第一题

代码：

import cv2  
import numpy as np  
from matplotlib import pyplot as plt  
import math  
  
#找黑白棋子  
img\_rgb = cv2.imread(&apos;.. eckerboard .png&apos;)  
img\_gray = cv2.cvtColor(img\_rgb, cv2.COLOR\_BGR2GRAY)  
template = cv2.imread(&apos;../white.jpg&apos;,0)  
template1 = cv2.imread(&apos;..ack.jpg&apos;,0)  
w, h = template.shape[::-1]  
res = cv2.matchTemplate(img\_gray,template,cv2.TM\_CCOEFF\_NORMED)#模块匹配  
threshold =0.8#设置阈值  
loc1 = np.where( res >= threshold)#参数为条件、x、y    当匹配的模块大于等于阈值时，记录位置   返回的是两个数组，第一个是行信息，第二个是列信息  
for pt in zip(\*loc1[::-1]):#当满足条件时，画一个矩形   每读到一个记录的位置时，画一个矩形  
    cv2.rectangle(img\_rgb, pt, (pt[0] + w, pt[1] + h), (0,255,0), 1)  
  
w, h = template1.shape[::-1]  
res = cv2.matchTemplate(img\_gray,template1,cv2.TM\_CCOEFF\_NORMED)#模块匹配  
threshold =0.8#设置阈值  
loc2 = np.where( res >= threshold)#参数为判断条件、x、y，true取下，false取y   当匹配的模块大于等于阈值时，记录位置   返回的是两个数组，第一个是行信息，第二个是列信息  
for pt in zip(\*loc2[::-1]):#当满足条件时，画一个矩形   每读到一个记录的位置时，画一个矩形  
    cv2.rectangle(img\_rgb, pt, (pt[0] + w, pt[1] + h), (0,0,255), 1)  
  
#找最值  
max1=0  
for x in zip(\*loc1[::-1]):  
    x1,y1=x#注意，此处返回的是一个坐标，即每个数组的相应位置  
    for x2 in zip(\*loc1[::-1]):  
        x3,y3=x2  
        a = (x1 - x3) \* (x1 - x3) + (y1 - y3) \* (y1 - y3)  
        if a>max1:  
            max1=a  
print(math.sqrt(max1))  
  
max2=0  
for x in zip(\*loc2[::-1]):  
    x1,y1=x  
    for x2 in zip(\*loc2[::-1]):  
        x3,y3=x2  
        a = (x1 - x3) \* (x1 - x3) + (y1 - y3) \* (y1 - y3)  
        if a>max2:  
            max2=a  
print(math.sqrt(max2))  
  
#画距离最大的线  
max3=0  
for x in zip(\*loc1[::-1]):  
    x1,y1=x  
    for x2 in zip(\*loc2[::-1]):  
        x3,y3=x2  
        a=(x1-x3)\*(x1-x3)+(y1-y3)\*(y1-y3)  
        if a>max3:  
            max3=a  
            maxx1=x1  
            maxx3=x3  
            maxy1=y1  
            maxy3=y3  
cv2.line(img\_rgb,(maxx1+15,maxy1+15),(maxx3+15,maxy3+15),(255,0,0),3)  
  
#霍夫变换，检测黑线  
img=cv2.imread(&apos;.. eckerboard .png&apos;)  
img2=img.copy()  
gray=cv2.cvtColor(img,cv2.COLOR\_BGR2GRAY)  
edges=cv2.Canny(gray,50,150,apertureSize=3)#边缘检测  
lines=cv2.HoughLines(edges,1,np.pi/180,200)#返回值是（ρ,θ）。ρ是点到原点的距离，其单位是像素，θ的单位是弧度。第一个参数是一个二值化图像，所以在进行霍夫变换之前要首先进行二值化，或者进行Canny边缘检测。  
# 第二和第三个值分别代表ρ和θ的精确度。第四个参数是阈值，只有累加其中的值高于阈值时才被认为是一条直线，也可以把它看成能检测到的直线的最短长度（以像素点为单位）  
for each in range(len(lines)):  
    for rho,theta in lines[each]:  
        a=np.cos(theta)  
        b=np.sin(theta)  
        x0=a\*rho  
        y0=b\*rho  
        x1=int(x0+1000\*(-b))  
        y1=int(y0+1000\*(a))  
        x2=int(x0-1000\*(-b))  
        y2=int(y0-1000\*(a))  
        cv2.line(img,(x1,y1),(x2,y2),(0,150,255),2)  
cv2.imshow(&apos;kuang hei bai zi&apos;, img\_rgb)  
cv2.imshow(&apos;window&apos;,img)  
cv2.waitKey(0)  
cv2.destroyAllWindows()

效果图：

