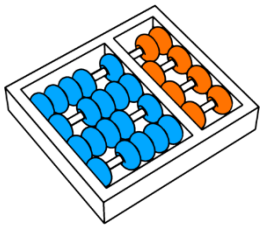


Universidade Estadual de Campinas

Instituto de Computação

Introdução ao Processamento Digital de Imagem (MC920A)

Professor: Hélio Pedrini

Relatório – Trabalho 2

Gabriel Augusto Bertolino Gomes

RA: 248287

26/04/2024

1. Introdução

A transformada de Fourier aplicada em processamento de imagem é capaz de explorar operações utilizando o domínio de frequência. Em especial a filtragem, que auxilia a mudança dos valores originais de entrada, visando realçar a imagem, destacar alguns pontos, reduzir ruídos, até mesmo evidenciar alguns contornos, entre outras possibilidades. Bem como, uma das aplicações importantes é a compressão, que é realizada eliminando os coeficientes de menor magnitude.

Esse trabalho tem por objetivo aplicar tais conceitos na prática, analisando os resultados e explorando algumas possibilidades.

2. Materiais e métodos

Os materiais usados para realizar as filtragens no domínio de frequência foram:

- Python 3.11.6
  - Bibliotecas
    - opencv-python (cv2) 4.9.0.80
    - numpy 1.26.1
    - matplotlib 3.8.4
- Visual Studio Code
- Imagens usadas: buttlerfly_cinza.png

Para além desses itens, abaixo encontra-se a estrutura de pasta como estão organizados o código e os arquivos anexos que são itens primordiais para o funcionamento do programa.

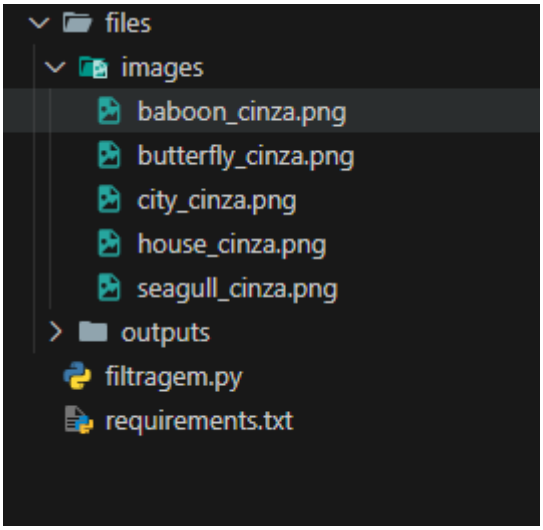


Figura 1 – Estrutura de pasta com os códigos fontes e anexos

Caso seja necessário a instalação de todas as dependências do programa, basta executar no terminal o seguinte comando:

- pip install -r requirements.txt

Isso será responsável por instalar todas as bibliotecas que são dependências do programa que realiza a filtragem usando a transformada de Fourier.

3. Modo de Uso

Com as dependências instaladas, agora deve-se executar o programa para então obter os resultados dos filtros aplicados, bem como das compressões realizadas. A execução do código filtragem.py deve seguir o seguinte formato, abra o terminal para executar o comando:

- python filtragem.py <imagem_entraga.png> <pasta_arquivo_saida> <raio_menor_filtro> <raio_maior_filtro>

    - **<imagem_entraga.png>:** Caminho para imagem de entrada que será usada como base das operações
    - **<pasta_arquivo_saida>:** Pasta, onde os arquivos de saída serão salvos
    - **<raio_menor_filtro>:** Raio menor que será usado na construção da máscara dos filtros
    - **<raio_maior_filtro>:** Raio maior que será usado na construção da máscara dos filtros

4. Resultados e Discussões

É necessário, agora, construir a lógica do algoritmo que aplica a Transformada de Fourier na imagem de entrada e cria os filtros: passa baixa, passa alta, passa faixa e rejeita faixa; e, a partir da inversa da Transformada de Fourier, obtém de volta a imagem filtrada. E, por fim, comprime todas as saídas.

4.1. Transformada Discreta de Fourier (DFT)

De início, é feita a leitura da imagem de entrada. A leitura é realizada de modo a garantir que a imagem obtida esteja em escala de cinza. Isto é, a imagem, terá em seu conteúdo valores entre 0 e 255, que são os possíveis níveis de cinza.

