi aplicado e opcionalmente gera a magnitude do domínio de frequência da imagem após a filtragem;
- `fft_compress.py`: Dado uma imagem, utiliza uma técnica de compressão baseada no espectro de Fourier para gerar uma nova imagem comprimida em um caminho provido pelo usuário.
- `fft_mag_spec.py`: Dado uma imagem, gera o espectro de Fourier desta imagem em caminho provido pelo usuário.

- `fft_utils.py`: Contém funções utilitárias para calcular a FFT de uma imagem, obter o espectro de magnitude e reconstruir a imagem a partir da FFT.
- `histogram.py`: Dado uma imagem, gera o histogram desta imagem em um caminho provido pelo usuário.

2 Interpretação da Transformada de Fourier aplicada a imagens

A Transformada de Fourier de uma imagem transforma a informação espacial da imagem em informação de frequência. Cada ponto na transformada representa uma determinada frequência espacial e sua magnitude representa a contribuição dessa frequência para a imagem original.

Componentes de baixa frequência (rigões mais claras do espectro de Fourier) na Transformada de Fourier correspondem a variações suaves na imagem, como áreas uniformes, enquanto componentes de alta frequência (regiões mais escuras) representam mudanças rápidas na intensidade, como bordas e detalhes finos. No centro da Transformada de Fourier estão as baixas frequências, e as altas frequências estão na periferia.

Além disso, a Transformada de Fourier pode revelar a orientação das características da imagem, sendo útil para analisar padrões direcionais como linhas ou bordas. O espalhamento espectral, que se refere à distribuição de energia no domínio da frequência, também é um conceito importante. Imagens mais homogêneas ou suaves terão mais energia concentrada em frequências baixas, enquanto aquelas com alto conteúdo de frequência terão uma energia mais distribuída em torno do centro.

Nota-se então que há tranformada de Fourier possui diversas aplicações em processamentos de imagens, contudo o presente trabalho focará em análisar apenas as seguintes aplicações:

- Filtragem de imagens: Utilizar a compreensão da distribuição das frequências da Transformada de Fourier para atenuar determinadas frequência possibilita a implementação dos filtros passa-alta, passa-baixa, passa-faixa e rejeita-faixa
- Compressão de imagens: Atenuando as baixas frequências do espcetro de fourier, dado um limiar, pode gerar uma imagem com caracteristicas semelhantes a original, todavia teóricamente menor, devido a redução de informação de seu conteúdo.

3 Implementação dos escripts utilizados para geração de imagens filtradas e comprimidas

3.1 fft_utils.py

O módulo fft_utils.py encapsula funções utilitárias relacionadas a manipulação de imagens por meio da transformada rapida de fourier, comuns entre os diferentes escripts presentes neste trabalho, evitando reescrita de código. Esse módulo faz uso da biblioteca numpy, referênciado no código como np, para manipular as imagens.

A função get_fft_from_img recebe uma matriz de imagem img como entrada e calcula a transformada de fourier da imagem. Primeiro, aplica a FFT, por meio da função np.fft.fft2, na imagem de entrada, transformando-a para o domínio da frequência. Em seguida, desloca o componente de frequência zero da FFT para o centro da matriz, com o uso da função np.fft.fftshift, para prepará-lo

para futuros processamentos adicionais. Por fim, retorna a Transformada de Fourier deslocada da imagem de entrada.

A função get_mag_from_fft recebe a transformada de fourier deslocada fft_shift de uma imagem como entrada. Calcula o espectro de magnitude a partir da FFT deslocada, representando o conteúdo de frequência da imagem. O espectro de magnitude é calculado usando a fórmula 20 * np.log(np.abs(fft_shift)), que computa o logaritmo do valor absoluto da FFT, escalado por 20 para fins de visualização. O espectro de magnitude resultante destaca a intensidade de diferentes componentes de frequência na imagem.

A função get_img_from_fft recebe a transformada de fourier deslocada fft_shift como entrada e reconstrói a imagem a partir dela. Primeiro, aplica a inversa da FFT, por meio da função np.fft.ifft2, na FFT deslocada para transformá-la de volta para o domínio espacial. Em seguida, aplica o inverso do deslocamento, através da função np.fft.ifftshift), para ajustar o componente de frequência zero de volta à sua posição original. Por fim, extrai a parte real da inversa da FFT, utilizando np.real, para obter a imagem reconstruída. Tomar a parte real remove quaisquer artefatos imaginários introduzidos durante as operações de FFT.

