《计算机视觉》实验报告

姓名：严昕宇 学号：20121802

实验 3

一. 任务 1

a) 核心代码：

第一题

1. 读取一张图片，将其转换为HSV空间

[1]:
```
import cv2

CatImg_Path = "./Cat.png"
Cat_Img = cv2.imread(CatImg_Path, cv2.IMREAD_COLOR)
# 转换为HSV空间
HSV_CatImg = cv2.cvtColor(Cat_Img,cv2.COLOR_BGR2HSV)

cv2.imshow("HSV Cat", HSV_CatImg)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

2. 分离原图片RGB通道及转换后的图片HSV通道

方法一：OpenCV里自带了分离三通道的函数cv2.split()，返回值依次是蓝色、绿色和红色通道的灰度图

[2]:
```
# 分离原图片RGB通道
Blue_Img, Green_Img, Red_Img = cv2.split(Cat_Img) # 注意顺序是BGR

cv2.imshow("Blue", Blue_Img)
cv2.imshow("Green", Green_Img)
cv2.imshow("Red", Red_Img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

方法二：手动分离，获取三个通道的子矩阵

[3]:
```
# 方法二：手动分离，获取三个通道的子矩阵
BlueImg = Cat_Img[:,:,0]
GreenImg = Cat_Img[:,:,1]
RedImg = Cat_Img[:,:,2]

cv2.imshow("Blue", BlueImg)
cv2.imshow("Green", GreenImg)
cv2.imshow("Red", RedImg)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

方法三：可以只提取RGB图像中的某一个颜色，其他颜色可设为0（另外一种效果）

[4]:
```
BlueImg = Cat_Img.copy()
BlueImg[:,:,1]=0
BlueImg[:,:,2]=0

GreenImg = Cat_Img.copy()
GreenImg[:,:,0]=0
GreenImg[:,:,2]=0

RedImg = Cat_Img.copy()
RedImg[:,:,0]=0
RedImg[:,:,1]=0

cv2.imshow("Blue", BlueImg)
cv2.imshow("Green", GreenImg)
cv2.imshow("Red", RedImg)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

