**ĐỀ THAM KHẢO**

**test full 5.**  
Đọc ảnh mầu the\_cancuoc\_congdan.jpg vào biến ma trận I.  
1. (2 điểm) Hiển thị kênh B của ảnh I.  
2. (4 điểm) Chuyển ảnh mầu I sang ảnh đa cấp xám (grayscale) theo công thức xác định mức độ xám từ tổ hợp các thành phần mầu (r,g,b) theo tỷ lệ (0.39,0.5,0.11),  
được ma trận ảnh Ig. Hiển thị ảnh Ig.  
3. (1 điểm). Chuyển ảnh Ig sang ảnh nhị phân Ib với ngưỡng Otsu.  
4. (1.5 điểm). Làm trơn ảnh Ig theo bộ lọc trung bình cộng với lân cận cửa sổ kích thước 5x5 thu được ảnh Im.  
5. (1.5 điểm). Xác định các contour của ảnh Im và vẽ vị trí các contour lên ảnh gốc I ban đầu.

import cv2  
  
path = r'C:\Users\vuduc\OneDrive\Pictures\ao8.jpg'  
I = cv2.imread(path)  
cv2.imshow('anh goc', I)  
  
B,G,R = cv2.split(I)  
cv2.imshow('kenh B', B)  
  
#Chuyển ảnh màu sang xám với tỷ lệ RGB là 0.39, 0.5, 0.11  
Ig = cv2.cvtColor(I, cv2.COLOR\_BGR2GRAY)  
Ig = R\*0.39 + G\*0.5 + B\*0.11  
Ig = Ig.astype('uint8')  
cv2.imshow('anh xam', Ig)  
  
#Chuyển Ig sang nhị phân Ib với OTSU  
ret, Ib = cv2.threshold(Ig, 0, 255, cv2.THRESH\_OTSU)  
cv2.imshow('anh nhi phan', Ib)  
  
#Làm trơn ảnh bằng tb cộng  
Im = cv2.blur(Ig, (5,5))  
cv2.imshow('Im', Im)  
  
#Xác định contour của ảnh Im và vẽ lên ảnh gốc  
contours, hirarchy = cv2.findContours(Im, cv2.RETR\_TREE, cv2.CHAIN\_APPROX\_SIMPLE)  
  
cv2.drawContours(I, contours, -1, (0, 255, 0), 2)  
cv2.imshow('Contour', I)  
  
cv2.waitKey()

**test full 6**  
  
Đọc ảnh mầu **I04.jpg** vào biến ma trận **I**.  
1. (2 điểm) Hiển thị tỷ lệ giữa giá trị độ cao và độ rông của ảnh **I**.  
2. (4 điểm) Hiệu chỉnh lại ảnh I với size mới là độ cao 256, độ rộng 256 được ảnh mới **I2**. Hiển thị ảnh **I2**.  
3. (1 điểm) Chuyển đôi ảnh I sang ảnh HSV được ma trận ảnh **Ihsv**. Hiển thị kênh S của ảnh **Ihsv**.  
4. (1.5 điểm) Xác định biên theo phương pháp Canny của kênh V của ảnh **Ihsv** được ảnh nhị phân **Ivb**.  
5. (1.5 điểm) Xác định histogram của kênh S của ảnh **Ihsv**.

import cv2  
import numpy as np  
from matplotlib import pyplot as plt  
  
path = r'C:\Users\vuduc\OneDrive\Pictures\ao8.jpg'  
I = cv2.imread(path)  
  
height = I.shape[0]  
width = I.shape[1]  
print("Tỷ lệ giữa giá trị độ cao và độ rộng của ảnh: ", width/height)  
  
# 2. (4 điểm) Hiệu chỉnh lại ảnh I với size mới là độ cao 256, độ rộng 256 được ảnh mới I2. Hiển thị ảnh I2.  
I2 = cv2.resize(I, (256,256))  
cv2.imshow("anh chinh sua",I2)  
  
# 3. (1 điểm) Chuyển đôi ảnh I sang ảnh HSV được ma trận ảnh Ihsv. Hiển thị kênh S của ảnh Ihsv.  
Ihsv = cv2.cvtColor(I, cv2.COLOR\_BGR2HSV)  
H,S,V = cv2.split(Ihsv)  
cv2.imshow("kenh S", S)  
  
# 4. (1.5 điểm) Xác định biên theo phương pháp Canny của kênh V của ảnh Ihsv được ảnh nhị phân Ivb.  
Ivb = cv2.Canny(V, 0, 255)  
cv2.imshow("anh nhi phan",Ivb)  
cv2.waitKey()  
cv2.destroyAllWindows()  
  
# 5. (1.5 điểm) Xác định histogram của kênh S của ảnh Ihsv.  
  
hist\_S = cv2.calcHist([S], [0], None, [256], [0,256] )  
plt.plot(hist\_S)  
plt.show()  
  
cv2.waitKey()  
cv2.destroyAllWindows()

**test full 7**  
Đọc ảnh mầu watch.jpg vào biến ma trận I.  
1. (2 điểm) Hiển thị kênh B của ảnh I.  
2. (4 điểm) Chuyển ảnh sang biểu diễn HSV được ma trận Ihsv. Hiển thị kênh S của Ihsv. Xác định giá trị mức sáng trung bình của kênh V của ảnh Ihsv.  
3. (1 điểm) Làm trơn ảnh kênh S của Ihsv theo bộ lọc trung bình cộng, kích thước cửa sổ lân cận là 5x5 được ảnh Is. Hiển thị ảnh Is.  
4. (1.5 điểm) Nhị phân hóa ảnh Is theo ngưỡng Otsu được ảnh nhị phân Ib.  
Xác định đường contour có chu vi lớn nhất của ảnh Ib. Vẽ đường contour trên ảnh I.  
5. (1.5 điểm) Tăng độ sáng của kênh V của ảnh Ihsv bằng phương pháp giãn tuyến tính các giá trị mức xám. Biến đổi ngược ảnh Ihsv về biểu diễn mầu RGB được ảnh I. Hiển thị lại ảnh I.