Com a imagem em mãos, é possível realizar a Transformação Discreta de Fourier, cuja fórmula segue:

$$F(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) e^{-i2\pi(\frac{ux}{M}+\frac{vy}{N})}$$

Em M × N é o tamanho da imagem digital, $u = 0, 1, \dots, M-1$ e $v = 0, 1, \dots, N-1$.

É claro, que a biblioteca utilizada (opencv-python) usa de um outro algoritmo chamado **Fast Fourier Transform (FFT)**, que é mais otimizado, porém utiliza do mesmo princípio. Por meio da fórmula da transformação, é perceptível que o número obtido para $F(u,v)$ é um número complexo, com uma componente real e uma componente complexa. Dessa forma, a imagem resultante de Fourier terá o seguinte shape $(M, N, 2)$, isso é importante pois é um ponto determinante no momento de construir os filtros mais à frente.

Posteriormente, com o DFT aplicado, é preciso shiftar o componente de zero frequência da transformada para o centro imagem. Por construção, o algoritmo posiciona o componente de menor frequência na parte esquerda superior da imagem. Para tanto, esse posicionamento dificulta a aplicação de máscaras, enquanto, as baixas frequências no meio ajudam e auxiliam nesse processo. O método utilizado para executar essa tarefa, encontra-se na biblioteca numpy e chama-se ***fft.fftshift().*** O centro da imagem DFT é dado por $(c_{x_{DFT}}, c_{y_{DFT}}) = (\frac{M}{2}, \frac{N}{2})$.

Para visualizar os resultados do processo, faz-se necessário obter a magnitude do espectro de Fourier, para isso, é aplicada a seguinte fórmula:

$$S(u,v) = \log|F(u,v)|$$

Em que S é a matriz que possui a magnitude do espectro de Fourier para posição (u,v) na matriz da transformada.

O resultado obtido com a transformação, juntamente com uma das imagens de entrada pode ser visto na figura 2, a seguir.

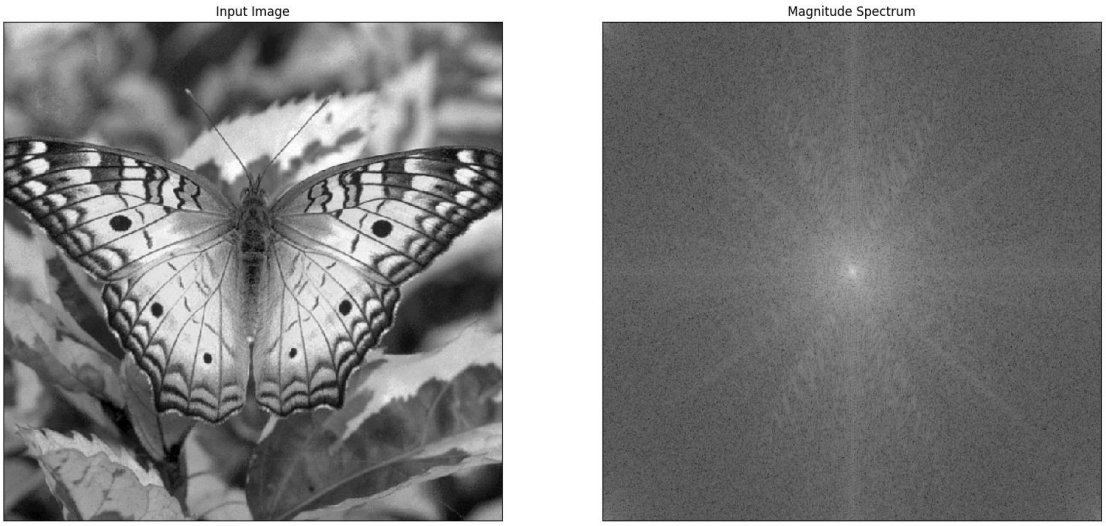


Figura 2 – Imagem de entrada e o resultado de sua Transformada de Fourier

4.2. Construção das Máscaras para Filtragem

A ideia inicial é construir quatro máscaras que serão responsáveis pela filtragem da imagem de entrada utilizando-se da Transformada de Fourier. Os filtros terão quatro categorias: filtro passa alta, filtro passa baixa, filtro passa faixa e filtro rejeita faixa. Para construção de tais artefatos, serão feitos na imagem dois círculos de raios r_l e r_b em que $r_l < r_b$.

