

Imagens Digitais

Visão Computacional em Robótica (BLU3040)

Prof. Marcos Matsuo (marcos.matsuo@ufsc.br)

Universidade Federal de Santa Catarina - UFSC

1 Imagens Digitais

- Imagem binária
- Imagem em escala de cinza
- Imagem colorida

2 Espaços de cores

- Espaço de cor RGB
- Espaço de HSV

3 Conversão de Imagens

- Conversão entre tipos de dados
- Conversão entre tipos de imagens

1 Imagens Digitais

- Imagem binária
- Imagem em escala de cinza
- Imagem colorida

2 Espaços de cores

- Espaço de cor RGB
- Espaço de HSV

3 Conversão de Imagens

- Conversão entre tipos de dados
- Conversão entre tipos de imagens

Imagens Digitais

- Uma **imagem digital** pode ser definida como uma representação de uma imagem através de **valores discretos**.
- Tais valores podem representar a **intensidade de uma cor**, **posição de um elemento gráfico**, etc.
- Tipos de imagens digitais:
 - Imagem **vetorial**.
 - Imagem **raster**.



Imagem Vetorial

Imagens Digitais

- Definida em termos de **elementos geométricos** (pontos, linhas, polígonos, etc).
- Cada elemento geométrico possui um conjunto de **atributos** relacionados com posição, cor, espessura, preenchimento, etc.
- Geradas através de *softwares* de criação e edição de imagens vetoriais (p.e., *CorelDraw* e *Inkscape*).

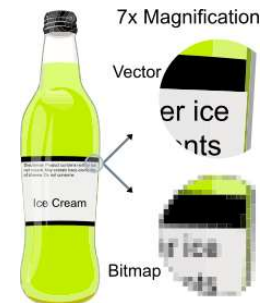


Imagem Raster

Imagens Digitais

- Possui um conjunto finito de valores discretos, chamados de **pixels** (*picture elements*).
- Os pixels são organizados em um formato matricial, com um número definido de linhas e colunas.
- Usualmente geradas por equipamentos como **câmeras digitais** e **scanners** ou através de *softwares* como *Photoshop* e *Gimp*.

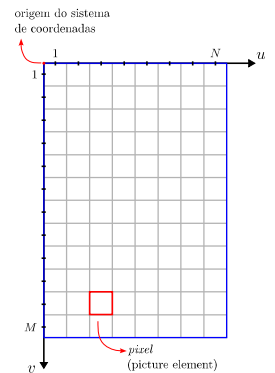


Imagem Raster

Imagens Digitais

Uma **imagem digital raster** com M linhas e N colunas pode ser representada como uma **matriz** de dimensão $M \times N$.

$$\mathbf{I} = \begin{bmatrix} I(1,1) & I(1,2) & \cdots & I(1,N) \\ I(2,1) & I(2,2) & \cdots & I(2,N) \\ \vdots & \vdots & \ddots & \vdots \\ I(M,1) & I(M,2) & \cdots & I(M,N) \end{bmatrix}$$

onde \mathbf{I} denota uma imagem/matriz e $I(v, u)$ representa o pixel/elemento da linha v e coluna u de \mathbf{I} .

Imagem Raster

Imagens Digitais

Em particular, o **valor** que cada pixel/elemento $I(v, u)$ pode assumir depende:

- do tipo de imagem (**binária**, escala de cinza ou colorida).
- do tipo de dado utilizado. Por exemplo,
 - logical $\rightarrow \{0, 1\}$ (0 ou 1)
 - uint8 $\rightarrow [0, 255]$ (números inteiros de 0 até 255)
 - double $\rightarrow [0.0, 1.0]$ (números reais de 0.0 até 1.0)

