UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

JOÃO LUIZ GRAVE GROSS
180171

Relatório – Laboratório 1

Trabalho da Disciplina Fundamentos de Processamento de Imagens

Prof. Manuel Menezes de Oliveira Neto

Porto Alegre, 26 de setembro de 2011.

LABORATÓRIO 1 – Questões

Questão 1. Leia sobre o comando **imread** digitando "*help imread*" e depois carregue a imagem 'cameraman' (ex: " I = imread('cameraman.tif') "). Para visualizá-la, utilize o comando **imshow** (ex: " imshow(I);").

O comando **imread** lê uma imagem passada a ela como parâmetro e retorna uma matriz bidimensional (M x N) para imagens preto e branca, ou tridimensional (M x N x 3) para imagens coloridas, sendo neste último caso cada pixel representado por 3 valores, das inensidades das cores vermelho (Red), verde (Green) e azul (Blue).

Executando a instrução " I = imread('cameraman.tif') " é atribuída à variável I uma matriz M x N, contendo em cada posição m_{ij} x n_{kl} um valor, correspondente à escala de preto e branco de cada pixel da imagem.

O comando **imshow** pega uma matriz e mostra na tela a imagem correspondente. Ao utilizar " imshow(I) " a imagem carregada à I usando imread foi exbida na tela.

Questão 2. Calcule o negativo de uma imagem em tons de cinza (negativo = 255 – imagem_original). Visualize as duas imagens lado a lado. Para tanto, utilize os comandos **figure**, **subplot** e **imshow** (digite *help <comando* para detalhes).

O negativo de uma imagem é o complemento do valor de cada pixel de uma imagem, ou seja, se todos os pixels de uma imagem são representados por valores com manetude variando de 0 a 255 e tivermos um pixel da imagem com valor 234, o negativo deste pixel será 255 – 234, ou 11.

Para calcular o negativo da imagem 'cameraman.tif' executei alguns comandos, obtendo o resultado esperado. Acompanhe:

```
I = imread('cameraman.tif');
J = 255 - I;
subplot(1,2,1), imshow(I);
subplot(1,2,2), imshow(J);
```