[5]:
```
# 分离转换后的图片HSV通道
Hue_Img, Saturation_Img, Value_Img = cv2.split(HSV_CatImg)
cv2.imshow("Hue", Hue_Img)
cv2.imshow("Saturation", Saturation_Img)
cv2.imshow("Value", Value_Img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

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

[6]:
```
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from matplotlib import cm
from matplotlib.ticker import LinearLocator, FormatStrFormatter
# 展示函数
def Display3D(img):
    fig = plt.figure(figsize=(16, 12))
    ax = fig.add_subplot(projection='3d')
    img_gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY) # 注意这里要将图片转化为灰度形式，不能为一维形式
    img_numpy = np.array(img_gray) # image类转numpy
    # 准备数据
    shape = img_numpy.shape
    h = int(shape[0]) # image height(rows)
    w = int(shape[1]) # image width(colums)
    x = np.arange(0, w, 1)
    y = np.arange(0, h, 1)
    x, y = np.meshgrid(x, y)
    z = img_numpy
    surf = ax.plot_surface(x, y, z, cmap=cm.coolwarm) # cmap指color map
    # 自定义 z 轴
    ax.set_zlim(-10, 260)
    ax.zaxis.set_major_locator(LinearLocator(10)) # z 轴网格线的疏密，刻度的疏密，20表示刻度的个数
    ax.zaxis.set_major_formatter(FormatStrFormatter('%.02f')) # 将z的value 字符串转为float，保留2位小数
    # 设置坐标轴的 label 和标题
    ax.set_xlabel('x', size=15)
    ax.set_ylabel('y', size=15)
    ax.set_zlabel('z', size=15)
    # 添加右侧的色卡条
    fig.colorbar(surf, shrink=0.6, aspect=8) # shrink表示整体收缩比例
    plt.show()
    return
Display3D(BlueImg)
Display3D(GreenImg)
Display3D(RedImg)
```

b) 实验结果截图

图 1 转换为 HSV 空间的图片

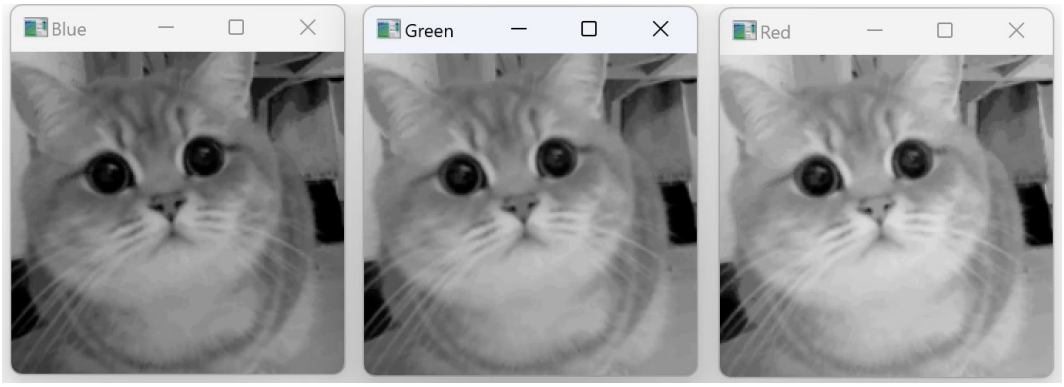

图 2 分离原图片 RGB 通道-1

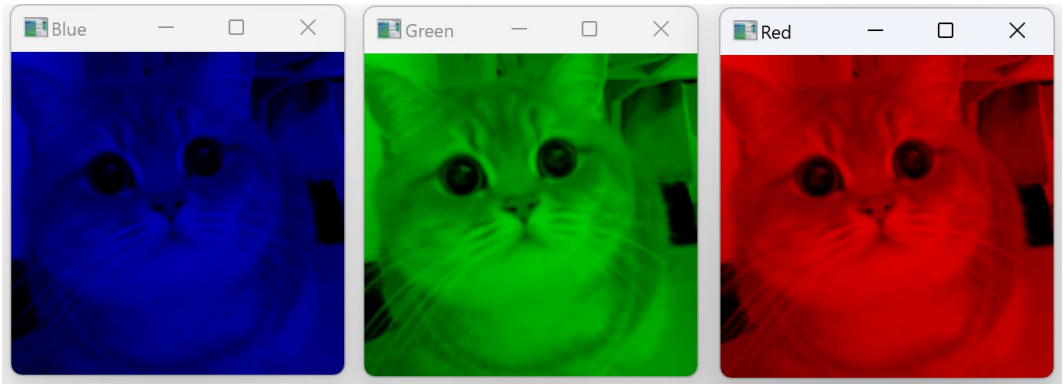

图 3 分离原图片 RGB 通道-2

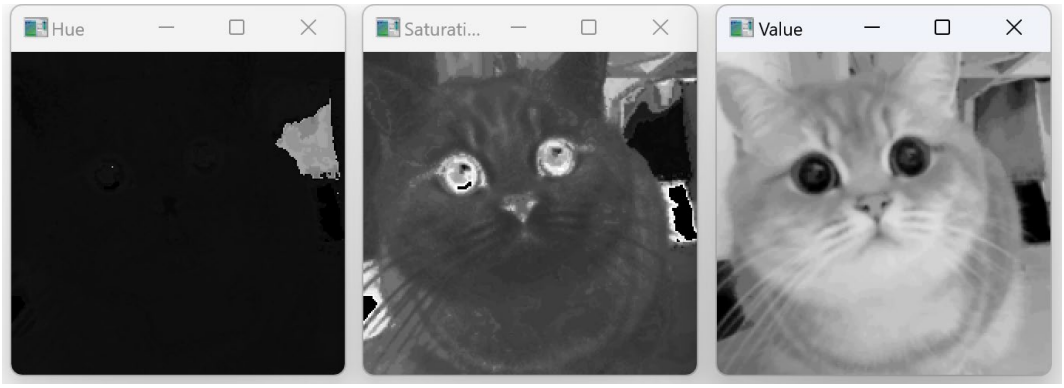

图 4 分离转换后的图片 HSV 通道

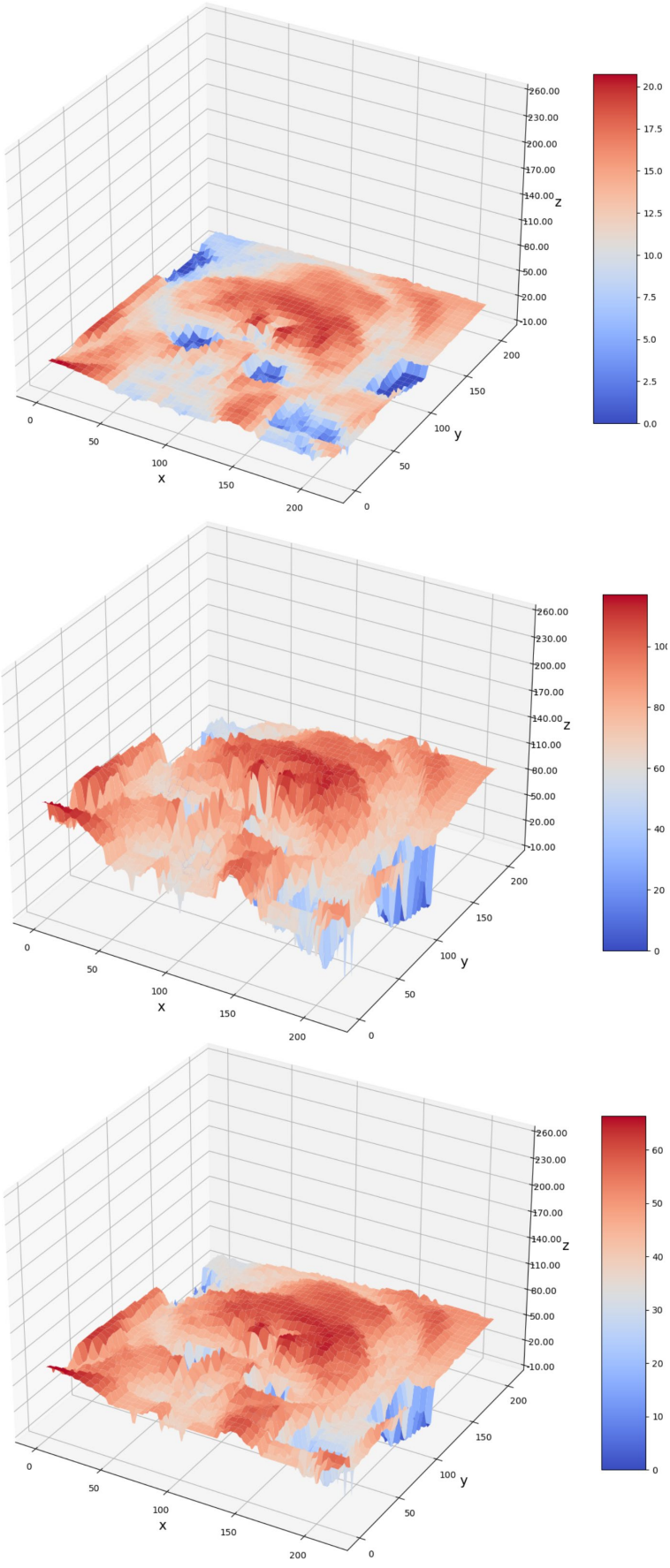

图 5 RGB 三个通道的三维图

c) 实验小结

颜色空间也称彩色模型(又称彩色空间或彩色系统)，它的用途是在某些标准下用通常可接受的方式对彩色加以说明。RGB 是由红色(R)，绿色(G)和蓝色(B)三个通道表示一幅图像。这三种颜色的不同组合可以形成几乎所有的其他颜色。但在图像处理中，使用较多的是 HSV 颜色空间，它更接近人们对彩色的感知经验，方便进行颜色的对比。

二. 任务 2

a) 核心代码：

第二题

1. 读取彩色图像home_color

[7]:
```
import cv2
import matplotlib.pyplot as plt
import numpy as np