import numpy as np  
from matplotlib import pyplot as plt  
import cv2  
  
path = r'C:\Users\vuduc\OneDrive\Pictures\ao8.jpg'  
I = cv2.imread(path)  
# 1. (2 điểm) Hiển thị kênh B của ảnh I.  
cv2.imshow("kenh B",I[:,:,0])  
# 2. (4 điểm) Chuyển ảnh sang biểu diễn HSV được ma trận Ihsv. Hiển thị kênh S của Ihsv. Xác định giá trị mức sáng trung bình của kênh V của ảnh Ihsv.  
Ihsv = cv2.cvtColor(I, cv2.COLOR\_RGB2HSV)  
cv2.imshow("kenh S",Ihsv[:,:,1])  
print("Mức sáng trung bình của Kênh V:", np.mean(Ihsv[:,:,2]))  
  
# 3. (1 điểm) Làm trơn ảnh kênh S của Ihsv theo bộ lọc trung bình cộng, kích thước cửa sổ lân cận là 5x5 được ảnh Is. Hiển thị ảnh Is.  
Is = cv2.blur(Ihsv[:,:,1],(5,5))  
cv2.imshow("lam tron kenh S",Is)  
  
# 4. (1.5 điểm) Nhị phân hóa ảnh Is theo ngưỡng Otsu được ảnh nhị phân Ib.  
thr, Ib = cv2.threshold(Is,0,255,cv2.THRESH\_OTSU)  
cv2.imshow("anh nhi phan",Ib)  
  
# Xác định đường contour có chu vi lớn nhất của ảnh Ib. Vẽ đường contour trên ảnh I.  
contours, \_ = cv2.findContours(Ib, cv2.RETR\_TREE, cv2.CHAIN\_APPROX\_SIMPLE)  
max\_per = 0.0  
for cnt in contours:  
 if max\_per <= cv2.arcLength(cnt, True):  
 max\_per = cv2.arcLength(cnt, True)  
 contour\_max = cnt  
cv2.drawContours(I, [contour\_max], -1, 255, 2)  
cv2.imshow("contour",I)  
  
# 5. (1.5 điểm) Tăng độ sáng của kênh V của ảnh Ihsv bằng phương pháp giãn tuyến tính các giá trị mức xám. Biến đổi ngược ảnh Ihsv về biểu diễn mầu RGB được ảnh I. Hiển thị lại ảnh I.  
def gian\_muc\_xam(Igray):  
 w=Igray.shape[1]  
 h=Igray.shape[0]  
 a=np.min(Igray)  
 b=np.max(Igray)  
 print([a,b])  
 aG=np.zeros(256,dtype='uint8')  
 for g in range(256):  
 aG[g]=(255\* g)//(b-a)  
 for i in range(h):  
 for j in range(w):  
 g=Igray[i][j]  
 Igray[i][j]=aG[g]  
 return a,b,Igray  
  
av,bv,Ihsv[:,:,2]=gian\_muc\_xam(Ihsv[:,:,2])  
  
I2 = cv2.cvtColor(Ihsv,cv2.COLOR\_HSV2RGB)  
cv2.imshow("tang do sang",I2)  
  
cv2.waitKey()  
cv2.destroyAllWindows()

**test full 8**  
Đọc ảnh mầu apple.jpg vào biến ma trận I.  
1. (2 điểm) Hiển thị kênh R của ảnh I.  
2. (4 điểm) Chuyển ảnh mầu I sang ảnh đa cấp xám (grayscale) theo công thức xác định mức độ xám từ tổ hợp các thành phần mầu (r,g,b) theo tỷ lệ (0.39,0.5,0.11), được ma trận ảnh Ig. Hiển thị ảnh Ig. Xác định mức xám trung bình của ảnh Ig.  
3. (1 điểm) Xác định ma trận gradient theo hướng x của Ig sử dụng toán tử Sobel và hiển thị ma trận kết quả.  
4. (1 điểm) Lấy biên của ảnh Ig theo phương pháp Canny được ảnh biên Ie là ảnh nhị phân nền đen. Kiểm tra các điểm ảnh của pixel có tọa độ dòng  y=100, cột x=120 có phải là điểm biên của Ig theo phương pháp dò biên Canny.  
5. (1.5 điểm). Nhị phân ảnh Ig theo ngưỡng Otsu được ảnh nhị phân nền đen Ib. Xác đường contour của ảnh Ib có diện tích lớn nhất. Vẽ đường contour tìm được trên lên ảnh gốc I.

import numpy as np  
import cv2  
from matplotlib import pyplot as plt  
  
path = r'C:\Users\vuduc\OneDrive\Pictures\ao8.jpg'  
I = cv2.imread(path)  
  
# 1. (2 điểm) Hiển thị kênh R của ảnh I.  
cv2.imshow("kenh R", I[:, :, 2])  
  
  
# 2. (4 điểm) Chuyển ảnh mầu I sang ảnh đa cấp xám (grayscale) theo công thức xác định mức độ xám từ tổ hợp các thành phần mầu (r,g,b) theo tỷ lệ (0.39,0.5,0.11), được ma trận ảnh Ig. Hiển thị ảnh Ig. Xác định mức xám trung bình của ảnh Ig.  
B,G,R = cv2.split(I)  
Ig = cv2.cvtColor(I, cv2.COLOR\_BGR2GRAY)  
Ig = R\*0.39 + G\*0.5 + B\*0.11  
Ig = Ig.astype('uint8')  
cv2.imshow('anh xam', Ig)  
cv2.imshow("anh Ig", Ig)  
print("muc xam trung binh cua anh Ig: ", np.mean(Ig))  
  
# 3. (1 điểm) Xác định ma trận gradient theo hướng x của Ig sử dụng toán tử Sobel và hiển thị ma trận kết quả.  
IgradientX = cv2.Sobel(Ig, cv2.CV\_64F, 1, 0, 3)  
plt.subplot(2, 2, 1), plt.imshow(IgradientX, cmap='gray')  
plt.title('Sobel X'), plt.xticks([]), plt.yticks([])  
plt.show()  
  
