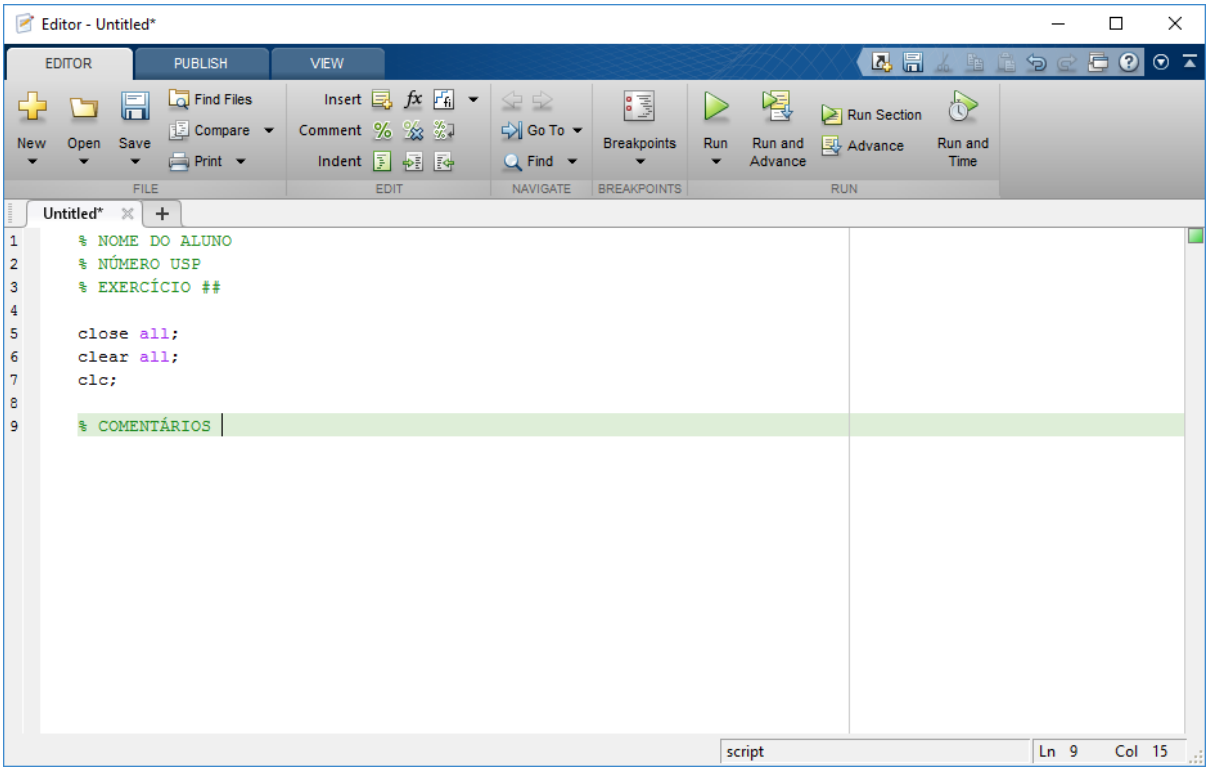
SEL5886 –Visão Computacional
Prof. Dr. Marcelo Andrade da Costa Vieira

Prática 2 – Processamento no Domínio do Espaço

Instruções:

- Essa prática consiste de 12 exercícios (E_1 a E_12).
- Deve ser gerado um arquivo no editor do Matlab (extensão *.m) para cada exercício pedido.
- Deve-se colocar comentários nos programas desenvolvidos.
- As perguntas devem ser respondidas também como comentários no arquivo.
- Deve-se tornar o diretório onde estão as figuras e os arquivos *.m como um diretório padrão do Matlab.
- Depois de terminado os exercícios, todos os arquivos *.m devem ser comprimidos em um único arquivo e enviado ao professor pelo site de *upload* da IRIS até a data máxima de entrega.

