

TRABALHO UM : CORREÇÃO DE CONTRASTE DE REGIÕES ESCURAS EM IMAGENS COLORIDAS

CORREÇÃO DE CONTRASTE DE REGIÕES ESCURAS EM IMAGENS COLORIDAS

O objetivo deste trabalho é projetar, implementar e testar um método para a correção de contraste em regiões escuras em imagens coloridas. Considere uma imagem que contenha um alto contraste global (por possuir regiões claras e escuras), mas que tenha regiões escuras com baixo contraste. A correção deve ser realizada somente nestas regiões escuras, enquanto as regiões claras não devem ser processadas. Basicamente, combine uma operação de thresholding com uma correção de contraste aplicada localmente (na região escura).

O trabalho, como diz o enunciado, deve ser aplicado em imagens coloridas. Você deve converter a imagem descrita acima para o sistema HSV e processar somente a banda V (não altere as bandas H e S). Após corrigir o contraste na banda V, você deve concatenar as bandas H, S (inalteradas) e V e fazer a conversão inversa para RGB.

Você deve explicar a sua abordagem e os métodos que você escolheu como suporte para a sua solução. Você deve desenvolver sua própria solução, mas pode utilizar funções auxiliares do OpenCV e do scikit-image em sua implementação. O método deve ser implementado em Python 3.x, usando-se a biblioteca Numpy. Use a biblioteca OpenCV ou scikit-image para carregar/salvar/mostrar as imagens. Priorize a utilização de fatiamento (slicing) no lugar de estruturas de repetição.

Você pode utilizar a função de Otsu, (não-paramétrico e disponível nas bibliotecas de imagens citadas) ou fazer o threshold manual (que requer um limiar escolhido pelo usuário). Para a correção, utilize a normalização do histograma ou a equalização do histograma. A sua escolha de correção de contraste deve ser implementada por você.

Você deve testar duas situações. Suponha que t seja o valor de limiar.

Na primeira situação, corrija o contraste da região escura para o intervalo [0, 255];

Na segunda situação, corrija para o intervalo [0, t].

Verificar em qual dos casos a correção é mais natural.

```

def normalizar_v(hsv_img):
    h = hsv_img[:, :, 0]
    s = hsv_img[:, :, 1]
    v = hsv_img[:, :, 2]
    v_min = v.min()
    v_max = v.max()
    print(f"v_min: {v_min:.2f}, v_max: {v_max:.2f}")
    if v_max - v_min == 0:
        v_norm = np.zeros_like(v)
    else:
        v_norm = (v - v_min) / (v_max - v_min) * 255

    hsv_normalizado = np.stack([h, s, v_norm], axis=2).astype(np.uint8)
    return hsv_normalizado