# 4. (1 điểm) Lấy biên của ảnh Ig theo phương pháp Canny được ảnh biên Ie là ảnh nhị phân nền đen. Kiểm tra các điểm ảnh của pixel có tọa độ dòng y=100, cột x=120 có phải là điểm biên của Ig theo phương pháp dò biên Canny.  
Ie = cv2.Canny(Ig, 0, 255)  
if Ie[100][120] == 255:  
 print("diem anh Igray(y=100,x=120) la diem bien theo Canny")  
else:  
 print("diem anh Igray(y=100,x=120) ko la điem bien theo Canny")  
  
# 5. (1.5 điểm). Nhị phân ảnh Ig theo ngưỡng Otsu được ảnh nhị phân nền đen Ib. Xác đường contour của ảnh Ib có diện tích lớn nhất. Vẽ đường contour tìm được trên lên ảnh gốc I.  
thr, Ib = cv2.threshold(Ig, 0, 255, cv2.THRESH\_OTSU)  
contours, \_ = cv2.findContours(Ib, cv2.RETR\_TREE, cv2.CHAIN\_APPROX\_SIMPLE)  
max\_area = 0.0  
contour\_max = []  
for cnt in contours:  
 if max\_area < cv2.contourArea(cnt):  
 max\_area = cv2.contourArea(cnt)  
 contour\_max = cnt  
  
cv2.drawContours(I, [contour\_max], -1, (0, 255, 0), 2)  
cv2.imshow("anh nhi phan", I)  
  
cv2.waitKey()  
cv2.destroyAllWindows();

**test full 9**  
Đọc ảnh mầu anh5.jpg vào biến ma trận I.  
1. (2 điểm) Hiển thị ảnh I.  
2. (4 điểm) Chuyển ảnh sang biểu diễn HSV được ma trận Ihsv. Hiển thị kênh S của Ihsv. Xác định giá trị mức sáng lớn nhất của kênh V của ảnh Ihsv.  
3. (1 điểm) Làm trơn ảnh kênh V của Ihsv theo bộ lọc trung bình cộng, kích thước cửa sổ lân cận là 3x3 được ảnh Iv. Hiển thị ảnh Iv.  
5. (1.5 điểm) Xác định đường contour có chu vi lớn nhất của ảnh Ib. Vẽ đường contour trên ảnh gốc I.  
6. (1.5 điểm) Tăng độ sáng của kênh V của ảnh Ihsv bằng phương pháp cân bằng histogram. Biến đổi ngược ảnh Ihsv về biểu diễn mầu RGB được ảnh I.

import cv2  
import numpy as np  
  
# Đọc ảnh vào biến ma trận I  
path = r'C:\Users\vuduc\OneDrive\Pictures\ao8.jpg'  
I = cv2.imread(path)  
  
# Hiển thị ảnh I  
cv2.imshow('I', I)  
cv2.waitKey(0)  
  
# Chuyển ảnh sang biểu diễn HSV và hiển thị kênh S  
Ihsv = cv2.cvtColor(I, cv2.COLOR\_BGR2HSV)  
cv2.imshow('S channel of Ihsv', Ihsv[:,:,1])  
cv2.waitKey(0)  
  
# Xác định giá trị mức sáng lớn nhất của kênh V của ảnh Ihsv  
max\_V = np.max(Ihsv[:,:,2])  
print('Max value of V channel: ', max\_V)  
  
# Làm trơn ảnh kênh V của Ihsv theo bộ lọc trung bình cộng, kích thước cửa sổ lân cận là 3x3  
Iv = cv2.blur(Ihsv[:,:,2], (3,3))  
  
# Hiển thị ảnh Iv  
cv2.imshow('Iv', Iv)  
cv2.waitKey(0)  
  
# Nhị phân hóa ảnh Iv thành ảnh Ib  
ret, Ib = cv2.threshold(Iv, 0, 255, cv2.THRESH\_BINARY+cv2.THRESH\_OTSU)  
  
# Tìm đường contour có chu vi lớn nhất của ảnh Ib  
contours, \_ = cv2.findContours(Ib, cv2.RETR\_TREE, cv2.CHAIN\_APPROX\_SIMPLE)  
max\_per = 0.0  
for cnt in contours:  
 if max\_per <= cv2.arcLength(cnt, True):  
 max\_per = cv2.arcLength(cnt, True)  
 contour\_max = cnt  
  
# Vẽ đường contour tìm được trên ảnh gốc I  
cv2.drawContours(I, [contour\_max], -1, (0,0,255), 2)  
cv2.imshow('I with largest contour', I)  
cv2.waitKey(0)  
  
# Tăng độ sáng của kênh V của ảnh Ihsv bằng phương pháp cân bằng histogram  
Ihsv[:,:,2] = cv2.equalizeHist(Ihsv[:,:,2])  
  
# Biến đổi ngược ảnh Ihsv về biểu diễn mầu RGB  
I = cv2.cvtColor(Ihsv, cv2.COLOR\_HSV2BGR)  
  
# Hiển thị ảnh I  
cv2.imshow('I after histogram equalization', I)  
cv2.waitKey(0)  
  
# Đóng tất cả các cửa sổ hiển thị  
cv2.destroyAllWindows()

**test full 10**  
Đọc ảnh mầu watch.jpg vào biến ma trận I.  
1. (2 điểm) Hiển thị kênh B của ảnh I.  
2. (4 điểm) Chuyển ảnh sang biểu diễn HSV được ma trận Ihsv. Hiển thị kênh S của Ihsv. Xác định giá trị mức sáng trung bình của kênh V của ảnh Ihsv.  
3. (1 điểm) Làm trơn ảnh kênh S của Ihsv theo bộ lọc trung bình cộng, kích thước cửa sổ lân cận là 5x5 được ảnh Is. Hiển thị ảnh Is.  
4. (1.5 điểm) Nhị phân hóa ảnh Is theo ngưỡng Otsu được ảnh nhị phân Ib.  
Xác định đường contour có chu vi lớn nhất của ảnh Ib. Vẽ đường contour trên ảnh I.  
5. (1.5 điểm) Tăng độ sáng của kênh V của ảnh Ihsv bằng phương pháp giãn tuyến tính các giá trị mức xám. Biến đổi ngược ảnh Ihsv về biểu diễn mầu RGB được ảnh I. Hiển thị lại ảnh I.

