**《计算机视觉》实验报告**

**姓名：刘远航 学号：22121883**

实验三

1. **任务1** 
   1. **实验内容**

1. 读取一张图片，将其转换为HSV空间

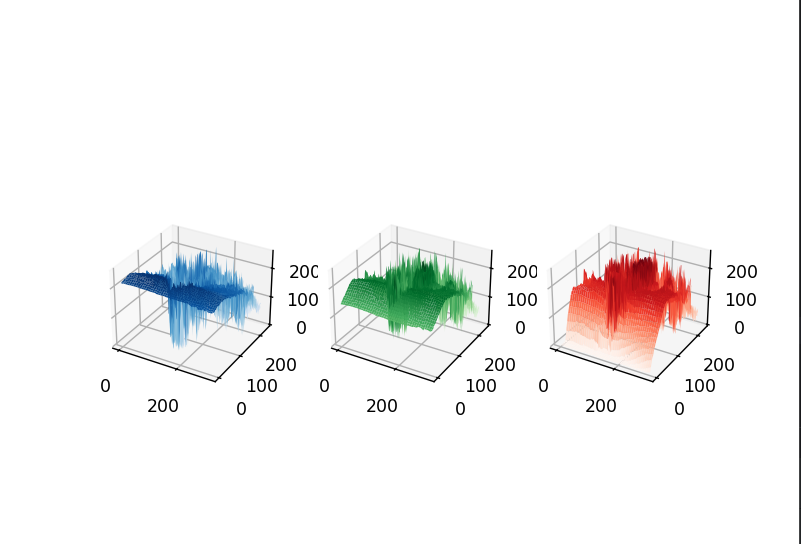
2. 分离原图片RGB通道及转换后的图片HSV通道

3. 对RGB三个通道分别画出其三维图（提示：polt\_sufface函数）

* 1. **核心代码：**

# 读取图片  
image = cv2.imread('image.jpg')  
# 将图片转换为HSV空间  
hsv\_image = cv2.cvtColor(image, cv2.COLOR\_BGR2HSV)  
# 分离原图RGB通道  
b, g, r = cv2.split(image)  
# 分离HSV图像的通道  
h, s, v = cv2.split(hsv\_image)  
# 获取图像尺寸  
height, width = image.shape[:2]  
x = np.linspace(0, width - 1, width)  
y = np.linspace(0, height - 1, height)  
X, Y = np.meshgrid(x, y)  
# 绘制RGB三维图  
fig = plt.figure()  
# 绘制蓝色通道的三维图  
ax1 = fig.add\_subplot(131, projection='3d')  
ax1.plot\_surface(X, Y, b, cmap='Blues')  
# 绘制绿色通道的三维图  
ax2 = fig.add\_subplot(132, projection='3d')  
ax2.plot\_surface(X, Y, g, cmap='Greens')  
# 绘制红色通道的三维图  
ax3 = fig.add\_subplot(133, projection='3d')  
ax3.plot\_surface(X, Y, r, cmap='Reds')  
plt.show()

