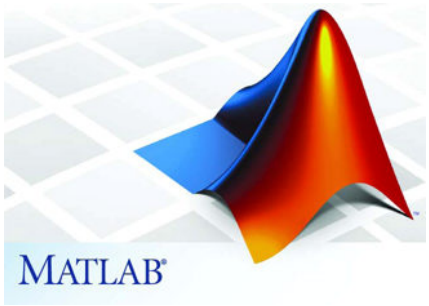
Faculdade de Ciências Aplicadas e Sociais de Petrolina – FACAPE

Processamento de Imagens com MATLAB



Prof. Sergio

Objetivos

- Dar uma visão geral dos fundamentos de processamento digital de imagens.
- Introdução a ferramentas analíticas atualmente usadas em processamento de imagens a fim de obter informações de imagens e processá-las.
- Desenvolver a habilidade de aplicar estas ferramentas em realce de imagens, segmentação e extração de características usando o ambiente de processamento de imagens do MATLAB.

Imagens no MATLAB

- MATLAB é um ambiente otimizado para operação com matrizes.
- Imagens são matrizes.
- Existem muitas funções embutidas no Toolbox Processamento de Imagens do MATLAB úteis ao tratamento de imagens.
- É muito fácil escrever suas próprias funções de processamento de imagens.

Imagens no MATLAB

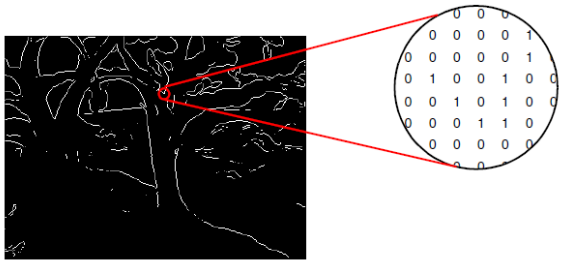
- O Image Processing Toolbox é uma coleção de funções MATLAB que ampliam a capacidade do ambiente de computação numérica do MATLAB.
- O toolbox oferece uma enorme quantidade de operações em processamento de imagens:
  - Operações geométricas
  - Operações ponto-a-ponto e orientada à vizinhança
  - Diversos tipos de filtragem
  - Transformadas
  - Análise e realce de imagens
  - Operações sobre imagens binárias, etc.

Imagens no MATLAB

- MATLAB pode importar/exportar diversos formatos de imagens:
  - BMP (Microsoft Windows Bitmap)
  - GIF (Graphics Interchange Files)
  - HDF (Hierarchical Data Format)
  - JPEG (Joint Photographic Experts Group)
  - PCX (Paintbrush)
  - PBM (Portable Bitmap)
  - PNG (Portable Network Graphics)
  - TIFF (Tagged Image File Format)
  - XWD (X Window Dump)
  - E diversos outros tipos de dados de imagens.
- Tipos de dados em MATLAB:
  - Double (64-bit double-precision floating point)
  - Single (32-bit single-precision floating point)
  - Int32 (32-bit signed integer)
  - Int16 (16-bit signed integer)
  - Int8 (8-bit signed integer)
  - Uint32 (32-bit unsigned integer)
  - Uint16 (16-bit unsigned integer)
  - Uint8 (8-bit unsigned integer)

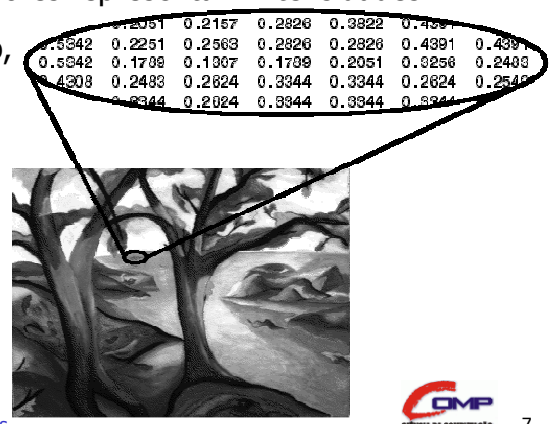
Imagens no MATLAB

- Imagens binárias
  - Em uma imagem binária, cada pixel assume apenas um de dois valores discretos possíveis: 0 (off) e 1 (on).