import cv2  
import numpy as np  
  
# Đọc ảnh mầu watch.jpg vào biến ma trận I.  
path = r'C:\Users\vuduc\OneDrive\Pictures\ao8.jpg'  
I = cv2.imread(path)  
  
# 1. (2 điểm) Hiển thị kênh B của ảnh I.  
cv2.imshow("1",I[:,:,2])  
  
# 2. (4 điểm) Chuyển ảnh sang biểu diễn HSV được ma trận Ihsv. Hiển thị kênh S của Ihsv. Xác định giá trị mức sáng trung bình của kênh V của ảnh Ihsv.  
Ihsv = cv2.cvtColor(I, cv2.COLOR\_BGR2HSV)  
print("Giá trị mức sáng trung bình của kênh V: ", np.mean(Ihsv[:,:,2]))  
  
# 3. (1 điểm) Làm trơn ảnh kênh S của Ihsv theo bộ lọc trung bình cộng, kích thước cửa sổ lân cận là 5x5 được ảnh Is. Hiển thị ảnh Is.  
Is = cv2.blur(Ihsv[:,:,1], (5,5))  
cv2.imshow("2",Is)  
  
# 4. (1.5 điểm) Nhị phân hóa ảnh Is theo ngưỡng Otsu được ảnh nhị phân Ib.Xác định đường contour có chu vi lớn nhất của ảnh Ib. Vẽ đường contour trên ảnh I.  
ret, Ib = cv2.threshold(Is, 0, 255, cv2.THRESH\_OTSU)  
contours, \_ = cv2.findContours(Ib, cv2.RETR\_TREE, cv2.CHAIN\_APPROX\_SIMPLE)  
max\_per = 0.0  
for cnt in contours:  
 if cv2.arcLength(cnt, True) > max\_per:  
 max\_per = cv2.arcLength(cnt, True)  
 max\_cnt = cnt  
I\_copy = I.copy()  
cv2.drawContours(I\_copy, [max\_cnt], -1 , (255,0,255), 3)  
cv2.imshow("3",I\_copy)  
  
# 5. (1.5 điểm) Tăng độ sáng của kênh V của ảnh Ihsv bằng phương pháp giãn tuyến tính các giá trị mức xám. Biến đổi ngược ảnh Ihsv về biểu diễn mầu RGB được ảnh I. Hiển thị lại ảnh I.  
def gian\_muc\_xam(Igray):  
 h = Igray.shape[0]  
 w = Igray.shape[1]  
 a = np.min(I)  
 b = np.max(I)  
  
 aG = np.zeros(256, dtype= 'uint8')  
 for i in range(256):   
 aG[i] = (256 \* i)   
 for i in range(h):  
 for j in range(w):  
 g = Igray[i][j]  
 Igray[i][j] = aG[g]   
 return a,b,Igray  
  
av,bv, I[:,:,2] = gian\_muc\_xam(I[:,:,2])  
I\_new = cv2.cvtColor(Ihsv, cv2.COLOR\_HSV2BGR)  
cv2.imshow("4",I\_new)  
cv2.waitKey()  
cv2.destroyAllWindows()

**test full 11**  
Đọc ảnh mầu apple.jpg vào biến ma trận I.  
1. (2 điểm) Hiển thị kênh R của ảnh I.  
2. (4 điểm) Chuyển ảnh mầu I sang ảnh đa cấp xám (grayscale) theo công thức xác định mức độ xám từ tổ hợp các thành phần mầu (r,g,b) theo tỷ lệ (0.39,0.5,0.11), được ma trận ảnh Ig. Hiển thị ảnh Ig. Xác định mức xám trung bình của ảnh Ig.  
3. (1 điểm) Xác định ma trận gradient theo hướng x của Ig sử dụng toán tử Sobel và hiển thị ma trận kết quả.  
4. (1 điểm) Lấy biên của ảnh Ig theo phương pháp Canny được ảnh biên Ie là ảnh nhị phân nền đen. Kiểm tra các điểm ảnh của pixel có tọa độ dòng  y=100, cột x=120 có phải là điểm biên của Ig theo phương pháp dò biên Canny.  
5. (1.5 điểm). Nhị phân ảnh Ig theo ngưỡng Otsu được ảnh nhị phân nền đen Ib. Xác đường contour của ảnh Ib có diện tích lớn nhất. Vẽ đường contour tìm được trên lên ảnh gốc I.

import numpy as np  
import cv2  
from matplotlib import pyplot as plt  
  
path = r'C:\Users\vuduc\OneDrive\Pictures\ao8.jpg'  
I = cv2.imread(path)  
  
# 1. (2 điểm) Hiển thị kênh R của ảnh I.  
cv2.imshow("kenh R", I[:, :, 2])  
  
  
# 2. (4 điểm) Chuyển ảnh mầu I sang ảnh đa cấp xám (grayscale) theo công thức xác định mức độ xám từ tổ hợp các thành phần mầu (r,g,b) theo tỷ lệ (0.39,0.5,0.11), được ma trận ảnh Ig. Hiển thị ảnh Ig. Xác định mức xám trung bình của ảnh Ig.  
B,G,R = cv2.split(I)  
Ig = cv2.cvtColor(I, cv2.COLOR\_BGR2GRAY)  
Ig = R\*0.39 + G\*0.5 + B\*0.11  
Ig = Ig.astype('uint8')  
cv2.imshow('anh xam', Ig)  
cv2.imshow("anh Ig", Ig)  
print("muc xam trung binh cua anh Ig: ", np.mean(Ig))  
  
# 3. (1 điểm) Xác định ma trận gradient theo hướng x của Ig sử dụng toán tử Sobel và hiển thị ma trận kết quả.  
IgradientX = cv2.Sobel(Ig, cv2.CV\_64F, 1, 0, 3)  
plt.subplot(2, 2, 1), plt.imshow(IgradientX, cmap='gray')  
plt.title('Sobel X'), plt.xticks([]), plt.yticks([])  
plt.show()  
  