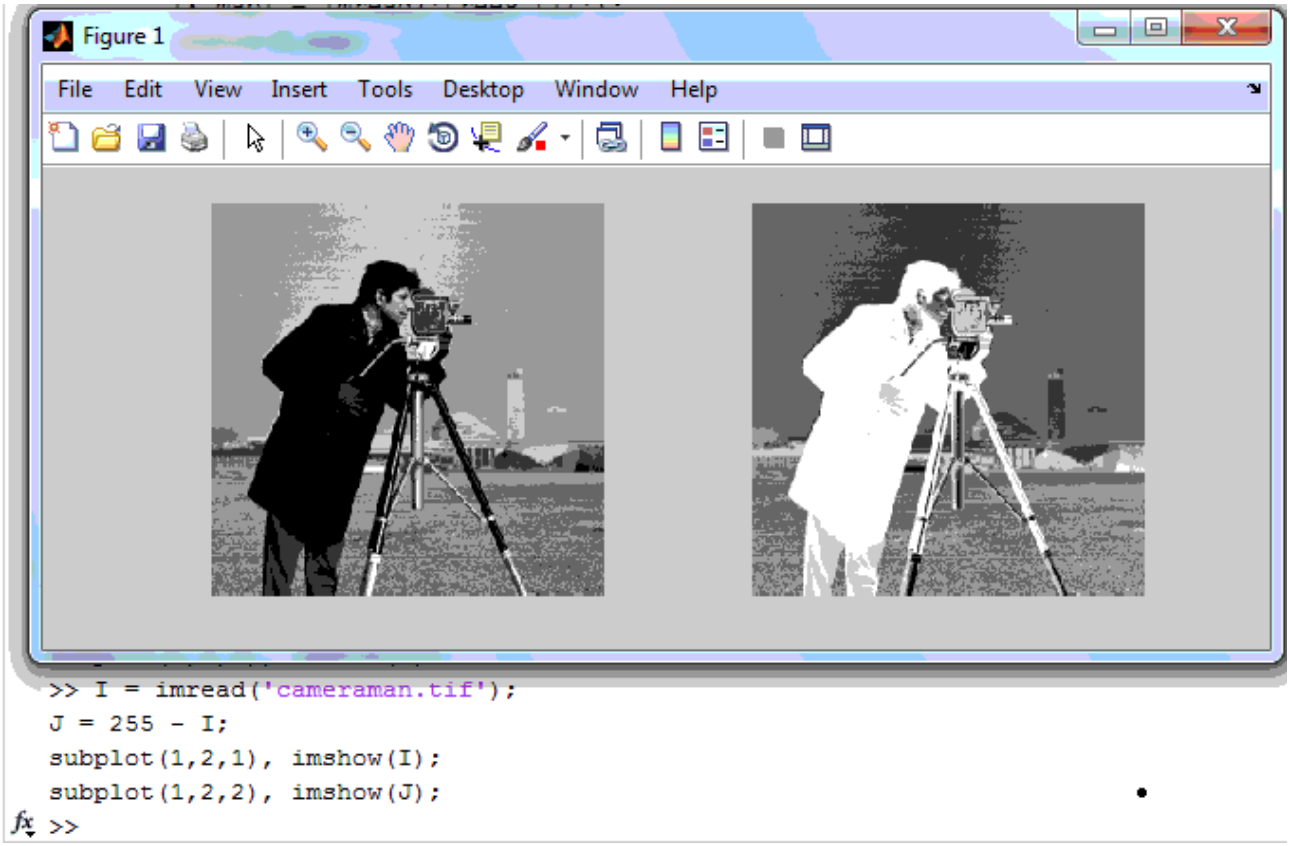

Figura 1: Imagem original e negativo do arquivo ‘'cameraman.tif’

O comando **subplot** seleciona um pedaço da janela de exibição para o qual será impressa a imagem. Como podemos ver no exemplo acima o uso do subplot fez com que pudessemos colocar duas imagens em uma mesma janela de exibição.

O comando **figure** é usado para criar novas janelas. Veja o exemplo abaixo:

```
I = imread('cameraman.tif');
J = 255 - I;
figure('Position', [100 100 300 300]), imshow(I);
figure('Position', [560 100 300 300]), imshow(J);
```

