MO443 - Introdução ao Processamento de Imagem Digital - Trabalho 2

Mauro Roberto Costa da Silva - 192800
maurorcsc@gmail.com

I. Introdução

Operações em processamento de imagens digitais podem ser aplicadas no domínio espacial ou no domínio de frequência. As técnicas desenvolvidas no domínio de frequência são fundamentadas no teorema da convolução, em que a transformada de Fourier de uma convolução de duas funções integráveis é igual ao produto ponto a ponto das transformadas de cada função.

O objetivo deste trabalho é aplicar a transformada rápida de Fourier (FFT) em imagens digitais, convertendo-as para o domínio de frequência. A filtragem das imagens no domínio de frequência possibilita a alteração de seus valores originais em novas informações, de forma a atenuar ruído nas imagens, suavizar os dados, aumentar o contraste, realçar detalhes das imagens, entre outras operações.

Neste trabalho, devem ser aplicados filtros passa-baixa, passa-alta e passa-faixa e as operações de compressão e rotação em imagens no domínio de frequência.

II. Execução do Código

Além deste relatório, foi entregue o código fonte e os demais arquivos citados no texto. O programa foi desenvolvido em *Python* 3.6.9, utilizando as bibliotecas *numpy* 1.19.5, *OpenCV* 3.2.0 e *SciPy* 1.5.4, e pode ser executado através do arquivo ***trabalho2.py***.

A. Parâmetros

Para a execução do programa, é necessário a utilização dos seguintes parâmetros:

- -i: imagem de entrada;
- -o: arquivo de saída (opcional);
- -a: ângulo de rotação da imagem (opcional);
- -c: valor da compressão (opcional).

Após o nome do arquivo *python* (*trabalho2.py*), é necessário inserir o nome do arquivo da imagem de entrada através do parâmetro **-i**. É possível inserir alguns outros parâmetros opcionais, são eles: **-o**, que indica o nome do arquivo de saída, por padrão esse nome é "*./output.png*"; **-a**, que indica o ângulo de rotação da imagem, por padrão esse valor é 45; e **-c**, que é a compressão da imagem, por padrão esse valor é 10.5.

Durante a execução do programa, todas as operações são executadas. Isto é, todos os filtros e as operações de compressão e rotação são aplicados. Cada operação gera uma imagem de saída, com o nome formado pelo valor passado no parâmetro *-o* concatenado ao nome da operação aplicada.

Segue um exemplo de como executar o programa:

```
python3 trabalho2.py -i ./imagens/baboon.png -o baboon.png
                                                -a 45 -c 10.5
```

Nesse exemplo, o programa recebe como imagem de entrada o arquivo *./imagens/baboon.png*, aplica os filtros passa-baixa, passa-alta e passa-faixa, aplica rotação à $45°$ e compressão com limiar $\epsilon = 10.5$. Uma imagem PNG para cada um dos espectros e cada uma das operações é gerada e salva com o nome *baboon* concatenado ao nome da operação. A entrada de dados foi testada com imagens cedidas pelo professor. Todas as imagens testadas estão no diretório "*./imagens*".

III. Processo da Implementação

Nesta seção, serão descritas as decisões que levaram à implementação final do programa.

A imagem de entrada é lida através do método **imread** da biblioteca *OpenCV* e armazenada em uma matriz do tipo *numpy.array*. Os arquivos de saída foram gerados utilizando também a biblioteca *OpenCV*, com o método **imwrite**.

A. Transformada de Fourier

A Transformada de Fourier foi obtida através do método **fft.fft2** da biblioteca *Numpy*. Essa transformada foi transladada para o centro através do método **numpy.fft.fftshift**. Foi utilizado o logaritmo da magnitude multiplicado por um fator para facilitar a visualização do espectro. O logaritmo foi obtido pelo método **numpy.log**, a magnitude pelo método **numpy.abs** e o fator utilizado foi 20. Foi necessário multiplicar por um fator pois o logaritmo em geral devolve valores pequenos e próximos, o que dificulta a visualização da imagem devido ao baixo contraste.

B. Criação e Aplicação dos Filtros

No código desenvolvidos, foram aplicados filtros ideais. Para o filtro passa-baixa, foi criada uma máscara do tamanho da imagem preenchida com 0s e um círculo de raio 60 no centro da imagem preenchido com 1s. No passa-alta, a máscara foi preenchida com 1s e foi criado um círculo no centro de raio 75 preenchido com 0s. Já no passa-faixa, a imagem foi preenchida com 0s e foram criados dois círculos no centro: o primeiro com raio 80 preenchido com 1s e o segundo com raio 20 preenchido com 0s. Os círculos foram criados com o método **cv2.circle**.

