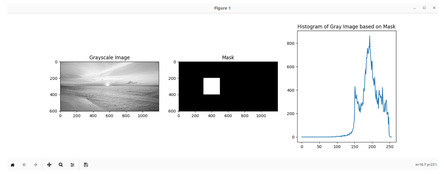
Histogramas de Imagem

O histograma de uma máscara particular ou região de interesse (ROI) pode ser útil em alguns casos para avaliar o histograma local em uma área específica.

```
codes > 🐍 mask_hist.py > ...
import cv2
import numpy as np
import matplotlib.pyplot as plt

img = cv2.imread('./images/sunset.jpg', 0)

# mask
mask = np.zeros(img.shape[:2], np.uint8)
mask[200:400, 300:500] = 255

# histogram based on mask
hist_mask = cv2.calcHist([img], [0], mask, [256], [0, 256])

# plt image, mask, histogram
fig, ax = plt.subplots(1, 3, figsize=(15, 5))

ax[0].imshow(img, cmap='gray')
ax[0].set_title('Grayscale Image')

ax[1].imshow(mask, cmap='gray')
ax[1].set_title('Mask')

ax[2].plot(hist_mask)
ax[2].set_title('Histogram of Gray Image based on Mask')

plt.show()
```

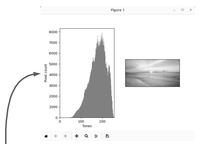


Conceito

O histograma de imagem exibe a distribuição dos valores de intensidade de pixels e ajuda a visualizar o contraste, brilho e alcance tonal geral de uma imagem.

O histograma revela a distribuição das tonalidades de cinza na imagem e permite identificar os pontos mais claros e escuros. Dessa forma, é possível verificar se a faixa de intensidade de pixels está sendo utilizada de forma adequada.

Uma análise do histograma e da distribuição espacial dos valores pode fornecer uma ideia do brilho da imagem, onde valores concentrados à esquerda indicam uma imagem mais escura, enquanto valores concentrados à direita indicam uma imagem mais clara.

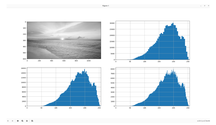



```
import cv2
import numpy as np
from matplotlib import pyplot as plt

# Load the image in grayscale
img = cv2.imread('./images/sunset.jpg', cv2.IMREAD_GRAYSCALE)

# Calculate histogram
hist, bins = np.histogram(img.flatten(), 256, [0, 256])

# Plot the histogram
plt.subplot(122)
plt.hist(img.flatten(), 256, [0, 256], color='gray')
plt.xlim([0, 256])
plt.title('Histogram')
plt.xlabel('Pixel Value')
plt.ylabel('Pixel count')

# Display the image
plt.subplot(121)
plt.imshow(img, cmap='gray')
plt.title('Grayscale Image')

# Show the plot
plt.show()
```

Terminologia

O histograma é uma representação gráfica do número de pixels em cada valor de pixel, dividido em subpartes chamadas 'bins', cujos valores são a soma das contagens de pixels. Os parâmetros histSize, dims e range, na documentação do OpenCV, referem-se, respectivamente, ao número de bins, ao número de valores de intensidade coletados e à faixa de valores de intensidade medidos.

Reduzir o número de compartimentos (BINS) pode ser usado para comparação de imagens e simplificação de histogramas.



Contraste e Faixa Dinâmica

Contraste é a diferença entre os valores mínimos e máximos de pixels em uma imagem, e uma imagem com alto contraste usa eficientemente toda a faixa de valores de intensidade disponíveis. A faixa dinâmica é o número de valores de pixel distintos na imagem, que idealmente deve abranger todos os valores utilizáveis .

Histogramas com mais de um byte por pixel

Usar 12 ou 16 bits em scanners pode capturar valores de intensidade mais precisos, mas tem benefícios limitados e pode causar problemas de compatibilidade. Além disso, os olhos humanos não conseguem discernir mais do que 256 níveis de cinza em uma tela de computador, então profundidades de bits mais altas não necessariamente melhoram a qualidade da imagem.

