



Faculdade de Tecnologia

Processamento Digital de Imagens

Lista 2

THIAGO TOMÁS DE PAULA

2023

Questão 1.1	2	Questão 1.4	5
Questão 1.2	3	Questão 1.5	6
Questão 1.3	4	Questão 1.6	7

LISTA

2

PARTE TEÓRICA

Este é o relatório as minhas respostas às questões na lista 2 de PDI. O texto aqui procura ser completo, mas é plenamente compreendido penas com o acompanhamento simultâneo dos códigos e referências relevantes. Estes serão indicados conforme a necessidade, e estão presentes no repositório *git* de onde este relatório se originou:

https://github.com/Thiago-TP>Listas_de_PDI/tree/main

Questão 1.1

Uma forma de obter a derivada discreta 2-D (aproximação) é calcular as diferenças $f(x+1, y) - f(x, y)$ e $f(x, y+1) - f(x, y)$.

- (a) Encontre o filtro equivalente no domínio da frequência ($H(u, v)$).
 - (b) Mostre que esse filtro é um passa-alta.
-

Programa relevante:

- `lista2_q1_1.m`

Resultados:

- Não salvos, mas o código gera imagens de validação. Isso é verdade para todas as questões da parte teórica.

- (a) Seja $F(u, v)$ a Transformada de Fourier de $f(x, y)$. Deseja-se encontrar $H(u, v)$ para a qual $G(u, v) = F(u, v)H(u, v)$, com $G(u, v)$ a transformada de $g(x, y) = [f(x+1, y) - f(x, y)] + [f(x, y+1) - f(x, y)]$. Digamos que a imagem filtrada tenha largura M e altura N . Pela fórmula da transformada

inversa, vem

$$f(x, y) = \frac{1}{MN} \sum_{u=1}^M \sum_{v=1}^N F(u, v) e^{j2\pi(\frac{ux}{M} + \frac{vy}{N})} \implies \quad (1.1)$$

$$g(x, y) = \frac{1}{MN} \sum_{u=1}^M \sum_{v=1}^N F(u, v) \{ [e^{j2\pi(\frac{u(x+1)}{M} + \frac{vy}{N})} - e^{j2\pi(\frac{ux}{M} + \frac{vy}{N})}] + \\ [e^{j2\pi(\frac{u(x)}{M} + \frac{v(y+1)}{N})} - e^{j2\pi(\frac{ux}{M} + \frac{vy}{N})}] \} \quad (1.2)$$

$$= \frac{1}{MN} \sum_{u=1}^M \sum_{v=1}^N F(u, v) e^{j2\pi(\frac{ux}{M} + \frac{vy}{N})} \{ e^{j2\pi \frac{u}{M}} + e^{j2\pi \frac{v}{N}} - 2 \} \quad (1.4)$$

Por definição,

$$g(x, y) = \frac{1}{MN} \sum_{u=1}^M \sum_{v=1}^N F(u, v) H(u, v) e^{j2\pi(\frac{ux}{M} + \frac{vy}{N})}.$$

Comparando esta relação com (1.4), é imediato que $H(u, v) = e^{j2\pi \frac{u}{M}} + e^{j2\pi \frac{v}{N}} - 2$.

- (b) A frequência de um ponto é grandeza proporcional à diferença de brilho/valor entre esse pixel e seus vizinhos. O filtro de derivada é passa-alta uma vez que uma sub-área de baixa frequência tem $|g|$ pequeno, ou nulo, enquanto sub-área de alta frequência tem $|g|$ grande. Visualmente, isso quer dizer que o filtro realça as bordas e contornos da imagem. A imagem abaixo valida essa afirmação. Nela, a imagem da esquerda é uma imagem de teste, e a da direita o resultado da filtragem na frequência dessa imagem pelo filtro $H(u, v)$.

Figura 1.1: Validação da questão 1.1.



Questão 1.2

Considere as imagens da Figura 1. A imagem da direita foi obtida por: (1) filtrando a imagem da esquerda com um filtro gaussiano passa-baixa e (2) filtrando o resultado com um filtro gaussiano passa-alta. As imagens tem 420×344 pixels e ambos os filtros tem uma frequência de corte $D0 = 25$.