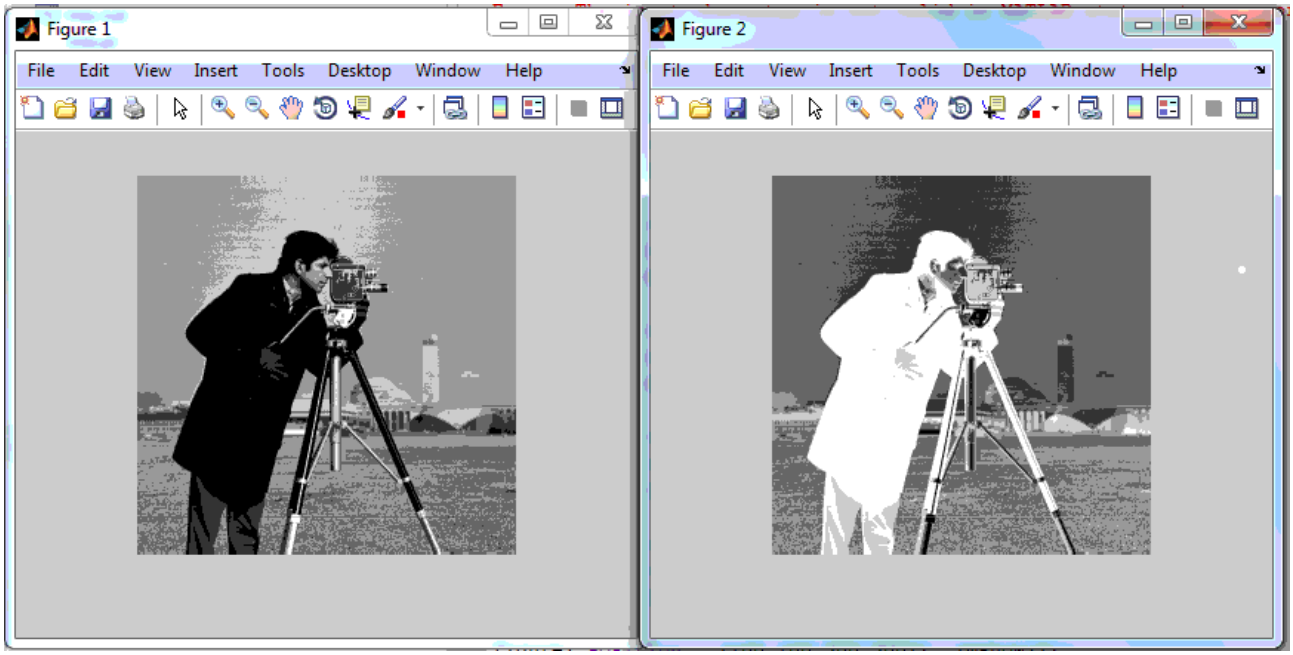

Figura 2: Imagem original e negativo usando o comando figure

Questão 3. Baixe a imagem colorida http://www.inf.ufrgs.br/~oliveira/Chateau_small.jpg" e salve no diretório "work" do MATLAB. Obtenha o seu negativo e visualize-o lado a lado com a imagem original.

Primeiramente acessei o link, baixei a imagem e salvei no meu diretório de trabalho. No MatLab alterei a pasta de trabalho vigente do programa, através do comando **cd** (change directory).

cd My' Dropbox'\UFRGS\2011-02\INF01046' - Fundamentos de Processamento de Imagens'\Laboratorio1\

Executei os mesmos comandos da questão 2, usando subplot e a imagem 'Chateau_small.jpg'. Obtive os seguintes resultados:

```
I = imread('Chateau_small.jpg');
J = 255 - I;
subplot(1,2,1), imshow(I);
subplot(1,2,2), imshow(J);
```

