Dados do relatório Lista 3 filtros espaciais

Matéria: Processamento Digital de Imagens (PDI) **código:** CMP1084 A01

Aluno: Vitor de Almeida Silva

Matrícula: 20161003305497

ENUNCIADO

1. Conceitue correlação e convolução.
2. Conceitue a filtragem espacial fuzzy baseada em regras.
3. Gere uma máscara para um filtro espacial usando o seguinte código:

   Tam = 3;

   Mascara = 1/(Tam*Tam) * ones(Tam,Tam);

   Realize a filtragem espacial com a máscara acima e exiba a imagem de saída.
4. Altere o tamanho (Tam) para 10 e 100, use a máscara recém-gerada para filtrar a imagem original.
5. Realize a filtragem espacial com as máscaras (Laplace e Sobel) abaixo:

1	1	1
1	-8	1
1	1	1

-1	-2	-1
0	0	0
1	2	1

6. Gere outra máscara de filtro usando o seguinte código:

   Mascara2 = [0 0 0 0 0; 1 1 1 1 1; 0 0 0 0 0; -1 -1 -1 -1 -1; 0 0 0 0 0];

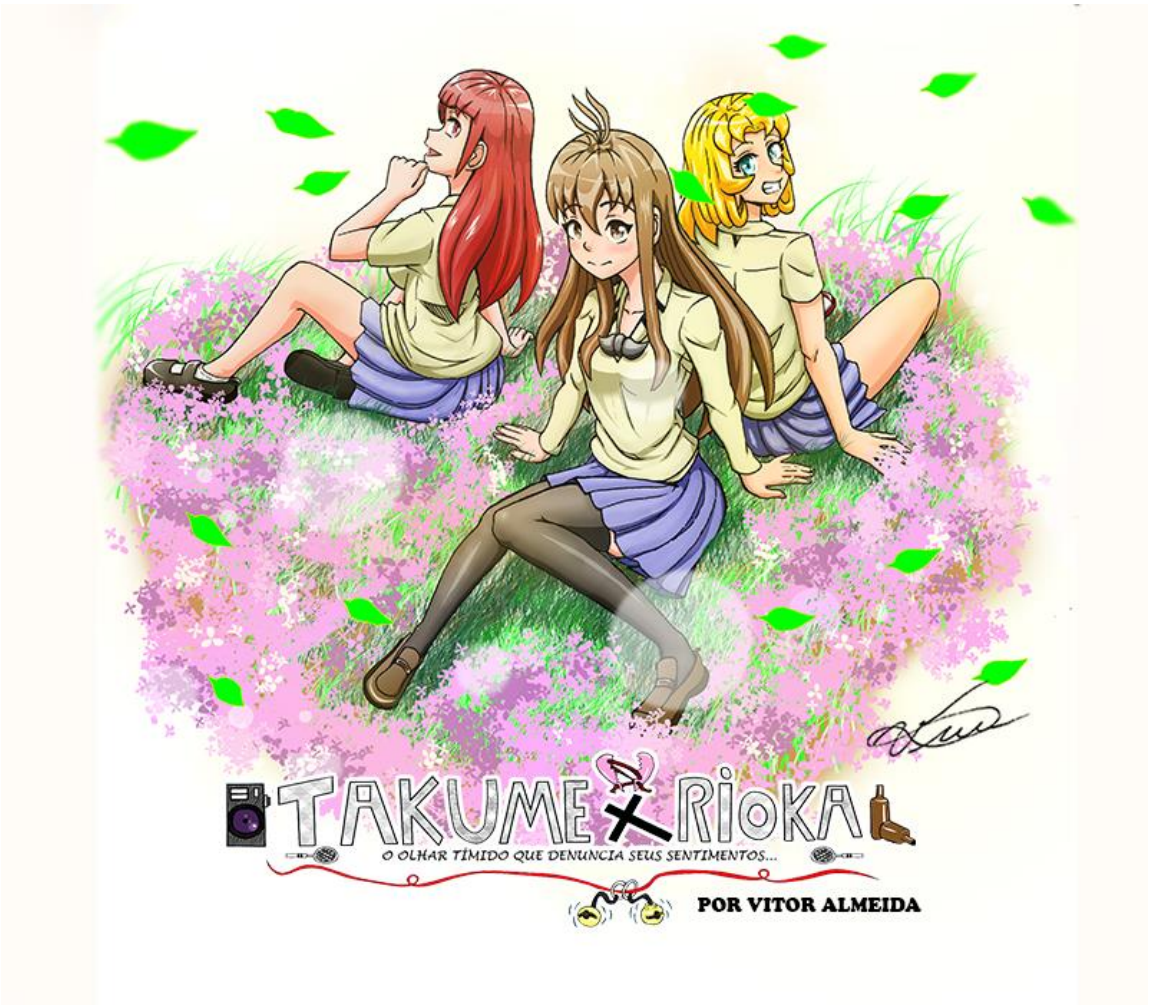
   angulo = 0;

   filtro = imrotate(mascara2, angulo, ‘crop’);

   Realize a filtragem espacial com a máscara acima e exiba a imagem de saída.
7. Altere o Ângulo para 90 e 196, exiba o resultado e explique a diferença entre diferentes saídas.
8. Utilize a máscara da questão 4, Tam igual á 10, na função *imfilter,* com os padding *symmetric, replicate, circular.* Explique a diferença de cada um deles.

A imagem utilizada ao longo da Lista é mostrada na Figura 1.

Figura 1: img1



Fonte: Mangá Takume x Rioka (Autoral)

1. Conceitue correlação e convolução.

Resposta:

Correlação e convolução, se referem as operações realizadas ao se passar uma máscara em uma matriz realizando soma de produtos e substituindo nas posições devidas. A diferença básica entre correlação e convolução, é que a correlação é rotacionada em 180 graus. Deste modo, ao se passar um filtro em uma matriz através do processo da convolução, nota-se na saída, que o resultado ficou rotacionado em 180 graus (ou espelhado) (GONZALEZ, WOODS, 2010).