3.2 fft_mag_spec.py

O script fft_mag_spec.py tem como objetivo gerar uma imagem que representa a magnitude do domínio de frequência de uma imagem, utilizando funções da biblioteca OpenCV e do módulo fft_utils.py. Ele recebe como parâmetros de execução o caminho da imagem de entrada, na qual serão realizadas as operações necessárias, e um caminho para salvar o resultado obtido das operações. O script lê a imagem utilizando a função imread, e em seguida aplica as funções get_fft_from_img e get_mag_from_fft em sequência. Essas funções geram, respectivamente, a FFT de uma imagem e a imagem de magnitude a partir da FFT. Por fim, a função imwrite é utilizada para escrever a imagem de magnitude no caminho fornecido pelo usuário

3.3 histogram.py

O script histogram.py recebe como parametros de execução o caminho a uma imagem, cujo o histagram será gerado, e o caminho no qual este histogram será salvo. Para realizar essa função são utilizados os metódos imread, hist, savefig das biblitecas OpenCV e matplotlib. Essas funções possuem respectivamente o papel de ler a imagem, gerar o histograma da imagem e salvar o histograma desta imagem.

3.4 fft_filters.py

O programa fft_filters.py implementa os filtros passa-alta, passa-baixa, passa-faixa e rejeita-faixa no domínio de frequência das imaggens, aproveitando a técnica de Transformada Rápida de Fourier (FFT). Com essa capacidade de manipular o conteúdo de frequência das imagens, ele oferece funcionalidades como redução de ruído, detecção de bordas e realce de características.

Ao ser executado, o script importa bibliotecas essenciais, incluindo sys, os, cv2 (OpenCV), numpy e o módulo fft_utils, que contém funções utilitárias para transformação de imagens.

Dentro do script, funções de utilidade são definidas para gerar máscaras circulares, fundamentais para operações de filtragem no domínio de frequência. Essas máscaras determinam quais componentes de frequência são mantidos ou atenuados durante o processo de filtragem.

O coração do script está em suas funções de filtragem, cada uma adaptada a um método específico de filtragem:

- `apply_lower_pass_filter`: Esta função atenua componentes de alta frequência enquanto preserva informações de baixa frequência. Isso é alcançado multiplicando a FFT da imagem por uma máscara que permite a passagem de conteúdo de baixa frequência.
- `apply_higher_pass_filter`: Por outro lado, esta função atenua componentes de baixa frequência enquanto retém detalhes de alta frequência. Isso é alcançado multiplicando a FFT da imagem por uma máscara circular que bloqueia o conteúdo de baixa frequência.
- `apply_band_pass_filter`: Esta função retém componentes de frequência dentro de uma faixa especificada enquanto atenua aqueles fora da faixa. Ela utiliza os parametros da chamada para combina as operações de apply_higher_pass_filter e apply_lower_pass_filter alcançando isso.
- `apply_reject_band_filter`: Similarmente, esta função atenua componentes de frequência dentro de uma faixa especificada enquanto retém aqueles fora da faixa. Ela utiliza a função apply_band_pass_filter para gerar um espectro que mantem valores da faixa a ser removida e após subtrai este espectro daquele de entrada.

Na função principal do script, as opções da linha de comando são analisadas utilizando o módulo getopt, o que permite aos usuários especificarem o método de filtragem, os parâmetros do filtro e se desejam gerar imagens do espectro de magnitude. Os métodos de filtragem disponíveis são: "low"(passa-baixa), "high"(passa-alto), "band"(passa-faixa) e "reject"(rejeita-faixa). Cada método representa a intenção do usuário ao aplicar o filtro no domínio de frequência. Quanto aos parâmetros, estes são os valores dos raios, representados por valores inteiros, dos círculos que serão utilizados como máscara na filtragem. Exemplos de commandos de execução estão presentes no arquivos README do diretório raiz deste projeto.

Após a validação das opções fornecidas, o script lê a imagem de entrada em escala de cinza utilizando a função imread do OpenCV e aplica a FFT para obter sua reperesentação no domínio de frequência por meio da função get_fft_from_img do módulo fft_utils. Além disso, utilizando a função get_mag_from_fft do mesmo módulo, o script obtém a magnitude do espectro de Fourier a partir de sua representação no domínio de frequência.

Com base no método especificado, a função de filtro correspondente é aplicada à FFT da imagem e sua magnitude, modificando o conteúdo de frequência de acordo com a técnica de filtragem escolhida. Isso significa que os coeficientes da FFT que representam as frequências indesejadas são modificados ou atenuados de acordo com as características do filtro selecionado. Uma vez concluído o processo de filtragem no domínio de frequência, o script reconstrói a imagem filtrada a partir da FFT modificada, através do comando get_img_from_fft do módulo fft_utils, retornando-a ao domínio espacial por meio da inversa da FFT. Esse processo restaura a imagem filtrada à sua representação original, onde as alterações no conteúdo de frequência são refletidas como mudanças nas intensidades dos pixels. Assim, a imagem resultante mantém as características desejadas após o processo de filtragem no domínio de frequência.

O manuseio de saída envolve a criação de diretórios para salvar as imagens filtradas e, opcionalmente, salvar a imagen do espectro de magnitude processada. Com isso o programa termina sua execução e prove a usúario a imagem filtrada e opcionalmente a representação da magnitude de seu aspectro de fourier no path output_image/metedo de filtragem/.