Foram testados diferentes valores de raios para os filtros desenvolvidos e foram escolhidos os

valores que deram melhores resultados. Para o filtro passa-baixa foram testados os valores do conjunto $\{40, 60, 80\}$, para o passa-faixa foram testados os valores $\{(80, 20), (80, 30), (60, 20), (60, 30), (70, 20), (70, 30)\}$ e para o passa-alta os valores $\{90, 80, 75, 70, 60\}$.

Para cada filtro, a máscara foi aplicada à Transformada de Fourier da imagem usando o operador $*$. O resultado foi transladado de volta para posição original usando o método **numpy.fft.ifftshift** e a transformada inversa foi obtida com o método **numpy.fft.ifft2**. A imagem de saída foi obtida através do valor absoluto da transformada inversa, usando o método **numpy.abs**.

Já o núcleo do filtro foi obtido pela multiplicação da máscara pelo espectro de magnitude da imagem, utilizando o operador $*$.

C. Rotação de Imagens

A rotação foi feita no domínio espacial da imagem, usando o método **ndimage.rotate** da biblioteca *SciPy*. Para ele é passada a imagem, o ângulo de rotação e uma variável *reshape* com valor *True* para indicar que a imagem deve ser redimensionada quando necessário, evitando a perda de informações.

Foi aplicada a Transformada de Fourier na imagem rotacionada, como na Seção III-A, para obter o espectro de magnitude.

D. Compressão de Imagens

Para a compressão, primeiro foi obtida a transformada da imagem e o logaritmo da magnitude, como na Seção III-A. Foi aplicada uma operação à transformada do tipo $fft[magnitude < \epsilon] = 0$, em que fft é a transformada, $magnitude$ é o valor da magnitude e ϵ é o limiar da compressão.

A imagem de saída foi obtida através do valor absoluto da transformada inversa, usando os métodos **numpy.fft.ifft2** e **numpy.abs**.

IV. Resultados e discussão

Para a apresentação dos resultados, será utilizada a imagem *baboon.png* (Figura 1), disponibilizada pelo professor. Em seguida, serão apresentados os resultados das operações e o que foi observado.

A Figura 2 apresenta o espectro de magnitude da imagem utilizada transladado para o centro. Os *pixels* mais distantes do centro representam as frequências mais altas e os mais próximos do centro as frequências mais baixas. E a Figura 3 apresenta o resultado da transformada inversa aplicada a esse espectro de magnitude.

Na Figura 4, podemos observar o núcleo do filtro passa-baixa no espectro de magnitude. Nela, observamos que apenas as frequências em um determinado raio do centro (as baixas frequências) permanecem (passam), enquanto as demais são zeradas.

A Figura 5 apresenta o núcleo do filtro passa-faixa no espectro de magnitude. É possível observar na imagem que apenas as frequências em uma determinada faixa de frequências permanecem (passam), enquanto as demais são zeradas.