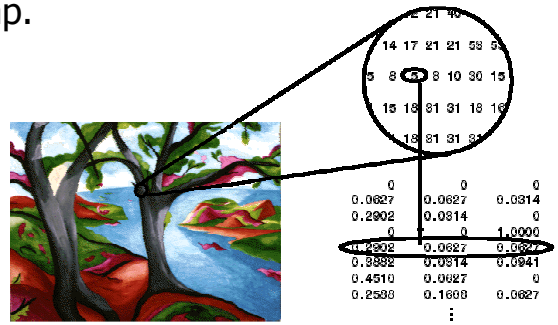
Imagens no MATLAB

- Imagens de intensidade (em escala de cinza)
  - Uma imagem de intensidade consiste apenas de uma matriz, I, cujos valores representam intensidades dentro de alguma faixa. Por exemplo, [0 1] (double) ou [0 255] (uint8).



Imagens no MATLAB

- Imagens indexadas
  - Uma imagem indexada consiste de uma matriz de dados, X, e uma matriz de mapa de cores, map.



Imagens no MATLAB

- Imagens RGB
  - Uma imagem RGB é armazenada em MATLAB como dados mxnx3 onde cada camada mxn define as componentes R (red), G (green) e B (blue) para cada pixel.

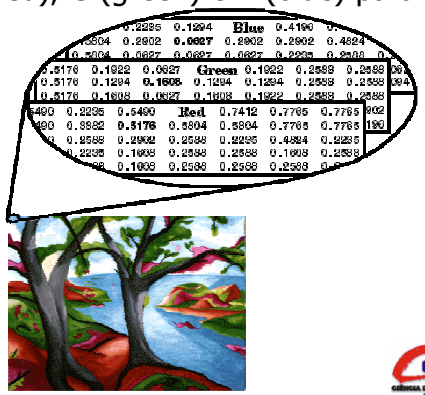


Image Viewer

- Comando **imshow**
  - Visualização de imagens.
  - Protótipo:
    - imshow(nome_do_arquivo)
  - Exemplo:
    - `imshow('eight.tif')`
  - Alternativas ao imshow
    - imagesc(I)
    - imtool(I)
    - image(I)

Image Viewer

- Comando **imzoom**
  - Zoom in ou zoom out em uma imagem.
  - Protótipo:
    - imzoom(fator)
  - Exemplo:
    - `>> imshow('eight.tif')`
    - `>> imzoom(2)`

E/S de Arquivos de Imagem

- Comando **imread**
  - Lê um arquivo de imagem.
  - Protótipo:
    - A = imread(filename)
  - Exemplo:

```
>> clear
>> I = imread('lena.gif');
>> I(1:5, 1:5)
ans =
    127   127   127   126   127
    127   127   127   126   127
    127   127   127   126   127
    127   127   127   126   127
    127   127   127   126   127
```

E/S de Arquivos de Imagem

- Leitura de Imagem BMP.
  - `[A, MAP] = imread('nome.bmp');`
- Para imagens coloridas:
  - `A = imread('nome.bmp');`
    - Gera uma matriz mxnx3 (onde 3 é a quantidade de planos).
  - `R = A(:, :, 1); % Matriz de tons vermelhos`
  - `G = A(:, :, 2); % Matriz de tons verdes`
  - `B = A(:, :, 3); % Matriz de tons azuis`

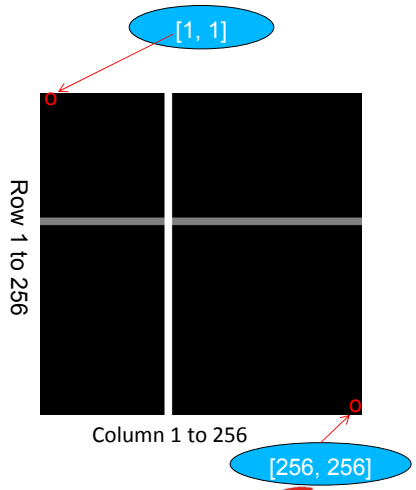
E/S de Arquivos de Imagem

- Comando **imwrite**
  - Escreve uma imagem para um arquivo gráfico.
  - Protótipo:
    - imwrite(A, filename, FMT)
      - FMT = formato
  - Exemplo:
    - imwrite(A, 'eight', 'tif')

Imagens e Matrizes

- Como construir uma matriz (ou imagem) de intensidades?