* 1. **实验结果截图**

****

* 1. **实验小结**

通过任务一了解了RGB与HSV的区别，HSV更适用于图像处理，RGB更适用于工业，通过绘制三维图对图片有了更直观的了解。

1. **任务2** 
   1. **实验内容**

1、读取彩色图像home\_color；

2、画出灰度化图像home\_gray的灰度直方图，并拼接原灰度图与结果图；

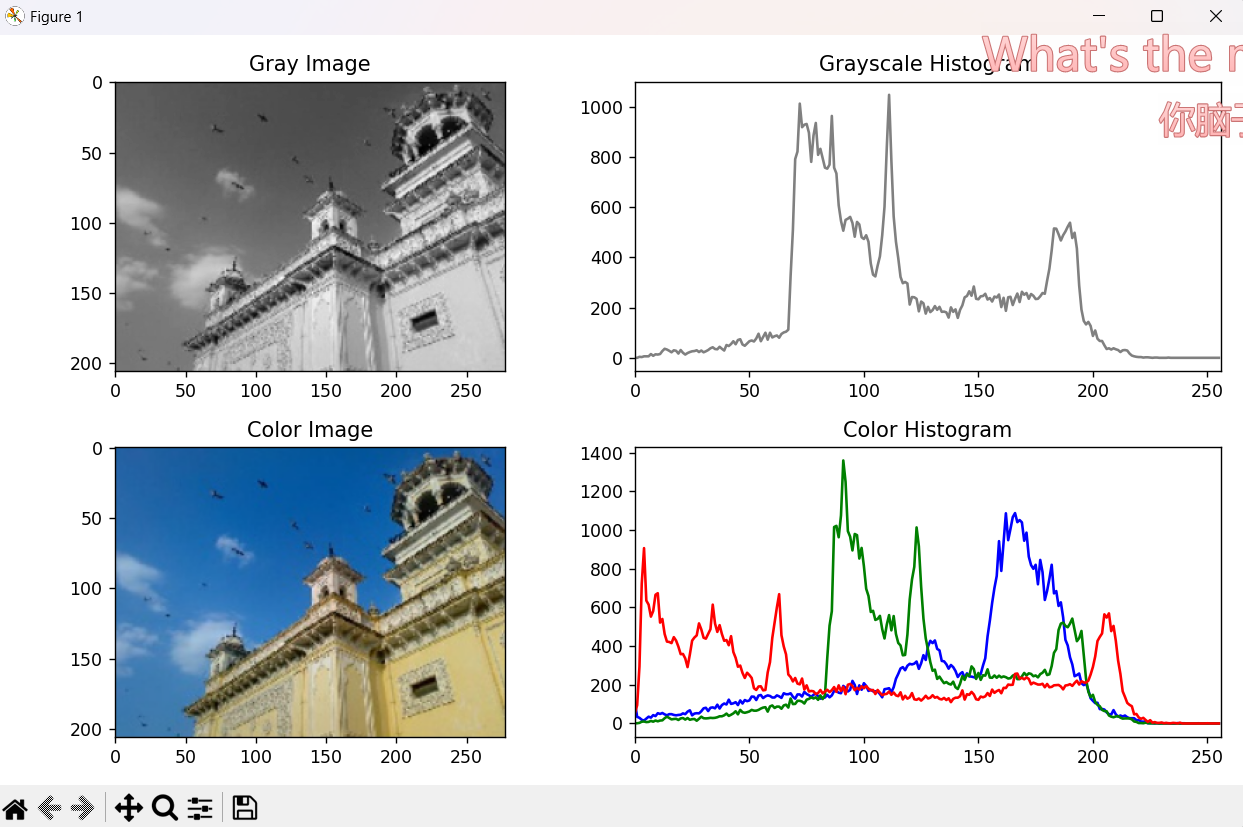
3、画出彩色home\_color图像的直方图，并拼接原彩色图与结果图，且与上一问结果放在同一个窗口中显示；

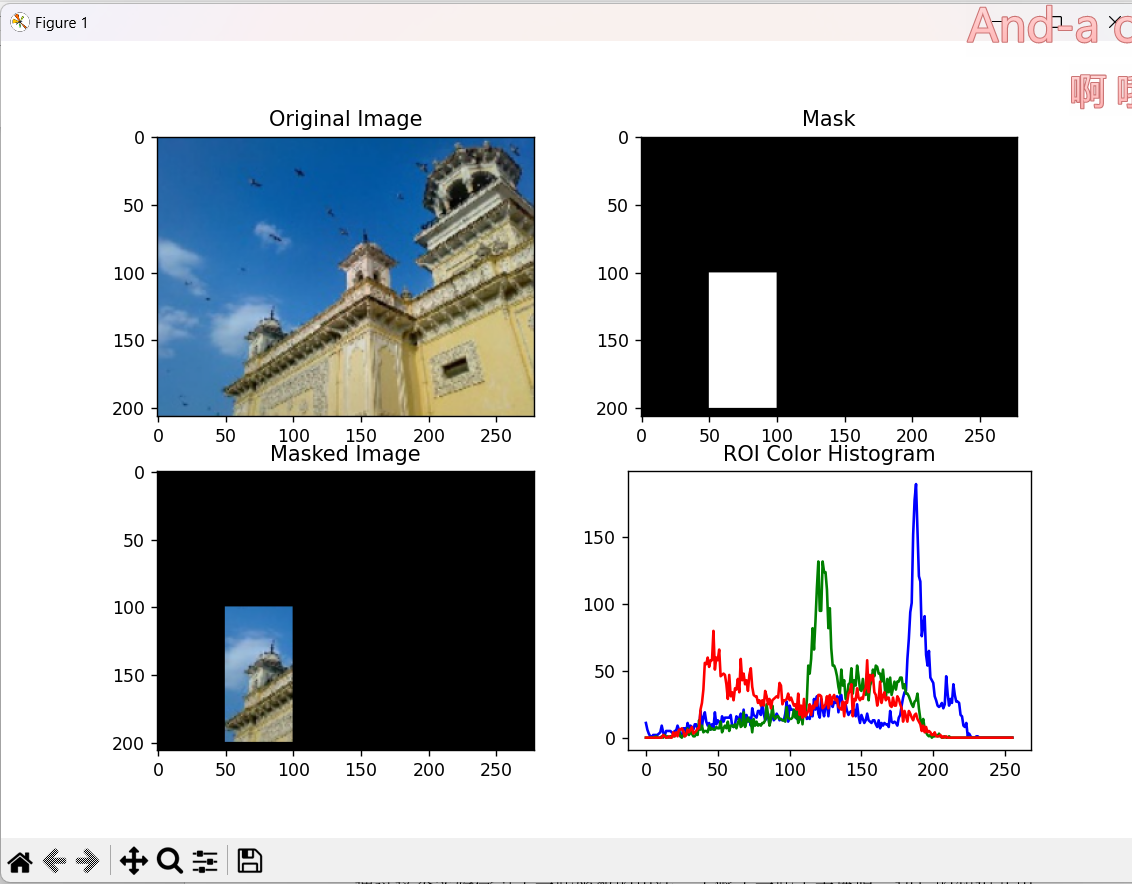
4、画出ROI（感兴趣区域 ）的直方图，ROI区域为 x：50-100，y：100-200，将原图home\_color，ROI的mask图，ROI提取后的图及其直方图放在一个窗口内显示。