HomeImg_Path = "./home_color.png"
Home_Img = cv2.imread(HomeImg_Path, cv2.IMREAD_COLOR)

cv2.imshow("home_color", Home_Img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

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

[8]:
```
Home_GrayImg = cv2.imread(HomeImg_Path, cv2.IMREAD_GRAYSCALE)

plt.subplot(121)
plt.title("home_gray")
plt.imshow(Home_GrayImg, cmap="gray")
# 灰度图像的直方图
plt.subplot(122)
gray_hist = cv2.calcHist([Home_GrayImg], [0], None, [256], [0, 256])
plt.plot(gray_hist)
plt.title("home_gray hist")
plt.show()
```

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

[9]:
```
plt.subplot(121)
plt.title("home_color")
plt.imshow(cv2.cvtColor(Home_Img, cv2.COLOR_BGR2RGB))
# 原图的直方图
plt.subplot(122)
color = ("b", "g", "r")
for i, col in enumerate(color):
    histr = cv2.calcHist([Home_Img], [i], None, [256], [0, 256])
    plt.plot(histr, color=col)
    plt.xlim([0, 256])
plt.title("home_color hist")

plt.show()
```

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。

[10]:
```
img = cv2.imread(HomeImg_Path)
img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
mask = np.zeros(img.shape[:2], np.uint8)

mask[50:100, 100:200] = 255  # ROI区域为x：50-100，y：100-200
mask_img = cv2.bitwise_and(img, img, mask=mask)

hist_full = cv2.calcHist([img], [0], None, [256], [0, 256])
hist_mask = cv2.calcHist([img], [0], mask, [256], [0, 256])

plt.figure()
plt.subplot(221)
plt.imshow(img, "gray")
plt.subplot(222)
plt.imshow(mask, "gray")
plt.subplot(223)
plt.imshow(mask_img, "gray")
plt.subplot(224)
plt.plot(hist_full)
plt.plot(hist_mask)
plt.xlim([0, 256])
plt.show()
```