# 4. (1 điểm) Lấy biên của ảnh Ig theo phương pháp Canny được ảnh biên Ie là ảnh nhị phân nền đen. Kiểm tra các điểm ảnh của pixel có tọa độ dòng y=100, cột x=120 có phải là điểm biên của Ig theo phương pháp dò biên Canny.  
Ie = cv2.Canny(Ig, 0, 255)  
if Ie[100][120] == 255:  
 print("diem anh Igray(y=100,x=120) la diem bien theo Canny")  
else:  
 print("diem anh Igray(y=100,x=120) ko la điem bien theo Canny")  
  
# 5. (1.5 điểm). Nhị phân ảnh Ig theo ngưỡng Otsu được ảnh nhị phân nền đen Ib. Xác đường contour của ảnh Ib có diện tích lớn nhất. Vẽ đường contour tìm được trên lên ảnh gốc I.  
thr, Ib = cv2.threshold(Ig, 0, 255, cv2.THRESH\_OTSU)  
contours, \_ = cv2.findContours(Ib, cv2.RETR\_TREE, cv2.CHAIN\_APPROX\_SIMPLE)  
max\_area = 0.0  
contour\_max = []  
for cnt in contours:  
 if max\_area < cv2.contourArea(cnt):  
 max\_area = cv2.contourArea(cnt)  
 contour\_max = cnt  
  
cv2.drawContours(I, [contour\_max], -1, (0, 255, 0), 2)  
cv2.imshow("anh nhi phan", I)  
  
cv2.waitKey()  
cv2.destroyAllWindows();

**test full 12**  
Đọc ảnh mầu hat1.png vào biến ma trận I.  
1. (2 điểm) Hiển thị kênh B của ảnh I.  
2. (4 điểm) Chuyển ảnh sang biểu diễn HSV được ma trận Ihsv. Hiển thị kênh H của Ihsv. Xác định giá trị mức sáng trung bình của kênh S của ảnh Ihsv.  
3. (1 điểm) Xác định và vẽ histogram của kênh S của ảnh Ihsv.  
4. (1.5 điểm) Làm trơn ảnh kênh V của Ihsv theo bộ lọc trung bình cộng, kích thước cửa sổ lân cận là 3x3 được ảnh Is. Hiển thị các giá trị mức xám trong lân cận 5x5 của điểm ảnh có tọa độ dòng y=9, cột x=11.  
5. (1.5 điểm) Nhị phân hóa ảnh Is theo ngưỡng Otsu được ảnh nhị phân Ib.  
Xác định đường contour có chu vi lớn nhất của ảnh Ib. Vẽ đường contour đó trên ảnh gôc I.

import cv2  
import numpy as np  
from matplotlib import pyplot as plt  
  
path = r'C:\Users\vuduc\OneDrive\Pictures\ao8.jpg'  
I = cv2.imread(path)  
cv2.imshow("kenh B", I[:,:,0])  
  
Ihsv = cv2.cvtColor(I, cv2.COLOR\_BGR2HSV)  
H,S,V = cv2.split(Ihsv)  
cv2.imshow("kenh H cua anh Ihsv", H)  
print("mức sáng trung bình của kênh S ảnh Ihsv:",np.mean(S))  
  
hist\_S = cv2.calcHist([S], [0], None, [256], [0,256] )  
plt.plot(hist\_S)  
plt.show()  
  
Is = cv2.blur(Ihsv[:,:,1], (3,3))  
height = Is.shape[0]  
width = Is.shape[1]  
y=100  
x=150  
for k in range(-1,6):  
 for l in range(-1,6):  
 if ((y + k )>= 0 & ((y + k) <= height - 1)) & ((x + l)>=0) &((x + l ) <= width - 1):  
 print("Các mức độ xám của của sổ lân cận 7x7 điểm ảnh (y="+str(y)+",x="+str(x)+")",Ihsv[y + k, x + l])  
  
\_, Ib = cv2.threshold(Is, 0, 255, cv2.THRESH\_OTSU)  
contours,\_ = cv2.findContours(Ib, cv2.RETR\_TREE, cv2.CHAIN\_APPROX\_SIMPLE)  
max = 0.0  
max\_contours = []  
for cnt in contours:  
 if max <= cv2.arcLength(cnt,True):  
 max = cv2.arcLength(cnt,True)  
 max\_contours = cnt  
I\_copy = I.copy()  
cv2.drawContours(I\_copy, [max\_contours], -1, 255, 3)  
cv2.imshow("I",I\_copy)  
  
cv2.waitKey()  
cv2.destroyAllWindows()

**test full 13**  
Đọc ảnh mầu coins.jpg vào biến ma trận I.  
1. (2 điểm) Hiển thị ảnh I và độ cao, độ rộng của ảnh.  
2. (4 điểm) Chuyển ảnh mầu I sang ảnh đa cấp xám (grayscale) theo công thức xác định mức độ xám từ tổ hợp các thành phần mầu (r,g,b) theo tỷ lệ (0.39,0.5,0.11), được ma trận ảnh Ig. Hiển thị ảnh Ig. Xác định mức xám trung bình của ảnh Ig.  
3. (1 điểm) Hiển thị các độ xám của của cửa sổ lân cận 5x5 của pixel có tọa độ dòng y=109, cột x=130 của ảnh Ig.  
4. (1.5 điểm) Lấy biên của ảnh Ig theo phương pháp Canny được ảnh biên Ie là ảnh nhị phân nền đen. Hiển thị ảnh Ie. Kiểm tra các điểm ảnh của pixel có tọa độ dòng  y=109, cột x=130 có phải là điểm biên của Ig theo phương pháp dò biên Canny.  
5. (1.5 điểm). Nhị phân ảnh Ig theo ngưỡng Otsu được ảnh nhị phân nền đen Ib. Xác định các đường contour của ảnh Ib gần tương tự với đường tròn. Vẽ các đường contour trên lên ảnh gốc I. Hiển thị ảnh I.

import cv2  
import numpy as np  
from matplotlib import pyplot as plt  
  
# Đọc ảnh mầu coins.jpg vào biến ma trận I.  
path = r'C:\Users\vuduc\OneDrive\Pictures\ao8.jpg'  
I = cv2.imread(path)  
  
# 1. (2 điểm) Hiển thị ảnh I và độ cao, độ rộng của ảnh.  
H = I.shape[0]  
W = I.shape[1]  
print("Do cao: ",H)  
print("Do rong: ",W)  
  