- Utilizar o padrão mostrado na Figura abaixo para seus arquivos *.m:
  - Colocar um cabeçalho contendo seu nome, número USP e o número do exercício correspondente (E1, E2, E3...);
  - Iniciar todos os exercícios com os 3 comandos mostrados na Figura abaixo, que servem para limpar as variáveis e as figuras abertas, além de limpar a tela de comando do Matlab;
  - Colocar comentários nas linhas de programa.



1) Mostrando uma Imagem de Intensidades

Carregue a imagem dada pela Matriz E, em escala de cinza, e observar os resultados.

```
E = [0  2  4  6  8  10  12  14  16  18  20  22  24  26  28  30  32  34  36  38  40
     0  2  4  6  8  10  12  14  16  18  20  22  24  26  28  30  32  34  36  38  40
     0  2  4  6  8  10  12  14  16  18  20  22  24  26  28  30  32  34  36  38  40
     0  2  4  6  8  10  12  14  16  18  20  22  24  26  28  30  32  34  36  38  40
     0  2  4  6  8  10  12  14  16  18  20  22  24  26  28  30  32  34  36  38  40
     0  2  4  6  8  10  12  14  16  18  20  22  24  26  28  30  32  34  36  38  40
     0  2  4  6  8  10  12  14  16  18  20  22  24  26  28  30  32  34  36  38  40
     0  2  4  6  8  10  12  14  16  18  20  22  24  26  28  30  32  34  36  38  40
     0  2  4  6  8  10  12  14  16  18  20  22  24  26  28  30  32  34  36  38  40
     0  2  4  6  8  10  12  14  16  18  20  22  24  26  28  30  32  34  36  38  40
     0  2  4  6  8  10  12  14  16  18  20  22  24  26  28  30  32  34  36  38  40
     0  2  4  6  8  10  12  14  16  18  20  22  24  26  28  30  32  34  36  38  40
     0  2  4  6  8  10  12  14  16  18  20  22  24  26  28  30  32  34  36  38  40
     0  2  4  6  8  10  12  14  16  18  20  22  24  26  28  30  32  34  36  38  40
     0  2  4  6  8  10  12  14  16  18  20  22  24  26  28  30  32  34  36  38  40
     0  2  4  6  8  10  12  14  16  18  20  22  24  26  28  30  32  34  36  38  40
     0  2  4  6  8  10  12  14  16  18  20  22  24  26  28  30  32  34  36  38  40];

IE = mat2gray(E);
```

E_1: Usar a função *imshow* para mostrar a imagem IE acima na sua escala de cinza. Observar os resultados.

a) Comentar os resultados mostrados em cada caso.

```
imshow(IE, [])
imshow(IE, [0.2, 0.6])
imshow(IE, [0 2])
```

Dica: Para melhor visualização, acrescente o parâmetro 'InitialMagnification' ao imshow.
Ex.: `imshow(..., 'InitialMagnification', 'fit')`

Converter a Imagem de Intensidades *IE* para a classe *uint8:* (A Figura 1 mostra as funções para conversão entre as classes de dados de Imagens)

```
F = im2uint8(IE);
```

Mostrar a imagem F e responder:

b) Qual o intervalo dos valores de intensidade para os pixels da imagem IE?
c) Qual o intervalo dos valores de intensidade para os pixels da imagem F?

Funções de conversão entre classes de dados de imagens

Name	Converts Input to:	Valid Input Image Data Classes
`im2uint8`	`uint8`	`logical`, `uint8`, `uint16`, and `double`
`im2uint16`	`uint16`	`logical`, `uint8`, `uint16`, and `double`
`mat2gray`	`double` (in range [0, 1])	`double`
`im2double`	`double`	`logical`, `uint8`, `uint16`, and `double`
`im2bw`	`logical`	`uint8`, `uint16`, and `double`

2) Convertendo uma Imagem de Intensidades.