- (a) Explique por que razão a parte central da imagem a direita (o anel) tem um aspecto sólido e brilhante, tendo em conta que as principais características da imagem filtrada são as linhas na borda exterior dos objetos (por exemplo, dedos, ossos, etc.) com uma área mais escura entre essas linhas. Por outras palavras, não seria de se esperar que o filtro passa-alta produzisse uma área constante no interior do anel, dado que o passa-alta elimina o termo DC?

(b) Se invertêssemos a ordem de aplicação dos filtros, o resultado seria o mesmo?

Programa relevante:

- `lista2_q1_2.m`

Resultados:

- Imagem de validação.

- (a) Para o passa-baixas, frequências além da de corte são atenuadas, e frequências abaixo, preservadas. Para o passa-altas, ocorre o contrário, frequências acima se preservam e abaixo se atenuam. Para um elemento de cor sólida mais clara rodeada de cor sólida mais escura, como é o caso do anel e dos dedos, o passa-baixas forma uma borda cinza borrada por volta do elemento. Da zona escura até o ponto mais claro, essa borda cria um gradiente de cinza que será preservado pelo passa-altas desde que esteja acima da frequência de corte. O que acontece com o anel que não acontece com os ossos é que, sendo o anel mais fino/esbelto, grande parte desse elemento após o passa-baixas está no gradiente comentado, que para $D_0 = 25$ foi preservado pelo passa-altas. Nos ossos, o “miolo” atingido após se passar da borda cinza do passa-baixas é uma cor sólida, e portanto será atenuada pelo passa-alta, escurecendo essa parte. Com isso dito, é possível notar um ligeiro acinzentamento no interior do anel, indicando atenuação do passa-altas.
- (b) Não. Ao se aplicar o passa-altas primeiro, não existiria a borda cinza comentada, e portanto poucos pontos no interior dos ossos/anel estariam dentro de um gradiente preservado pelo passa-altas. Em outras palavras, o anel e dedos ficariam escuros imediatamente. Em seguida, filtrar esse resultado pelo passa-baixas apenas boraría a imagem, dificultando a visualização. O código desta questão valida as respostas dadas neste item e no anterior através de filtragens por gaussianos passa-baixas e passa altas com raios variáveis feitas sobre uma imagem similar ao da Figura 1 (raio X de mão com anel).

Questão 1.3

As barras brancas na imagem apresentada na Figura 2 tem 7 pixels de largura e 210 pixels de altura.

A separação entre as barras tem 17 pixels de largura. Como ficaria a imagem se fossem aplicados:

- (a) Filtros de média aritmética de tamanho 3×3 e 7×7 .
 - (b) Filtros de média geométrica de tamanho 3×3 e 7×7 .
 - (c) Filtros contra-harmônicos com $Q = 1$ e $Q = -1$, e com tamanho 3×3 e 7×7 .
 - (d) Filtros de mediana de tamanho 3×3 e 7×7 .
-

Programa relevante:

- `lista2_q1_3.m`

Resultados:

- Imagens de validação.

- (a) **3x3** Parte preta largamente inalterada, contornos cinzas borrados nas barras brancas.
7x7 Mesmo efeito do kernel anterior, mas como a largura do kernel se torna igual à da barra, o borrado cinza é bem maior, e apenas a linha no centro da barra continua branca (isto é, com valor 255).
- (b) **3x3** Parte preta original permanece a mesma, mas barras brancas diminuem. Isso ocorre porque o filtro zera o resultado caso haja ao menos um pixel preto no kernel. Sendo o filtro 3×3 , apenas os pontos brancos no perímetro foram zerados. Alternativamente, pode-se dizer que a imagem sofreu erosão com estruturante 3×3 de 1.

- 7x7** Resultado análogo ao anterior, mas como o kernel aqui é maior, remove-se um contorno maior de cada barra (espessura de 3 pixels, para ser mais exato). Das barras originais, restam uma linha vertical com 1 pixel de largura e 204 de altura.
- (c) **3x3, Q = 1** De maneira geral, se houver $n > 0$ pixels brancos no kernel contra-harmônico, o resultado sera $n \times 255^2 / n \times 255 = 255$. Caso não exista nenhum pixel branco, o resultado é 0. Essa filtragem corresponde portanto exatamente a uma dilatação com elemento estruturante 3x3 de 1. Dessa maneira, o resultado da filtragem sera a ampliação de cada barra em 1 pixel em cada direção (irão de 210x7 para 212x9).

