Universidade Federal de Campina Grande

Centro de Ciências e tecnologias

Relatório da segunda missão (ras)

openCV

Aluno: Rodrigo Milani Rocha matrícula: 122110425

Período: 2

Campina grande,2023

1 introdução

O openCV busca dar a visão a uma maquina e facilitar a troca de informações entre a maquina e ser humano que a está operando, transformando a imagem em dados mais “entendíveis” para o computador facilitando a troca dessas informações e possibilitando então o armazenamento das mesmas

1.1 material utilizado

Utilizando de um computador com python 3.10 instado, com o compilador pycharm 2022.2.3 instalado e utilizando a biblioteca cv2 derivada do openCV, fiz os exemplos pedidos.

1.2 objetivos

O principal objetivo dessa missão é melhor entender e começar a ser capaz de manipular e ser capaz de interpretar imagens através de uma maquina que entende as imagens através de matrizes, nesse caso pelo python

2 desenvolvimentos

Após levar algum tempo considerável para baixar as bibliotecas necessárias o que levei um tempo a te descobrir uma dependência do compilador de ser informado das novas bibliotecas a serem importadas, então comecei a me aventurar pelo livro guia disponibilizado *"Introdução a Visão Computacional com Python e OpenCV"* e seguindo os exemplos formei diversas imagens.

Depois da instalação das bibliotecas, decidi testar como o computador me devolveria a imagem fornecida para ele, onde o mesmo me devolveu uma sequencia de matrizes onde deixarei algumas para melhor interpretar

[255 255 255]

[255 255 255]

[255 255 255]]

...

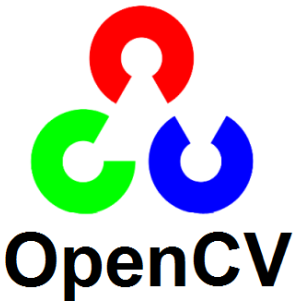
[[255 255 255]

[255 255 255]

[255 255 255]

Depois comecei de fato a testar os exemplos fornecidos.

Entrando com a imagem ao lado no código:




```
# testando o primeiro exemplo do livro "Introdução a Visão Computacional com Python e
OpenCV"
# onde testamos as funções basicas da biblioteca numa imagem (que nesse caso usei a logo
do opencv)
import cv2
# Leitura da imagem com a função imread()
imagem = cv2.imread('OpenCV_Logo.png')

print('Largura em pixels: ', end='')
print(imagem.shape[1]) #largura da imagem

print('Altura em pixels: ', end='')
print(imagem.shape[0]) #altura da imagem

print('Qtde de canais: ', end='')
print(imagem.shape[2])

# Mostra a imagem com a função imshow
cv2.imshow("Nome da janela", imagem)
cv2.waitKey() #espera pressionar qualquer tecla para continuar o programa
# Salvar a imagem no disco com função imwrite()
cv2.imwrite("saida.jpg", imagem)
```

Que utiliza as funcionalidades básicas da biblioteca, que são coisas simples como, receber a imagem, descobrir em pixel a altura e largura da imagem, apresentar a imagem e salvar a imagem, pode-se começar a entender como interpretar a imagem, o qual é o objetivo do primeiro capítulo