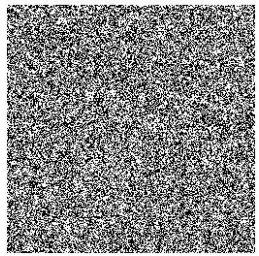
```
row = 256;
col = 256;
img = zeros(row, col);
img(100:105, :) = 0.5;
img(:, 100:105) = 1;
figure;
imshow(img);
```



Imagens e Matrizes

- Imagem binária

```
row = 256;
col = 256;
img = rand(row, col);
img = round(img);
figure;
imshow(img);
```



Dimensões de uma imagem

- Comando **size**
  - `[lin, col] = size(im);`
    - Para imagens em tons de cinza, e preto e branco.
  - `[lin, col, plan] = size(im);`
    - Para imagens coloridas (plan = 3, indicando que é uma estrutura com 3 matrizes). Para imagens em tons de cinza ou preto-e-branco, se usado esse parâmetro, ele terá valor 1.

Conversão entre tipos

- Comando **im2bw**
  - Converte uma imagem para preto-e-branco.
  - Protótipo:
    - BW = im2bw(X, MAP, level)
      - X = imagem original
      - MAP = paleta de cores da imagem original
      - level = valor de corte (threshold): $0 \leq level \leq 1$
  - Exemplo 1:

```
[A,MAP] = tiffread('eight.tif');
bw = im2bw(A,MAP,0.4);
imshow(bw);
```


Conversão entre tipos

- Exemplo 2:

```
level = graythresh(I3);
bw = im2bw(I3, level);
figure, imshow(bw)
```

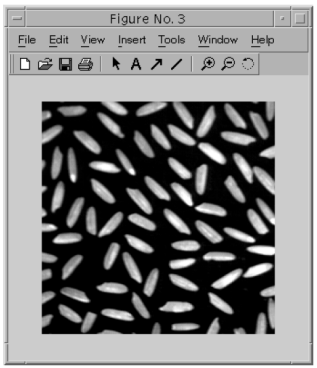


Imagem original

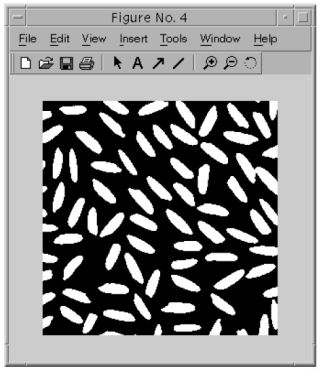


Imagem binarizada

Conversão entre tipos

- Comando **rgb2gray**
  - Converte uma imagem RGB para uma imagem em tons de cinza.
  - Protótipo:
    - I = rgb2gray(RGB)
      - RGB = imagem original true color
  - Exemplo:

```
A = imread('flowers.tif');
I = rgb2gray(A);
imshow(I);
```

Conversão entre tipos

- Comando **rgb2ind**
  - Converte uma imagem RGB para uma imagem indexada.
  - Protótipo:
    - [X, NEWMAP] = rgb2ind(RGB, n)
      - NEWMAP = paleta de cores final
      - n = nº de cores

Conversão entre tipos - Resumo

- dither
- gray2ind
- grayslice
- im2bw
- ind2gray
- ind2rgb
- mat2gray
- rgb2gray
- rgb2ind

Tipos de imagens

- Alguns comandos que verificam os tipos de certas imagens:

isbw

verdadeiro para imagens B&W.

isgray

verdadeiro para imagens em tons de cinza.

isind

verdadeiro para imagens indexadas.