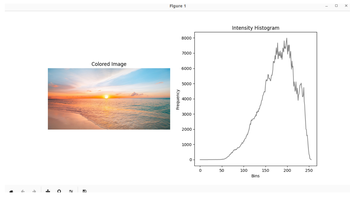
Em áreas como medicina e astronomia, uma maior profundidade é crucial para medições e análises precisas, resultando em descobertas potencialmente salvadoras de vidas. Embora não seja necessário para aplicações cotidianas, o aumento de profundidade é crucial em áreas especializadas.

Histogramas de Imagens Coloridas

Histogramas de Intensidade

O histograma de intensidade de uma imagem colorida é equivalente ao histograma da imagem em escala de cinza correspondente, obtida por uma soma ponderada dos canais de cor, baseada na teoria da percepção de cores.

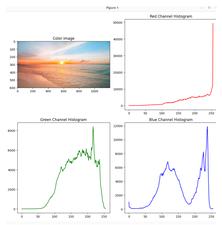
```
codes > 🐍 intensity_hist.py > ...
import cv2
import numpy as np
import matplotlib.pyplot as plt

# load colored image
img = cv2.imread('./images/sunset.jpg')

# get the gray image based on the color one
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

# calculate histogram
hist = cv2.calcHist([gray], [0], None, [256], [0, 256])

# plot
fig, ax = plt.subplots(1, 2, figsize=(12,6))
ax[0].imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
ax[0].set_title('Colored Image')
ax[0].axis('off')
ax[1].plot(hist, color='gray')
ax[1].set_title('Intensity Histogram')
ax[1].set_xlabel('Bins')
ax[1].set_ylabel('Frequency')
plt.show()
```



Histogramas de cada canal de cor

Os histogramas de luminância em cada componente ajudam a avaliar iluminação, contraste e saturação de cada canal, mas não representam a distribuição de cor real da imagem, pois são baseados em canais de cor individuais, não na combinação de canais que formam um pixel.

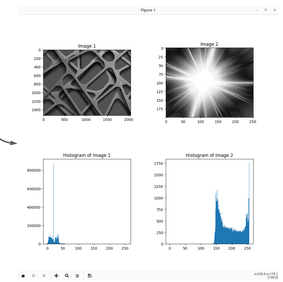
```
import cv2
import numpy as np
import matplotlib.pyplot as plt

# Load color image
img = cv2.imread('./images/sunset.jpg')

# Split image into three color channels
b,g,r = cv2.split(img)

# Calculate histograms for each color channel
hist_r = cv2.calcHist([r],[0],None,[256],[0,256])
hist_g = cv2.calcHist([g],[0],None,[256],[0,256])
hist_b = cv2.calcHist([b],[0],None,[256],[0,256])

# Plot color image and histograms in one figure
fig, axs = plt.subplots(nrows=2, ncols=2, figsize=(10,10))
plt.gray()

axs[0,0].imshow(img[:,:,::-1])
axs[0,0].set_title('Color Image')

axs[0,1].plot(hist_r, color='r')
axs[0,1].set_title('Red Channel Histogram')

axs[1,0].plot(hist_g, color='g')
axs[1,0].set_title('Green Channel Histogram')

axs[1,1].plot(hist_b, color='b')
axs[1,1].set_title('Blue Channel Histogram')

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



Fotografia

Fotógrafos usam o histograma ao vivo para ajustar configurações da câmera e obter uma visão de como a luz foi capturada pelo sensor. Isso permite melhorar a exibição da imagem ao utilizar o canal de cor ou escala de cinza de forma ideal.

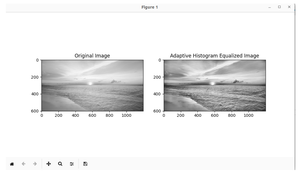
```
import cv2
from matplotlib import pyplot as plt