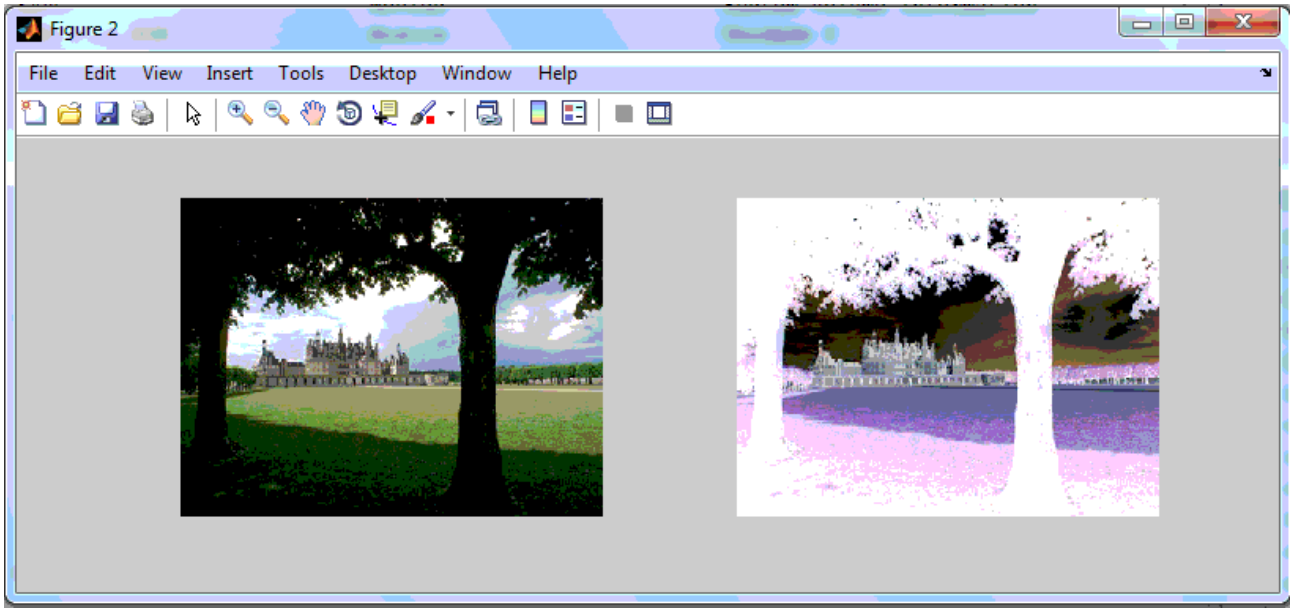

Figura 3: Imagens original e negativo do arquivo ‘Chateau_small.jpg’

Questão 4. Estude o comando **conv2** do MATLAB. Em seguida crie matrizes (3x3) representando cada um dos filtros descritos abaixo, convolva-os com a imagem do cameraman e descreva o esultado obtido (não esqueça de converter o resultado da convolução para o tipo uint8):

O comando **conv2** é utilizado no MATLAB para fazer convoluções de duas dimensões.

(a) **Filtro Gaussiano** (passa baixas):
G = [0.0625 0.125 0.0625**;** 0.125 0.25 0.125**;** 0.0625 0.125 0.0625]

O filtro gaussiano é um filtro passa baixa, ou seja, um filtro que só deixa frequencias baixas serem exibidas. A aplicação desse filtro na imagem (convolução usando kernel semelhante a G) a deixa um pouco borrada, pois os contornos dos objetos (áreas de altas frequencias) são modificados de modo a fazer a transição entre lado externo do objeto, contorno e lado interno do objeto mais suaves.

A aplicação de tal filtro pode ser vista a seguir:

```
I = imread('cameraman.tif');
kernel = [0.0625 0.125 0.0625; 0.125 0.25 0.125; 0.0625 0.125 0.0625];
gauss = conv2(double(I),kernel);
subplot(1,2,1), imshow(I);
subplot(1,2,2), imshow(uint8(gauss));
```