Bordas de imagem

- Comando **edge**
  - Realiza a extração de bordas de uma imagem.
  - Protótipo:
    - BW = edge(A, 'method');
      - method: sobel, roberts, prewitt, log, zerocross
  - Exemplo:

```
>> A = imread('rice', 'tif');
>> BW = edge(A, 'sobel');
>> imshow(BW);
```

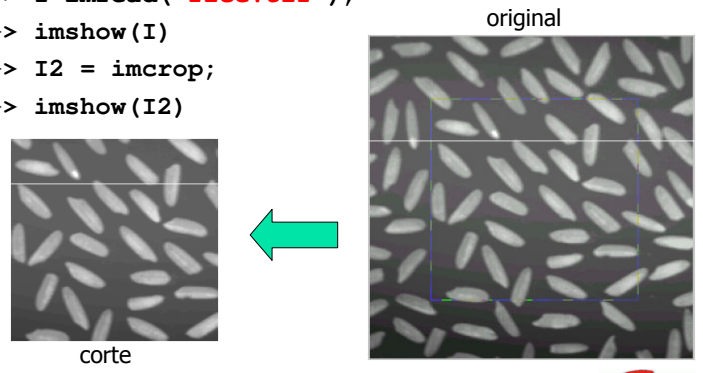
Operações geométricas

- Comando **imcrop**
  - usado para definir uma subimagem retangular da atual imagem.
  - Exemplo 1:
    - imshow('eight.tif')
    - B = imcrop;
      - Seleciona uma área com o mouse.

Operações geométricas

- Comando **imcrop**
  - Exemplo 2:

```
>> I=imread('rice.tif');
>> imshow(I)
>> I2 = imcrop;
>> imshow(I2)
```



Operações geométricas

- Comando **imresize**
  - Altera o tamanho de uma imagem.
  - Protótipo:
    - B = imresize(A, M, 'method')
      - 'method' =
        - nearest = vizinho mais próximo
        - bilinear = interpolação bilinear
        - bicubic = interpolação bicúbica
    - Retorna uma matriz que é M vezes maior (ou menor) que A.
  - Exemplo 1:

```
>> A = imread('eight', 'tif');
>> B = imresize(A, 0.5, 'nearest');
>> imshow(B)
```

  - Exemplo 2:

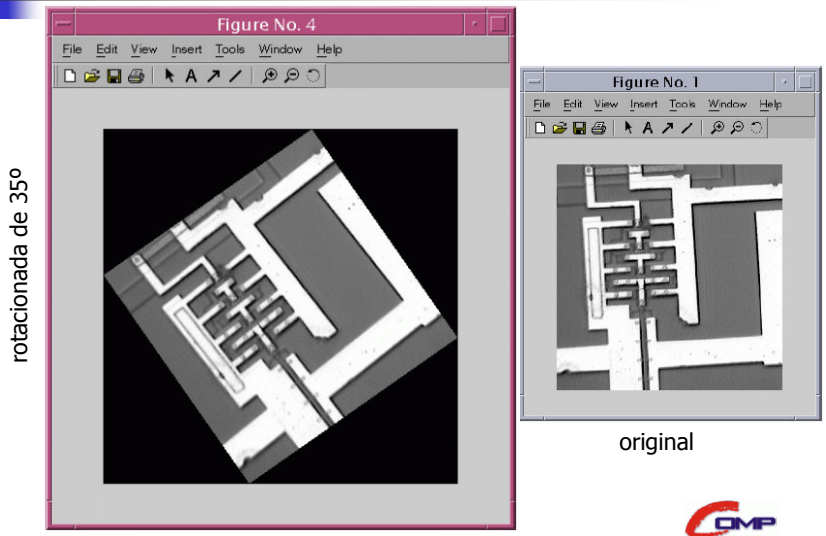
```
>> Ismall = imresize(A, [100 100], 'bilinear');
```


Operações geométricas

- Comando **imrotate**
  - Rotaciona uma imagem.
  - Protótipo:
    - B = imrotate(A, Ângulo, 'method');
      - Method = nearest, bilinear ou bicubic
  - Exemplo:

```
>> A = imread('ic.tif');
>> B = imrotate(A, 35, 'bilinear');
>> imshow(A), figure, imshow(B)
```