1 Imagens Digitais

- Imagem binária
- Imagem em escala de cinza
- Imagem colorida

2 Espaços de cores

- Espaço de cor RGB
- Espaço de HSV

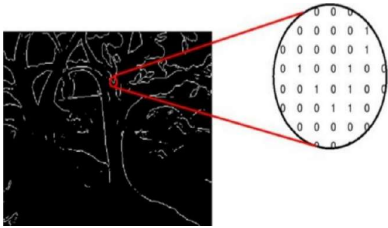
3 Conversão de Imagens

- Conversão entre tipos de dados
- Conversão entre tipos de imagens

Imagem Binária

Imagem Raster

- Também conhecida como **imagem preto e branca**.
- Cada **pixel** pode assumir somente um dos seguintes valores:
 - 0 (preto) ou 1 (branco) → dados do tipo `logical`
 - 0 (preto) ou 255 (branco) → dados do tipo `uint8`
 - 0.0 (preto) ou 1.0 (branco) → dados do tipo `double`



Exemplo 1: criação de imagem binária no Matlab

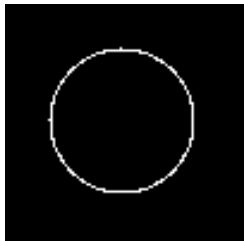
Solução 2

```
1 M = 200;
2 N = 400;
3
4
5 % cria matriz de M linhas e N colunas com elementos do
6 % tipo logical
7 I = zeros(M,N,'logical');
8
9 % comando para acessar todos os elementos da região desejada
10 % (acessa todos os elementos das linhas 51 até 150 e colunas
11 % 21 até 380)
12 I(51:150,21:380) = 1;
13
14 % apresentação da imagem binária gerada
15 figure;
16 imshow(I);
```

Após execução do código acima digite na Janela de Comandos `whos I`, para verificar quantos bytes são utilizados para armazenar a variável `I`.

Exemplo 2: criação de imagem binária no Matlab

Enunciado: crie no Matlab uma imagem binária de 100 linhas e 100 colunas de fundo preto, contendo uma circunferência branca com raio $r = 30$ pixels centralizado em $v_0 = 50$ e $u_0 = 50$. Um exemplo de resultado esperado é mostrado abaixo.



Exemplo 1: criação de imagem binária no Matlab

Enunciado: crie no Matlab uma imagem binária de 200 linhas e 400 colunas de fundo preto, contendo um retângulo branco (centralizado na imagem) de 100 linhas e 360 colunas. Um exemplo de resultado esperado é mostrado abaixo.



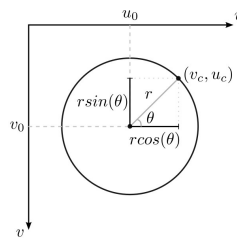
Exemplo 1: criação de imagem binária no Matlab

Solução 3

```
1 M = 200;
2 N = 400;
3
4
5 % cria matriz de M linhas e N colunas com elementos do
6 % tipo uint8 (inteiro sem sinal de 8 bits)
7 I = zeros(M,N,'uint8');
8
9 % atribui valor 255 (branco) para os elementos da
10 % região retangular desejada
11 I(51:150,21:380) = 255;
12
13 % apresentação da imagem binária gerada
14 figure;
15 imshow(I);
```

Após execução do código acima digite na Janela de Comandos `whos I`, para verificar quantos bytes são utilizados para armazenar a variável `I`.

Exemplo 2: criação de imagem binária no Matlab



Conforme ilustrado na figura acima, as coordenadas (v_c, u_c) dos pixels pertencentes a **circunferência** de raio r com centro em (v_0, u_0) podem ser calculadas como

$$v_c = v_0 + r \sin(\theta)$$

$$u_c = u_0 + r \cos(\theta)$$

Exemplo 1: criação de imagem binária no Matlab

Solução 1

```
1
2 M = 200;
3 N = 400;
4
5 % cria matriz de M linhas e N colunas com elementos do
6 % tipo logical
7 I = zeros(M,N,'logical');
8
9 % laços de repetição aninhados para acessar todos os
10 % elementos da região retangular desejada
11 for v = 51:150
12     for u = 21:380
13         I(v,u) = 1;
14     end
15 end
16
17 % apresentação da imagem binária gerada
18 figure;
19 imshow(I);
```