Outra condição é em relação ao tamanho da imagem, de modo que, se o filtro tem tamanho **m**, se deve preencher os lados da imagem (M+m0 e (N+m) com 0s. O motivo, é a necessidade de realizar a soma dos produtos para cada elemento da imagem, posteriormente, pode-se realizar um processo de recortar as regiões com 0s para voltar a imagem ao tamanho original (GONZALEZ, WOODS, 2010). Deste modo, às equações de convolução e correlação são mostradas nas Figuras 1 e 2.

Figura 1:Equação de convolução

$$w(x,y) \bigstar f(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) f(x-s, y-t)$$

Fonte: (GONZALEZ, WOODS, 2010)

Figura 2:Equação de Correlação

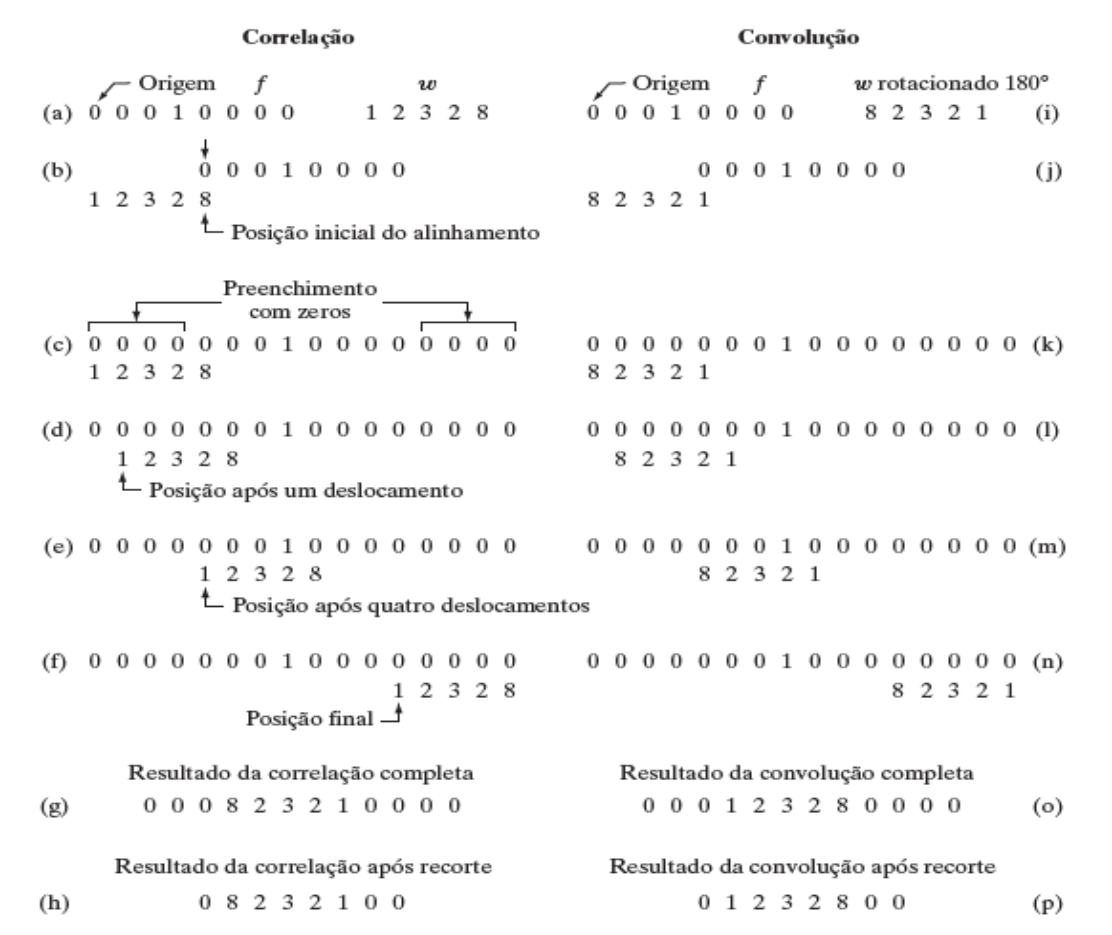
$$w(x,y) \star f(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) f(x+s, y+t)$$

Fonte: (GONZALEZ, WOODS, 2010)

Sendo a=(m-1)/2 e b=(n-1)/2. Para praticidade m e n, são números inteiros impares. A operação de convolução ou correlação, são utilizadas para aplicação de filtros espaciais em imagens, o importante é que a máscara do filtro utilizada em uma dada tarefa de filtragem seja especificada de forma a corresponder com a operação pretendida (GONZALEZ, WOODS, 2010).

A Figura 3 mostra um exemplo de aplicação de correlação em comparação com a convolução utilizando vetores.