3.5 fft_compress.py

O programa fft_compress.py foi desenvolvido para realizar a compressão de imagens por meio da aplicação da Transformada Rápida de Fourier (FFT), implementando uma técnica baseada em remover os coeficientes menos significativos da aspectro de fourier de uma imagem. Essa operação é

feita com base em um threshold fornecido pelo usúario e resulta em uma imagem semelhante aquela fornecida por ele, porém menor em quesitos de tamanho na memória.

Ao executar o script, ele primeiro importa as bibliotecas essenciais, incluindo sys, os, numpy e cv2 (OpenCV). Essas bibliotecas fornecem as funcionalidades necessárias para manipulação de arquivos, processamento de imagens e operações numéricas.

A funcionalidade central do script está encapsulada na função main, que é a entrada principal do programa. Esta função é responsável por processar os argumentos passados pela linha de comando e executar o algoritmo de compressão de imagens.

O conjunto de instruções iniciais da função main tem como objetivo analisar os argumentos da linha de comando. Nesta etapa, o script verifica se há argumentos suficientes para realizar a compressão da imagem. Em seguida, são extraídos o nível de compressão desejado, o caminho para a imagem de entrada, na qual o algoritmo de compressão será aplicado, e o caminho onde a imagem comprimida será salva. O nível de compressão é um valor de ponto flutuante que varia entre 0 e 1. O valor 0 indica que nenhum nível de compressão será aplicado, ou seja, nenhum coeficiente do espectro de Fourier da imagem será removido, enquanto o valor 1 significa que todos os coeficientes serão removidos. Qualquer outro valor dentro deste intervalo define o nível de compressão a ser aplicado. Antes de prosseguir com a compressão, o script valida se o nível de compressão fornecido está dentro do intervalo válido.

Em seguida, o script lê a imagem de entrada em escala de cinza utilizando a função imread do OpenCV. A imagem é então transformada no domínio da frequência aplicando a FFT, por meio da função get_fft_from_img do modúlo fft_utils.

Após a transformação no domínio da frequência, o script procede determinando um limiar com base no nível de compressão especificado pelo usuário. Esse limiar é uma referência que permite ao algoritmo decidir quais coeficientes da FFT serão mantidos e quais serão removidos para alcançar o nível desejado de compressão. O cálculo desse limiar é baseado na magnitude dos coeficientes da FFT da imagem. Coeficientes de frequência abaixo do limiar são definidos como zero, reduzindo assim a quantidade de dados necessários para representar a imagem. Observe que o nível de compressão é um parâmetro usado para definir a porcentagem dos coeficientes de maior magnitude que devem ser mantidos na imagem.

Posteriormente, a imagem comprimida é reconstruída a partir da FFT modificada, utilizando uma função get_img_from_fft do módulo fft_utils. Esta etapa restaurar a imagem original a partir de sua representação no domínio da frequência. Por fim, a imagem processada é escrita no arquivo de saída especificado utilizando a função cv.imwrite do OpenCV, e o processo de compressão é concluído.

4 Análise dos Resultados Obtidos

4.1 Filtragem no espectro de frequência

As imagens de entradas utilizadas nos testes estão presentes no diretório input_images. A suas versões processadas, analisadas nesta seseção, são aquelas filtradas pelo script fft_filter.py, cujo estão localizadas no diretório output_images, organizadas de modo que cada método de filtragem no espectro de frequência tenha um subdiretório para armazenar as imagens resultantes da aplicação desse método, assim como as imagens das magnitudes do espectro de frequência de cada uma dessas imagens. Portanto, os subdiretórios low, high, band e reject, presentes no caminho output_images, contêm as imagens filtradas com diferentes parâmetros de cada método.

4.1.1 Filtro Passa-Baixa

O processo de filtragem passa-baixa utilizando a transformada de Fourier consiste em atenuar as altas frequências (bordas e transições abruptas) do espectro de Fourier, ou seja, eliminar as componentes periféricas ao centro da transformada. Esse método teoricamente provoca uma suavização das bordas, o que gera um efeito de borramento no conteúdo original da imagem.

Dada a característica radial do espectro de Fourier, a seguinte fórmula foi aplicada na forma de uma mascara sobre o espectro da imagem de entrada:

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

Onde $D(u,v)$ representa a distância entre um ponto (u,v) na frequência e o centro da transformada, e D_0 é o raio de corte.

As imagens resultantes do filtro passa-baixa estão presentes no diretório output_images/low/. Observe, por exemplo, a Figura 4, uma das imagens contidas neste diretório, na qual foi aplicado o filtro passa-baixa em seu domínio de frequência com raio de corte de 80 pixels, representado na figura 5. Ao comparar esta imagem com sua versão de entrada, figura 1, nota-se que o resultado do filtro passa-baixa gera uma espécie de borramento, provocando a sensação de menos nitidez na imagem.