7x7, Q = 1 A lógica aqui é a mesma do caso 3x3. A diferença no resultado é que a expansão/dilatação será de 3 pixels em cada direção, uma vez que o kernel/elemento estruturante é maior (barras irão de 210x7 para 216x13).

3x3, Q = -1 Para $Q = -1$, o filtro contra-harmônico é equivalente ao harmônico, cujo resultado por sub-área é a média harmônica dos pixels naquela região. Essa média é equivalente a dividir o produto dos valores de brilho pela soma desses mesmos valores. Logo, se existe ao menos 1 pixel preto o resultado sera 0 (no caso especial em que todos os pixels são nulos o resultado também é tomado como 0). Assim, o resultado sera 255 se e só se todos os pixels da sub-área são brancos. Em suma, esta filtragem corresponde a uma erosão com elemento estruturante 3x3 de 1, e o resultado será a reformatação das barras brancas de 210x7 para 208x5.

7x7, Q = -1 Mesma lógica do item anterior, mas com uma erosão ligeiramente maior, dado o maior tamanho do kernel: as barras irão de 210x7 para 204x1.

- (d) **3x3** Apenas as quinas de cada barra mudaram, isto é, foram zeradas. Em qualquer ponto da parte escura da imagem original o resultado local da mediana daria resultado nulo, uma vez que pelo menos 6 dos pixels no kernel seriam 0. Pela mesma lógica, isso acontece no interior e limites das barras exceto nas quinas, onde apenas 4 dos 9 pixels do kernel seriam brancos, resultando em preto.

7x7 Efeito similar ao caso 3x3, mas além das quinas outros vizinhos brancos também são removidos. Sendo o kernel 7x7, os vizinhos removidos são aqueles duas posições a esquerda/direita, ou abaixo/acima, a depender da quina analisada.

O código desta questão valida as respostas dadas.

Questão 1.4

Mostre que a subtração do Laplaciano de uma imagem é proporcional ao 'mascaramento unsharp'. Utilize a definição de Laplaciano dada pela equação (3.6-3).

Programa relevante:

- `lista2_q1_4.m`

Resultados:

- Imagem de validação.

Primeiro redefine-se o problema de maneira matemática. Seja $f(x, y)$ uma imagem, isto é, sinal digital bidimensional com imagem nos inteiros não negativos menores que 256. Seja $g(x, y)$ o resultado de $f(x, y)$ após ser filtrada por média aritmética de kernel 3x3, $A_{3 \times 3}$; deseja-se mostrar que existem $c = c(k)$ para o qual $f_{\text{unsharp}}(x, y) = f(x, y) + k(f(x, y) - g(x, y))$ e $f_{\text{laplace}}(x, y) = f(x, y) + c\hat{g}(x, y)$ são iguais, onde $\hat{g}(x, y)$ é o resultado da convolução entre f e o kernel abaixo.

$$\nabla^2 = \begin{pmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{pmatrix} \quad (1.5)$$

A demonstração é imediata pela definição de g . Fixado k , vem

$$\begin{aligned}
 f_{\text{sharp}}(x, y) &= f(x, y) + k(f(x, y) - f(x, y) * A_{3 \times 3}) \\
 &= f(x, y) + f(x, y) * (\mathbb{I}_{3 \times 3}k - A_{3 \times 3}) \\
 &= f(x, y) + f(x, y) * \begin{pmatrix} -k/9 & -k/9 & -k/9 \\ -k/9 & k - k/9 & -k/9 \\ -k/9 & -k/9 & -k/9 \end{pmatrix} \\
 &= f(x, y) - (k/9)f(x, y) * \nabla^2,
 \end{aligned}$$

e portanto basta tomar $c = -k/9$. $\mathbb{I}_{3 \times 3}$ é uma matriz 3×3 esparsa cujo único valor não nulo é o centro, que vale 1. De maneira geral, se o *unsharp masking* é feito por média de $n \times n$ (n ímpar), então $c = -k/n^2$. Neste caso, entende-se que ∇^2 é uma matriz $n \times n$ de 1 exceto no centro, onde vale $1 - n^2$. O código da questão valida essa relação.