Exemplo 1: criação de imagem binária no Matlab

Solução 4

```
1 M = 200;
2 N = 400;
3
4
5 % cria matriz de M linhas e N colunas com elementos do
6 % tipo double (números reais)
7 I = zeros(M,N,'double');
8
9 % atribui valor 1 (branco) para os elementos da
10 % região retangular desejada
11 I(51:150,21:380) = 1;
12
13 % apresentação da imagem binária gerada
14 figure;
15 imshow(I);
```

Após execução do código acima digite na Janela de Comandos `whos I`, para verificar quantos bytes são utilizados para armazenar a variável `I`.

Exemplo 2: criação de imagem binária no Matlab

```
1 M = 100;
2 N = 100;
3
4 I = zeros(M,N,'uint8');
5
6 % parâmetros da circunferência
7 v0 = 50;
8 u0 = 50;
9 r = 30;
10
11 for theta = 0:360
12     vc = ceil(v0 + r*sind(theta));
13     uc = ceil(u0 + r*cosd(theta));
14     I(vc,uc) = 255;
15 end
16
17 figure; imshow(I);
```

A função `ceil(x)` recebe como argumento de entrada um número real x e retorna o inteiro mais próximo maior ou igual a x .

Exercício 1: criação de imagem binária no Matlab

Enunciado: crie no Matlab uma imagem binária de 100 linhas e 100 colunas de fundo preto, contendo um círculo branco com raio $R = 30$ pixels centralizado em $v_0 = 50$ e $u_0 = 50$. Um exemplo de resultado esperado é mostrado abaixo.



Navigation icons

1 Imagens Digitais

- Imagem binária
- Imagem em escala de cinza
- Imagem colorida

2 Espaços de cores

- Espaço de cor RGB
- Espaço de HSV

3 Conversão de Imagens

- Conversão entre tipos de dados
- Conversão entre tipos de imagens

Navigation icons

Exemplo 4: criação de imagem em escala de cinza



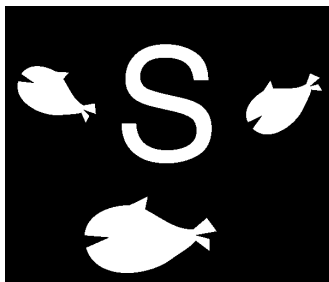
Assumindo que $y(u)$ denota o nível de cinza em função da coordenada de coluna u , tem-se

$$y(1) = 0 \quad (1) \quad y(N) = 1 \quad (2)$$

Navigation icons

Exemplo 3: leitura de imagem no Matlab

Enunciado: carregue no Matlab a imagem binária sharks.png (disponível no Moodle e apresentada abaixo). Apresente a imagem utilizando a função `imshow(.)`. Na sequência, utilize a ferramenta `imtool(.)` para analisar a imagem.

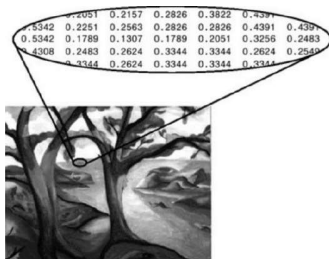


Navigation icons

Imagem em Escala de Cinza

Imagem Raster

- Cada **pixel** pode assumir valores nos seguintes intervalos:
 - 0 (preto) até 255 (branco) → dados do tipo `uint8`
 - 0.0 (preto) até 1.0 (branco) → dados do tipo `double`