```

Código 1 - Código fonte da função de normalização

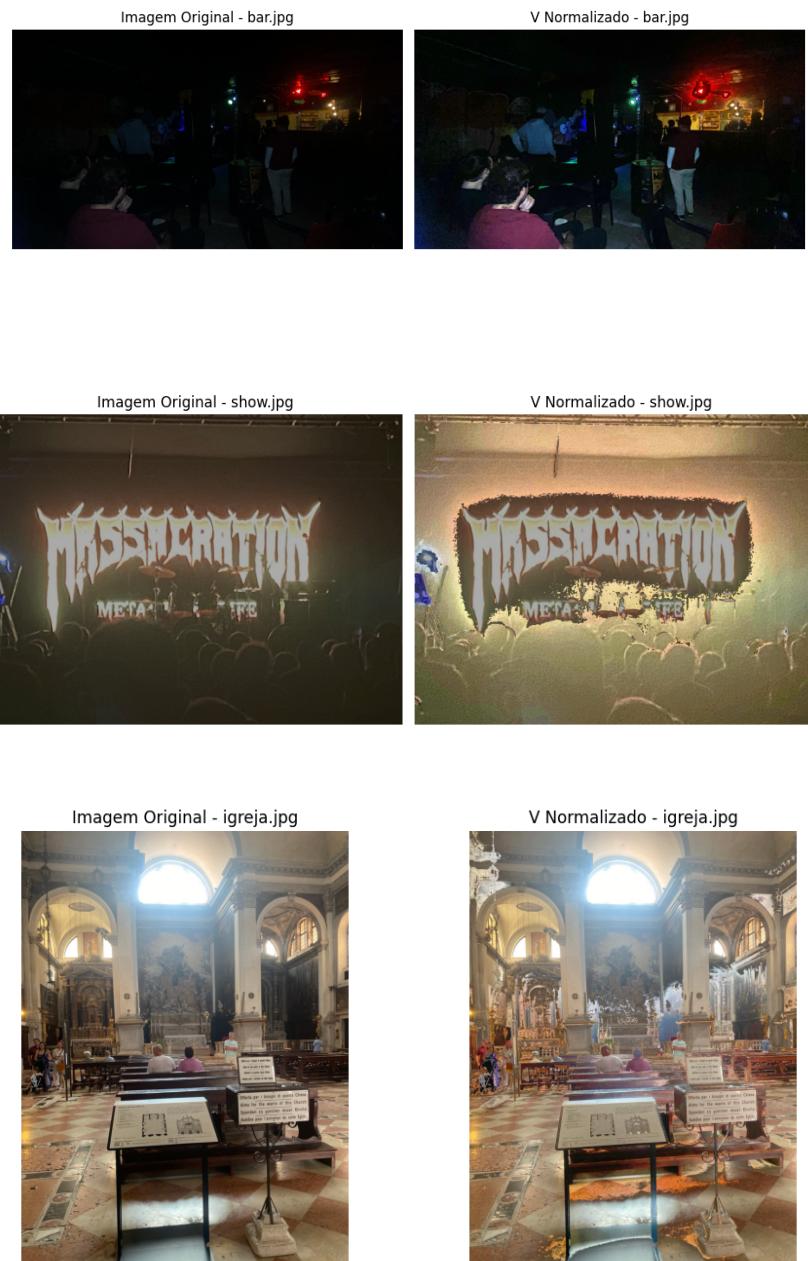


Figura 2 - Imagens Utilizadas antes de depois da normalização com limiar em V = 80

Uma terceira variação foi feita limitando o valor de V na saída das áreas escuras. Na versão 2 a saída poderia estar entre 0 e 255, na terceira versão a saída das áreas escuras utilizaram um limite de 130% do limitar utilizado. As imagens saídas estão na figura 3 e diferença entre as funções estão no código 2.

```
def normalizar_v_com_limite_limitado(hsv_img, limite_v):
    h = hsv_img[:, :, 0]
    s = hsv_img[:, :, 1]
    v = hsv_img[:, :, 2]
    v_min = v.min()
    v_max = v.max()
    print(f"v_min: {v_min:.2f}, v_max: {v_max:.2f}, limite: {limite_v}")
    v_norm = v.copy()
    mascara = v < limite_v
    if np.any(mascara):
        v_sub = v[mascara]
        v_sub_min = v_sub.min()
        v_sub_max = v_sub.max()
        if v_sub_max - v_sub_min == 0:
            v_norm[mascara] = 0
        else:
            v_norm[mascara] = ((v_sub - v_sub_min) / (v_sub_max - v_sub_min)) * limite_v * 255
    OU
            v_norm[mascara] = ((v_sub - v_sub_min) / (v_sub_max - v_sub_min)) * limite_v * 1.3
    hsv_normalizado = np.stack([h, s, v_norm], axis=2).astype(np.uint8)
    return hsv_normalizado
```

Código 2 - Código fonte da função de normalização das áreas escuras

Fica evidente o *tradeoff* entre o aumento de contraste e a naturalidade da imagem. A escolha do limiar e do limite superior para as áreas escuras é fundamental para esta naturalidade.

A terceira variação foi a escolha automática do limitar através do limiar_otsu limitando a saída em 80% do limiar calculado.

Os valores calculados para cada figura são apresentados na Tabela 1 e os resultados na figura 4.

Arquivo	Limiar Otsu	V Mínimo	V Máximo	Limite (1.8 × Otsu)
bar.jpg	45	0.00	255.00	81.0
show.jpg	101	10.00	177.00	181.8
igreja.jpg	120	0.00	255.00	216.0

