

Cap 2. Dados Multimídia

Prática II: Manipulação de Imagens com Python

INE5431 SISTEMAS MULTIMÍDIA
PROF. ROBERTO WILLRICH (INE/UFSC)
ROBERTO.WILLRICH@UFSC.BR
[HTTPS://MOODLE.UFSC.BR](https://moodle.ufsc.br)

Aula Prática II

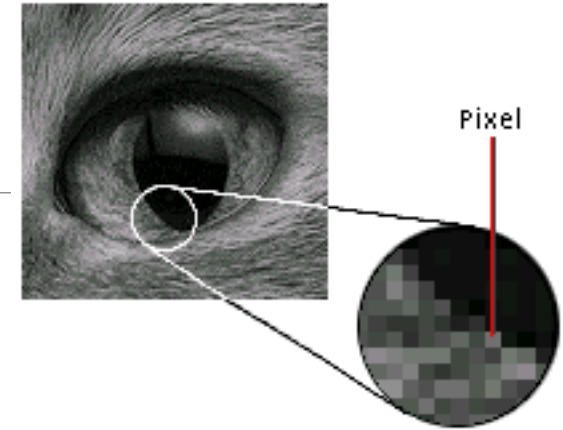
Objetivos

- Reforçar conceitos básicos em representação digital de imagens
- Entender mais sobre o manipulação de imagens digitais usando Python
- Suposições
 - Conhecimento em Python

Imagens Digitais

Formatos de Imagens

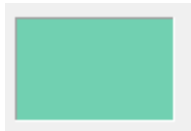
- Imagens no computador são representadas por bitmaps
 - bitmap = matriz espacial bidimensional de elementos de imagem chamados de pixels
 - pixel é o menor elemento de resolução da imagem
 - tem um valor numérico chamado amplitude
 - define ponto preto e branco, nível de cinza, ou atributo de cor (3 valores)
 - Expresso por um número de bits
 - 1 para imagens P&B, 2, 4, 8, 12, 16 ou 24 bits
 - "Resolução" da imagem é o número de elementos que a imagem possui na horizontal e na vertical



Imagens Digitais


Convertendo imagens RGB em tons de cinza

- $Y = 0.3R + 0.59G + 0.11B$;



Vermelho:	113
Verde:	208
Azul:	177

$$Y = 0,3*113+0,59*208+0,11*177=176$$



Convertendo imagens tons de cinza em binárias

- Pixel é preto se tom de cinza é abaixo da metade da escala e branco se o tom de cinza é acima da metade da escala
 - Exemplo: imagens de 256 tons:
 - $y \geq 127$ é branco
 - $Y < 127$ é preto

Imagens em Python

Pacotes/módulos utilizados

- *from urllib.request import urlopen*
 - *urlopen*: para leitura de um recurso a partir de um URL
- *from PIL import Image*
 - PIL é a biblioteca de imagens do Python, com várias funções para manipular imagens. **Não utilizem nenhuma função além daquelas usadas no código entregue**
- *import math*
 - Módulo que fornece acesso às funções matemáticas definidas pelo padrão C

Imagens em Python

Funções usadas do modulo Image

- *img = Image.new(mode,size, color=0)*
 - mode: “RGB” para true color, “L” para tons de cinza, e “1” para binária
 - size: tuple (largura,altura) em pixels
 - Color: cor usada para a imagem (default é preto)
- *img.show()*
 - Apresentação da imagem
- *img.size[0]* e *img.size[1]*
 - Permite ler a largura e altura de uma imagem
- *raster = img.load()*
 - Função que aloca armazenamento para a imagem e carrega os pixels (raster)
 - Dado pode ser manipulado usando *raster[i,j]* para ler e escrever o valor do píxel na posição (i,j)
 - *raster[i,j] = (220,219,97,255)* # R=220,G=219,B=98,alfa=255
- *(r, g, b) = img.getpixel((0, 0))*
 - Leitura do píxel na posição (0,0)