Navigation icons

Exemplo 4: criação de imagem em escala de cinza

Assumindo variação linear de $y(u)$ em relação à u , tem-se

$$y(u) = \alpha u + \beta \quad (3)$$

Utilizando (1) e (2) em (3), obtém-se

$$y(u) = \frac{1}{N-1}u - \frac{1}{N-1}. \quad (4)$$

Navigation icons

Exemplo 3: leitura de imagem no Matlab

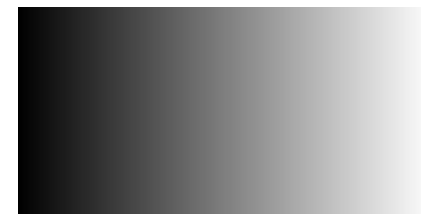
Enunciado: carregue no Matlab a imagem binária sharks.png (disponível no Moodle e apresentada abaixo). Apresente a imagem utilizando a função `imshow(.)`. Na sequência, utilize a ferramenta `imtool(.)` para analisar a imagem.

```
1 % lê arquivo de imagem
2 I = imread('sharks.png');
3
4 % apresenta imagem I
5 figure;
6 imshow(I);
7
8 % carrega ferramenta imtool para analisar a imagem I
9 imtool(I);
```

Navigation icons

Exemplo 4: criação de imagem em escala de cinza

Enunciado: crie no Matlab uma imagem em escala de cinza de $M = 200$ linhas e $N = 400$ colunas, com efeito de gradiente de nível de cinza similar ao apresentado na figura abaixo. Observe que o nível de cinza das colunas da imagem variam de preto para branco a medida que a coordenada de coluna u varia de 1 para N .



Navigation icons

Exemplo 4: criação de imagem em escala de cinza

Solução 1

```
1 M = 200;
2 N = 400;
3
4 I = zeros(M,N,'double');
5
6 for u = 1:N
7     % nível de cinza para a coluna u
8     y = 1/(N-1) * u - 1/(N-1);
9
10    % atribui o nível de cinza para todos os elementos da coluna u
11    I(:,u) = y;
12 end
13
14 figure;
15 imshow(I);
```

Navigation icons

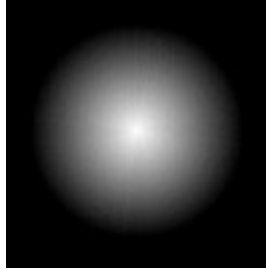
Exemplo 4: criação de imagem em escala de cinza

Solução 2

```
1 M = 200;
2 N = 400;
3
4 I = zeros(M,N,'uint8');
5
6 for u = 1:N
7     % nível de cinza para a coluna u
8     y = 1/(N-1) * u - 1/(N-1);
9
10    % atribui o nível de cinza para todos os elementos da coluna u
11    I(:,u) = round(y*255);
12 end
13
14 figure;
15 imshow(I);
```

Exercício 2: criação de imagem em escala de cinza

Enunciado: crie no Matlab uma imagem em escala de cinza de 100 linhas e 100 colunas, contendo um efeito de gradiente radial, similar a imagem apresentada abaixo.



Exemplo 5: leitura de imagem no Matlab

Enunciado: carregue no Matlab a imagem em escala de cinza castle.jpg (disponível no Moodle e apresentada abaixo). Apresente a imagem utilizando a função `imshow(.)`. Na sequência, utilize a ferramenta `imtool(.)` para analisar a imagem.



Exemplo 5: leitura de imagem no Matlab

Enunciado: carregue no Matlab a imagem em escala de cinza castle.jpg (disponível no Moodle e apresentada abaixo). Apresente a imagem utilizando a função `imshow(.)`. Na sequência, utilize a ferramenta `imtool(.)` para analisar a imagem.

```
1 % lê arquivo de imagem
2 I = imread('castle.jpg');
3
4 % apresenta imagem I
5 figure;
6 imshow(I);
7
8 % carrega ferramenta imtool para analisar a imagem I
9 imtool(I);
```

Imagens Digitais

- 1 Imagens Digitais
 - Imagem binária
 - Imagem em escala de cinza
 - Imagem colorida

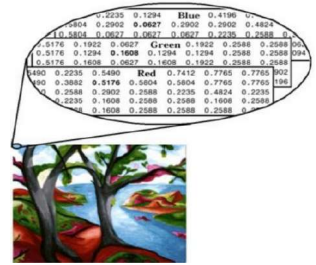
- 2 Espaços de cores
 - Espaço de cor RGB
 - Espaço de HSV

- 3 Conversão de Imagens
 - Conversão entre tipos de dados
 - Conversão entre tipos de imagens

Imagem Colorida

Imagem Raster

- Formada por três camadas/matrices.
- Cada **pixel** é formado por 3 valores, que em conjunto representam uma cor.
- Os significados dos valores que formam o pixel, dependem do **espaço de cor** utilizado.