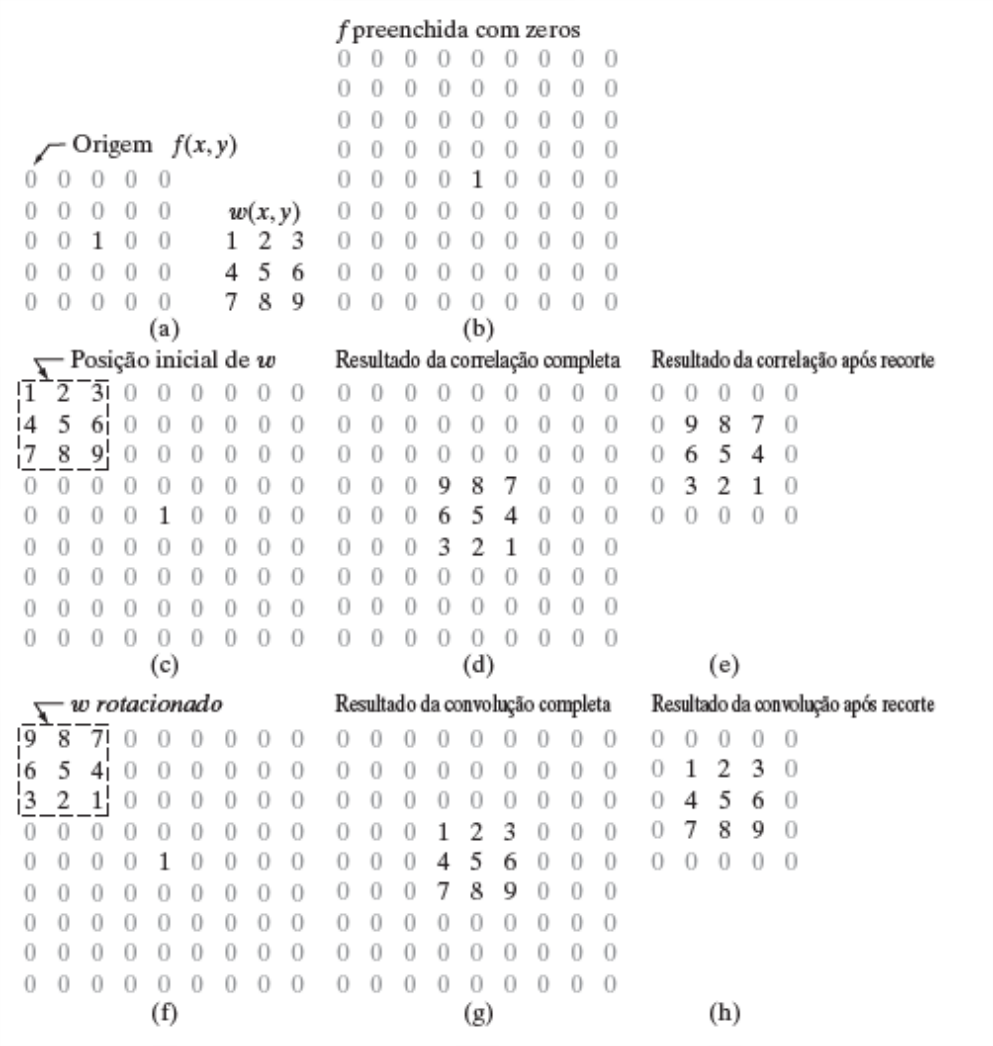
Figura 3: Correlação vs convolução



Fonte: (GONZALEZ, WOODS, 2010)

A Figura 4, mostra a aplicação da correlação (linha do meio) e da convolução (linha de baixo) em uma matriz utilizando uma máscara w(x,y).

Figura 4: aplicação de convolução e correlação em matriz



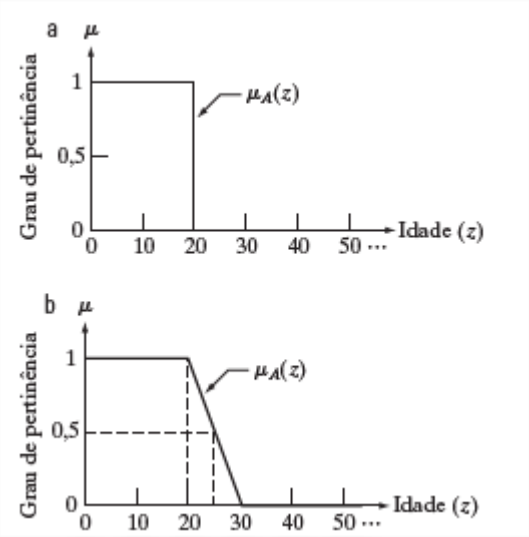
Fonte: (GONZALEZ, WOODS, 2010)

2. Conceitue a filtragem espacial fuzzy baseada em regras.

Resposta:

A teoria de conjuntos de lógica fuzzy, se basea na questão de tratar graus de pertinência de um determinado elemento em relação a pertencer a algum conjunto fuzzy. Lógica Fuzzy, é utilizada para tratar incertezas nos valores de entrada, de modo que, possa classificar um elemento X de um conjunto T, baseado em seu grau de pertinência a aquele conjunto (pertencer). A Figura 5 mostra a comparação entre a teoria de pertinência em um conjunto crisp (conjunto rígido 0 ou 1) e ou conjunto fuzzy, que trada os graus de pertinência entre valores 0 e 1.

Figura 5: Comparação de conjunto *Crisp* com conjunto *Fuzzy*



Fonte: (GONZALEZ, WOODS, 2010)

Ao aplicarmos conjunto *Fuzzy* à filtragem espacial, a metodologia básica é definir propriedades de vizinhança que “capturem” a essência do que os filtros devem detectar. Para exemplificar isso, pode-se imaginar um filtro de extração de fronteira com base em uma regra *Fuzzy* mostrada da seguir:

- Se um pixel pertencer a uma região uniforme, faça com que ele seja banco; senão, faça com que ele seja preto, (preto e branco são conjuntos *fuzzy*)*;*

De acordo com essa única rega, pode-se estabelecer outras regras para limitar as operações de conjuntos fuzzys e definir qual valor atribuir ao pixel, a Figura 6 mostra as regras estabelecida.

Figura 6: regras fuzzy

SE d_2 for *zero* E d_6 for *zero*, ENTÃO z_5 é *branco*
SE d_6 for *zero* E d_8 for *zero*, ENTÃO z_5 é *branco*
SE d_8 for *zero* E d_4 for *zero*, ENTÃO z_5 é *branco*
SE d_4 for *zero* E d_2 for *zero*, ENTÃO z_5 é *branco*
SENÃO z_5 é *preto*

Fonte: (GONZALEZ, WOODS, 2010)

O conjunto de regra mostrado na Figura 6, define uma forma de separar os pixels em dois conjuntos branco (pixels uniformes) e preto (pixels de fronteira). Dado as regras Fuzzy, é possível se realizar filtragens espaciais em imagens ponto a ponto, de modo a tratar ou extrair alguma informação desejada.