Questão 1.5

-
- (a) Quantos níveis de cinza existem no sistema de cores RGB, tendo em conta que uma imagem tem 8 bits?
 (b) Em uma imagem RGB, os componentes R, G e B têm um perfil horizontal de intensidades, como apresentado no diagrama da Figura 3. Qual é a cor da coluna do meio desta imagem? Qual é a cor dos limites da imagem?

Programa relevante:

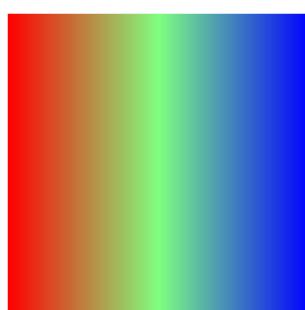
- `lista2_q1_5.m`

Resultados:

- `q1.5_perfil.png`

- (a) Uma imagem de 8 bits tem $2^8 = 256$ níveis de cinza, indo do preto (0) ao branco (255).
 (b) **Cor no meio.** Assume-se que a imagem tenha largura N , de forma que os perfis se refiram a N colunas, indexadas de 0 a $N - 1$, da esquerda para a direita. Ademais, entende-se que a cor de pixels numa mesma coluna são idênticas. Sob essas hipóteses, a coluna do meio, de índice $N/2$ ou $(N - 1)/2$, terá RGB igual a $[0.5, 1, 0.5] \times 255 = [127, 255, 127]$, que corresponde a um verde claro pálido.
Cor nos limites. Ainda sob as mesmas hipóteses da resposta anterior, entende-se como “limites da imagem” as colunas de índice 0 e $N - 1$. Nesse caso, tem-se que a cor no índice 0 é $[1, 0, 0] \times 255 = [255, 0, 0]$, vermelho puro. Por outro lado, a cor no índice $N - 1$ é $[0, 0, 1] \times 255 = [0, 0, 255]$, azul puro. O código da questão gera figura $N \times N$ com perfil de cor igual ao da Figura 3, validando as respostas dadas. Verifique-a a seguir.

Figura 1.2: Validação da questão 1.5.



Questão 1.6

- (a) Explique em detalhes o algoritmo JPEG. Inclua um diagrama de blocos na sua explicação.
(b) Explique em detalhes o algoritmo MPEG. Inclua um diagrama de blocos na sua explicação.

Programa relevante:

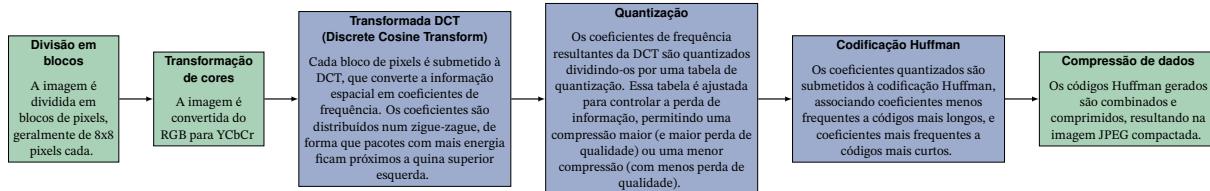
- lista2_q1_6.m

Resultados:

- Nenhum.

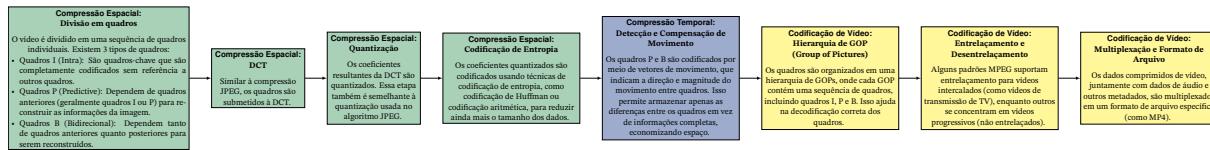
- (a) A figura abaixo resume a parte de codificação do algoritmo JPEG. Para a decodificação, basta realizar a operação inversa de cada etapa, começando pela última até chegar na primeira. Caso a letra seja pequena demais para a leitura, note que este fluxograma está presente no repositório sob o nome fluxograma_JPEG.pdf

Figura 1.3: Fluxograma do algoritmo JPEG (codificação).