4.2.1. Filtro Passa Alta

Essa filtragem tem por finalidade passar as frequências altas dentro da imagem. Ou seja, atenuar às frequências ditas baixas e favorecer as frequências altas. Tais frequências encontram-se na imagem em bordas, contornos, ruídos etc.

Fazendo uso da Transformada de Fourier, sabe-se que as frequências de maior magnitude da imagem ficam na periferia da transformação, ou seja, as frequências que ficam longe do centro. Sabendo disso, para construir o filtro passa alta, basta construir um círculo preto no centro da DFT.

Para isso, inicialmente, constrói-se uma imagem que vai ser usada como máscara com shape da DFT $(M, N, 2)$, completa por valores 1. E com isso, define-se a área do círculo da seguinte forma,

$$\left(x - c_{x_{DFT}}\right)^2 + \left(y - c_{y_{DFT}}\right)^2 \leq r_b$$

E para todo o par (x, y) que estiver nesta área deve ser setado para 0. Dessa forma, é concluída a máscara para o filtro passa alta.

4.2.2. Filtro Passa Baixa

O filtro passa baixa tem por finalidade atenuar os sinais de maior frequência e favorecer e evidenciar as menores frequências.

Dito isso, tem-se que esse filtro é o contrário do filtro passa alta. Uma vez que, os valores com menor magnitude de frequência estão no centro da imagem DFT. Com isso,

basta inverter os valores da máscara do filtro passa alta. A posição que tem valor 1, torna-se 0 na máscara do filtro passa baixa e quem tem valor 0, torna-se 1.

4.2.3. Filtro Passa Faixa

O filtro passa faixa, por sua vez, tem por finalidade atenuar as baixas e as altas frequência e evidenciar as frequências que estão dentro desse intervalo.

Para tanto, inicialmente, é preciso copiar a máscara do filtro passa baixa, da forma que foi construído no item anterior 4.2.2. Após a cópia, assim como no filtro passa alta, basta desenhar um círculo preto com raio menor. Analogamente, a área desse círculo pode ser definida da seguinte maneira,

$$\left(x - c_{x_{DFT}}\right)^2 + \left(y - c_{y_{DFT}}\right)^2 \leq r_l$$

Com isso, o par (x, y) que estiver nessa área, deve ser passado para 0. Portanto, obtém-se a máscara do filtro passa faixa.

4.2.4. Filtro Rejeita Faixa

Por fim, o filtro rejeita faixa tem o objetivo oposto ao passa faixa. Favorecer e evidenciar as baixas e altas frequências e negar as frequências que estão dentro desse intervalo.

Dito isso, para construir o filtro rejeita faixa, é preciso somente inverter os valores da máscara do filtro passa faixa e assim, obter a máscara do filtro rejeita faixa.

4.2.5. Visualização do Núcleo dos filtros

Para visualizar como os filtros irão afetar a imagem do DFT, basta multiplicar todas as máscaras pela imagem que representa a magnitude do espectro de Fourier. Todos esses resultados podem ser vistos na figura 3.