3. Gere uma máscara para um filtro espacial usando o seguinte código:

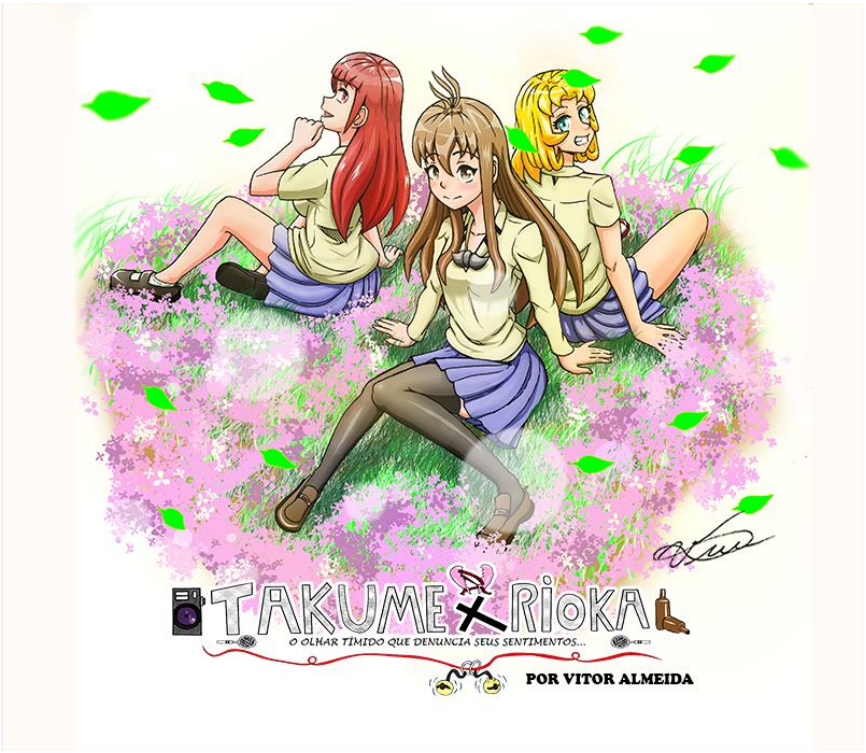
Tam = 3;

Mascara = 1/(Tam*Tam) * ones(Tam,Tam);

Realize a filtragem espacial com a máscara acima e exiba a imagem de saída.

Dados que as operações de convolução e correlação foram explicadas na questão 1, basta dizer que uma filtragem espacial é feita por meio da convolução de uma matriz de coeficientes com a matriz da imagem. Deste modo, as Figura 7 e 8 mostram a imagem original e o resultado da filtragem espacial respectivamente.

Figura 7: imagem original



Fonte: Autoral

Figura 8: imagem filtrada

Fonte: Autoral

É importante destacar que, a imagem filtrada é menor que a imagem original em x-m e y-m, devido a operações feitas para realização da filtragem.

Código Questão 3:

```
%leitura da imagem
img1=imread('img1.png');
imshow(img1(:,:,:));
title('imagem original');
figure();

Tam = 3;
Mascara = 1/(Tam*Tam) * ones(Tam,Tam);

imgFiltred= imfilter(img1,Mascara,'replicate');
imshow(imgFiltred);
title('img filtrada função criada');
figure();


imgFiltred2=convolucao(img1,Mascara);
imshow(imgFiltred2(:,:,:));
title('img filtrada função criada');
figure();


%essa fucnção foi criada baseada nas definições expostas no livro do
%Gonzales pág 97 . Importante lembrar que a imagem final da função é menos
%que a original (x-m-1).
function [imgNova3] = convolucao(img,masck)
[x,y,k] = size(img);
[m,n] = size(masck);
%rotaciona a mascara em 180 graus para realizar a convolução
h=rot90(masck,2);
%cria uma matriz x+m+1 por y+n+1 preenchida de zero
imgNova=zeros(x+m+1,y+n+1,3,'double');
[X1,Y1,~] = size(imgNova);

%coloca a imagem original centralizada dentro da matriz de zeros
for i=1: x
    for j=1: y
        imgNova(i+m-1,j+m-1,:)= img(i,j,:);
    end
end

imgNova2=imgNova;
%realiza o cálculo da soma dos produtos dos pixels com a mascara
for i= 1:X1-m+1
    for j=1:Y1-n+1
        pixelNovo=1;
        for s=1: m
            for t=1: n
                pixelNovo= pixelNovo + h(s,t)*imgNova(i+s-1,j+t-1,:);
            end
        end
        imgNova2(i+round(m/2)-1,j+ round(n/2)-1,:)= round(pixelNovo);
    end
end

imgNova3=uint8(imgNova2(m+1:end-m-1,n+1:end-n-1,:));

end
```

4. **Altere o tamanho (Tam) para 10 e 100, use a máscara recém-gerada para filtrar a imagem original.**

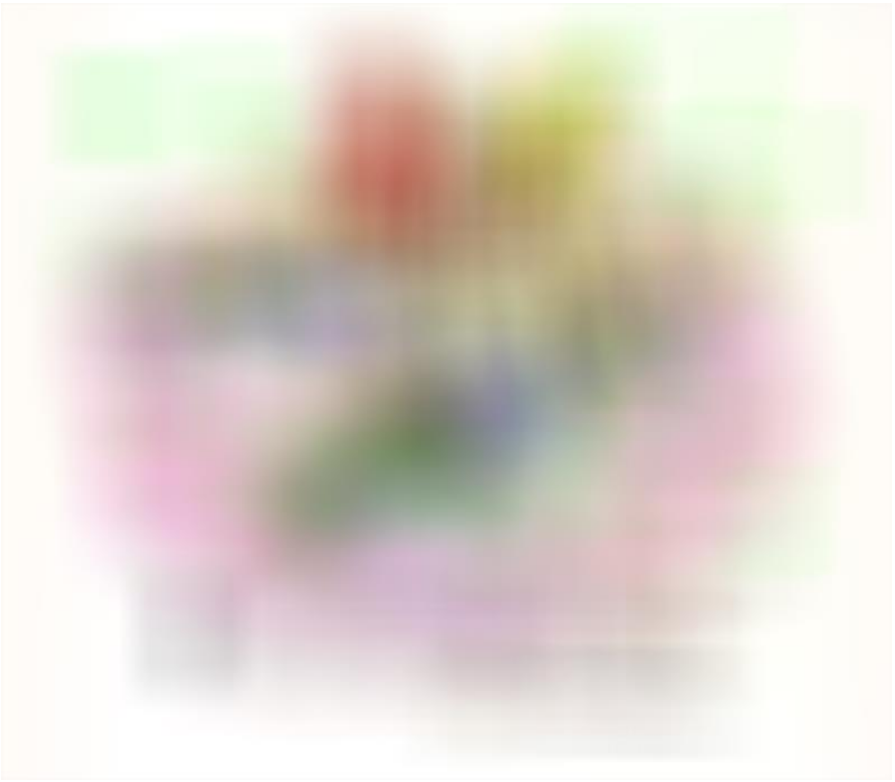
Resposta:

As Figura 9 e 10, mostram os resultados obtidos com a alteração da variável *Tam* para 10 e 100 respectivamente.

Figura 9: Tam=10

Fonte: Autoral

Figura 10: Tam100



Fonte: Autoral

Observando as Figuras 10, percebe-se que o efeito de desfoque provocado pelo filtro, ficou tão intenso que fez a imagem da Figura 10 ficar irreconhecível em comparação com a imagem original.

Para este código, foi utilizado a filtragem presente no matlab, isso foi feito por conta da demora na execução da convolução feita manualmente para um Tam =100. Em aula virtual do dia 26/03/2020, foi indicado que poderia se fazer isto caso já tivesse implementado a função de convolução.

Código Questão 4:

```
%leitura da imagem
img1=imread('img1.png');
imshow(img1(:,:,:));
title('imagem original');
figure();

Tam = 10;
Mascara = 1/(Tam*Tam) * ones(Tam,Tam);

imgFiltred= imfilter(img1,Mascara,'replicate');
imshow(imgFiltred(:,:,:));
title('img filtrada Tam=10 com matlab');
figure();

Tam = 100;
Mascara = 1/(Tam*Tam) * ones(Tam,Tam);

imgFiltred= imfilter(img1,Mascara,'replicate');
imshow(imgFiltred(:,:,:));
title('img filtrada Tam=100 com matlab');
figure();
```

5. Realize a filtragem espacial com as máscaras (Laplace e Sobel) abaixo:

1	1	1
1	-8	1
1	1	1

-1	-2	-1
0	0	0
1	2	1

Resposta:

Filtros de Laplace e Sobel, são filtros baseados em derivadas parciais de duas variáveis e vetores gradientes que servem para realizar os realces de bordas da imagem. O filtro de Laplace, realça de forma melhor as bordas diagonais e verticais. O filtro de Sobel realça melhor as bordas horizontais. As Figuras 11 e 12, mostram os resultados obtidos para os dois filtros Laplace e Sobel respectivamente.

Figura 11: Filtro Laplace

Fonte: Autoral

Figura 12: Filtro Sobel

Fonte: Autoral

Analisando se as Figuras 11 e 12, é possível notar que o Filtro de Laplace obteve melhor precisão nas bordas diagonais e verticais, enquanto que o Sobel realçou melhor as bordas horizontais, isso pode ser visto, por exemplo, vendo que o contorno horizontal das folhas e das letras foi realçado mais claramente no filtro de Sobel.

Código Questão 5:

```
%leitura da imagem
img1=imread('img1.png');
imshow(img1(:,:,:));
title('imagem original');
figure();

MascaraLaplace = [1 1 1 ; 1 -8 1 ; 1 1 1];
MascaraSobel = [-1 -2 -1 ; 0 0 0 ; 1 2 1];

imgFiltred= imfilter(img1,MascaraLaplace,'replicate');
imshow(imgFiltred(:,:,:));
title('img filtrada Laplace matlab');
figure();

imgFiltred= imfilter(img1,MascaraSobel,'replicate');
imshow(imgFiltred(:,:,:));
title('img filtrada Sobel com matlab');
figure();

imgFiltred=convolucao(img1,MascaraLaplace);
imshow(imgFiltred(:,:,:));
title('img filtrada Laplace função feita');
figure();

imgFiltred=convolucao(img1,MascaraSobel);
imshow(imgFiltred(:,:,:));
title('img filtrada Sobel função feita');
figure();

%essa fucnção foi criada baseada nas definições expostas no livro do
%Gonzales pág 97 . Importante lembrar que a imagem final da função é menos
%que a original (x-m-1).
function [imgNova3] = convolucao(img,masck)
[x,y,k] = size(img);
[m,n] = size(masck);
%rotaciona a mascara em 180 graus para realizar a convolução
h=rot90(masck,2);
%cria uma matriz x+m+1 por y+n+1 preenchida de zero
imgNova=zeros(x+m+1,y+n+1,3,'double');
[X1,Y1,~] = size(imgNova);

%coloca a imagem original centralizada dentro da matriz de zeros
for i=1: x
    for j=1: y
        imgNova(i+m-1,j+m-1,:)= img(i,j,:);
    end
end

imgNova2=imgNova;
%realiza o cálculo da soma dos produtos dos pixels com a mascara
for i= 1:X1-m+1
    for j=1:Y1-n+1
        pixelNovo=1;
        for s=1: m
            for t=1: n
                pixelNovo= pixelNovo + h(s,t)*imgNova(i+s-1,j+t-1,:);
            end
        end
        imgNova2(i+round(m/2)-1,j+ round(n/2)-1,:)= round(pixelNovo);
    end
end
imgNova3=uint8(imgNova2(m+1:end-m-1,n+1:end-n-1,:));
end
```

6. **Gere outra máscara de filtro usando o seguinte código:**

Mascara2 = [0 0 0 0 0; 1 1 1 1 1; 0 0 0 0 0; -1 -1 -1 -1 -1; 0 0 0 0 0];
angulo = 0;

filtro = imrotate(mascara2, angulo, 'crop');

Realize a filtragem espacial com a máscara acima e exiba a imagem de saída.