Quando adentramos o segundo capitulo começamos a aprender como manipular os pixels e a localiza-los na imagem. Entrando com a mesma imagem do capitulo anterior no código a seguir. Que se baseia nos exemplos do livro só mudando os valores do RGB (ou como utilizado no programa BGR)

```
# testando o segundo conjunto do livro "Introdução a Visão Computacional com Python e
OpenCV"
# aqui eu testo os exemplos do segundo capitulo

import cv2
imagem = cv2.imread('saida.jpg')
imagem2 = cv2.imread('OpenCV_Logo.png')

#alterei os valores para criar algo diferente do livro

#exemplo 2(o primeiro exemplo ele só iria passar pixel por pixel "pintando")
for y in range(0, imagem2.shape[0]): #percorre linhas
 for x in range(0, imagem2.shape[1]): #percorre colunas
    imagem2[y, x] = (x%56,y%140,x%156)
cv2.imshow("Imagem modificada", imagem2)
cv2.waitKey()
cv2.imwrite("ex2cap2.jpg", imagem2)
#terceiro exemplo
for y in range(0, imagem.shape[0], 1):
 for x in range(0, imagem.shape[1], 1):
    imagem[y, x] = (0,(x*y)%256,0)
cv2.imshow("Imagem modificada-2", imagem)
cv2.waitKey(0)
cv2.imwrite("ex3cap2.jpg", imagem)
#quarto e ultimo exemplo, onde pegarei a imagem alterada e farei um pequeno esquema
tendo o quarto exmplo como base

for y in range(0, imagem2.shape[0], 10):
    for x in range(0, imagem2.shape[1], 60):
        imagem2[y:y+5, x: x+5] = (255,65,187)

for y in range(0, imagem2.shape[0], 47):
    for x in range(0, imagem2.shape[1], 3):
        imagem2[y:y+5, x: x+5] = (125,165,18)

cv2.imshow("Imagem modificada", imagem2)
cv2.waitKey(0)
cv2.imwrite("abstrata.jpg", imagem2)
```

Ele retorna as seguintes imagens respectivamente no segundo, terceiro e quarto exemplo depois de efetuar o código acima.

Figura 1: chamada no código de ex2cap2

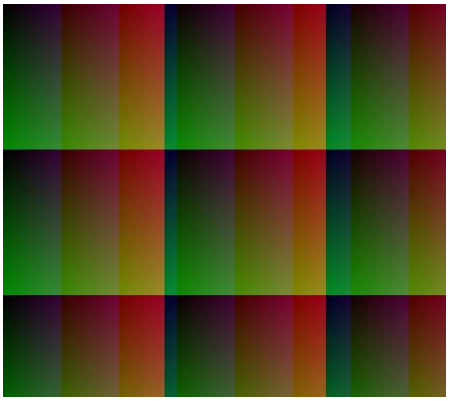


Figura 2: chamada no código de ex3

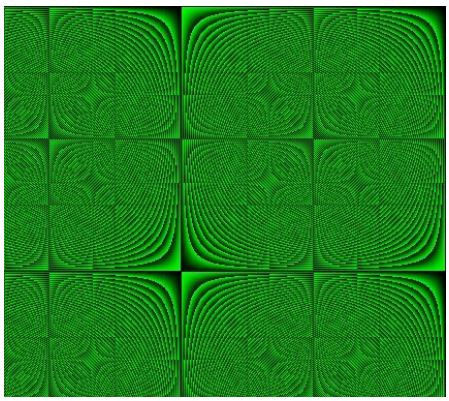
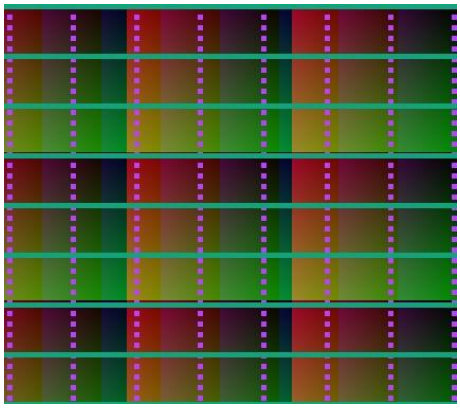


Figura 3: chamada no código de abstrata



3 conclusão

Com isso comecei a adquirir a habilidade de manipular imagens através do openCV, podendo assim depois de desenvolver essa habilidade cada vez mais, ser capaz de criar softwares capazes de através das imagens de uma câmera, interpretar o mundo podendo ter melhor dados para ser capaz de tomar melhores decisões de acordo com os parâmetros parâmetros fornecidos.