Além da figura 4, no iretório output_images/low/, estão presentes outras imagens filtradas, tendo como base a figura 1. O que se nota é que quanto menor o raio do círculo aplicado na filtragem do domínio de frequência, menos nítida a imagem fica. Ou seja, muitas das frequências consideradas altas estão sendo atenuadas, enquanto as regiões homogêneas estão sendo mantidas nas imagens.

4.2 Filtro Passa-Alta

O filtro passa-alta é uma técnica de processamento de imagem que enfatiza as altas frequências, realçando bordas e detalhes finos, enquanto atenua as baixas frequências, suavizando regiões homogêneas. Isso é realizado eliminando as componentes centrais do espectro de Fourier e mantendo as frequências periféricas.

Dada a característica radial do espectro de Fourier, a fórmula utilizada para o filtro passa-alta é complementar à do filtro passa-baixa. Enquanto o filtro passa-baixa atua permitindo apenas as frequências dentro de um determinado raio e atenuando as demais, o filtro passa-alta faz o oposto, atenuando as frequências dentro de um raio e permitindo as demais.

Assim, a fórmula para o filtro passa-alta pode ser definida como:

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

Onde $D(u,v)$ representa a distância entre um ponto (u,v) na frequência e o centro da transformada, e D_0 é o raio de corte.

As imagens resultantes do filtro passa-alta estarão presentes no diretório output_images/high/. Observe as figuras 6 e 7, imagens contidas neste diretório, e a imagem original de entrada do script de filtro, figura 1. Nota-se que ao contrário do filtro passa-baixa, ao aplicar o filtro passa-alta, as frequências baixas que representam regiões homogêneas são atenuadas, enquanto as frequências altas, como bordas e detalhes finos, são mantidas, resultando em uma imagem com bordas realçadas em comparação a imagem original.

Analisando outras imagens deste diretório, percebe-se que um raio muito grande pode levar a efeitos indesejados, como a amplificação de ruído ou a perda de detalhes sutis. No caso da imagem 1, ao aplicar um raio de 180 pixel, apenas as componentes com extrema alta frequência, os pelos da macaco, são mantidas. O caso oposto, aquele cujo um raio muito pequeno é aplicado pode provocar um efeito contrário ao desejado de um filtro passa-alta, dado que as componentes de baixa frequência podem não ser atenuadas, devido o tamanho do circulo aplicado. O que percebe-se então é que um raio a ser utilizado na aplicação deste filtro, deve ser selecionados com base em experimentos para cada imagem.

4.3 Filtro Passa-Faixa

O filtro passa-faixa é uma técnica de processamento de imagem que permite a passagem apenas de um intervalo específico de frequências, enquanto atenua as frequências fora desse intervalo. Isso é útil quando se deseja realçar ou destacar características de uma imagem que estão contidas em uma faixa de frequência particular.

Para implementar o filtro passa-faixa, pode-se utilizar uma combinação de filtros passa-baixa e passa-alta. Primeiramente, um filtro passa-baixa é aplicado para atenuar as altas frequências e manter as baixas frequências dentro de uma determinada faixa. Em seguida, um filtro passa-alta é aplicado para atenuar as baixas frequências e realçar as altas frequências dentro dessa mesma faixa.

A fórmula para o filtro passa-faixa pode ser representada como a multiplicação das funções de transferência dos filtros passa-baixa e passa-alta:

$$H_{\text{passa-faixa}}(u, v) = H_{\text{passa-baixa}}(u, v) \cdot H_{\text{passa-alta}}(u, v)$$

Onde $H_{\text{passa-baixa}}(u, v)$ e $H_{\text{passa-alta}}(u, v)$ são as funções de transferência dos filtros passa-baixa e passa-alta, respectivamente. Nesta formulação estão implicitos os raios de corte utilizados na filtragem passa-alta e passa-baixa.

As imagens resultantes do filtro passa-faixa estão presentes no diretório output_images/band/. Observe as figuras 8 e 9, imagens contidas neste diretório, e a imagem original utilizada como entrada ao script de filtro, figura 1. Nota-se que ao aplicar o filtro passa-faixa, apenas as frequências contidas na faixa especificada serão mantidas, resultando em uma imagem que destaca características nessa faixa de frequência específica.

É importante ajustar os parâmetros dos filtros passa-baixa e passa-alta de acordo com as características da imagem e o intervalo de frequências desejado. Experimentos com diferentes valores de raio e faixa de frequência são recomendados para encontrar o melhor resultado para uma determinada aplicação.

4.4 Filtro Rejeita-Faixa

O filtro rejeita-faixa, é uma técnica de processamento de imagem que atenua ou remove seletivamente uma faixa específica de frequências, enquanto permite a passagem das frequências fora dessa faixa. Ele pode ser útil para remover ruídos ou artefatos que estão presentes em frequências específicas, sem afetar o restante da imagem.

Similar ao filtro passa-faixa, o filtro rejeita-faixa pode ser implementado combinando os princípios dos filtros passa-baixa e passa-alta. No entanto, ao invés de apenas multiplicar as funções de transferência dos dois filtros, é necessário remover essa componente da imagem original, para criar o efeito de rejeição na faixa de frequência desejada.