Resposta:

A mascara sugerida consiste em um filtro de Sobel de ordem 5. Deste modo, a Figura 13 mostra o resultado das operações.

Figura 13: Filtro exercício 6

Fonte: Autoral

Analisando a Figura 13, nota-se que, o filtro realçou, com mais intensidade, as bordas horizontais da imagem. Para o filtro indicado no exercício, caso se execute a função ***imfilter()*** do matlab, é possível notar que o filtro ficou invertido, isso se da por conta de que a função utilizada na resolução foi uma

convolução, isto é, o filtro é rotacionado em 180 graus, o matlab por sua vez utiliza uma correlação.

Código questão 6:

```
%leitura da imagem
img1=imread('img1.png');
imshow(img1(:,:,:));
title('imagem original');
figure();

Mascara2 = [0 0 0 0 0; 1 1 1 1 1; 0 0 0 0 0; -1 -1 -1 -1 -1; 0 0 0 0 0];
angulo = 0;
filtro = imrotate(Mascara2, angulo, 'crop');


imgFiltred= imfilter(img1,filtro,'replicate');
imshow(imgFiltred(:,:,:));
title('img filtrada matlab');
figure();


imgFiltred=convolucao(img1,filtro);
imshow(imgFiltred(:,:,:));
title('img filtrada Laplace função feita');
figure();

function [imgNova3] = convolucao(img,masck)
[x,y,k] = size(img);
[m,n] = size(masck);
%rotaciona a mascara em 180 graus para realizar a convolução
h=rot90(masck,2);
%cria uma matriz x+m+1 por y+n+1 preenchida de zero
imgNova=zeros(x+m+1,y+n+1,3,'double');
[X1,Y1,~] = size(imgNova);

%coloca a imagem original centralizada dentro da matriz de zeros
for i=1: x
    for j=1: y
        imgNova(i+m-1,j+m-1,:)= img(i,j,:);
    end
end

imgNova2=imgNova;
%realiza o cálculo da soma dos produtos dos pixels com a mascara
for i= 1:X1-m+1
    for j=1:Y1-n+1
        pixelNovo=1;
        for s=1: m
            for t=1: n
                pixelNovo= pixelNovo + h(s,t)*imgNova(i+s-1,j+t-1,:);
            end
        end
        imgNova2(i+round(m/2)-1,j+ round(n/2)-1,:)= round(pixelNovo);
    end
end
imgNova3=uint8(imgNova2(m+1:end-m-1,n+1:end-n-1,:));
end
```

7. **Altere o Ângulo para 90 e 196, exiba o resultado e explique a diferença entre diferentes saídas.**

Resposta:

As Figuras 14 e 15, mostram os resultados para as filtragens com ângulo de 90 graus e 196 graus respectivamente.

Figura 14: Filtragem com ângulo de 90 graus

Fonte: Autoral

Figura 15: Filtragem com ângulo de 196 graus

Fonte: Autoral

Filtros de Sobel, são construídos baseados em teoria de vetores gradientes, a Figura 16 mostra a definição disponível em (GONZALEZ, WOODS, 2010) para o vetor gradiente utilizado para aguçamento de imagens.

Figura 16: definição vetor gradiente

$$\nabla f \equiv \text{grad}(f) \equiv \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

Fonte: (GONZALEZ, WOODS, 2010)

Esse vetor tem a propriedade geométrica de apontar na direção da maior taxa de variação de f na posição (x,y). Essa valor, presente em (x,y), pode ser calculado através da norma do vetor gradiente supracitado.

Da mesma forma, as derivadas definidas na Figura 16, não são invariantes em rotação (isotrópicas), porém, a magnitude (norma) é. Com isso, é possível se ter mascaras gradientes que sejam isotrópicas somente em um número limitado de incremento de rotações.

Como indicado em (GONZALEZ, WOODS, 2010), as mascaras mais populares utilizadas como uma aproximação do gradiente são isotrópicas em múltiplos de 90 graus. Deste modo, observa-se que na Figura 14, o filtro não foi isotrópico, sendo assim, a alteração do ângulo realçou com mais intensidade as bordas verticais. A Figura 16, por sua vez, não sofreu efeito do ângulo de 196 graus, realçando assim, as bordas horizontais.

Código Questão 7:

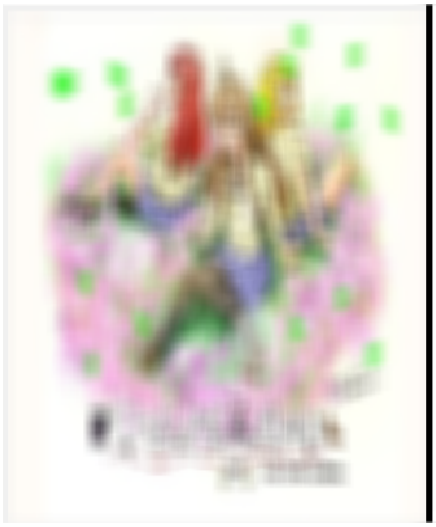
```
%leitura da imagem
img1=imread('img1.png');
imshow(img1(:,:,:));
title('imagem original');
figure();

Mascara2 = [0 0 0 0 0; 1 1 1 1 1; 0 0 0 0 0; -1 -1 -1 -1 -1; 0 0 0 0 0];
angulo = 90;
filtro = imrotate(Mascara2, angulo, 'crop');

imgFiltred=convolucao(img1,filtro);
imshow(imgFiltred(:,:,:));
title('img filtrada Laplace função feita angulo=90');
figure();

angulo = 196;
filtro = imrotate(Mascara2, angulo, 'crop');

imgFiltred=convolucao(img1,filtro);
imshow(imgFiltred(:,:,:));
title('img filtrada Laplace função feita, angulo=196');
figure();


function [imgNova3] = convolucao(img,masck)
[x,y,k] = size(img);
[m,n] = size(masck);
%rotaciona a mascara em 180 graus para realizar a convolução
h=rot90(masck,2);
%cria uma matriz x+m+1 por y+n+1 preenchida de zero
imgNova=zeros(x+m+1,y+n+1,3,'double');
[X1,Y1,~] = size(imgNova);

%coloca a imagem original centralizada dentro da matriz de zeros
for i=1: x
    for j=1: y
        imgNova(i+m-1,j+m-1,:)= img(i,j,:);
    end
end

imgNova2=imgNova;
%realiza o cálculo da soma dos produtos dos pixels com a mascara
for i= 1:X1-m+1
    for j=1:Y1-n+1
        pixelNovo=1;
        for s=1: m
            for t=1: n
                pixelNovo= pixelNovo + h(s,t)*imgNova(i+s-1,j+t-1,:);
            end
        end
        imgNova2(i+round(m/2)-1,j+ round(n/2)-1,:)= round(pixelNovo);
    end
end
imgNova3=uint8(imgNova2(m+1:end-m-1,n+1:end-n-1,:));
end
```