# 2. (4 điểm) Chuyển ảnh mầu I sang ảnh đa cấp xám (grayscale) theo công thức xác định mức độ xám từ tổ hợp các thành phần mầu (r,g,b) theo tỷ lệ (0.39,0.5,0.11), được ma trận ảnh Ig. Hiển thị ảnh Ig. Xác định mức xám trung bình của ảnh Ig.  
B,G,R = cv2.split(I)  
Ig = cv2.cvtColor(I, cv2.COLOR\_BGR2GRAY)  
Ig = R\*0.39 + G\*0.5 + B\*0.11  
Ig = Ig.astype('uint8')  
cv2.imshow('anh xam', Ig)  
cv2.imshow("anh Ig", Ig)  
print("muc xam trung binh cua anh Ig: ", np.mean(Ig))  
  
# 3. (1 điểm) Hiển thị các độ xám của của cửa sổ lân cận 5x5 của pixel có tọa độ dòng y=109, cột x=130 của ảnh Ig.  
height = Ig.shape[0]  
width = Ig.shape[1]  
y = 109  
x = 130  
for k in range(-1, 4):  
 for l in range(-1, 4):  
 if ((y + k) >= 0) & ((y + k) <= height - 1) & ((x + l) >= 0) & ((x + l) <= width - 1):  
 print("Các mức độ xám của của sổ lân cận 5x5 điểm ảnh Igray(y=109,x=130)",Ig[y + k, x + l])  
  
# . (1.5 điểm) Lấy biên của ảnh Ig theo phương pháp Canny được ảnh biên Ie là ảnh nhị phân nền đen. Hiển thị ảnh Ie. Kiểm tra các điểm ảnh của pixel có tọa độ dòng y=109, cột x=130 có phải là điểm biên của Ig theo phương pháp dò biên Canny.  
Ie = cv2.Canny(Ig, 0, 255)  
if Ie[109][130] == 255:  
 print("Điểm ảnh Igray(y=109,x=130) là điểm điểm biên theo Canny")  
else:  
 print("Điểm ảnh Igray(y=109,x=130) không là điểm điểm biên theo Canny")  
  
# 5. (1.5 điểm). Nhị phân ảnh Ig theo ngưỡng Otsu được ảnh nhị phân nền đen Ib. Xác định các đường contour của ảnh Ib gần tương tự với đường tròn. Vẽ các đường contour trên lên ảnh gốc I. Hiển thị ảnh I.  
I\_copy = I.copy()  
\_, Ib = cv2.threshold(Ig, 0, 255, cv2.THRESH\_OTSU)  
coutours, \_ = cv2.findContours(Ib, cv2.RETR\_TREE, cv2.CHAIN\_APPROX\_SIMPLE)  
cv2.drawContours(I\_copy,coutours,-1, 255, 3)  
cv2.imshow("anh I",I\_copy)  
  
cv2.waitKey()  
cv2.destroyAllWindows()

**test full 14**  
Đọc ảnh mầu hat1.png vào biến ma trận I.  
1. (2 điểm) Hiển thị kênh ảnh B của I.  
2. (4 điểm) Chuyển ảnh sang biểu diễn HSV được ma trận Ihsv. Hiển thị kênh V của Ihsv. Xác định giá trị mức sáng trung bình của kênh S của ảnh Ihsv.  
3. (1 điểm) Xác định và vẽ histogram của kênh V của ảnh Ihsv.  
4. (1.5 điểm) Làm trơn ảnh kênh S của Ihsv theo bộ lọc trung bình cộng, kích thước cửa sổ lân cận là 5x5 được ảnh Is. Hiển thị ảnh Is.  
5. (1.5 điểm) Xác định đường contour có tỉ lệ giữa chu vi và diện tích là lớn nhất của ảnh Ib. Vẽ đường contour trên ảnh gôc I Hiển thị ảnh I

import numpy as np  
import cv2  
from matplotlib import pyplot as plt  
  
# Đọc ảnh mầu hat1.png vào biến ma trận I.  
path = r'C:\Users\vuduc\OneDrive\Pictures\ao8.jpg'  
I = cv2.imread(path)  
  
# 1. (2 điểm) Hiển thị kênh ảnh B của I.  
cv2.imshow("kenh B",I[:,:,2])  
  
# 2. (4 điểm) Chuyển ảnh sang biểu diễn HSV được ma trận Ihsv. Hiển thị kênh V của Ihsv. Xác định giá trị mức sáng trung bình của kênh S của ảnh Ihsv  
Ihsv = cv2.cvtColor(I, cv2.COLOR\_RGB2HSV)  
H,S,V = cv2.split(Ihsv)  
cv2.imshow("kênh V",V)  
print("Mức sáng trung bình của Kênh S: ", np.mean(V))  
  
# 3. (1 điểm) Xác định và vẽ histogram của kênh V của ảnh Ihsv.  
  
hist\_V = cv2.calcHist([V], [0], None, [256], [0,256])  
plt.plot(hist\_V)  
plt.show()  
  
# 4. (1.5 điểm) Làm trơn ảnh kênh S của Ihsv theo bộ lọc trung bình cộng, kích thước cửa sổ lân cận là 5x5 được ảnh Is. Hiển thị ảnh Is.  
Is = cv2.blur(Ihsv[:,:,1],(5,5))  
cv2.imshow("kênh S",Is)  
  
# 5. (1.5 điểm) Xác định đường contour có tỉ lệ giữa chu vi và diện tích là lớn nhất của ảnh Ib. Vẽ đường contour trên ảnh gôc I Hiển thị ảnh I  
Ig = cv2.cvtColor(I, cv2.COLOR\_RGB2GRAY)  
\_, Ib = cv2.threshold(Ig, 0, 255, cv2.THRESH\_OTSU)  
coutours, \_ = cv2.findContours(Ib, cv2.RETR\_TREE, cv2.CHAIN\_APPROX\_SIMPLE)  
max = 0.0  
for cnt in coutours:  
 if cv2.arcLength(cnt, True) != 0 and (cv2.contourArea(cnt)) != 0:  
 ti\_so = (cv2.arcLength(cnt, True))/(cv2.contourArea(cnt))  
 if ti\_so > max:  
 max = ti\_so  
 contour\_max = cnt  
I\_copy = I.copy()  
cv2.drawContours(I\_copy,[contour\_max], -1, 255, 2)  
cv2.imshow("anh I",I\_copy)  
  
cv2.waitKey()  
cv2.destroyAllWindows()

