INSTITUTO FEDERAL
São Paulo

Exercícios #01

Instituto Federal de São Paulo (IFSP)
Campus Caraguatatuba
Tópicos Avançados (TPA A6)

13 de Outubro de 2021

Objetivos Essa é a primeira lista de exercícios da disciplina de "Tópicos Avançados". Essa lista tem como objetivo introduzir os conceitos básicos de programação Python e das bibliotecas de algebra linear Numpy e de visão computacional OpenCV. Você irá trabalha com conversão de espaço de cores (i.e., RGB e HSV), separação e manipulação de canais de cores, e processamento de imagens e vídeos utilizando o OpenCV.

Exercício 1.01. Nesse exercício, você irá converter imagens JPG e o *stream* de vídeo fornecido por sua *webcam* em dois espaço de cores distintos (i.e., RGB e HSV). Você também irá mostrar como cada canal contribui com a representação das cores na imagem ilustrada. Primeiro, abra o arquivo `Ex101_color_space.py` disponível no material de suporte dessa lista de exercícios e, então, leia com atenção a estrutura do código-fonte disponível. Você irá desenvolver seu código diretamente nesse arquivo Python.

(a) **Conversão de espaço de cores:** Use a função `cv2.cvtColor()` [1] para converter uma imagem colorida em um espaço de cor definido. Esta função possui dois parâmetros obrigatórios, a saber: `src`, a imagem de entrada; e `code`, um valor inteiro ou uma enumeração com o código de conversão do espaço de cores. Essa função retorna um objeto do tipo `dst` contendo uma nova imagem convertida para o espaço de cores selecionado. Você deve criar duas novas imagens com os seguintes códigos:

- `image_rgb`: para criar uma imagem RGB, utilize o código `cv2.COLOR_BGR2RGB`. *Dica*: por padrão, o OpenCV manipula imagens digitais em uma ordem reversa (i.e. BGR); e
- `image_hsv`: para criar uma imagem HSV, utilize o código `cv2.COLOR_BGR2HSV`.

Dica: Você pode obter mais informações sobre qualquer função de bibliotecas disponível no seu ambiente de desenvolvimento Python, utilizando o comando `help` no interpretador Python. Por exemplo: `help(cv2.cvtColor)`.

[1] Uma função da biblioteca OpenCV utilizada para converter uma imagem de um espaço de cor para um outro espaço distinto.

(b) **Separação de canais de cores:** Depois de converter a imagem de entrada em espaço de cores distintos, você deve separar a imagem de múltiplos canais em três imagens de canais simples (e.g., `channel_r`, `channel_g` e `channel_b` para uma imagem RGB). Para responder esse exercício, você tem duas opções:

- Separe diretamente o *array* do Numpy (e.g. `image_rgb[:, :, 0]` – isto significa, selecione todas as colunas e todas as linhas do canal vermelho); ou
- Separe os canais da imagem utilizando a função `cv2.split()` [2]. Esta função tem somente um parâmetro (i.e. `image`) e ela retorna cada canal como uma imagem distinta de canal simples.

(c) **Mude a representação do canal de cor:** Quando você separa uma imagem de múltiplos canais, cada canal irá parecer como uma imagem em tons de cinza. Nesse exercício, você deve mudar a representação do canal de cor para demonstrar como cada componente contribui para formar a imagem colorida. As instruções a seguir mostram como mudar a representação do canal "verde" em uma imagem RGB.

- Crie um *array* Numpy (chamado `single`) com a mesma dimensão da imagem de entrada (`width` e `height`) e preencha todos os elementos do *array* com valores iguais a zero (0). Utilize a função `numpy.zeros()` [3] e informe a resolução da imagem como primeiro parâmetro e defina o valor do parâmetro `dtype` como `uint8` (i.e., uma representação numérica que aceita apenas valores entre 0 e 255);
- Utilize a função `cv2.merge()` [4] para unir várias imagens de canal simples em uma image de múltiplos canais (e.g., preencha os canais *vermelho* e *azul* com zeros e una-os com o canal *verde*: `cv2.merge([single, channel_g, single])`).

Você deve mudar a representação dos canais para os espaço de cores RGB e HSV. Por exemplo, a Figura 1 ilustra a tradicional foto da *Lena* em três canais distintos:

Figure 1: Exemplo da representação RGB.

[2]Uma função disponível no OpenCV para dividir uma imagem de múltiplos canais em vários *arrays* de canais simples.

[3]Uma função da biblioteca Numpy para criar um novo *array* com todos os elementos iguais a zero.

[4]Uma função da biblioteca OpenCV para criar um *array* de múltiplos canais a partir de um conjunto de imagens de canal simples.

INSTITUTO FEDERAL
São Paulo

Exercício 1.02. Nesse exercício, você irá repetir todos os passos do exercício anterior, no entanto, você irá trabalhar com o *stream* de vídeo da sua *webcam*. Abra o arquivo `Ex102_video_stream.py` disponível no material de suporte dessa lista de exercícios e, então, leia com atenção a estrutura do código-fonte disponível. Você irá desenvolver seu código diretamente nesse arquivo Python. *Dica*: Caso o seu computador não tenha uma câmera embutida, você pode utilizar um arquivo de vídeo para responder esse exercício. Na pasta `inputs` do material de suporte você irá encontrar um arquivo de vídeo chamado `PPAP.mp4`.