8. **Utilize a máscara da questão 4, Tam igual á 10, na função *imfilter,* com os padding (circular), *symmetric, replicate, circular.* Explique a diferença de cada um deles.**

Resposta:

Para melhor visualização dos resultados as dimensões da imagem foram reduzidas para 210x241x3 (isso foi feito tirando um print da imagem orinial e salvando na mesma pasta como img3.jpg). Os resultados da aplicação dos tipos de cálculos indicados são mostrados nas Figuras 16, 17 e 18, sendo que a esquerda está a imagem e a direita parte da matriz representada numericamente.

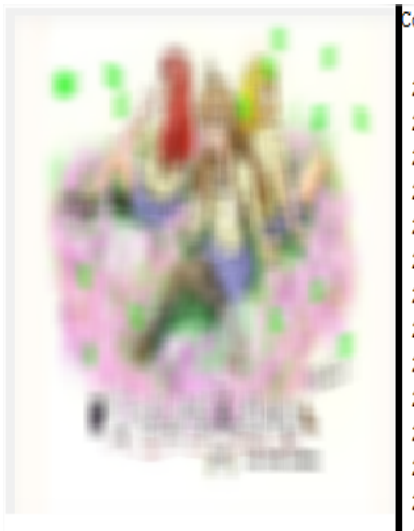
Figura 16: Circular



```
Columns 1 through 18

 248   248   248   249   250   251   251   252   252   252   252   252   252   252   252   252   251   251
 248   248   248   249   250   251   251   251   252   252   252   252   252   252   252   251   251   251
 248   248   248   249   250   251   251   251   252   252   252   252   252   252   252   251   251   251
 248   248   248   249   250   251   251   252   252   252   252   252   252   252   252   251   251   251
 248   248   249   250   251   252   252   252   252   252   252   252   252   252   252   251   251   251
 249   249   250   251   252   253   253   253   253   253   253   253   253   253   253   252   252
 250   250   251   252   253   254   254   254   255   255   254   255   254   254   254   254   253   253
 250   250   251   252   253   254   255   255   255   255   255   255   255   254   254   254   254   254
 250   250   250   252   253   254   255   255   255   255   255   255   255   255   254   254   254   254
 250   250   250   252   253   254   254   254   255   255   255   255   255   255   255   254   254   254
 250   250   250   252   253   254   254   254   254   255   255   255   255   255   255   254   254   254
 250   250   250   252   253   254   254   254   254   255   255   255   255   255   255   255   255   255
 250   250   250   252   253   254   254   254   254   255   255   255   255   255   255   255   255   255
 250   250   250   252   253   254   254   254   254   255   255   255   255   255   255   255   255   255
```

Fonte: Autoral

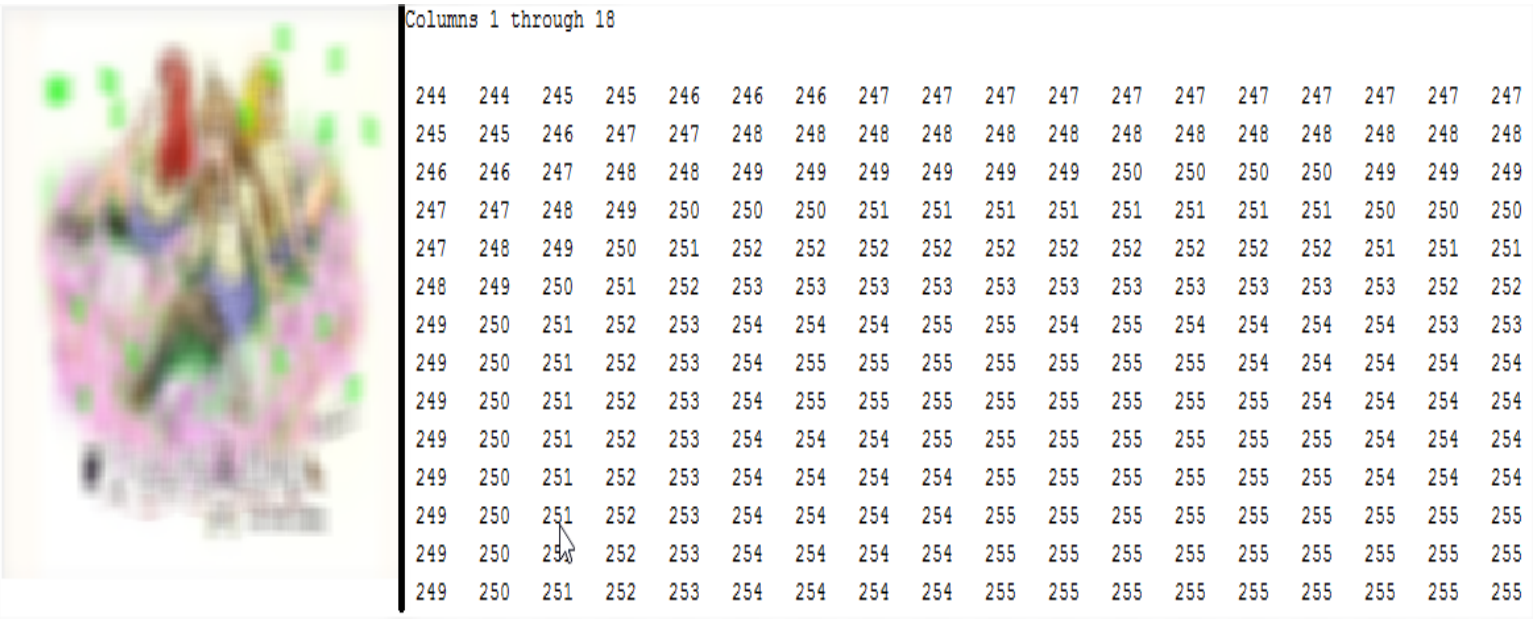
Figura 17: symmetric



```
Columns 1 through 18

 247   247   247   247   248   249   249   249   249   249   249   250   250   249   249   249   249   248
 247   247   247   248   248   249   249   249   249   249   249   250   249   249   249   249   249   248
 247   247   248   248   248   249   249   249   249   249   250   250   250   250   249   249   249   249
 249   249   249   249   250   250   250   251   251   251   251   251   251   251   251   250   250   250
 250   250   250   250   251   252   252   252   252   252   252   252   252   252   252   251   251   251
 251   251   251   251   252   253   253   253   253   253   253   253   253   253   253   253   252   252
 252   252   252   252   253   254   254   254   255   255   254   255   254   254   254   254   253   253
 252   252   252   252   253   254   255   255   255   255   255   255   255   254   254   254   254   254
 252   252   252   252   253   254   255   255   255   255   255   255   255   255   254   254   254   254
 252   252   252   252   253   254   254   254   255   255   255   255   255   255   255   254   254   254
 252   252   252   252   253   254   254   254   254   255   255   255   255   255   255   254   254   254
 252   252   252   252   253   254   254   254   254   255   255   255   255   255   255   255   255   255
 252   252   252   252   253   254   254   254   254   255   255   255   255   255   255   255   255   255
 252   252   252   252   253   254   254   254   254   255   255   255   255   255   255   255   255   255
```