- (b) A figura abaixo resume a parte de codificação do algoritmo MPEG. Enquanto o JPEG codifica imagens, o MPEG codifica vídeos, e neste contexto surge uma interessante otimização que pode ser feita: a temporal. Esta é uma das principais componentes do algoritmo. De resto, a codificação é em geral similar ao JPEG. Caso a letra seja pequena demais para a leitura, note que este fluxograma está presente no repositório sob o nome fluxograma_MPEG.pdf

Figura 1.4: Fluxograma do algoritmo MPEG (codificação).



Questão 2.1.....
Questão 2.2.....

8 Questão 2.3..... 9
8 Questão 2.4..... 12

L I S T A

2

PARTE PRÁTICA

Questão 2.1

Desenvolva um algoritmo em Matlab/Octave que realize o watermarking de imagens a partir de uma imagem que será marcada (f) e uma marca d'água (ω). Para tanto, utilize a expressão:

$$f_\omega = (1 - \alpha)f + \alpha\omega$$

onde f_ω é a imagem watermarked e α é uma constante que controla a visibilidade relativa da marca d'água na imagem. Realize os testes com as imagens que preferir.

Programa relevante:

- lista2_q2_1.m

Resultados:

- q2.1_fw1.png
- q2.1_fw2.png

A programa da questão é uma aplicação direta da fórmula no enunciado. Foram considerados dois pares de imagens de teste, q2.1_f1.jpg e q2.1_w1.jpg, q2.1_f2.jpg e q2.1_w2.jpg. Os arquivos “f” são as imagens originais e os “w” as marcas d’água. Para o primeiro par, $\alpha = 0.2$ gerava uma combinação mais agradável, exibida na imagem q2.1_fw1.png. Para o segundo, um valor menor de α , 0.1, era suficiente para o resultado desejado, capturado no arquivo q2.1_fw2.png.

Figura 2.1: Tease dos resultados da questão 2.1.

(a) Primeiro par



(b) Segundo par



Questão 2.2

Baixe as imagens fig_lista3_1.bmp e fig_lista3_2.bmp, use-as como referências para os itens que se seguem.

- Implemente um algoritmo de equalização utilizando uma abordagem global, ou seja, uma abordagem baseada no histograma da imagem completa. Calcular os histogramas das imagens originais. Aplicar o método de equalização global para as duas imagens. Apresentar as imagens originais e as suas versões equalizadas, juntamente com os histogramas correspondentes.
- Implemente um algoritmo de equalização usando uma abordagem local, ou seja, em vez de usar o histograma da imagem completa, aplique o método de equalização em pequenas sub-áreas (sub-imagens) da imagem. Aplique o método de equalização local para as duas imagens, considerando áreas de tamanhos 5x5 e 7x7. Visualize as imagens originais e suas versões equalizadas, juntamente com os histogramas correspondentes. Compare os resultados com os obtidos em (a).

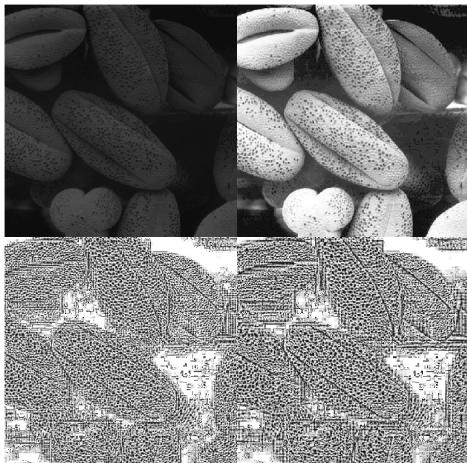
Programa relevante:

- lista2_q2_2.m

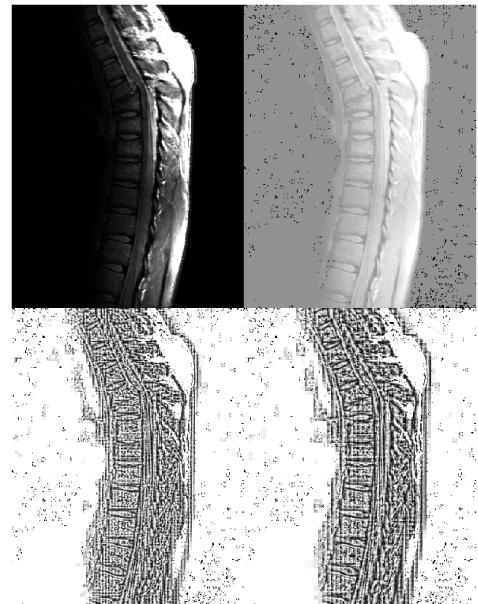
(a), (b) O código da questão contém os algoritmos implementados, e apresenta os histogramas de cada imagem de entrada/saída. Estas são apresentadas nos dois conjuntos de imagens a seguir. Para cada conjunto, a imagem original está no topo, à esquerda; o resultado da equalização global está no topo, à direita; as equalizações locais estão abaixo, sendo a da esquerda com área 5x5 e a da direita, 7x7.

Figura 2.2: Tease dos resultados da questão 2.2.

(a) fig_lista3_1.bmp e equalizações.



(b) fig_lista3_2.bmp e equalizações.



Em comparação com a equalização global, as locais são de difícil leitura, e destroem os detalhes originais. Esses efeitos indesejáveis advém da reescala no algoritmo de equalização: impõe na equalização que tanto o menor quanto o maior valor de brilho sejam atingidos na imagem/sub-área resultante (o algoritmo usado é descrito em https://en.wikipedia.org/wiki/Histogram_equalization). Isso dá à equalização local um aspecto de contraste exagerado, violento.

Questão 2.3

Baixe o arquivo "imagensruidosas". Este arquivo comprimido contém um conjunto de imagens com ruído. Implemente 2 filtros (a sua escolha) para remover o ruído. Utilize esses filtros para remover o ruído em cada uma das imagens. Discuta os resultados obtidos. Quando possível, descreva o tipo de ruído presente na imagem.

Programa relevante:

- lista2_q2_3.m

Resultados:

- q2.3_JFK_filtragens.png

- q2.3_lena_filtragens.png
- q2.3_lena_salt_pepper_filtragens.png
- q2.3_lena_sigma25_filtragens.png

A análise da transformada de Fourier do canal Y da imagem JFKreg.jpg revela presença de ruído periódico na frequência, talvez similar ao padrão de Moiré. A foto em lena.png não possui ruído, enquanto lena-sp.png e lena_sigma25.png tem ruído *salt and pepper* e gaussiano, respectivamente. O que indica o ruído gaussiano é a comparação do histograma dessa imagem com o da Lena sem ruído (esses histogramas são mostrados na execução do código).

Os filtros escolhidos para remoção de ruído foram o filtro de média aritmética e o filtro da mediana. Os resultados são mostrados na Figura 2.4. Para cada tripla de imagens, a da esquerda é a original, a do meio a original filtrada por média, e a da direita a original filtrada por mediana.

Análise – JFKreg. Embora nenhum filtro tenha sido capaz de atenuar o ruído de maneira eficaz, o filtro da media restringe os padrões ruidosos às baixas frequências, como mostra Figura 2.3. O filtro da mediana consegue eliminar uma pequena porção dos padrões periódicos (segmentos verticais), mas a maior parte desse padrão fica intocado (segmentos diagonais). Além disso, os artefatos gerados na filtragem de mediana são, subjetivamente, mais desagradáveis ao olho humano que os artefatos da filtragem por media.

Análise – lena. Não existe ruído na imagem original então não há análise a ser feita aqui.

Análise – lena-sp. O ruído salt & pepper é eficazmente removido pelo filtro da mediana, embora com artefatos inevitáveis, intrínsecos dessa filtragem. Isso era o esperado, uma vez que os pontos brancos e pretos do ruído raramente são a mediana dos pontos considerados, e mesmo quando estão na posição de mediana, isso significa simplesmente que a maior parte da área considerada é ou preta ou branca, e a interferência do ruído é mínima.

Análise – lena_sigma25. Ambos os filtros atenuam o ruído de forma a obter resultados similares, mas os artefatos deixados pelo da mediana são, subjetivamente, mais desagradáveis ao olho humano.

Figura 2.3: Transformada de Fourier da Figura 2.4a.