Operações geométricas



original

Operações aritméticas

- imabsdiff
- imadd
- imcomplement
- imdivide
- imlincomb
- immultiply
- imsubtract

Operações aritméticas

Comando **iadd**: adicionando imagens

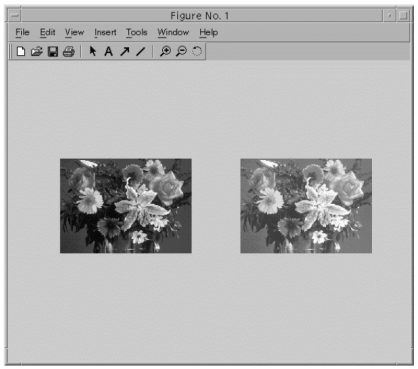
```
I = imread('rice.tif');
J = imread('cameraman.tif');
K = imadd(I, J);
imshow(K)
```

imagem I imagem J imagem K

Operações aritméticas

Clareando imagens

```
RGB = imread('flowers.tif');
RGB2 = imadd(RGB, 50);
subplot(1, 2, 1); imshow(RGB);
subplot(1, 2, 2); imshow(RGB2);
```

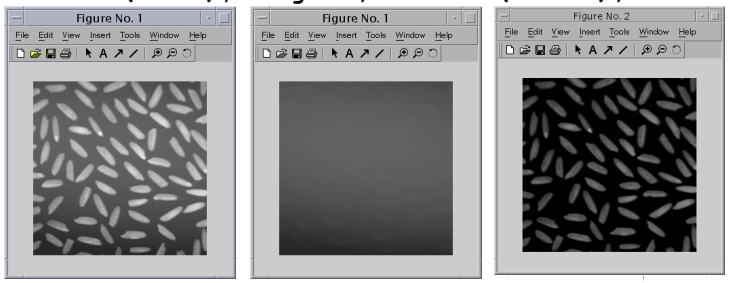


Operações aritméticas

Comando **imsubtract**: subtraindo imagens.

Fundo de uma cena.

```
rice = imread('rice.tif');
background = imopen(rice, strel('disk', 15));
rice2 = imsubtract(rice, background);
imshow(rice), figure, imshow(rice2);
```



Cálculos estatísticos

- Comando **imhist**
  - Calcula e mostra o histograma de uma imagem.
    - Protótipo:
      - imhist(A): histograma de 256 cores
      - imhist(A,N): histograma de N cores
    - Exemplo:

```
>> figure, imhist(I)
```


Cálculos estatísticos

- Comando **mean2**
  - Calcula a média de uma matriz bidimensional.
  - Protótipo:
    - M = mean2(A)
- Comando **std2**
  - Calcula o desvio padrão bidimensional.
  - Protótipo:
    - D = std2(A)

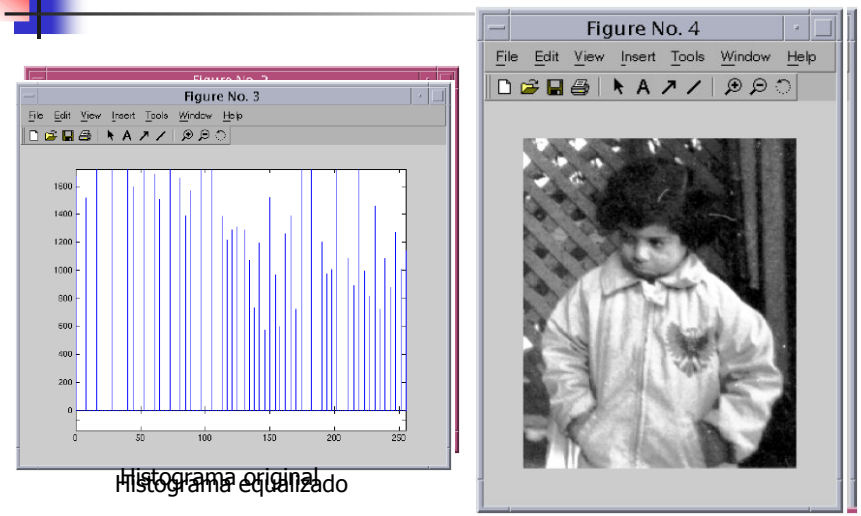
Realce de imagem

- Comando **histeq**
  - Calcula a equalização de histograma.
  - Exemplo:

```
>> I2 = histeq(I);
>> figure, imshow(I2)
>> figure, imhist(I2)
```

- Comando **imadjust**
  - Realiza a especificação de histograma (atribui o histograma de uma imagem A a outra imagem B).