Imagens Digitais

- 1 Imagens Digitais
 - Imagem binária
 - Imagem em escala de cinza
 - Imagem colorida

- 2 Espaços de cores
 - Espaço de cor RGB
 - Espaço de HSV

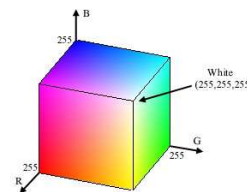
- 3 Conversão de Imagens
 - Conversão entre tipos de dados
 - Conversão entre tipos de imagens

Espaço RGB

Espaços de cores

Cada cor é representada pela combinação de diferentes níveis de **vermelho** (R), **verde** (G) e **azul** (B).

- vermelho:** [1.0, 0.0, 0.0] ou [255, 0, 0]
- verde:** [0.0, 1.0, 0.0] ou [0, 255, 0]
- azul:** [0.0, 0.0, 1.0] ou [0, 0, 255]



Exemplo 6: criação de imagem colorida

Enunciado: crie no Matlab uma imagem colorida de 200 linhas e 400 colunas de fundo preto, contendo um retângulo amarelo (centralizado na imagem) de 100 linhas e 360 colunas. Um exemplo de resultado esperado é mostrado abaixo.



Exemplo 6: criação de imagem colorida

Exemplo 6 criação de imagem colorida

Solução 1

```
1 M = 200;
2 N = 400;
3
4 % cria matriz com 3 camadas
5 I = zeros(M,N,3,'uint8');
6
7 % camada vermelha
8 I(51:150,21:380,1) = 255;
9
10 % camada verde
11 I(51:150,21:380,2) = 255;
12
13 figure;
14 imshow(I);
```

Solução 2

```
1 M = 200;
2 N = 400;
3
4 % cria matriz com 3 camadas
5 I = zeros(M,N,3,'double');
6
7 % camada vermelha
8 I(51:150,21:380,1) = 1;
9
10 % camada verde
11 I(51:150,21:380,2) = 1;
12
13 figure;
14 imshow(I);
```

1 Imagens Digitais

- Imagem binária
- Imagem em escala de cinza
- Imagem colorida

2 Espaços de cores

- Espaço de cor RGB
- Espaço de HSV

3 Conversão de Imagens

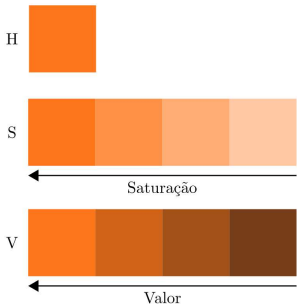
- Conversão entre tipos de dados
- Conversão entre tipos de imagens

Espaço HSV

Espaços de cores

No espaço de cor HSV, cada cor é representada pelos seguintes parâmetros:

- **Hue/Matiz:** parâmetro relacionado com a cor.
- **Saturação:** parâmetro relacionada com a pureza da cor (↑ Saturação ↑ Cor pura)
- **Valor:** parâmetro relacionado com o brilho da cor (↑ Valor ↑ Brilho)

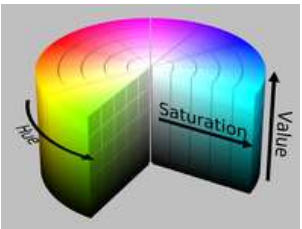


Espaço HSV

Espaços de cores

Intervalo dos valores dos parâmetros do espaço de cor HSV:

- **Hue/Matiz:** 0° até 360°.
- **Saturação:**
1 → alta saturação
0 → baixa saturação.
- **Valor:**
1 → alto brilho
0 → baixo brilho.

