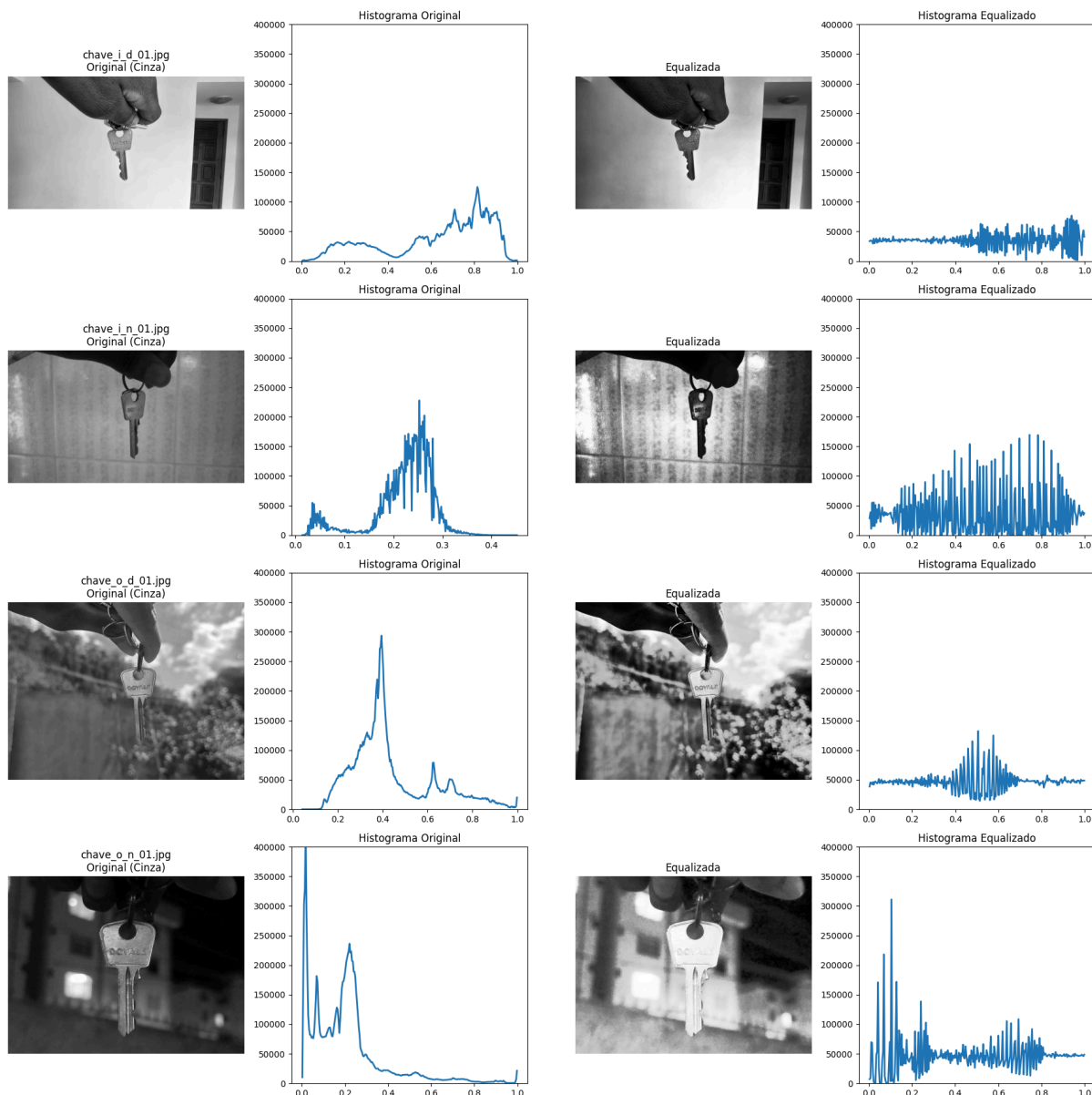


Atividade 5

Histogramas



- **chave_i_d_01 (Interno, Dia):** O histograma original era desbalanceado devido ao fundo muito claro. A equalização distribuiu melhor os tons por toda a faixa, o que resultou em um aumento visível do contraste e dos detalhes da chave.


- **chave_i_n_01 (Interno, Noite):** O histograma original estava todo comprimido na zona escura, indicando uma foto subexposta. A equalização o expandiu, mas acabou por realçar principalmente o ruído e a textura do fundo, piorando a qualidade visual da chave.
- **chave_o_d_01 (Externo, Dia):** O histograma original já era bem distribuído, mas com um pico de brilho do fundo. A equalização equilibrou a imagem, reduzindo a intensidade das áreas mais claras e, com isso, melhorou a definição da textura da chave.
- **chave_o_n_01 (Externo, Noite):** O histograma original mostrava um pico no preto (fundo) e os pixels da chave em uma faixa estreita. A equalização esticou a faixa de intensidade da chave, fazendo-a parecer muito mais clara e destacando seus detalhes contra o fundo escuro, porém introduzindo grandes oscilações no processo, o que justifica o resultado não tão eficiente em realçar a chave.

Imagens Utilizadas


Github

GitHub - andreb308/processamento-de-imagens

Contribute to andreb308/processamento-de-imagens development by creating an account on GitHub.

 <https://github.com/andreb308/processamento-de-imagens/tree/master>

andreb308/
processamento-de-...



1 Contributor


0 Issues

0 Stars

0 Forks

Google Drive

grayscale - Google Drive

 https://drive.google.com/drive/u/0/folders/1mWVvEW_MXpuPE_8nWBIGyuhLAFzXMp_9

Código

```
import matplotlib.pyplot as plt
import numpy as np
import os
```

```

from skimage import io, color, util
from skimage.exposure import equalize_hist, histogram

# Salvando os nomes de cada arquivo de imagem em uma lista
image_files = [f for f in os.listdir('./images') if os.path.isfile(os.path.join('./images', f))]
fig, axes = plt.subplots(4, 4, figsize=(16, 16))
fig, axes = plt.subplots(4, 4, figsize=(18, 18))

# Selecionando as imagens de chave (índices 4 a 7)
for idx, filename in enumerate(image_files[4:8]):
    filepath = os.path.join('./images', filename)
    image = io.imread(filepath)
    grayscale_img = color.rgb2gray(image)

    # Histograma da imagem original
    hist, hist_centers = histogram(grayscale_img)

    # Equalização do histograma
    equalized_img = equalize_hist(grayscale_img)
    hist_eq, hist_centers_eq = histogram(equalized_img)

    row = idx

    # Imagem original em cinza
    axes[row, 0].imshow(grayscale_img, cmap='gray')
    axes[row, 0].set_title(f'{filename}\nOriginal (Cinza)')
    axes[row, 0].axis('off')

    # Histograma da imagem original
    axes[row, 1].plot(hist_centers, hist, lw=2)
    axes[row, 1].set_title('Histograma Original')
    axes[row, 1].set_ylim([0, 400000])

    # Imagem equalizada
    axes[row, 2].imshow(equalized_img, cmap='gray')
    axes[row, 2].set_title('Equalizada')
    axes[row, 2].axis('off')

```

```
# Histograma da imagem equalizada
axes[row, 3].plot(hist_centers_eq, hist_eq, lw=2)
axes[row, 3].set_title('Histograma Equalizado')
axes[row, 3].set_ylim([0, 400000])

plt.tight_layout()
plt.show()
```