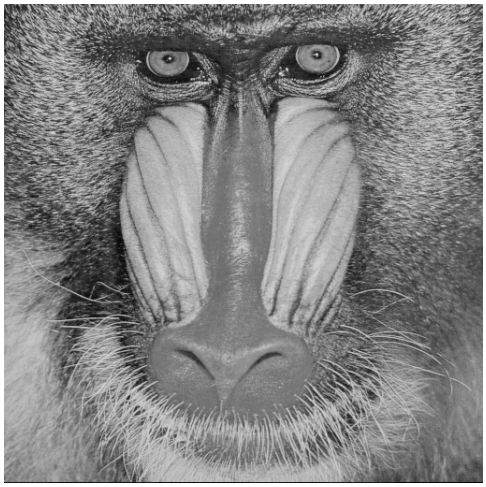

Figure 1: Imagem original monocromática

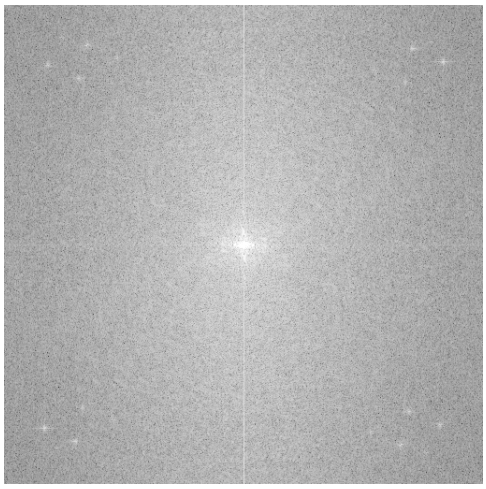

Figure 2: Espectro de magnitude

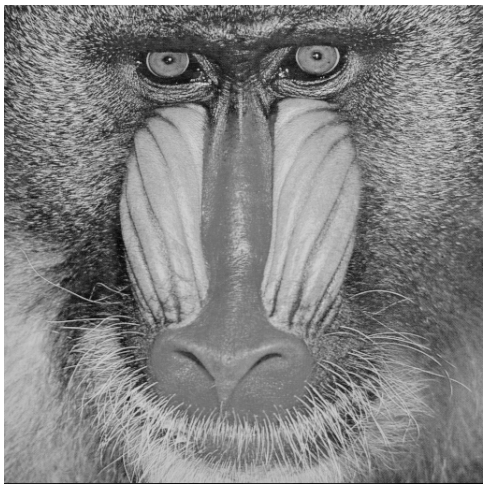

Figure 3: Transformada inversa de Fourier

Já a Figura 6 apresenta o núcleo do filtro passa-alta no espectro de magnitude. Nela, podemos observar que apenas

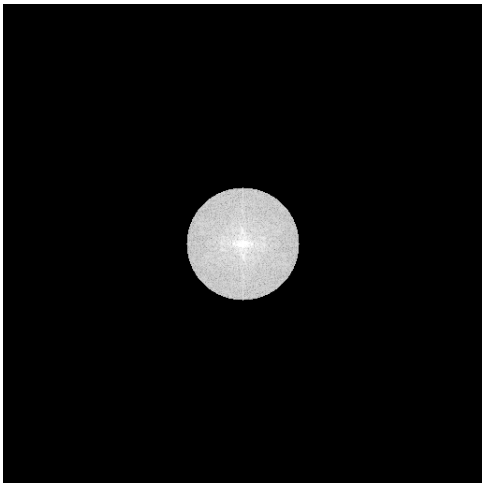

Figure 4: Núcleo do filtro passa-baixa

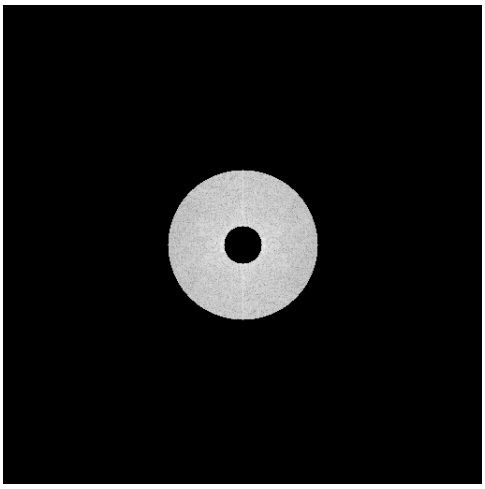

Figure 5: Núcleo do filtro passa-faixa

as frequências acima de um determinado raio a partir do centro permanecem (passam), enquanto as dentro desse raio são zeradas.

O resultado da aplicação do filtro passa-baixa na imagem original pode ser visto na Figura 7. Por atenuar as frequências altas, a imagem perdeu nitidez de detalhes e os ruídos foram atenuados.

Na Figura 8, podemos observar a aplicação do filtro passa-faixa na imagem original. O resultado é uma imagem com perda de nitidez e sem realce das bordas com frequências altas.

A Figura 9 apresenta o resultado da aplicação do filtro passa-alta na imagem original. Esse filtro realçou as bordas e os detalhes da imagem.

O resultado da compressão aplicada na imagem original pode ser observado na Figura 10. Essa operação foi aplicada com valor 10.5 para o limiar de compressão e resultou em uma perda de qualidade na imagem. Quanto maior o limiar utilizado, maior a perda de qualidade na imagem resultante.

A Figura 11 apresenta uma rotação à $45°$ aplicada na imagem original. Essa imagem foi redimensionada para que a rotação não resultasse em cortes na imagem original.