* 1. **核心代码**

# 读取彩色图像  
home\_color = cv2.imread('image3.png')  
# 1. 灰度化图像  
home\_gray = cv2.cvtColor(home\_color, cv2.COLOR\_BGR2GRAY)  
# 2. 画出灰度化图像的直方图  
hist\_gray = cv2.calcHist([home\_gray], [0], None, [256], [0, 256])  
# 3. 画出彩色图像的直方图  
colors = ('b', 'g', 'r')  
hist\_color = []  
for i, color in enumerate(colors):  
 hist = cv2.calcHist([home\_color], [i], None, [256], [0, 256])  
 hist\_color.append(hist)  
# 绘制灰度直方图  
plt.figure(figsize=(10, 6)) ,plt.subplot(2, 2, 1),plt.imshow(home\_gray, cmap='gray')  
plt.title('Gray Image')  
plt.subplot(2, 2, 2), plt.plot(hist\_gray, color='gray')  
plt.title('Grayscale Histogram'),plt.xlim([0, 256])  
# 绘制彩色直方图  
plt.subplot(2, 2, 3)  
plt.imshow(cv2.cvtColor(home\_color, cv2.COLOR\_BGR2RGB))  
plt.title('Color Image')  
plt.subplot(2, 2, 4)  
for i, color in enumerate(colors):  
 plt.plot(hist\_color[i], color=color)  
plt.title('Color Histogram')  
plt.xlim([0, 256])  
plt.tight\_layout()  
plt.show()  
#ROI  
roi = home\_color[100:200, 50:100]  
mask = np.zeros(home\_color.shape[:2], np.uint8)  
mask[100:200, 50:100] = 255  
masked\_img = cv2.bitwise\_and(home\_color, home\_color, mask=mask)  
  
plt.figure(figsize=(12, 8)), plt.subplot(221)  
plt.imshow(cv2.cvtColor(home\_color, cv2.COLOR\_BGR2RGB)), plt.title('Original Image')  
plt.subplot(222), plt.imshow(mask, 'gray'), plt.title('Mask')  
plt.subplot(223)  
plt.imshow(cv2.cvtColor(masked\_img, cv2.COLOR\_BGR2RGB)), plt.title('Masked Image')  
color\_hist\_roi = []  
for i, col in enumerate(['b', 'g', 'r']):  
 hist = cv2.calcHist([roi], [i], mask[100:200, 50:100], [256], [0, 256])  
 color\_hist\_roi.append(hist)  
plt.subplot(224)  
for i, col in enumerate(['b', 'g', 'r']):  
 plt.plot(color\_hist\_roi[i], color=col)  
plt.title('ROI Color Histogram')  
plt.show()