# load bright and dark images
img1 = cv2.imread('./images/dark.jpg', cv2.IMREAD_GRAYSCALE)
img2 = cv2.imread('./images/bright.jpeg', cv2.IMREAD_GRAYSCALE)

# create a figure for the subplots
fig, axs = plt.subplots(2, 2, figsize=(10, 10))

# show first image
axs[0, 0].imshow(img1, cmap='gray')
axs[0, 0].set_title('Image 1')

# show histogram of first image
axs[1, 0].hist(img1.ravel(), 256, [0, 256])
axs[1, 0].set_title('Histogram of Image 1')

# show second image
axs[0, 1].imshow(img2, cmap='gray')
axs[0, 1].set_title('Image 2')

# show histogram of second image
axs[1, 1].hist(img2.ravel(), 256, [0, 256])
axs[1, 1].set_title('Histogram of Image 2')

# space plots
plt.subplots_adjust(left=0.1, right=0.9, bottom=0.1, top=0.9, wspace=0.4, hspace=0.4)

plt.show()
```



Histograma Adaptativo

O histograma adaptativo é uma técnica de processamento de imagem que melhora o contraste e o brilho ajustando o histograma de forma local.

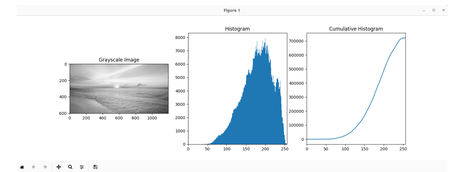
```
codes > 🐍 adaptivehist.py > ...
import cv2
import numpy as np
from matplotlib import pyplot as plt

img = cv2.imread('./images/sunset.jpg', 0)

# adaptive histogram
clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8,8))
clahe_img = clahe.apply(img)

# plot
fig, axs = plt.subplots(1, 2, figsize=(10, 5))
axs[0].imshow(img, cmap='gray')
axs[0].set_title('Original Image')
axs[1].imshow(clahe_img, cmap='gray')
axs[1].set_title('Adaptive Histogram Equalized Image')

plt.show()
```



Histograma Acumulativo

O histograma cumulativo é a soma dos valores do histograma até um determinado nível de intensidade, e é útil em operações de imagem como a equalização de histograma. É uma ferramenta útil em processamento de imagem para melhorar o contraste e ajustar o brilho.

```
codes > 🐍 cumulative_hist.py > ...
import cv2
import numpy as np
import matplotlib.pyplot as plt

img = cv2.imread('./images/sunset.jpg', cv2.IMREAD_GRAYSCALE)

# normal hist
hist, bins = np.histogram(img.ravel(), 256, [0, 256])

# cumulative hist
cumulative_hist = np.cumsum(hist)

# plotting
fig, axs = plt.subplots(1, 3, figsize=(15, 5))

axs[0].imshow(img, cmap='gray')
axs[0].set_title('Grayscale Image')

axs[1].hist(img.ravel(), 256, [0, 256])
axs[1].set_title('Histogram')

axs[2].plot(cumulative_hist)
axs[2].set_xlim([0, 256])
axs[2].set_title('Cumulative Histogram')

plt.show()
```



Equalização de Histograma

A equalização de histograma é uma técnica de processamento de imagem que busca obter uma distribuição uniforme das intensidades de pixel, melhorando o contraste e tornando detalhes mais visíveis.

```
codes > 🐍 equalization_hist.py > ...
import cv2
import matplotlib.pyplot as plt

# Load image in grayscale mode
img = cv2.imread('./images/sunset.jpg', cv2.IMREAD_GRAYSCALE)

# Apply histogram equalization
equalized_img = cv2.equalizeHist(img)

# Plot original and equalized image side by side
fig, axes = plt.subplots(nrows=1, ncols=2, figsize=(10, 5))
axes[0].imshow(img, cmap='gray')
axes[0].set_title('Original')
axes[1].imshow(equalized_img, cmap='gray')
axes[1].set_title('Equalized')
plt.show()
```