A fórmula para o filtro rejeita-faixa pode ser representada como:

$$H_{\text{rejeita-faixa}}(u, v) = 1 - H_{\text{passa-faixa}}(u, v)$$

Onde $H_{\text{passa-faixa}}(u, v)$ é a função de transferência do filtro passa-faixa. Nesta formulação estão implicitos os limiares que definem o intervalo de frequência a ser mantido pela função passa-faixa.

As imagens resultantes do filtro rejeita-faixa estão presentes no diretório output_images/reject/. Observe as figuras 10 e 11, imagem contidas neste diretório, e a imagem original utilizada como entrada para o script de filtro, figura 1. Nota-se que ao aplicar o filtro rejeita-faixa, as frequências contidas dentro da faixa especificada serão removidas, enquanto as frequências fora dessa faixa serão preservadas.

Assim como nos outros tipos de filtros no domínio de frequência, a escolha dos parâmetros para o filtro rejeita-faixa é crucial e depende das características da imagem e do tipo de ruído ou artefato a ser removido. Experimentos com diferentes valores de raio e faixa de frequência são recomendados para encontrar os melhores resultados para uma determinada aplicação.

4.5 Compressão

As imagens de entrada utilizadas nos testes estão presentes no diretório input_images. Os histogramas e magnitudes do espectro de Fourier destas imagens estão presentes, respectivamente, nos diretórios output_images/hist e output_images/mag. As versões processadas, analisadas nesta seção, são aquelas processadas pelo script fft_compress.py, presentes no diretório output_images/compress, juntamente com seus histogramas e magnitudes do espectro de Fourier, gerados pelos scripts fft_mag_spec.py e histrogram.py.

Dado um limiar, a técnica de compressão utilizada elimina todos os coeficientes abaixo dele. A intenção é manter os coeficientes mais significativos para representação da imagem e remover aqueles de menor frequência.

Em teoria, esse processo reduziria o espaço em memória ocupado pelos objetos comprimidos. Isso na prática tornou-se verdadeiro. Observe as imagens no diretório output_images/compress, com prefixo baboon e sufixo um número, geradas a partir da imagem presente na figura 1. Estas imagens estão nomeadas de modo a descrever qual imagem e nível de compressão foram utilizados para gerá-las. O nível de compressão está dado nos últimos dígitos. Por exemplo, 9999 da figura 15 significa que o nível de compressão utilizado foi de 0,999. Nota-se que quanto maior o nível de compressão dessas imagens, menor é o espaço que elas ocupam em disco. As figuras 1, 12 e 15 ocupam, respectivamente, 234KB, 203KB e 79KB em disco.

O custo de se reduzir o tamanho através da técnica de compressão é notável. Ao observar as figuras citadas anteriormente, percebe-se que o maior nível de compressão (0,999), figura 15, apesar de reduzir quase três vezes o tamanho da imagem, a torna incompreensível. Conclui-se então que uma análise experimental é necessária para encontrar o nível adequado às necessidades vigentes.

Além disso, a análise dos histogramas das figuras 3, 14 e 17, nos permite ter uma noção do comportamento do algoritmo. É perceptível que quanto maior o nível de compressão, mais estreita é a faixa na qual os valores do histograma estão dispostos, indicando que a imagem está ficando menos nítida e mais borrada. Interessante também é analisar as figuras 2, 13 e 16, que representam a magnitude do espectro de Fourier. Nelas, é visível que quanto maior o nível de compressão, menores são as baixas frequências presentes na imagem.

5 Figuras

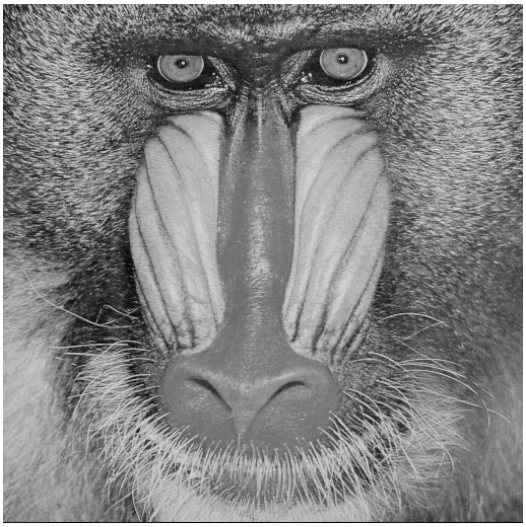

Figura 1: Imagem input_images/baboon.png, utilizada como base para aplicação de filtros e compressão