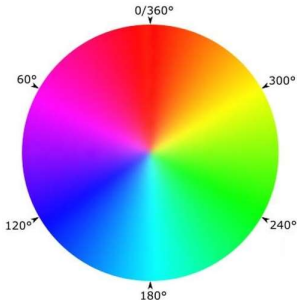


Espaço HSV

Espaços de cores

Intervalo dos valores dos parâmetros do espaço de cor HSV:

- **Hue/Matiz:** 0° até 360°.
- **Saturação:**
1 → alta saturação
0 → baixa saturação.
- **Valor:**
1 → alto brilho
0 → baixo brilho.



Conversão entre tipos de dados

Conversão de Imagens

1 Imagens Digitais

- Imagem binária
- Imagem em escala de cinza
- Imagem colorida

2 Espaços de cores

- Espaço de cor RGB
- Espaço de HSV

3 Conversão de Imagens

- Conversão entre tipos de dados
- Conversão entre tipos de imagens

Conversão de imagem com dados do tipo uint8 para double

```
1 % I1: imagem com dados do tipo uint8
2 % I2: imagem com dados do tipo double
3 I2 = double(I1)/255;
4
5 % Segunda opção
6 I2 = im2double(I1);
```

Conversão de imagem com dados do tipo double para uint8

```
1 % I1: imagem com dados do tipo double
2 % I2: imagem com dados do tipo uint8
3 I2 = uint8(I1*255);
```

Conversão entre tipos de dados

Conversão de Imagens

1 Imagens Digitais

- Imagem binária
- Imagem em escala de cinza
- Imagem colorida

2 Espaços de cores

- Espaço de cor RGB
- Espaço de HSV

3 Conversão de Imagens

- Conversão entre tipos de dados
- Conversão entre tipos de imagens

Conversão entre tipos de imagens

Conversão de Imagens

Em diversas aplicações é necessário realizar a conversão entre tipos de imagens. As conversões mais comuns são listadas abaixo.

- Colorida RGB → Colorida HSV;
- Colorida HSV → Colorida RGB;
- Colorida RGB → Escala de cinza;
- Escala de cinza → Binária;
- Colorida → Binária.

RGB → RGB e RGB → RGB

Conversão entre tipos de imagens

Conversão de imagem colorida RGB para HSV

```
1 % I1: imagem colorida do tipo RGB
2 % I2: imagem colorida do tipo HSV
3 I2 = rgb2hsv(I1);
```

Conversão de imagem colorida HSV para RGB

```
1 % I1: imagem colorida do tipo HSV
2 % I2: imagem colorida do tipo RGB
3 I2 = hsv2rgb(I1);
```

RGB → Escala de cinza

Conversão entre tipos de imagens

Conversão de imagem colorida RGB para escala de cinza

```
1 % I1: imagem colorida do tipo RGB
2 % I2: imagem em escala de cinza
3 R = I1(:, :, 1);
4 G = I1(:, :, 2);
5 B = I1(:, :, 3);
6
7 % Método da média ponderada (coeficientes obtidos da norma
8 % Rec.ITU-R BT.601-7)
9 I2 = 0.2989*R + 0.5870*G + 0.1140*B;
10
11 % Segunda opção
12 I2 = rgb2gray(I1);
```

Escala de cinza → Binária

Conversão entre tipos de imagens

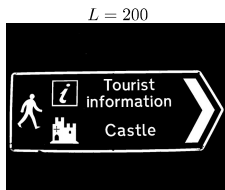
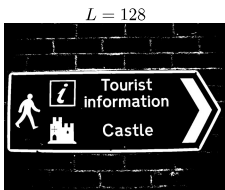
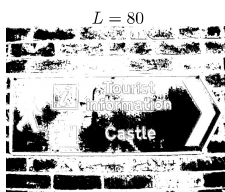
Em diversas aplicações é necessário converter uma imagem em escala de cinza para **binária**.