b) 实验结果截图

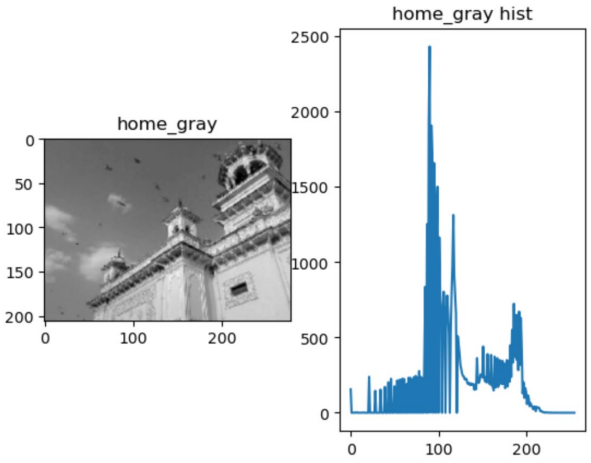

图 6 灰度化图像 home_gray 及其直方图

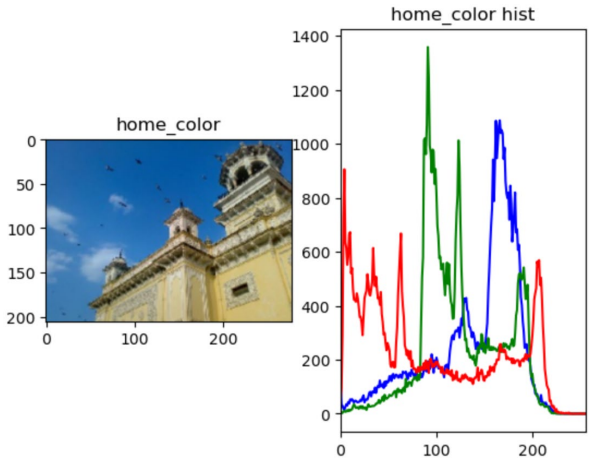

图 7 彩色图像 home_color 及其直方图

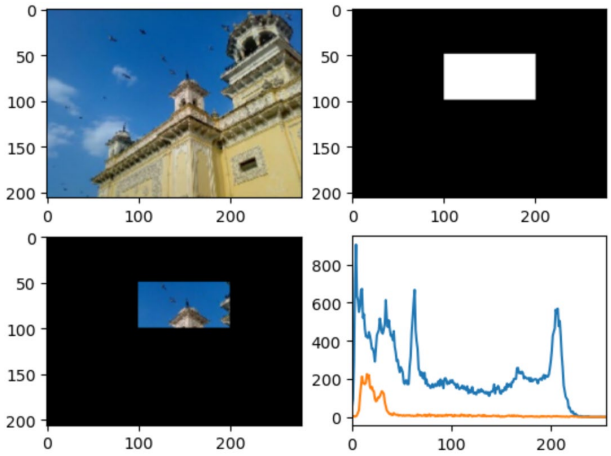

图 8 ROI 的 Mask 图、ROI 提取后的图及其直方图

c) 实验小结

直方图是可以对整幅图的灰度分布进行整体了解的图示，通过直方图我们可以对图像的对比度、亮度和灰度分布等有一个直观了解。计算直方图可用多种函数，如下：

- 使用 OpenCV 中的函数 cv2.calcHist(images, channels, mask, histSize, ranges):
- 使用 numpy 的函数 np.bincount():
- 使用 matplotlib 自带的绘制工具 plt.hist()绘制。

三. 任务 3

a) 核心代码：

1. 编程实现直方图均衡化，给出测试效果

[11]:
```
import cv2
import matplotlib.pyplot as plt

HomeImg_Path = "./home_color.png"
Home_Img = cv2.imread(HomeImg_Path, cv2.IMREAD_COLOR)
B, G, R = cv2.split(Home_Img)  # src参数必须是8比特的单通道图像
Equal_B = cv2.equalizeHist(B)
Equal_G = cv2.equalizeHist(G)
Equal_R = cv2.equalizeHist(R)
Equal = cv2.merge((Equal_B, Equal_G, Equal_R))  # merge
cv2.imshow("Home_Img", Home_Img)
cv2.imshow("Equal_Home_Img", Equal)
cv2.waitKey(0)
cv2.destroyAllWindows()

# 直方图对比
Hist_EqualB=cv2.calcHist([Equal_B],[0],None,[256],[0,256])
Hist_EqualG=cv2.calcHist([Equal_G],[0],None,[256],[0,256])
Hist_EqualR=cv2.calcHist([Equal_R],[0],None,[256],[0,256])
Hist_B=cv2.calcHist([B],[0],None,[256],[0,256])
plt.plot(Hist_EqualB,'b');
plt.plot(Hist_B,'r');
plt.show()
```

b) 实验结果截图

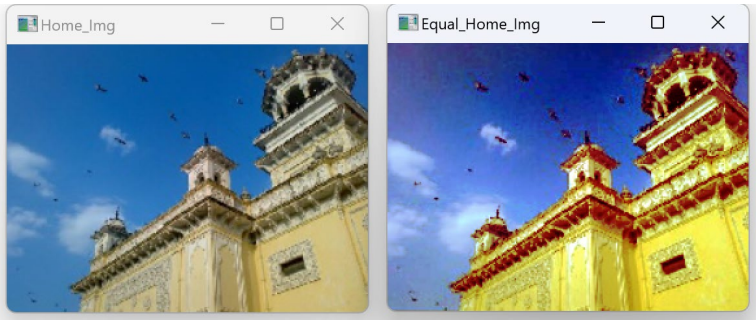

图 9 直方图均衡化结果

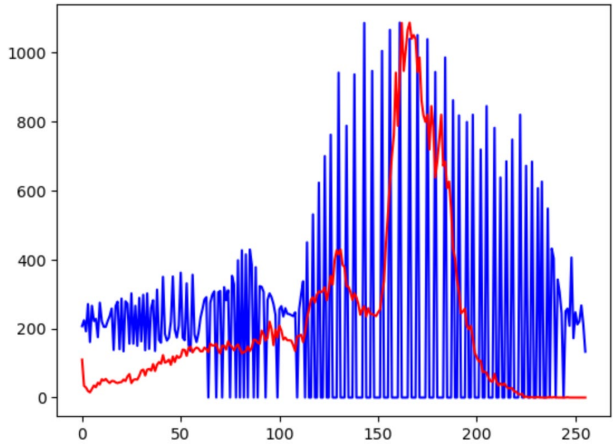

图 10 直方图对比

c) 实验小结

直方图优化算法的基本思想是把原始图像的灰度统计直方图变换成均匀分布的形式，这样就增强了像素灰度值的动态范围，从而达到增强图像整体对比度的效果。