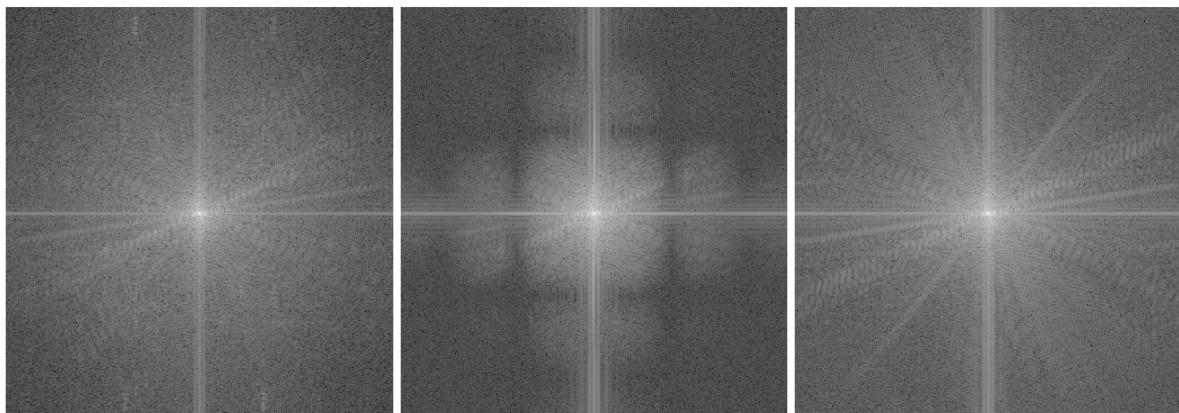
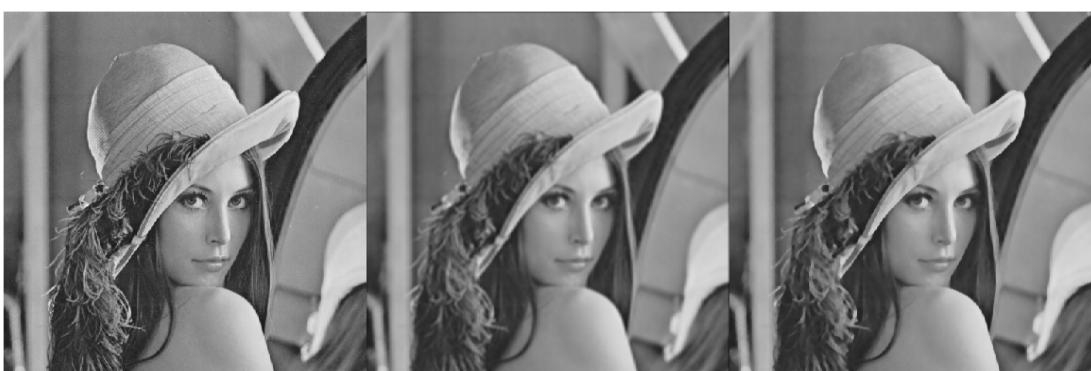


Figura 2.4: Resultados da questão 2.3.

(a) JFKreg.jpg



(b) lena.png



(c) lena-sp.png



(d) lena_sigma25.png



Questão 2.4

- (a) Implemente um algoritmo de processamento de cor que gere cores falsas (pseudocores). Nesse sistema, deve-se especificar dois intervalos de níveis de cinza na imagem de entrada. O algoritmo deve gerar uma imagem colorida (RGB) com pixels com uma cor específica para cada intervalo de níveis de cinza da imagem de entrada. Os pixels restantes da imagem de entrada (fora dos dois intervalos) devem manter os níveis de cinza originais.
- (b) Baixe a imagem da Figura 4 (Figura 1.10 (4) do livro texto). Usando o algoritmo desenvolvido em (a), processe a imagem de forma a que o rio no centro da imagem apareça amarelo, enquanto o resto da imagem mantém os níveis originais da escala de cinzas. É aceitável ter um pequeno número de pixels amarelos isolados na imagem de saída.

Programa relevante:

- `lista2_q2_4.m`

Resultados:

- `q2.4_pseudocor_rio.png`

(a), (b) O algoritmo pedido no item (a) está implementado no código da questão. Procurando atender ao item (b), encontrou-se por tentativa e erro que $[0, 40]$ é um bom intervalo de cinza para se colorir o rio sem colorir muito do resto da imagem. O rio colorido é mostrado abaixo, à direita do rio original.

Figura 2.5: Resultados da questão 2.4.