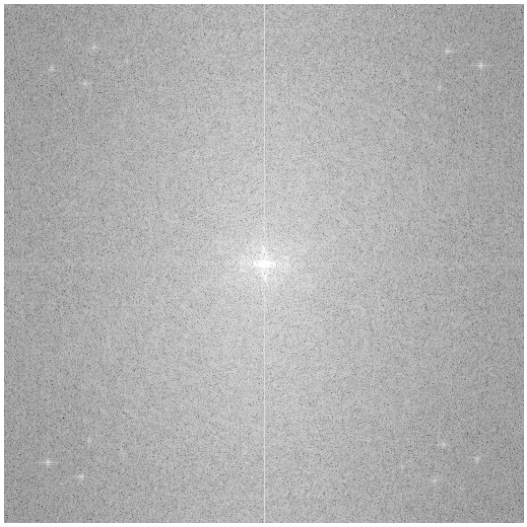

Figura 2: Imagem output_images/mag/baboon_mag.png. Mangnitude do espectro fourier da figura 1

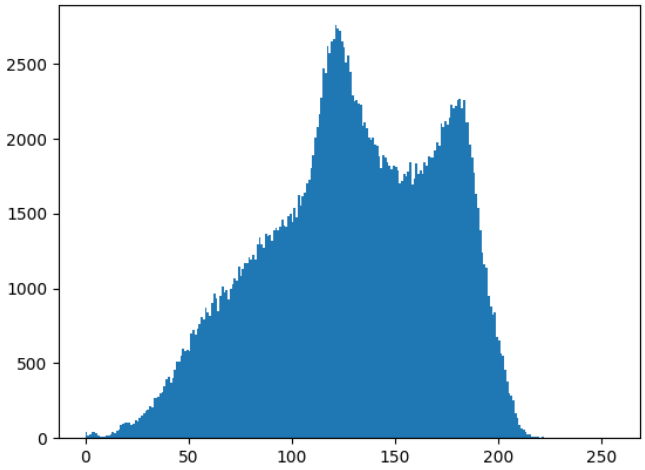

Figura 3: Histograma output_images/hist/baboon_hist.png. Este é o historgrama da figura 1

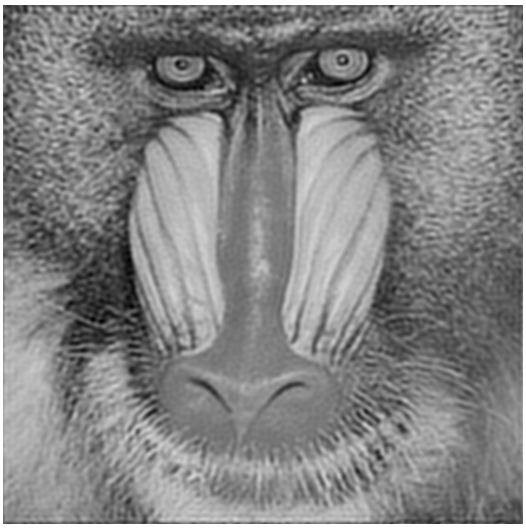

Figura 4: Imagem out_imges/low/baboon_low_80.png.Figura 1 após apliação de um filtro passa-baixa de raio 80 pixel

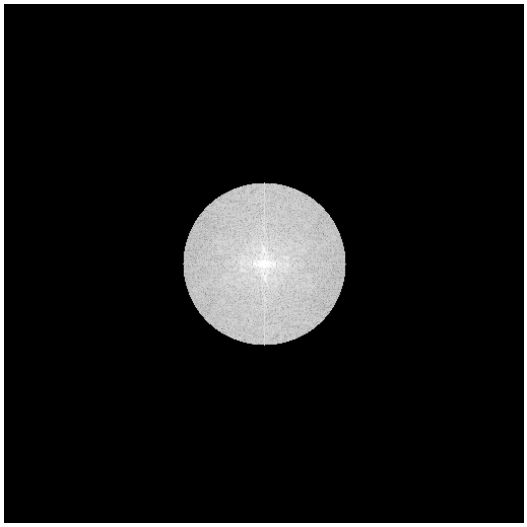

Figura 5: Filtro com raio de 80 pixel aplicado no espectro de frequência da figura 1, que gera a figura 4

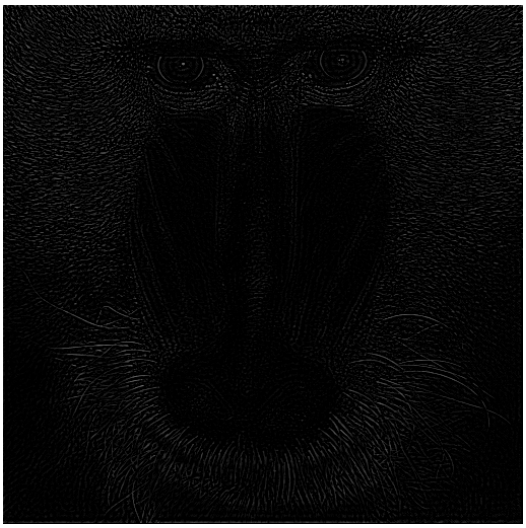

Figura 6: Imagem out_imges/high/baboon_high_80.png. Figura 1 após apliação de um filtro passa-alta de raio 80 pixel