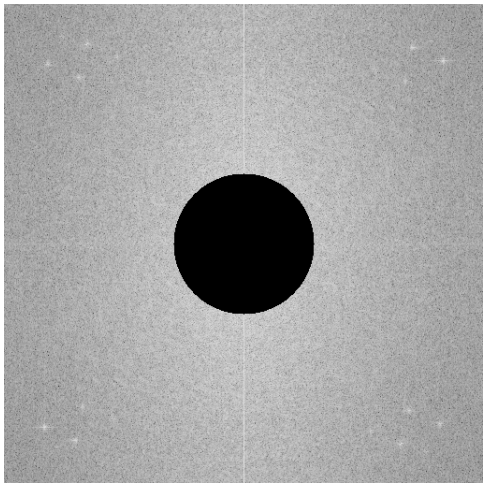

Figure 6: Núcleo do filtro passa-alta

Figure 7: Imagem após filtro passa-baixa

Figure 8: Imagem após filtro passa-faixa

Por fim, a Figura 12 apresenta o espectro de magnitude da

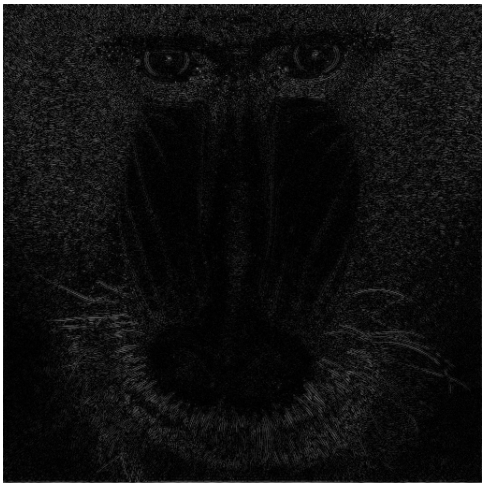

Figure 9: Imagem após filtro passa-alta

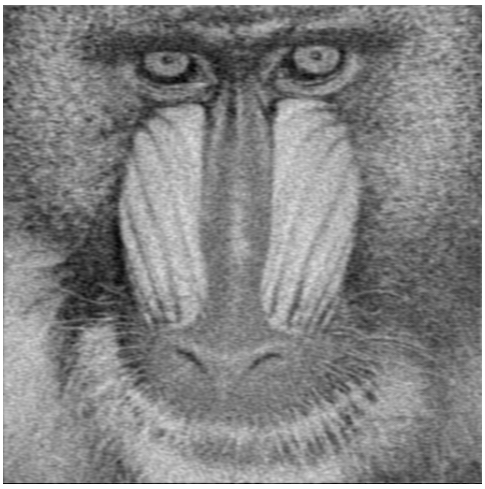

Figure 10: Imagem após compressão

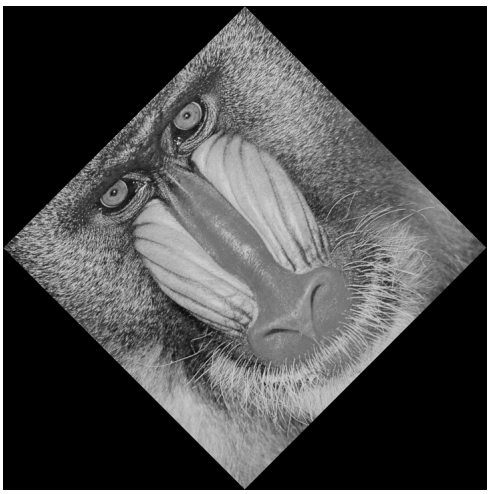

Figure 11: Imagem após rotação de $45°$

imagem rotacionada. Esse espectro é semelhante ao da imagem original, porém rotacionado à $45°$.

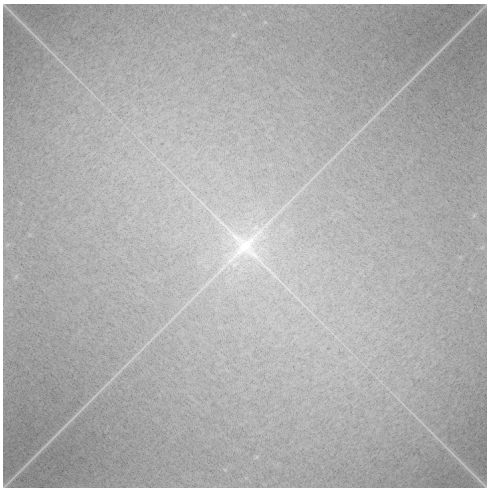

Figure 12: Espectro de magnitude da imagem rotacionada

V. Conclusão

A aplicação de filtros no domínio de frequência permite um ganho de desempenho em relação ao processo de convolução aplicado no Trabalho 1. Isso porque, uma vez que a Transformada de Fourier da imagem é obtida, a aplicação do filtro pode ser feita com apenas uma multiplicação de matrizes.

Pudemos perceber a importância de operações aplicadas à imagens no domínio de frequência. Foram testados diferentes valores para os raios dos filtros aplicados e para o limiar de compressão, e foram escolhidos os que obtiveram melhores resultados.

Os resultados foram satisfatórios, já que as operações implementadas obtiveram saídas semelhantes aos exemplos apresentados na descrição do trabalho.