E_2: Converter a Matriz G abaixo:

```
G = [-0.7    1.2    0.4   -0.6   -0.4    1.2
     -1.6   -0.6    0.4    0.8    0.0    0.8
      1.5   -0.5    0.1    0.2   -0.8    1.1
      2.1    0.9    0.5   -0.5   -1.6   -0.7
      0.1   -1.4    1.1   -1.2    0.2   -0.9
     -1.0   -2.0    1.0    0.6   -0.1   -1.3];
```

a) Para Imagem de Intensidade com valores no intervalo [0, 1].
b) Para Imagem Binária com Limiar de 25%.
c) Para Imagem com valores no intervalo [0, 255].
d) Comente os resultados encontrados em a, b, c.
e) Qual a diferença entre as duas conversões abaixo? Explique e comente.

- `G1 = im2double(im2uint8(G))`
- `G2 = mat2gray(G)`

3) Avaliação de mudanças na resolução espacial.

E_3: Modificar resolução espacial de uma imagem.

a) Carregar a o arquivo de imagem "*`frutas.bmp`*"
b) Utilizando a função *`imresize`*, reduza o tamanho da imagem de 512x512 para
    a. 256x256
    b. 128x128
    c. 64x64
    d. 32x32
c) Mostrar todas as imagens usando janelas do mesmo tamanho e comentar os resultados**.**

Dica: Para manter as imagens do mesmo tamanho use o parâmetro 'InitialMagnification' do imshow. Ex.: `imshow(..., 'InitialMagnification', 'fit')`

4) Avaliação de mudanças na resolução de escala de cinza.

E_4: Modificar resolução de níveis de cinza da imagem abaixo:

a) Carregar a o arquivo de imagem "*frutas.bmp*"
b) Reduzir a escala de níveis de cinza da imagem de 256 níveis para:
    a. 128
    b. 64
    c. 16
    d. 4
    e. 2

Usando no seu código os comandos abaixo:

```
t = <nova escala de níveis de cinza desejada>;
resultado = round(double(img).*(t-1)/255);

figure, imshow(resultado, [0 t-1])
```

c) Comente o código acima e explique o que cada comando faz.
d) Mostrar todas as imagens (no mesmo tamanho) e comentar os resultados.

Dica: Para manter as imagens do mesmo tamanho use o parâmetro 'InitialMagnification' do imshow. Ex.: `imshow(..., 'InitialMagnification', 'fit')`

5) Visualização de Histogramas

E_5: Ler a imagem "*mammogram.bmp*" **e gerar seu histograma.**

```
f = imread('mammogram.bmp');
imshow(f)
figure, imhist(f)
a = imfinfo ('mammogram.bmp')
```

6) Transformação de intensidades

As técnicas de processamento no domínio espacial operam diretamente nos pixels da imagem. A expressão geral para a **Função de Transformação** nos níveis de cinza pode ser dada por:

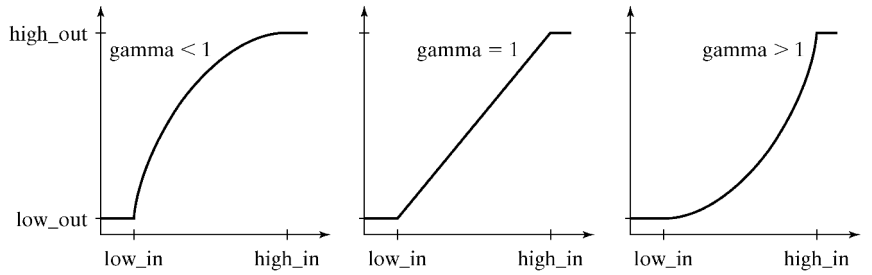
$$g(x,y) = T[f(x,y)]$$

onde ***f(x,y)*** é a imagem de entrada e ***g(x,y)*** é a imagem de saída ou imagem processada.
T é um operador em ***f.***

A função do MatLab que realiza transformações de intensidade nos níveis de cinza de uma imagem é a ***imadjust*** que tem a seguinte sintaxe:

g = imadjust(f, [low_in high_in], [low_out high_out], gamma)

cuja função de transformação é vista nos gráficos a seguir:



a b c

FIGURE 3.2 The various mappings available in function `imadjust`.