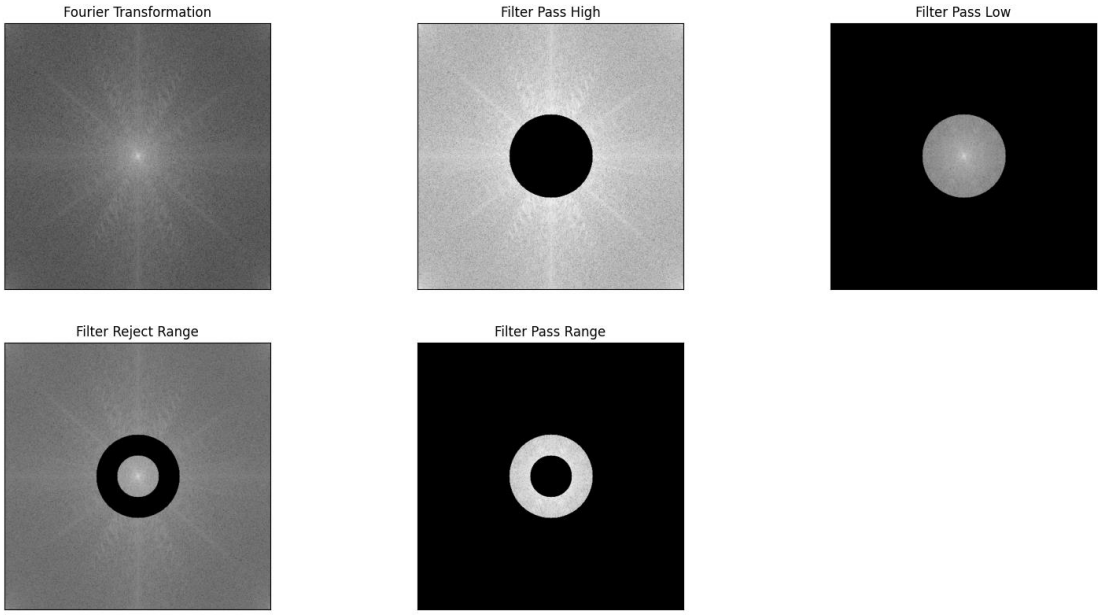


Figura 3 – Núcleo dos filtros

4.3. Aplicação da Transformada Inversa

A filtragem no domínio de frequência consiste em modificar a transformada de Fourier de uma imagem e, em seguida, calcular a transformada inversa para obter a imagem de saída com o filtro desejado.

Para realizar a modificação, é preciso somente realizar a multiplicação da transformada por alguma das máscaras construídas acima. Sabendo disso, considere $H(u,v)$ qualquer uma das máscaras discutidas nos itens anteriores e $F(u,v)$ a transformada de Fourier calculada no item 4.2. A aplicação dessa máscara no domínio de frequência consiste em realizar uma multiplicação da forma,

$$F_n(u,v) = F(u,v) \times H(u,v)$$

Em que F_n trata-se da transformada de Fourier alterada pela máscara de alguns dos filtros.

Caso o sistema estivesse no domínio espacial, a operação teria de ser uma convolução para aplicar a filtragem. Uma operação muito mais cara e mais complexa que uma simples multiplicação.

Após realizar tal tarefa, agora resta então calcular a transformada inversa de Fourier para obter novamente a imagem original com alguma das filtragens aplicadas. A transformada inversa discreta de Fourier pode ser descrita matematicamente como,

$$f(x,y) = \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u,v) e^{i2\pi(\frac{ux}{M} + \frac{vy}{N})}$$

Para aplicar essa função na imagem do DFT, é preciso antes desfazer o shift realizado nos passos iniciais. Isso deve ser feito pois a transformada inversa só tem o efeito desejado, se aplicada em uma matriz com a estrutura similar à saída do DFT, que não possui o shift.

Feito isso, tem-se a imagem com seu filtro aplicado. Porém, ela possui valores de ponto flutuante em seu conteúdo. Para salvar a saída de forma adequada é preciso normalizar os valores, de modo que o menor valor seja 0 e o maior 255, em uma escala de nível de cinza. O tipo de normalização utilizado foi o MINMAX, para adequar-se às saídas da transformada inversa de Fourier.

4.3.1. Análise das imagens filtradas

Utilizando dos passos acima, obtêm-se por fim os resultados das aplicações dos filtros na imagem de entrada, usando da transformada de Fourier. Para gerar tais imagens, precisou-se dos seguintes parâmetros de entrada, imagem de entrada: buttlefly_cinza.png, os raios para gerar as máscaras foram: $r_l = 40$ e $r_b = 80$

