# Representação de imagens digitais e transformações geométricas

Osmar Tormena Junior, Prof. Dr.*

## 1 Introdução

Aquisição, análise e processamento de imagens digitais possuem inúmeras aplicações nas mais diversas áreas. Nos dias de hoje, é difícil encontrar uma pessoa na rua que não possua uma câmera em seu celular, e este, por sua vez, não possua alguma capacidade de processamento, análise (e até classificação) de imagens.

Nesta disciplina serão abordados os conceitos de:

**Processamento de imagens:** Operações realizadas sobre a imagem, no intuito de atenuar ruído e/ou artefatos de imagem, ao mesmo tempo que trazendo maior distinção às estruturas desejáveis da imagem.

**Análise de imagens:** Dados que podem ser averiguados sobre uma imagem, afim de determinar uma melhor ferramento de processamento, ou para parametrizar características importantes da imagem.

Todo o desenvolvimento teórico, bem como das atividades práticas, será realizado com suporte do Matlab®.

## 2 Imagens digitais

Diferente do paradigma das disciplinas de Sinais e Sistemas e Processamento Digital de Sinais, que em seu desenvolvimento, tinham um enfoque em sinais de uma única dimensão e canal, imagens digitais são sinais bidimensionais (duas dimensões de espaço) e até três canais (um para cada cor).

Outro aspecto importante está no fato de as imagens digitais serem definidas pela representação binária de números inteiros, em contraste com a representação em ponto fixo ou flutuante comumente encontrada em outras aplicações de PDS.

Essas diferenças fundamentais produzem um caráter único na análise e processamento de imagens digitais.

## 3 Representação de imagens digitais

Como qualquer sinal digital, a imagem é discretizada em suas variáveis independentes e dependentes. Diferentes tipos de imagens são representadas em formatos específicos, otimizados para seu próprio caso.

As funções `imread` e `imwrite` são utilizadas para importar e exportar imagens no ambiente Matlab®.

**Exemplo 1** (Carregar imagens:)**:** O código a seguir ilustra a entrada e saída básica de imagens.

```
% Leitura e escrita de imagens
close all
clear
clc
```

*tormena@utfpr.edu.br

```
% ler imagem
A = imread('ngc6543a.jpg');
% mostrar imagem em figura
imshow(A)
% gravar imagem, especificando a
% qualidade
imwrite(A,'teste.jpeg','Quality',75)
```

Neste exemplo, é possível definir a qualidade da compressão JPEG (1–100) e averiguar o tamanho do arquivo resultante e a perda de detalhamento na imagem. □

**Exercício 1:** A função `checkerboard` pode ser utilizada para produzir um padrão simples como imagem. Isto pode auxiliar na visualização de efeitos do processamento da imagem. Modifique o código do Exemplo 1, de maneira a produzir uma imagem JPEG através da função `checkerboard`.

## 3.1 Cores

De uma forma geral, as imagens coloridas são representadas como três páginas num *array* RGB (dependendo do formato, pode ser um *array* com mais páginas). Um outro espaço de cores comumente utilizado é o YCbCr.

As funções `rgb2ycbcr` e `ycbcr2rgb` podem ser utilizadas para conversão entre os espaços de cor. Como a percepção de croma, nos canais Cb e Cr são menos acuradas que a luminância Y, é possível sub-amostrar e quantizar mais agressivamente nos canais de cor, sem degeneração apreciável da qualidade.

Imagens coloridas podem ser reduzidas para imagens preto-e-branco. As informações de cor nas páginas do *array* são sumarizadas em uma única matriz na escala de cinza. A função `rgb2gray` é utilizada para gerar uma imagem preto e branco a partir de uma colorida (RGB).

**Exercício 2:** Obtenha uma imagem em escala de cinza a partir de uma colorida. Compare os tamanhos resultantes dos arquivos.

# 4 Transformações geométricas

Muito embora as transformações geométricas tenham efeitos simples de entender, na imagem visualizada, é importante atentar para efeitos "inesperados" em sua representação digital.

Através da função `imcrop`, é possível recortar intervalos da imagem. Esta ferramenta pode ser utilizado de forma iterativa, ou com um retângulo de corte definido (dado como `[xmin ymin width height]`, em pixels). Esta é uma ferramenta útil ao trabalhar com imagens onde só uma parte é relevante para a análise, pois seu corte, antes do processamento, oferece grandes economias do custo computacional.

**Exemplo 2** (Redimensionando imagens:)**:** A função `imresize` pode ser utilizada para reescalar imagens.

```
% Reescalamento de imagens
close all
clear
clc

% ler imagem
A = imread('peppers.png');
% reescalar
B = imresize(A,3);
% visualizar
figure
imshow(A)
figure
imshow(B)
```

Os pixels da imagem reescalada são resultados da interpolação dos pixels originais. Diferentes métodos de interpolação podem ser utilizados, com diferentes resultados na qualidade "aparente" do resultado. □

Alternativamente, para reescalamentos sequenciais, duplicando ou dividindo à metade (aproximadamente) do tamanho original, pode ser utilizada a função `impyramid`.

**Exercício 3:** Crie um código que recorte um detalhe de uma imagem, reproduzindo

este detalhe no tamanho da imagem original.

A rotação de imagens é outra operação geométrica comumente aplicada. Muito embora a rotação em ângulos retos seja um simples remapeamento da representação da imagem, rotações com ângulos arbitrários podem necessitar de interpolação. Nesse caso, também há a necessidade de definir como será recortada a imagem rotacionada. A função `imrotate` é utilizada para rotacionar imagens.

**Exercício 4:** Tomando uma imagem, que contenha linhas retas ortogonais (e.g. uma placa de circuito impresso), rotacione a imagem no intuito de alinhar as linhas da imagem às bordas.

De forma similar, as imagens podem ser transladadas através da função `imtranslate`. Esta função é útil caso seja necessário alinhar duas imagens sobrepostas.

**Exercício 5:** Através das transformações geométricas, e de espaço de cor, apresentadas, verifique a diferença de qualidade ao subamostrar e quantizar uma imagem RGB e uma imagem YCbCr. Utilize um formato com compressão sem perdas, para uma comparação fidedigna.