**test full 15**  
Đọc ảnh mầu I04.jpg vào biến ma trận I.  
1. (2 điểm) Hiển thị tỷ lệ giữa giá trị độ cao và độ rông của ảnh I.  
2. (4 điểm) Hiệu chỉnh lại ảnh I với size mới là độ cao 256, ảnh giữ nguyên tỷ lệ so với ảnh gốc, được ảnh mới I2. Hiển thị ảnh I2.  
3. (1 điểm) Chuyển đôi ảnh I sang ảnh HSV được ma trận ảnh Ihsv. Hiển thị kênh S của ảnh Ihsv.  
4. (1.5 điểm) Làm trơn kênh S của ảnh Ihsv với bộ lọc median kích thước cửa sổ 3x3. Biến đổi ngược ảnh Ihsv về biểu diễn mầu RGB được ảnh I3. Hiển thị ảnh I3.  
5. (1.5 điểm) Cân bằng histogram của  kênh S của ảnh Ihsv. Biến đổi ngược ảnh Ihsv về biểu diễn mầu RGB được ảnh I4. Hiển thị ảnh I4.

import numpy as np  
import cv2  
from matplotlib import pyplot as plt  
  
# Đọc ảnh mầu I04.jpg vào biến ma trận I.  
path = r'C:\Users\vuduc\OneDrive\Pictures\ao8.jpg'  
I = cv2.imread(path)  
  
# 1. (2 điểm) Hiển thị tỷ lệ giữa giá trị độ cao và độ rông của ảnh I.  
h = I.shape[0]  
w = I.shape[1]  
print("tỉ lệ: ", h/w)  
  
  
# 2. (4 điểm) Hiệu chỉnh lại ảnh I với size mới là độ cao 256, ảnh giữ nguyên tỷ lệ so với ảnh gốc, được ảnh mới I2. Hiển thị ảnh I2.  
width = I.shape[1]  
I2 = cv2.resize(I, (width, 256))  
cv2.imshow('Resize',I2)  
  
# 3. (1 điểm) Chuyển đôi ảnh I sang ảnh HSV được ma trận ảnh Ihsv. Hiển thị kênh S của ảnh Ihsv.  
Ihsv = cv2.cvtColor(I, cv2.COLOR\_RGB2HSV)  
cv2.imshow("kenh S",Ihsv[:,:,1])  
  
# 4. (1.5 điểm) Làm trơn kênh S của ảnh Ihsv với bộ lọc median kích thước cửa sổ 3x3. Biến đổi ngược ảnh Ihsv về biểu diễn mầu RGB được ảnh I3. Hiển thị ảnh I3.  
Ihsv[:,:,1] = cv2.medianBlur(Ihsv[:,:,1], 3)  
I3 = cv2.cvtColor(Ihsv, cv2.COLOR\_HSV2RGB)  
cv2.imshow("anh I3",I3)  
  
# 5. (1.5 điểm) Cân bằng histogram của kênh S của ảnh Ihsv. Biến đổi ngược ảnh Ihsv về biểu diễn mầu RGB được ảnh I4. Hiển thị ảnh I4.  
Ihsv[:,:,1] = cv2.equalizeHist(Ihsv[:,:,1])  
I4 = cv2.cvtColor(Ihsv, cv2.COLOR\_HSV2RGB)  
cv2.imshow("anh I4",I4)  
cv2.waitKey()  
cv2.destroyAllWindows()

**test full 16**  
Đọc ảnh mầu **anh5.jpg** vào biến ma trận **I**.  
1. (2 điểm) Hiển thị ảnh là trung bình cộng của 2 kênh R và B của ảnh I.  
2. (4 điểm) Chuyển ảnh sang biểu diễn HSV được ma trận Ihsv. Hiển thị kênh H của Ihsv. Xác định giá trị mức sáng lớn nhất của kênh S của ảnh Ihsv.  
3. (1 điểm) Làm trơn ảnh kênh S của **Ihsv** theo bộ lọc trung bình cộng, kích thước cửa sổ lân cận là 7x7 được ảnh **Is**. Hiển thị ảnh **Is**.  
4. (1.5 điểm) Nhị phân hóa của ảnh **255**-**Is** theo ngưỡng Otsu được ảnh nhị phân **Ib**.  
Xác định đường contour có chu vi lớn nhất của ảnh **Ib**. Vẽ đường contour trên ảnh gôc **I** và hiển thị ảnh **I**.  
5. (1.5 điểm) Tăng độ sáng của kênh **V** của ảnh **Ihsv**bằng phương pháp giãn tuyến tính mức xám. Biến đổi ngược ảnh **Ihsv** về biểu diễn mầu RGB được ảnh **I** mới.

import cv2  
import numpy as np  
  
I = cv2.imread("cake.jpg")  
TBC = (I[:,:,2]+I[:,:,0]) / 2  
cv2.imshow("TBC cau 2 kenh R va B",TBC)  
  
Ihsv = cv2.cvtColor(I, cv2.COLOR\_BGR2HSV)  
cv2.imshow("kenh H", Ihsv[:,:,0])  
print("mức sáng lớn nhất kênh S:", np.max(Ihsv[:,:,1]))  
  
# 3. (1 điểm) Làm trơn ảnh kênh S của Ihsv theo bộ lọc trung bình cộng, kích thước cửa sổ lân cận là 7x7 được ảnh Is. Hiển thị ảnh Is.  
Is = cv2.blur(Ihsv[:,:,1], (7,7))  
cv2.imshow("anh Is",Is)  
  
# 4. (1.5 điểm) Nhị phân hóa của ảnh 255-Is theo ngưỡng Otsu được ảnh nhị phân Ib.  
\_, Is = cv2.threshold(Is, 0, 255, cv2.THRESH\_OTSU)  
Ib = 255 - Is  
cv2.imshow("anh nhi phan",Ib)  
  