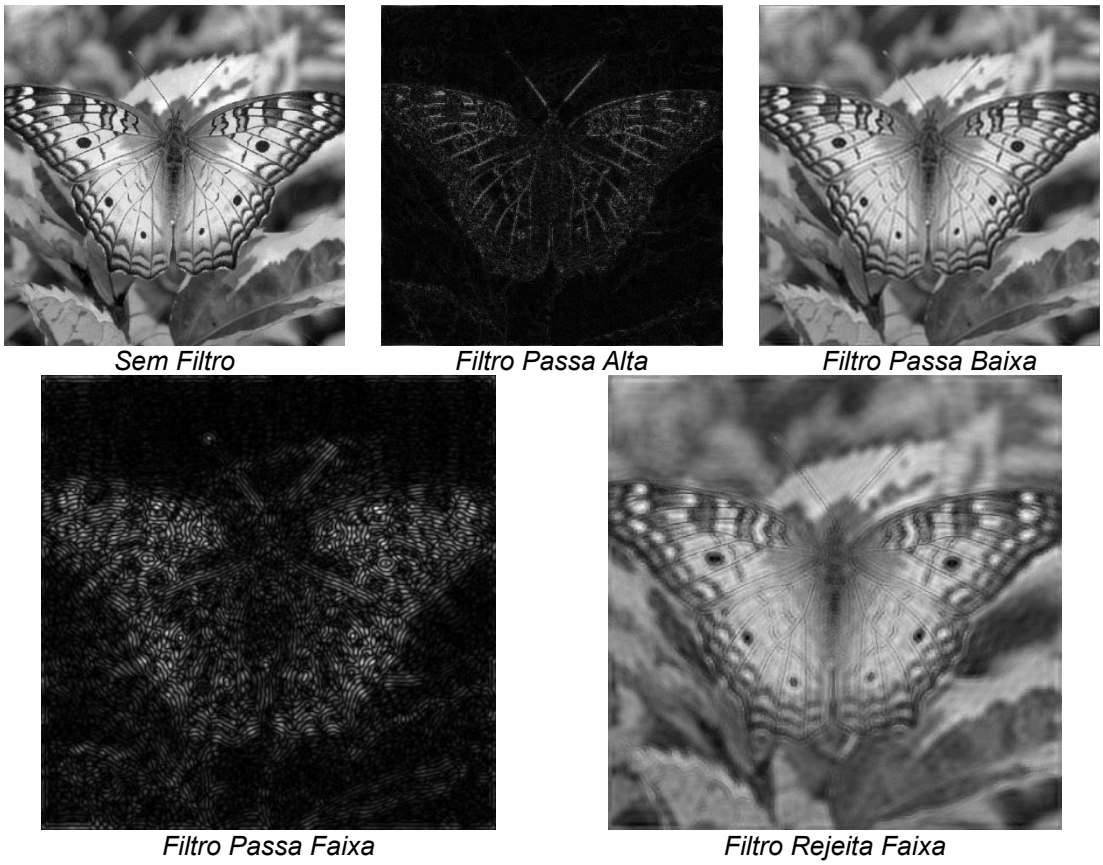


Figura 4 – Resultados das aplicações dos filtros

A primeira imagem, trata-se da transformada inversa de Fourier aplicada sobre a própria transformada de Fourier da imagem de entrada, sem nenhuma filtragem ou compressão. Logo como não há alteração no conteúdo da transformada, espera-se que a imagem de saída seja a mesma que a imagem utilizada como entrada, desconsiderando algumas incertezas de cálculos. E utilizando como base a Figura 2, pode-se ver que o resultado obtido é plenamente satisfatório, haja visto que a imagem obtida é muito similar à que foi dada como entrada.

Já para o filtro passa alta, ocorre uma imagem que ressalta os contornos e bordas da imagem da borboleta. Isso é um comportamento esperado, uma vez que as frequências de maior magnitude estão justamente nas bordas, contornos e ruídos de uma imagem.

O filtro passa baixa, por sua vez, apresenta um leve *Blur* na imagem, e é perceptível que os contornos e bordas não estão tão claros e bem definidos, uma vez que essa bloqueia as altas frequências.

O filtro passa faixa tem por definição bloquear as maiores e menores frequências, deixando passar somente uma faixa de frequência. Isso pode ser percebido pela imagem de saída que usa desse filtro. Os contornos não são bem definidos, e as áreas de baixa frequência (locais onde normalmente tem cor homogênea, sem grandes perturbações) também não é possível visualizar. O que é passível de ser entendido dentro da imagem, é a silhueta da borboleta com alguns pontos de referência, mas nenhuma área bem definida.

Por fim, o filtro rejeita faixa, que passa as altas e baixas frequências e bloqueia algumas das frequências que estão nesse meio termo. É possível ver que os contornos da imagem estão lá, juntamente com as regiões de níveis de cinza homogêneo, isto é, as baixas frequências. Contudo, é perceptível, também, a ocorrência de um *Blur* mais

acentuado, tal efeito é vindo do bloqueio da faixa de magnitude devido à máscara aplicada (Figura 3).

4.3.2. Compressão no domínio de frequência

A ideia de compressão no domínio de frequência é bastante similar às aplicações dos filtros. Uma vez que, a compressão empregada nesse contexto foi, a partir de um determinado limiar de magnitude, bloquear ou passar aquela faixa de valores a fim de comprimir a imagem de saída, com perda de informação.