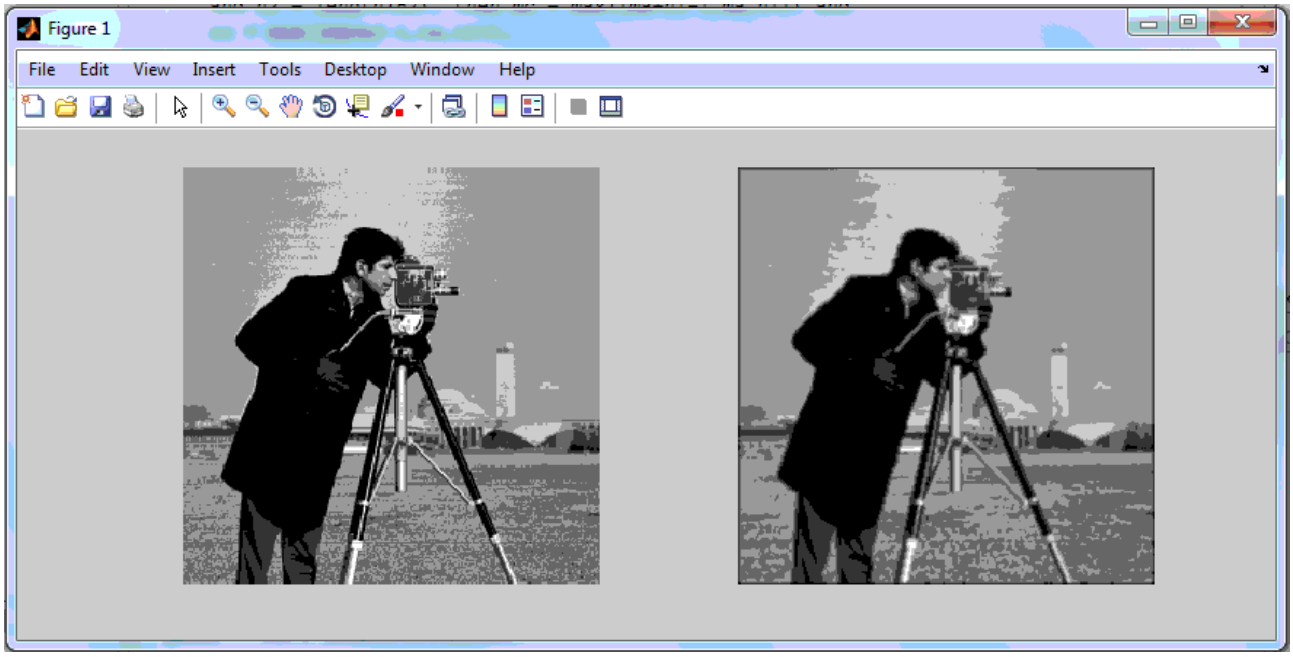

Figura 4: Imagem original e com a aplicação do filtro de gauss

(b) Utilize o filtro Gaussiano 3 vezes seguidas, reaplicando-o ao resultado do passo anterior.

```
I = imread('cameraman.tif');
kernel = [0.0625 0.125 0.0625; 0.125 0.25 0.125; 0.0625 0.125 0.0625];
gauss1 = conv2(double(I),kernel);
gauss2 = conv2(double(gauss1),kernel);
gauss3 = conv2(double(gauss2),kernel);
subplot(2,2,1), imshow(I);
subplot(2,2,2), imshow(uint8(gauss1));
subplot(2,2,3), imshow(uint8(gauss2));
subplot(2,2,4), imshow(uint8(gauss3));
```