# Xác định đường contour có chu vi lớn nhất của ảnh Ib. Vẽ đường contour trên ảnh gôc I và hiển thị ảnh I.  
coutours, \_ = cv2.findContours(Ib, cv2.RETR\_TREE, cv2.CHAIN\_APPROX\_SIMPLE)  
max = 0.0  
max\_per = []  
for cnt in coutours:  
 if cv2.arcLength(cnt, True) > max:  
 max = cv2.arcLength(cnt,True)  
 max\_per = cnt  
I\_copy = I.copy()  
cv2.drawContours(I\_copy,[cnt], -1, 255, 2)  
cv2.imshow("contours",I\_copy)  
  
# 5. (1.5 điểm) Tăng độ sáng của kênh V của ảnh Ihsv bằng phương pháp giãn tuyến tính mức xám. Biến đổi ngược ảnh Ihsv về biểu diễn mầu RGB được ảnh I mới.  
def gian\_muc\_xam(Igray):  
 w=Igray.shape[1]  
 h=Igray.shape[0]  
 a=np.min(Igray)  
 b=np.max(Igray)  
 print([a,b])  
 #chinh lai  
 aG=np.zeros(256,dtype='uint8')  
 for g in range(256):  
 aG[g] = (255\* g)//(b-a)  
 for i in range(h):  
 for j in range(w):  
 g=Igray[i][j]  
 Igray[i][j]=aG[g]  
 return a,b,Igray  
  
I[:,:,2],\_,\_ = gian\_muc\_xam(I[:,:,2])  
I\_new = cv2.cvtColor(Ihsv, cv2.COLOR\_HSV2BGR)  
cv2.imshow("anh moi",I\_new)  
cv2.waitKey()  
cv2.destroyAllWindows()

**test full 17**  
  
Đọc ảnh mầu **coins.jpg** vào biến ma trận **I**.  
1. (2 điểm) Hiển thị ảnh **I**.  
2. (4 điểm) Chuyển ảnh mầu **I** sang ảnh đa cấp xám (grayscale) theo công thức xác định mức độ xám từ tổ hợp các thành phần mầu (r,g,b) theo tỷ lệ (0.39,0.5,0.11), được ma trận ảnh **Ig**. Hiển thị ảnh **Ig**. Xác định mức xám trung bình của ảnh Ig.  
3. (1 điểm) Hiển thị các độ xám của của cửa sổ lân cận 5x5 của pixel có tọa độ dòng y=100, cột x=120 của ảnh **Ig**.  
4. (1.5 điểm) Lấy biên của ảnh Ig theo phương pháp Canny được ảnh biên **Ie** là ảnh nhị phân nền đen. Kiểm  
tra các điểm ảnh của pixel có tọa độ dòng  y=100, cột x=120 có phải là điểm biên của Ig theo phương pháp dò biên  
Canny.  
5. (1.5 điểm). Nhị phân ảnh **Ig** theo ngưỡng Otsu được ảnh nhị phân nền đen **Ib**. Xác định các đường contour của ảnh **Ib**. Vẽ các đường contour trên lên ảnh gốc **I**.

import cv2  
import numpy as np  
  
# 1. (2 điểm) Hiển thị ảnh I.  
path = r'C:\Users\vuduc\OneDrive\Pictures\ao8.jpg'  
I = cv2.imread(path)  
  
# 2. (4 điểm) Chuyển ảnh mầu I sang ảnh đa cấp xám (grayscale) theo công thức xác định mức độ xám từ tổ hợp các thành phần mầu (r,g,b) theo tỷ lệ (0.39,0.5,0.11), được ma trận ảnh Ig. Hiển thị ảnh Ig. Xác định mức xám trung bình của ảnh Ig.  
B,G,R = cv2.split(I)  
Ig = cv2.cvtColor(I, cv2.COLOR\_BGR2GRAY)  
Ig = R\*0.39 + G\*0.5 + B\*0.11  
Ig = Ig.astype('uint8')  
cv2.imshow('anh xam', Ig)  
cv2.imshow("anh Ig", Ig)  
print("muc xam trung binh cua anh Ig: ", np.mean(Ig))  
  
# 3. (1 điểm) Hiển thị các độ xám của của cửa sổ lân cận 5x5 của pixel có tọa độ dòng y=100, cột x=120 của ảnh Ig.  
height = Ig.shape[0]  
width = Ig.shape[1]  
y = 100  
x = 150  
for k in range(-1, 6):  
 for l in range(-1, 6):  
 if ((y + k) >= 0) & ((y + k) <= height - 1) & ((x + l) >= 0) & ((x + l) <= width - 1):  
 print("Các mức độ xám của của sổ lân cận 7x7 điểm ảnh Igray(y=100,x=120)",Ig[y + k, x + l])  
  
# 4. (1.5 điểm) Lấy biên của ảnh Ig theo phương pháp Canny được ảnh biên Ie là ảnh nhị phân nền đen. Hiển thị ảnh Ie. Kiểm tra  
Ie = cv2.Canny(Ig, 0, 255)  
if Ie[109][130] == 255:  
 print('Điểm ảnh của pixel có tọa độ dòng y=109, cột x=130 là điểm biên của Ig theo phương pháp dò biên Canny. ')  
else:  
 print('Điểm ảnh của pixel có tọa độ dòng y=109, cột x=130 không phải là điểm biên của Ig theo phương pháp dò biên Canny. ')  
  
# 5. (1.5 điểm). Nhị phân ảnh Ig theo ngưỡng Otsu được ảnh nhị phân nền đen Ib. Xác định các đường contour của ảnh Ib gần tương tự với đường tròn. Vẽ các đường contour trên lên ảnh gốc I. Hiển thị ảnh I.  
I\_copy = I.copy()  
\_,Ib = cv2.threshold(Ig, 0, 255, cv2.THRESH\_OTSU)  
contours, \_ = cv2.findContours(Ib, cv2.RETR\_TREE,cv2.CHAIN\_APPROX\_SIMPLE)  
cv2.drawContours(I\_copy, contours, -1, 255, 3)  
cv2.imshow("contours",I\_copy)  
cv2.waitKey()  
cv2.destroyAllWindows()