Como a compressão é no domínio de frequência, ela deve ser realizada antes de aplicar a transformada inversa de Fourier. Com isso, obtém-se a magnitude da transformada, seu conteúdo é normalizado, de modo que os valores fiquem entre 0 e 255 níveis de cinza.

Usando da imagem de magnitude normalizada, determina-se um limiar ou *threshold*, e com isso gera-se uma máscara que seleciona somente os níveis de cinza que estejam acima desse valor. Tem por objetivo bloquear um certo número de coeficientes da magnitude. Com a máscara de compressão em mãos, aplica-se ela nas duas bandas da transformada original, na banda real e na banda complexa. Dessa forma, a imagem de saída da transformada inversa de Fourier estará comprimida com perda de informação. Uma vez que, ao bloquear as frequências abaixo do *threshold*, os pixels da imagem original que dependiam desses valores serão alterados e seus valores iniciais serão perdidos.

Para verificar as mudanças que a compressão realiza na imagem, é possível analisar como o histograma da imagem de saída sem compressão comporta-se e como o histograma da imagem de saída com compressão com limiar de 170 comporta-se.

Na figura 5, pode-se ver que o histograma da imagem sem compressão é esparso, de modo que praticamente todos os níveis de cinza são ocupados. O modulo de alguns níveis de cinza tem um valor bem expressivo. Enquanto para o histograma da imagem com compressão, os pixels estão concentrados nos níveis de cinza centrais, bem diferente da imagem sem compressão. Bem como, nenhum nível de cinza tem um valor discrepante dos demais.

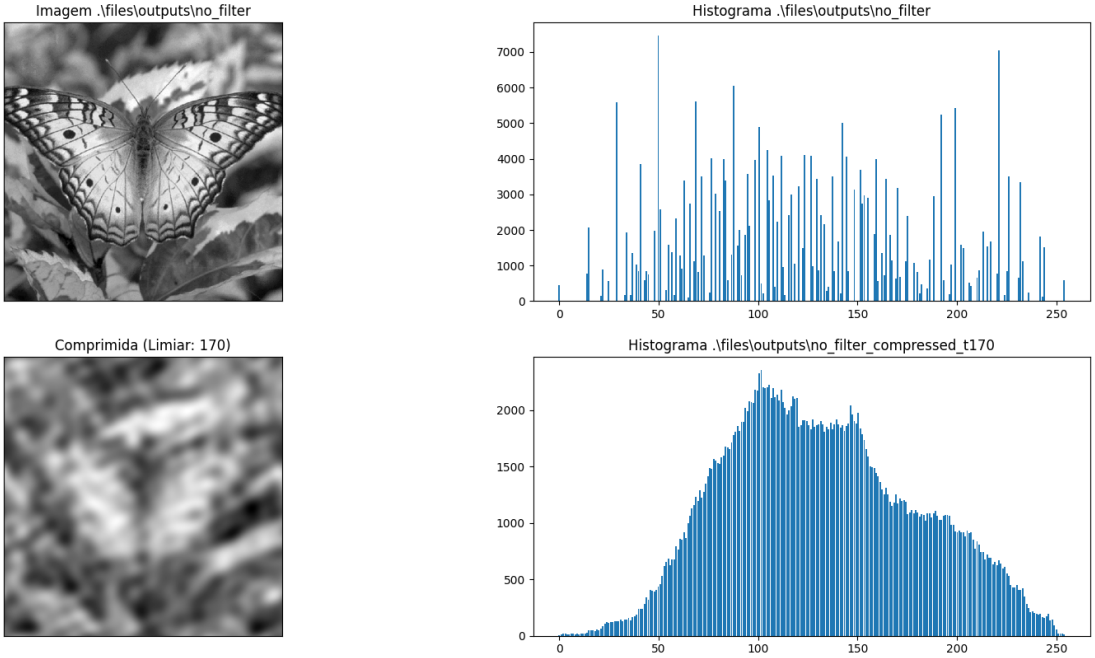
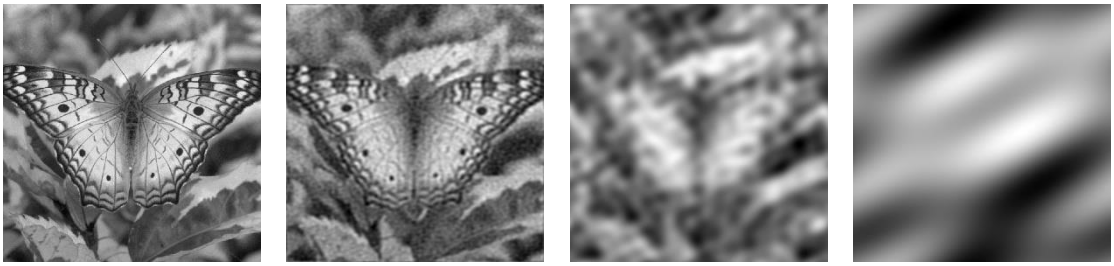