* 1. **实验结果截图**

****



* 1. **实验小结**

通过这个实验学习了直方图的画法以及什么是直方图，对掩膜切片也有了更深的认识，在画直方图的时候也可以在三个通道画，对图像处理认识的更深了一些。

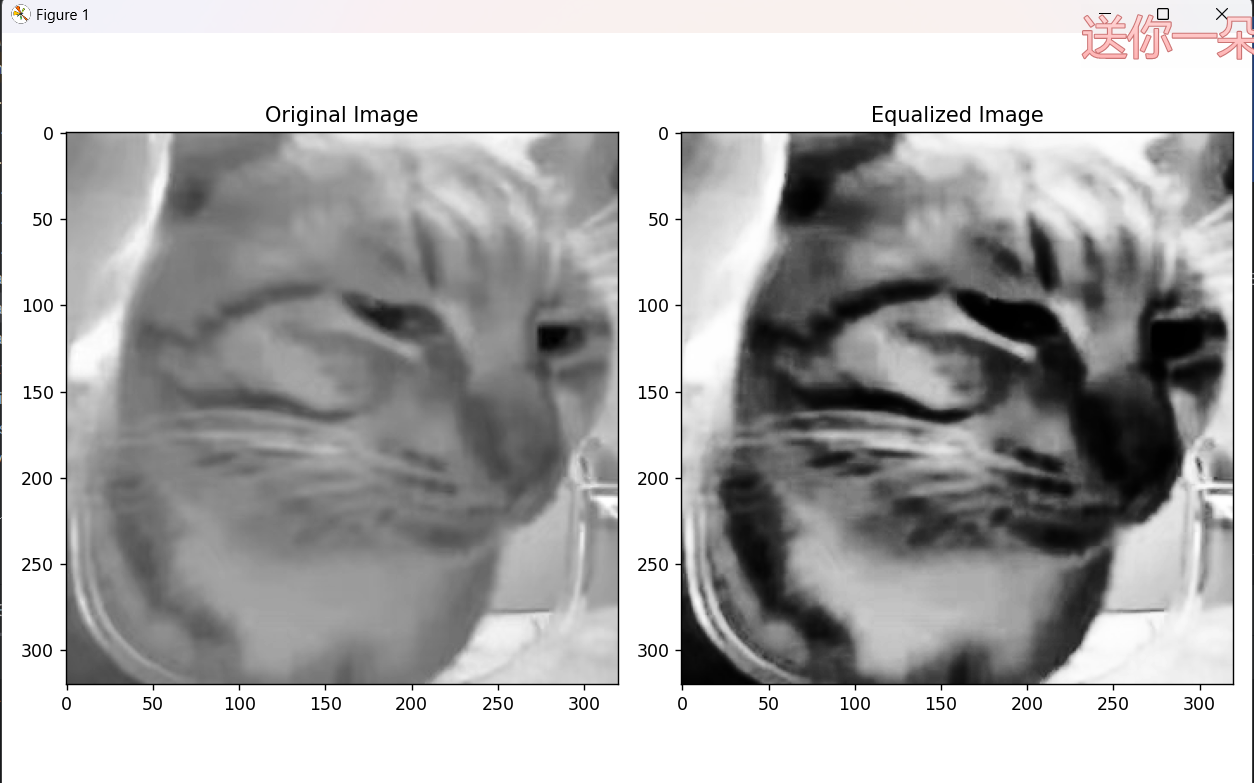
1. **任务3** 
   1. **实验内容**

1.编程实现直方图均衡化，给出测试效果

* 1. **核心代码**

def histogram\_equalization(image):  
 # 计算图像的直方图  
 hist, bins = np.histogram(image.flatten(), 256, [0, 256])  
  
 # 计算累积分布函数  
 cdf = hist.cumsum()  
 cdf\_normalized = cdf \* hist.max() / cdf.max()  
  
 # 使用累积分布函数进行均衡化  
 cdf\_masked = np.ma.masked\_equal(cdf, 0)  
 cdf\_masked = (cdf\_masked - cdf\_masked.min()) \* 255 / (cdf\_masked.max() - cdf\_masked.min())  
 cdf\_equalized = np.ma.filled(cdf\_masked, 0).astype('uint8')  
  
 # 将均衡化后的值映射回原始图像  
 equalized\_image = cdf\_equalized[image]  
  
 return equalized\_image

* 1. **实验结果截图**

****

* 1. **实验小结**

明白了什么叫直方图均衡化，是通过调整像素大小来使整个图像的像素分布更均匀，通过编程了解了均衡化的原理。