E_6: Processe a imagem "*face.bmp*" usando as funções de transformação abaixo:

a) g1 = imadjust(f, [0 1], [1 0])
b) g2 = imadjust(f, [0.5 0.75], [0 1])
c) g3 = imadjust(f, [ ], [ ], 2)

Mostre as três imagens resultantes e comente os resultados encontrados.

E_7: *Operadores ponto a ponto*.

a) Modificar a imagem "face.bmp" agora usando as seguintes relações*:
   a. g=2f - 40
   b. g=30log10(f+1)
   c. g=f^1.4
   d. g=f^0.6

b) Mostrar as imagens e os histogramas resultantes de cara uma das transformações acima. **Comente os resultados encontrados**.

*Dica: passar para *double* antes de aplicar as transformações e voltar para *uint8* antes de mostrar a imagem processada e calcular seu histograma. Não usar *im2double* nem *im2uint8*, pois essas funções fazem uma re-escala na imagem.

7) Equalização de Histograma.

A equalização de histogramas no MatLab é implementada através da função:

$$g = histeq(f, nlev)$$

onde ***f*** é a imagem de entrada e ***nlev*** é o número de níveis de intensidades especificados para a imagem de saída.

E_8: Processar a imagem "`polem.bmp`" **utilizando**:

a) Equalização de histograma com 256 níveis de cinza
b) Equalização de histograma com 64 níveis de cinza
c) Comente sobre as diferenças visuais das figuras equalizadas com 256 e 64 níveis de cinza.
d) Alargamento de contraste (espalhe o histograma entre 0 e 255 sem saturar nenhum nível de cinza)
e) Comente sobre as diferenças visuais entre a equalização de histograma com 256 níveis de cinza e o alargamento de contraste
f) O que acontece se a imagem equalizada for equalizada novamente? Teste e comente os resultados.

8) Filtragem Espacial: Filtro Passa Baixa e Passa Alta.

E_9: Filtro Passa Alta.

Para a imagem "*moon.tif*", aplicar os filtros passa-alta (*w4*) e (*w8*).

```
f = imread('Moon.tif');
w4 = fspecial('laplacian',0);
w8 = [-1 -1 -1; -1 +8 -1; -1 -1 -1]
g4 = imfilter(f, w4);
g8 = imfilter(f, w8);
imshow(f)
figure, imshow(g4)
figure, imshow(g8)
g4i = imcomplement(g4);
g8i = imcomplement(g8);
figure, imshow(g4i)
figure, imshow(g8i)
```

a) **Comente todas as linhas desse código e explique o que o programa faz.**
b) **Comente sobre as diferenças visuais dos dois filtros utilizados**

E_10: Filtro Passa-Baixa

Para a imagem "*moon.tif*", aplicar agora o filtro da média ('average') com matrizes 3x3, 9x9 e 13x13. Utilizar as funções *fspecial* e *imfilter*.

a) **Comente sobre as diferenças visuais nos resultados entre os diferentes tamanhos das máscaras.**

9) Pixels da borda

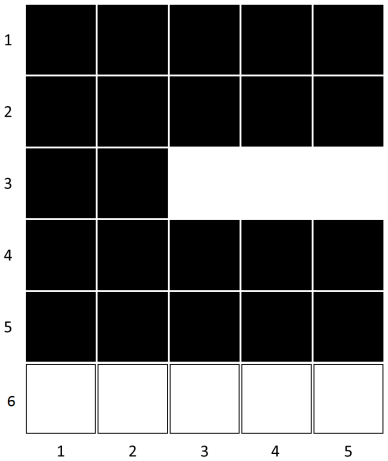


Imagem exemplo

E_11: Variação da borda na convolução

a) Construir a imagem 6x5 acima (classe *uint8*) e apresente em um tamanho suficiente para observação.

