Exercício nº 7

"Paralelização da atividade de ajuste de brilho e contraste de imagens"

O objetivo deste exercício é modificar um código de processamento de imagens, especificamente para contraste e brilho, de tal modo que realize o processamento de maneira paralela.

As técnicas voltadas para a análise de dados multidimensionais, adquiridos por diversos tipos de sensores recebem o nome de processamento digital de imagens, ou seja, é a manipulação de uma imagem por computador de modo onde a entrada e a saída do processo são imagens.

> **Pixels:** são normalmente usados para se referir à resolução de exibição de um monitor ou tela de computador. Quanto maiores os pixels, maior o detalhe da imagem.
>
> **Brilho:** a intensidade ou brilho é a medida de energia total envolvida em todos os comprimentos de onda e, portanto, responsável pela sensação de brilho dessa energia incidente sobre o olho. Ou seja, quando o brilho é ajustado, toda a gama de tons na imagem é aumentada ou diminuída de acordo.
>
> **Contraste:** a técnica de realce de contraste tem por objetivo melhorar a qualidade das imagens sob os critérios subjetivos do olho humano. Quando o ajuste de contraste é aumentado, os tons médios são eliminados. A imagem terá uma porcentagem maior de tons escuros ou pretos e brancos ou realces com tons médios mínimos. A manipulação do contraste consiste numa transferência radiométrica em cada "pixel", com o objetivo de aumentar a discriminação visual entre os objetos presentes na imagem. Realiza-se a operação ponto a ponto, independentemente da vizinhança.

Modifique o código "gamma1.py" para que funcione com *multiprocessing*, utilizando pelo menos 2 (dois) subprocessos. Capture os tempos de execução do código sequencial e do paralelo para efeito de análise.

Utilize uma mesma imagem para análise.

Código de exemplo:

```
'''
Este código carrega uma imagem jpg e realiza um ajuste de brilho e contraste.

22/08/2022
'''

import numpy as np
from PIL import Image
from time import perf_counter


# matriz = imagem para operação; alpha = contraste ;  beta= brilho ;
def define_brilho(matriz, alpha, beta):
    # Cria-se uma imagem temporária, "vazia", com as dimensões e tipo da original (passada por parâmetro)
    imagem_result = np.zeros(matriz.shape, matriz.dtype)

    # Obtem-se as dimensões e quantidade de canais da imagem original (passada por parâmetro)
    # No nosso caso, vamos obter as dimensões em pixels (y, x) e a quantidade de canais (RGB = 3)
    y, x, c = matriz.shape

    # Percorre todos os pares x,y da imagem em todos os canais
    for ypos in range(y):
        for xpos in range(x):
            for canal in range(c):
                # Realiza-se os cálculos necessários para cada ponto e canal, mantendo o valor resultante dentro
                # do limite de 0 a 255
                imagem_result[ypos, xpos, canal] = np.clip((alpha * matriz[ypos, xpos, canal]) + beta, 0, 255)

    # Retorna a matriz modificada da imagem
    return imagem_result




if __name__ == '__main__':
    start_time = perf_counter()

    # Carrega uma imagem de um arquivo
    img = Image.open('c:\\temp\\img001.jpg')

    # Converte a imagem em uma matriz RGB
    imagem_original = np.asarray(img)

    # Executa a função de contraste e brilho
    imagem_brilho = define_brilho(imagem_original, 3, 100)

    # Salva a nova imagem (matriz) em um novo arquivo
    imagem2 = Image.fromarray(imagem_brilho)
    imagem2.save('c:\\temp\\img002.jpg')

    end_time = perf_counter()
    print(f'As tarefas levaram {end_time- start_time: 0.2f} segundo(s) para executar.')
```