Imagens em Python

Main

```
# Leitura de uma imagem
img = Image.open(urlopen("https://www.inf.ufsc.br/~roberto.willrich/INE5431/RGB.jpg"))

# apresentação da imagem
img.show()

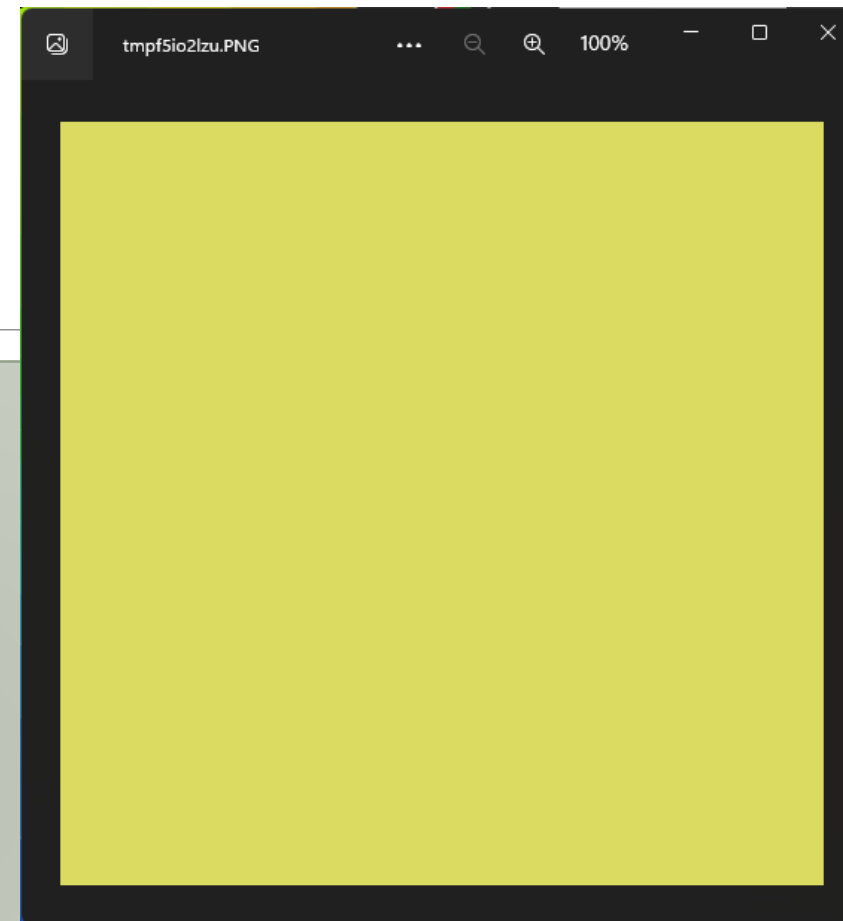
# Cria e apresenta uma imagem True Color
criarImagemRGB().show()

#Cria e apresenta uma imagem Tons de Cinza
criarImagemCinza().show()

#Cria e apresenta uma imagem Binária
criarImagemBinaria().show()
```

Imagens em Python

```
def criarImagemRGB():  
    img = Image.new( "RGB", (512,512))  
    raster = img.load()  
    for i in range(img.size[0]):  
        for j in range(img.size[1]):  
            raster[i,j] = (220,219,97,255)  
    (r, g, b) = img.getpixel((0, 0))  
    print(r, g, b)  
    return img
```



Imagens em Python

```
def criarImagemCinza():  
    img = Image.new( "L", (256,512))  
    raster = img.load()  
    for i in range(img.size[0]):  
        for j in range(img.size[1]):  
            raster[i,j] = i  
    y = img.getpixel((5, 5))  
    print(y)  
    return img
```



Imagens em Python

```
def criarImagemBinaria():  
    # checkerboard pattern.  
    img = Image.new("1", (250,150))  
    raster = img.load()  
    for i in range(img.size[0]):  
        for j in range(img.size[1]):  
            if ((int(i/50)+int(j/50)) % 2 == 0):  
                raster[i,j] = 0  
            else:  
                raster[i,j] = 1  
    return img
```