Fonte: Autoral

Figura 18: replicate



Columns 1 through 18

244	244	245	245	246	246	246	247	247	247	247	247	247	247	247	247	247	247
245	245	246	247	247	248	248	248	248	248	248	248	248	248	248	248	248	248
246	246	247	248	248	249	249	249	249	249	249	250	250	250	250	249	249	249
247	247	248	249	250	250	250	251	251	251	251	251	251	251	251	250	250	250
247	248	249	250	251	252	252	252	252	252	252	252	252	252	252	251	251	251
248	249	250	251	252	253	253	253	253	253	253	253	253	253	253	253	252	252
249	250	251	252	253	254	254	254	255	255	254	255	254	254	254	254	253	253
249	250	251	252	253	254	255	255	255	255	255	255	255	254	254	254	254	254
249	250	251	252	253	254	255	255	255	255	255	255	255	255	254	254	254	254
249	250	251	252	253	254	254	254	255	255	255	255	255	255	255	254	254	254
249	250	251	252	253	254	254	254	254	255	255	255	255	255	255	254	254	254
249	250	251	252	253	254	254	254	254	255	255	255	255	255	255	255	255	255
249	250	251	252	253	254	254	254	254	255	255	255	255	255	255	255	255	255
249	250	251	252	253	254	254	254	254	255	255	255	255	255	255	255	255	255

Fonte: Autoral

Analisando as Figuras 16, 17 e 18, nota-se que, visualizando a matriz como uma imagem, não é possível perceber diferença nas imagens para cada tipo de tratamento. Por outro lado, quando se visualiza as imagens na forma numérica, é possível notar diferença nos valores, especialmente quando estão próximo a borda da imagem, ou seja, nos cantos.

Esse efeito, é explicado pela diferença no tratamento das informações no canto da imagem que cada um *circular*, *replicate* e *symetric* fornece, a informação ***padding*** **(preenchimento)**, se refere ao parâmetro que define a forma de preencher os cantos e não a uma das formas, isso foi corrigido para o **circular**. Como pode ser visto em Mathworks (2020), as operações são as seguintes:

- ***Circular*** **(circular):** Os valores de entrada que estão fora das dimensões da matriz são calculados assumindo implicitamente que a matriz é periódica;
- ***Symmetric*** ***(*****simétrico):** Os valores de entrada que estão fora das dimensões da matriz são calculados pela reflexão da matriz através das bordas;

- ***Replicate* (replicar):** Valores de entrada fora dos limites da matriz são assumidos como iguais ao valor de borda mais próximo.

As afirmações anteriores, indicam os tipos de tratamento de valores nos limites da imagem, que estão disponíveis na função *imfilter()* do Matlab. De acordo com cada um dos tratamen, se tem valores diferentes, porém aproximados, para as bordas da imagem e por conta dessa aproximação, é difícil notar o efeito visual na imagem.

Código questão 8:

```
%leitura da imagem
img1=imread('img3.jpg');
imshow(img1(:,:,:));
title('imagem original');
figure();

Tam = 10;
Mascara = 1/(Tam*Tam) * ones(Tam,Tam);

imgFiltred1= imfilter(img1,Mascara,'circular');
imshow(imgFiltred1(:,:,:));
title('img filtrada Tam=10, Cirular');
figure();

imgFiltred2= imfilter(img1,Mascara,'symmetric');
imshow(imgFiltred2(:,:,:));
title('img filtrada Tam=10, symmetric');
figure();

imgFiltred3= imfilter(img1,Mascara,'replicate');
imshow(imgFiltred3(:,:,:));
title('img filtrada Tam=10, replicate');
figure();
```

REFERÊNCIAS

MATHWORKS (org.). **Imfilter.** 2020. Disponível em: https://www.mathworks.com/help/images/ref/imfilter.html. Acesso em: 29 mar. 2020.

GONZALEZ, Rafael C., WOODS, Richard E. **Processamento Digtal de Imagem.** São Paulo: Sv, 2010.