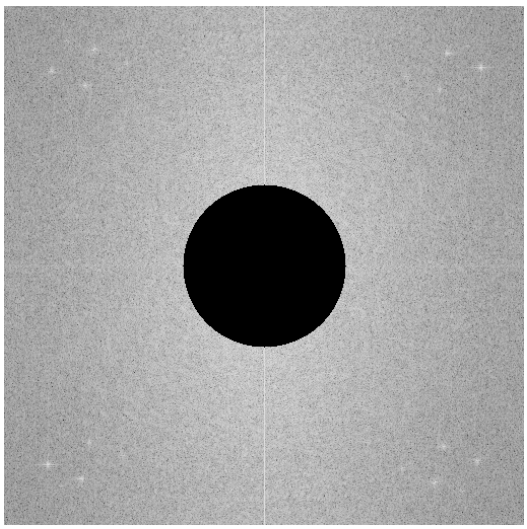

Figura 7: Filtro passa-alta com raio de 80 pixel aplicado no espectro de frequência da figura 1, que gera a figura 6

Figura 8: Imagem out_imges/band/baboon_band_70_80.png. Figura 1 após apliação de um filtro passa-faixa no intervalo radial de 30 a 80 pixel

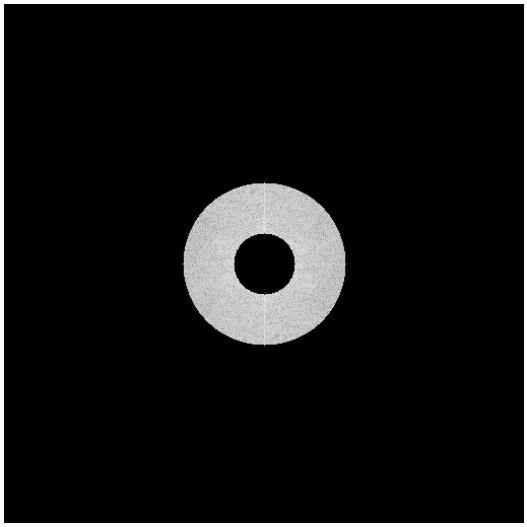

Figura 9: Filtro passa-faixa com intervalo radial de 30 a 80 pixel aplicado no espectro de frequência da figura 1, que gera a figura 8

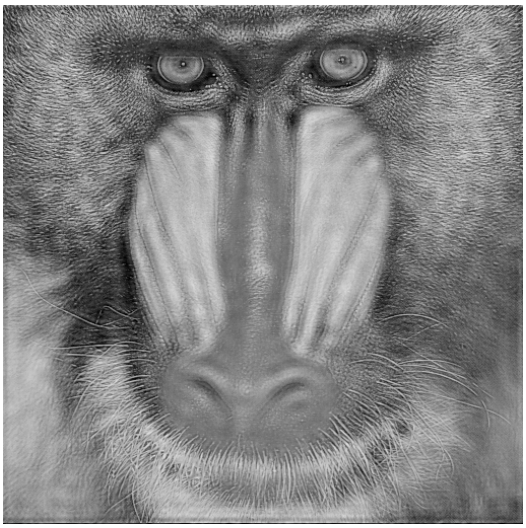

Figura 10: Imagem out_imges/reject/baboon_reject_30_80.png. Figura 1 após apliação de um filtro rejeita-faixa no intervalo radial de 30 a 80 pixel

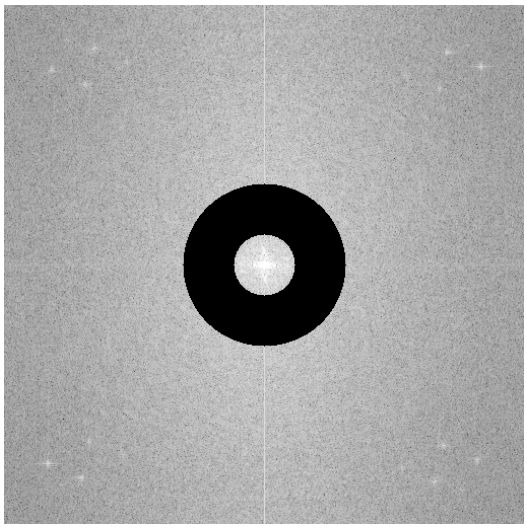

Figura 11: Filtro rejeita-faixa com intervalo radial de 30 a 80 pixel aplicado no espectro de frequência da figura 1, que gera a figura 10

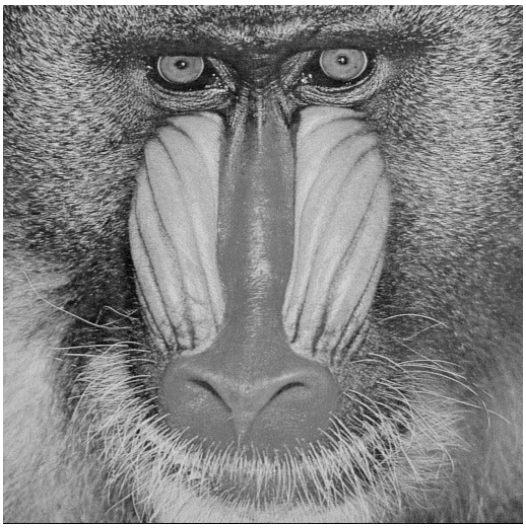

Figura 12: Imagem out_imges/compress/baboon_0_5.png. Figura 1 após compressão com nivel de 0.5