b) Filtre a imagem utilizando um filtro da média 3x3. Utilizar as funções *fspecial* e *imfilter*. Utilize as seguintes soluções para a borda:

a) *Padding* com zeros
b) Convolução simétrica
c) Convolução circular

c) Mostre as imagens resultantes e comente as diferenças encontradas

10) Detecção de Bordas – Função *edge*

Linhas Verticais, Horizontais ou em $\pm 45^{o}$ podem ser detectadas através da convolução da imagem com filtros derivativos (gradiente) de primeira ordem:

O Toolbox de Processamento de Imagens do MatLab possui uma função que realiza a detecção de bordas. A sintaxe desta função é:

[g, t] = edge(f, 'method', parameters)

Onde ***f*** é a Imagem de entrada, ***g*** é a Imagem de saída (arranjo lógico com 1 nas posições de bordas detectadas e 0 nas demais) e ***'method'*** corresponde ao tipo de detector utilizado, conforme mostra a Tabela a seguir. O valor de ***t*** é opcional e corresponde ao *Threshold* que a função utilizará para definir uma borda.

Tabela- Detectores de Borda da função *edge*

Edge Detector	Basic Properties
Sobel	Finds edges using the Sobel approximation to the derivatives shown in Fig. 10.5(b).
Prewitt	Finds edges using the Prewitt approximation to the derivatives shown in Fig. 10.5(c).
Roberts	Finds edges using the Roberts approximation to the derivatives shown in Fig. 10.5(d).
Laplacian of a Gaussian (LoG)	Finds edges by looking for zero crossings after filtering $f(x, y)$ with a Gaussian filter.
Zero crossings	Finds edges by looking for zero crossings after filtering $f(x, y)$ with a user-specified filter.
Canny	Finds edges by looking for local maxima of the gradient of $f(x, y)$. The gradient is calculated using the derivative of a Gaussian filter. The method uses two thresholds to detect strong and weak edges, and includes the weak edges in the output only if they are connected to strong edges. Therefore, this method is more likely to detect true weak edges.

Por exemplo, a sintaxe para o Detector de **Sobel** usando a função ***edge*** é:

[g, t] = edge(f, 'sobel', T, dir)

onde ***T*** é um *Threshold* especificado e ***dir*** dá a direção preferencial das bordas detectadas (***vertical, horizontal, ou both*** (default)). Se ***T*** é especificado então ***t = T***. Se T não for especificado, a função ***edge*** usa um *Threshold* automático e retorna seu valor em ***t***.

E_12: Detectores de borda

Para os exercícios abaixo, apresentem as imagens resultantes no <u>negativo</u> *(imcomplement)*

a) Detecte as bordas horizontais e verticais da imagem da *wirebond_mask.tif* utilizando os filtros de Roberts, Prewitt e Sobel. Utilize a função *edge*. Mostre as imagens resultantes com as bordas detectadas (pelos três métodos) e comente os resultados. Qual a diferença entre os três detectores?
b) Repita o exercício **a)** utilizando agora a função *imfilter* ao invés da função *edge*. Mostre as imagens resultantes com as bordas detectadas (pelos três métodos) e comente os resultados. Qual a diferença entre as duas funções do Matlab?
c) Utilize um dos filtros passa-alta do exercício **E_9** para detectar as bordas da imagem *wirebond_mask.tif*. Utilize a função *imfilter*. Comente sobre as diferenças entre os detectores de bordas utilizados no item a) com esse usado neste exercício.
d) Repita o exercício **a)** variando o valor do *Threshold* da função *edge*. Observar o que acontece com as bordas detectadas. Mostre as imagens com um novo valor de *Threshold* e comente os resultados encontrados.

Dica: As sintaxes da função ***edge*** para os Detectores de **Roberts** e de **Prewitt** são idênticas à sintaxe para o Detector de **Sobel.**