Figura 5 – Histograma da imagem comprimida em comparação com a imagem original sem compressão

Agora, deve-se analisar como diferentes limiares alteram o resultado da compressão. Os limiares utilizados foram divididos em quatro valores. Um limiar baixo (threshold de 100), um limiar no meio termo para baixo (threshold de 140), um limiar no meio termo para cima (threshold de 170) e um limiar alto (threshold de 200).



Limiar 100 Limiar 140 Limiar 170 Limiar 200

Figura 6 – Compressões com diferentes limiares

Os resultados para os diferentes podem ser vistos acima, na figura 6. Para o limiar baixo com valor de 100, a imagem comprimida é bem próxima a imagem sem compressão, não é percebida qualquer mudança na imagem. Para o limiar de 140, já é perceptível uma leve perda dos valores na saída da compressão, contudo essa perda ainda não é tão drástica. Enquanto para o limiar de 170, já existe uma perda maior em relação às anteriores. Já não se percebe claramente algumas informações que estão na imagem. Por fim, para o limiar 200, a perda de informação é quase total, a imagem torna-se um borrão sem nenhuma definição.

4.4. Desafios na implementação

Um dos grandes desafios para a implementação dos algoritmos acima foi a definição do uso da forma de resposta da transformada de Fourier devolvida pela função dft do opencv. Existem algumas formas de devolução do resultado da transformada, dois deles são a forma real, com shape (M,N) e a forma complexa com shape (M,N,2). Escolheu-se a forma complexa, pois, quando a saída real foi escolhida, os resultados não estavam de acordo com o esperado, talvez devido a precisão e algumas incertezas combinadas.

Porém com a escolha da saída complexa, alguns problemas foram encontrados, principalmente com *broadcasting.* Já que, devido à escolha, a aplicação de qualquer máscara deveria ser nas duas posições da matriz. Mas isso foi rapidamente contornado e os resultados obtidos foram plenamente satisfatórios.

5. Conclusão

Dessa forma, conseguiu-se explorar grande parte do uso da transformada de Fourier, bem como, foi possível trabalhar no domínio de frequência realizando as ações de filtragem e compressão.

Quanto a filtragem, foram utilizados o filtro passa baixa, que suavizam a imagem e reduzem alguns ruídos, o filtro passa alta que realça detalhes da imagem como bordas e contornos, o filtro passa faixa, que destaca magnitudes intermediarias de frequência e o filtro rejeita faixa, que realça as bordas da imagem e ao mesmo tempo suaviza as regiões homogêneas.

A filtragem no domínio de frequência mostrou-se eficiente, uma vez que a aplicação de filtros, já obtendo a transformada em mãos, é simplesmente a execução de uma multiplicação. Não há necessidade explicita do uso de convoluções.

Tratando-se da compressão, foram usados quatro limiares distintos para visualizar e entender como é o comportamento para cada valor. Para valores baixos (100 e 140) a compressão foi mais branda, isso é perceptível pois o conteúdo da imagem foi pouco alterado, ocorrendo somente uma suavização. Porém, para valores mais altos (170 e 200), foi visto o contrário, para esses casos a perda de informação é mais perceptível, em especial para o caso do limiar de 200, que gerou um grande borrão na imagem. Enquanto para o 170, a informação ainda pode ser percebida, mas ela é notável que ela foi alterada de forma drástica.

Com isso, pode-se concluir que o algoritmo de filtragem usando o conceito da transformada de Fourier cumpre com seu papel, alterando a imagem de entrada com diferentes filtros e comprimindo-a utilizando diferentes limiares, tudo isso dentro do domínio de frequência.