Típico **problema de classificação** onde deseja-se verificar se cada pixel da imagem em escala de cinza pertence a uma região de interesse ou não.



Escala de cinza → Binária

Conversão entre tipos de imagens



Escala de cinza → Binária

Conversão entre tipos de imagens

Para realizar a conversão de imagem em **escala de cinza** para **binária** pode-se utilizar a abordagem conhecida como **limiarização**, onde

$$O(v, u) = \begin{cases} 1, & \text{se } I(v, u) \geq L \\ 0, & \text{se } I(v, u) < L \end{cases}$$

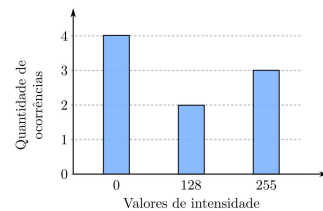
onde $O(v, u)$ e $I(v, u)$ denotam, respectivamente, os pixels com coordenada (v, u) das imagens de saída **O** e entrada **I**, e L é um **limiar** de comparação.

Escala de cinza → Binária

Conversão entre tipos de imagens

- Como definir o valor do **limiar** de comparação L ?
- Uma das formas é analisar o **histograma** da imagem.
- Um histograma de imagem é um gráfico de barras que apresentada a **quantidade de ocorrências** (frequência) dos **valores de intensidade** em uma imagem.

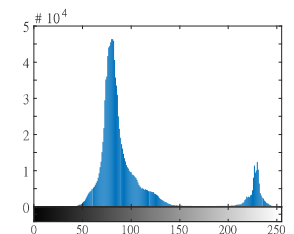
Imagem			
0	128	255	
128	0	0	
255	0	255	



Escala de cinza → Binária

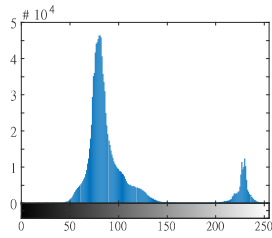
Conversão entre tipos de imagens

```
1 I = imread('castle.jpg');
2
3 figure; imshow(I);
4 figure; imhist(I);
```



Escala de cinza → Binária

Conversão entre tipos de imagens

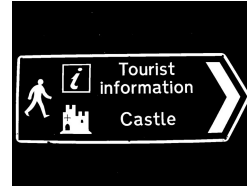
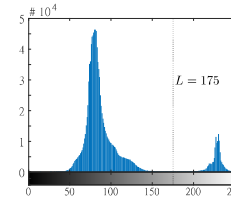


Observa-se que para a imagem em escala de cinza acima, a intensidade dos pixels são distribuídas em **duas** regiões bem definidas.

Há uma grande quantidade de pixels com valores menores (pixels mais **escuros**) e uma outra região menor com pixels com valores maiores (mais **claros**).

Escala de cinza → Binária

Conversão entre tipos de imagens



Escala de cinza → Binária

Conversão entre tipos de imagens

```
1 I = imread('castle.jpg');
2
3 L = graythresh(I);
4 I2 = (I >= L);
5
6 figure; imshow(I);
7 figure; imshow(I2);
```



Imagem colorida → Binária

Conversão entre tipos de imagens

Geração das imagens binárias para cada camada de cor

$$M_r(v, u) = \begin{cases} 1, & \text{se } r_f(1 - \delta) \leq \mathbf{R}(v, u) \leq r_f(1 + \delta) \\ 0, & \text{caso contrário} \end{cases}$$

$$M_g(v, u) = \begin{cases} 1, & \text{se } g_f(1 - \delta) \leq \mathbf{G}(v, u) \leq g_f(1 + \delta) \\ 0, & \text{caso contrário} \end{cases}$$

e

$$M_b(v, u) = \begin{cases} 1, & \text{se } b_f(1 - \delta) \leq \mathbf{B}(v, u) \leq b_f(1 + \delta) \\ 0, & \text{caso contrário} \end{cases}$$

Imagem binária final

$$M = M_r \& M_g \& M_b$$

Imagem colorida → Binária

Conversão entre tipos de imagens

Em algumas aplicações deseja-se **classificar** os pixels de uma imagem colorida como pertencentes (ou não) a uma dada região de interesse.