Realce de imagem



Histograma original

Histograma equalizado

Imagem original

Imagem equalizada

Filtragem de imagem

- Comando **filter2**
  - Corresponde a um filtro digital bidimensional.
  - Protótipo:
    - Filter2(B,X)
      - Filtra a imagem X usando o filtro FIR definido pela matriz B.
  - Exemplo:

```
>> I = imread('rice.tif');
>> imshow(I);
```


Filtragem de imagem

continuação...

```
>> h = [1 2 1; 0 0 0; -1 -2 -1];
>> I2 = filter2(h,I);
>> imshow(I2),colorbar
```

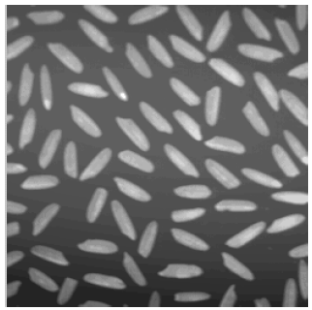


Imagem original

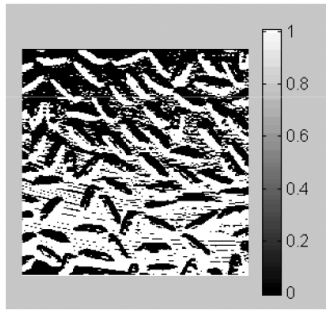


Imagem filtrada

Filtragem de imagem

- Comando **fspecial**
  - Cria um filtro 2D de um tipo específico.
    - gaussian
    - sobel
    - prewitt
    - laplacian
    - log
    - average
    - unsharp

Exemplo 1:

```
>> h=fspecial('laplacian', 5);
>> I2=uint8(round(filter2(h,I)));
>> imshow(I2)
```

Exemplo 2:

```
>> h=fspecial('sobel');
>> I2=filter2(h,I));
>> imshow(I2)
```

Filtragem de imagem

- Filtragem linear
  - conv2
    convolução bidimensional
  - convmtx2
    matriz de convolução bidimensional
  - convn
    convolução n-dimensional

Função MATLAB

- Uma função MATLAB tem parâmetros de entrada e saída.
- MATLAB pode retornar mais de uma variável no final de uma função.
- Variáveis no escopo de uma função MATLAB saem do escopo e são eliminadas quando a função MATLAB deixa de existir.
- Protótipo:
  - function [output] = function_name(input_arguments)
- Exemplo:

```
function [output] = square(input)
  output = input*input;
end
```


Função MATLAB

```
function im2 = teste(nome)
   im = imread(nome);
   [lin, col] = size(im);
   for i = 1:lin
        for j=1:col
              .....
        end
   end
   imwrite (im2, 'saida.bmp', 'bmp');
end
```


Exercício 1

1. Carregue o arquivo tree.tif no MATLAB. Que tipo de imagem é essa? Você pode comentar sobre isso antes de obter informações do sistema?
2. Mostre a imagem carregada.
3. Converta a imagem para uma imagem de intensidade (níveis de cinza).
4. Agora converta-o para uma imagem binária (preto e branco).

- Mais:

1. Use subplots para mostrar todas as três imagens.
2. Você encontra o ovo de páscoa na árvore?

Solução

```
>> im_info = imfinfo('trees.tif');
>> im_info(1).ColorType

>> [I,map] = imread('trees.tif');
>> subplot(2,2,1), subimage(I,map)

>> I_gray = ind2gray(I,map);
>> subplot(2,2,2), subimage(I_gray)

>> I_bw = im2bw(I,map,0.4);
>> subplot(2,2,3), subimage(I_bw)

% Easter egg
>> figure
>> [I2,map] = imread('trees.tif',2);
>> imshow(I2,map)
```