Tabela 1 - Valores de limiares e limites de cada imagem

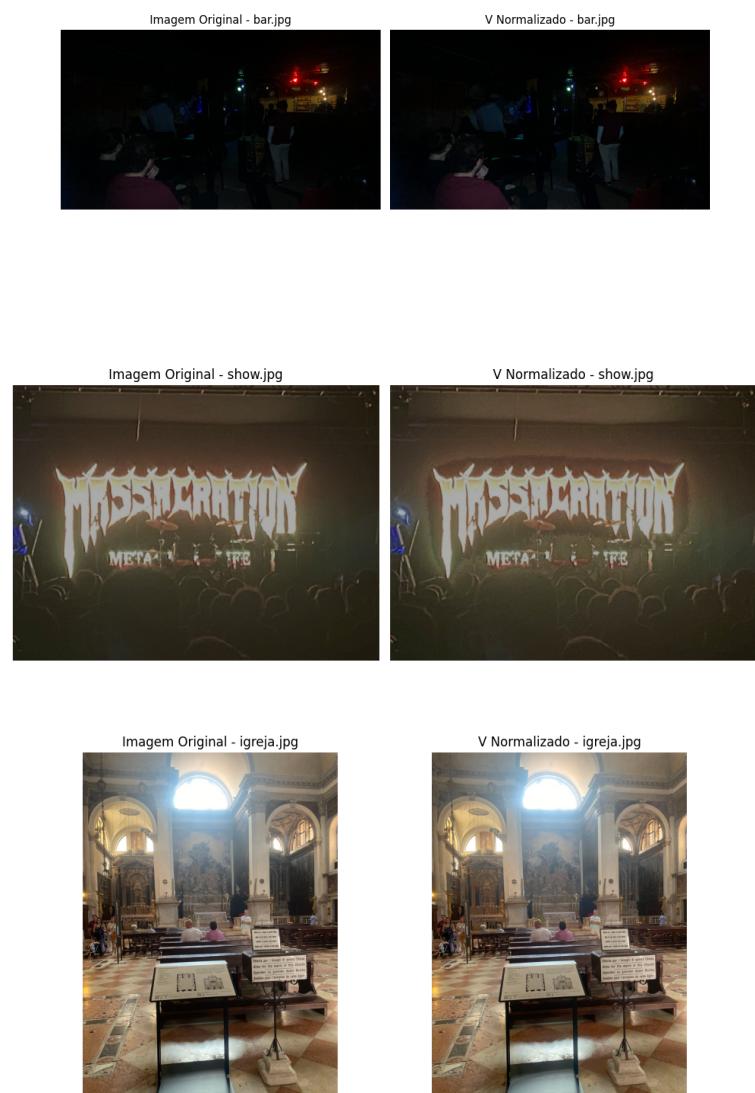


Figura 3 - Imagens Utilizadas antes de depois da normalização com limiar em $V = 80$ limitados as 130%

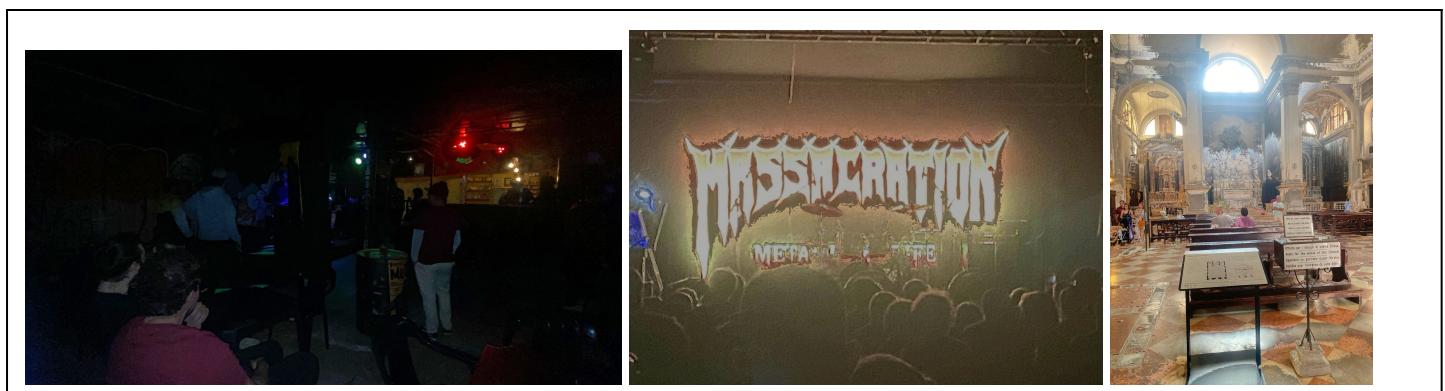


Figura 4 - Imagens Utilizadas depois da normalização com limiar em V por otsu limitados as 180%

Os resultados da variação com o limiar de cálculo automático se mostraram mais interessantes embora tenham causado distorções antinaturais. Todos os métodos, com exceção do tradicional necessitam de supervisão e ajustes para obter os resultados mais naturais.