A conversão de imagem colorida para binária também é conhecida como **segmentação baseada em cor**.

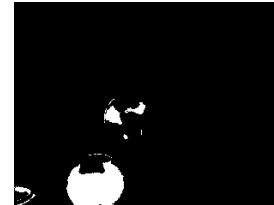
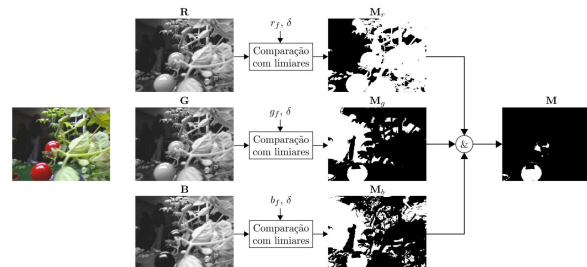


Imagem colorida → Binária

Conversão entre tipos de imagens

Diagrama das etapas de processamento



Escala de cinza → Binária

Conversão entre tipos de imagens

Uma forma de determinar o limiar L é através do **Método de Otsu***.

O método de Otsu determina o limiar L que **minimiza** a variância intraclasse, isto é,

$$\sigma^2(L) = \omega_0(L)\sigma_0^2(L) + \omega_1(L)\sigma_1^2(L)$$

onde $\omega_i(L)$ e $\sigma_i^2(L)$ denotam, respectivamente, a probabilidade de ocorrência da classe i e a variância da classe i para o limiar L .

*Otsu N., A Threshold Selection Method from Gray-Level Histograms. *IEEE Transactions on Systems, Man, and Cybernetics*. Vol. 9, Issue: 1, Jan. 1979.

Imagem colorida → Binária

Conversão entre tipos de imagens

- Uma abordagem para realizar a **segmentação** de uma imagem colorida com base na cor da região de interesse é aplicar **limitares de comparação** para cada camada de cor (de forma individual) e, depois, combinar os resultados.
- Neste caso, o projetista define a cor da região de interesse (r_f, g_f, b_f) (*cor de referência*) e um valor de margem δ , com $0 < \delta < 1$.
- Na sequência, para cada plano R, G e B verifica-se quais pixels se encontraram dentro de uma região em torno do respectivo valor de referência r_f, g_f ou b_f . Gerando para cada plano uma imagem binária.
- Por fim, as imagens binárias obtidas da etapa anterior são combinadas por uma lógica & (E) para gerar a imagem binária final.

Exercício 3: Segmentação de imagem baseada em cor

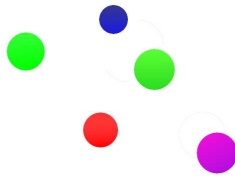
Enunciado: Implemente no Matlab o algoritmo descrito nos slides anteriores para identificar as regiões da imagem `tomatoe_124.jpg` (mostrada abaixo e disponível no Moodle) que possuem tomates.



Para facilitar o procedimento de seleção da *cor de referência* utilize a função `ginput` (`.`), discutida em aula. Para esclarecimentos de como utilizar essa função, acesse https://www.youtube.com/watch?v=Yb10_MbqiSY.

Exercício 4: Segmentação de imagem baseada em cor

Enunciado: Teste o algoritmo implementado no exercício anterior para imagem `circulos.png` (mostrada abaixo e disponível no Moodle).



Os resultados obtidos são bons? Tente desenvolver outro algoritmo para segmentação de imagem baseado em cor que produza melhores resultados de segmentação.