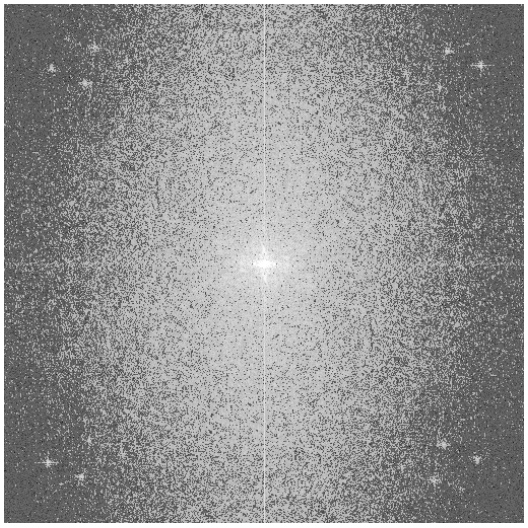

Figura 13: Imagem output_image/compress/baboon_0_5_mag.png. Magnitude do espectro de fourier da figura 12

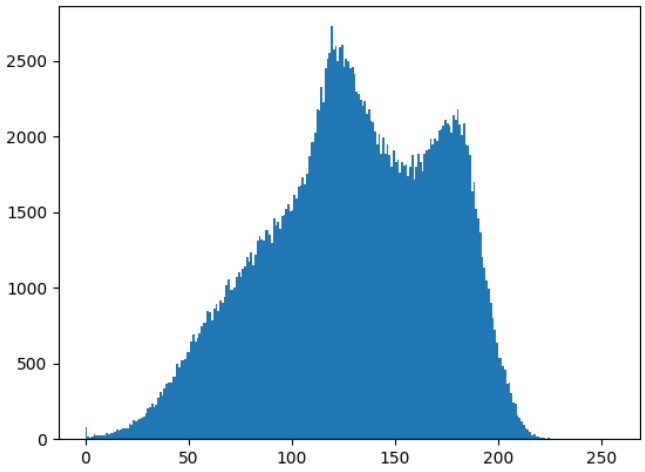

Figura 14: Histograma output_image/compress/baboon_0_5_hist.png. Histograma da figura 12

Figura 15: Imagem out_imges/compress/baboon_0_999.png. Figura 1 após compressão com nivel de 0.999

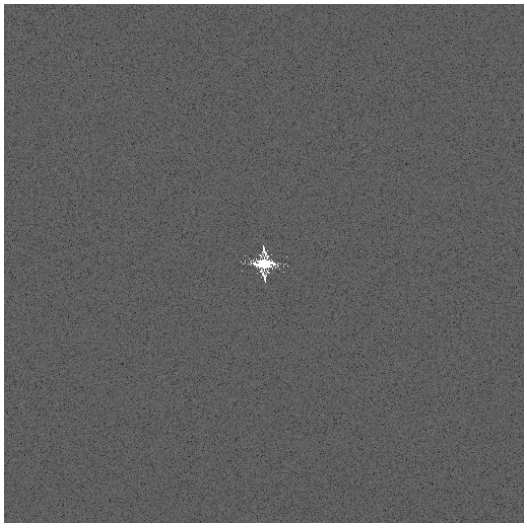

Figura 16: Imagem output_image/compress/baboon_0_999_mag.png. Magnitude do espectro de fourier da figura 15

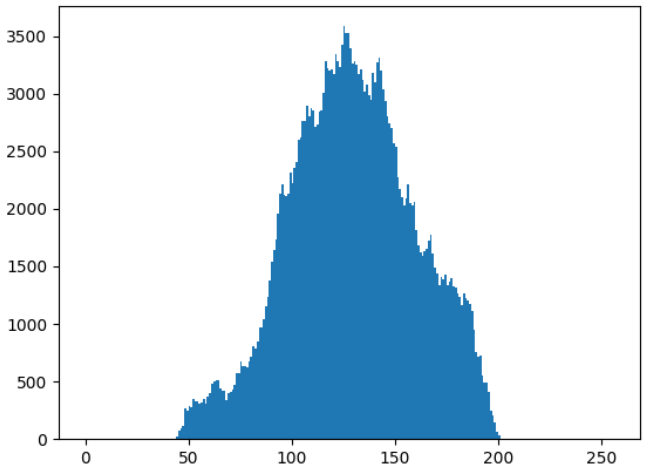

Figura 17: Histograma output_image/compress/baboon_0_999_hist.png. Histograma da figura 15