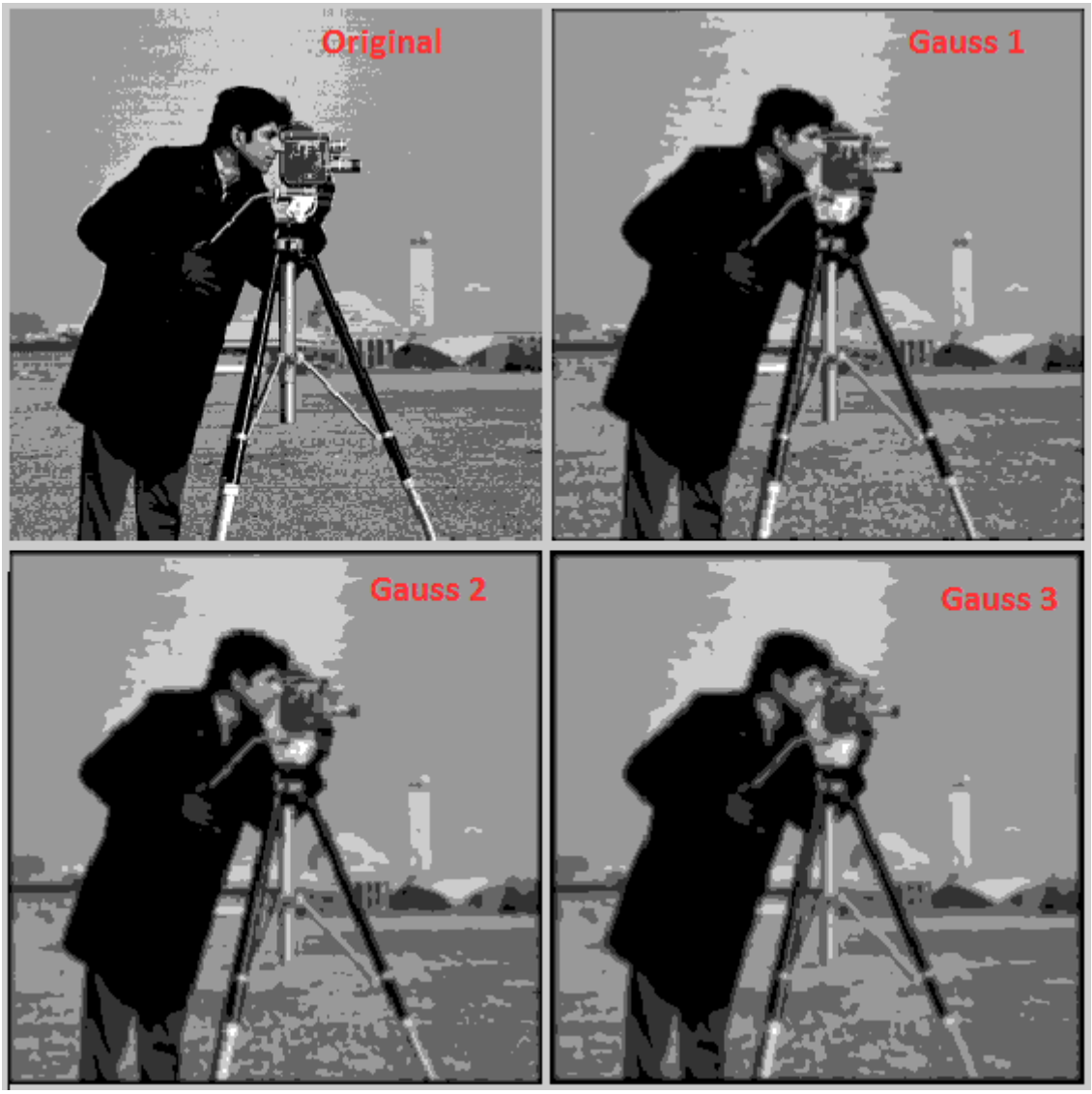

Figura 5: Imagem original e aplicação do filtro de gauss 3 vezes

(c) **Filtro Laplaciano** (passa altas):
L = [0 -1 0; -1 4 -1; 0 -1 0]

Assim como o filtro de gauss, o filtro laplaciano também só eia algumas frequencias passarem. Porém este filtro deixa passar apenas as frequencias altas, ou seja, os contornos dos objetos, onde há troca brusca de cores.

Os comandos executados foram os seguinte:

```
I = imread('cameraman.tif');
kernel = [0 -1 0; -1 4 -1; 0 -1 0];
laplace = conv2(double(I),kernel);
subplot(1,2,1), imshow(I);
subplot(1,2,2), imshow(uint8(laplace));
```

Figura 6: Imagem original e aplicação do filtro laplaciano

(d) **Remoção da Média**:
RM = [-1 -1 -1; -1 9 -1; -1 -1 -1]

```
I = imread('cameraman.tif');
kernel = [-1 -1 -1; -1 9 -1; -1 -1 -1];
rm = conv2(double(I),kernel);
subplot(1,2,1), imshow(I);
subplot(1,2,2), imshow(uint8(rm));
```

Figura 7: Imagem original e remoção de média

(e) **Prewitt horizontal**:
PH = [-1 0 1; -1 0 1; -1 0 1]

```
I = imread('cameraman.tif');
kernel = [-1 0 1; -1 0 1; -1 0 1];
prewit = conv2(double(I),kernel);
subplot(1,2,1), imshow(I);
subplot(1,2,2), imshow(uint8(prewit));
```

Figura 8: Imagem original e imagem com a aplicação do filtro prewit

(f) Inverta os sinais em PH e compare o resultado com o encontrado em (e)

```
I = imread('cameraman.tif');
kernel = [1 0 -1; 1 0 -1; 1 0 -1];
prewit = conv2(double(I),kernel);
subplot(1,2,1), imshow(I);
subplot(1,2,2), imshow(uint8(prewit));
```

Figura 9: Imagem original e imagem com a aplicação do filtro prewit com sinais invertidos

Analisando a imagem obtida em (e) e em (f) pode-se perceber que o resultado de ambas é praticamente idêntico, à excessão de uma coluna branca que aparece à direita em (e) e à esquerda em (f), isso pelo fato de que com a mudança dos sinais o efeito do filtro ocorre em sentidos diferentes.

(g) **Prewitt vertical**:
PV = [-1 -1 -1; 0 0 0; 1 1 1]

```
I = imread('cameraman.tif');
kernel = [-1 -1 -1; 0 0 0; 1 1 1];
prewit = conv2(double(I),kernel);
subplot(1,2,1), imshow(I);
subplot(1,2,2), imshow(uint8(prewit));
```

Figura 10: Imagem original e imagem com a aplicação do filtro prewit vertical

Caso fosse aplicado o filtro de prewit vertical com os sinais invertidos, ocorreria o mesmo efeito de (e) e (f), ou seja, a linha branca que vemos no pé da Figura 10 passaria a ficar no topo da imagem.

Questão 5. Sejam Iph e Ipv os resultados das convoluções da imagem do cameraman com os filtros PV e PH, respectivamente. Calcule a **magnitude to vetor gradiente**: *mag_grad(i,j) = sqrt(Iph(i,j)² + Ipv(i,j²₂)*. Neste caso, faça a iteração sobre os elementos da imagem por meio de laços utilizando o comando **for** do MATLAB. Obs.: Não esqueça de fazer as conversões para double e uint8, quando necessárias.

```
I = imread('cameraman.tif');
kernelV = [-1 -1 -1; 0 0 0; 1 1 1];
Ipv = conv2(double(I),kernelV);
kernelH = [-1 0 1; -1 0 1; -1 0 1];
Iph = conv2(double(I),kernelH);
for i=1:256
        for j=1:256
                mag_grad(i,j) = sqrt ( (Ipv(i,j)^2)+(Iph(i,j)^2));
        end
end
subplot(2,2,1), imshow(I);
subplot(2,2,2), imshow(uint8(mag_grad));
subplot(2,2,3), imshow(uint8(Ipv));
subplot(2,2,4), imshow(uint8(Iph));
```

Figura 11: magnitude do vetor gradiente

Questão 6. Utilize também o comando **.*** (ponto asterisco) que faz a multiplicação de elementos correspondentes de uma matrix (ou vetor) para, junto com o comando **sqrt**, calcular a magnitude do gradiente da imagem utilizando **apenas uma linha de comando** do MATLAB. Compare o resultado com o obtido na questão anterior, exibindo-os lado a lado em uma janela por meio do comando **subplot**.

```
I = imread('cameraman.tif');
kernelV = [-1 -1 -1; 0 0 0; 1 1 1];
Ipv = conv2(double(I),kernelV);
kernelH = [-1 0 1; -1 0 1; -1 0 1];
Iph = conv2(double(I),kernelH);
for i=1:256
        for j=1:256
                mag_grad1(i,j) = sqrt ( (Ipv(i,j)^2)+(Iph(i,j)^2));
        end
end
mag_grad2 = sqrt ( (Ipv .* Ipv)+(Iph .* Iph));
```

```
subplot(1,2,1), imshow(uint8(mag_grad1));
subplot(1,2,2), imshow(uint8(mag_grad2));
```

Figura 12: comparação dos vetores gradiente

Não percebi nenhuma diferença entre as imagens, a única diferenã é que o operador .* tornou o código mais curto, obtendo o mesmo resultado.