第一题的视频效果：



首先第一次看到这个题的的时候，我提前翻了翻学长发的opencv.pdf,然后自己在电脑上一个一个的写出来后运行，也就逐步了解了这道题所需要的所有的函数的大概作用，但是美中不足的是，pdf中有些是错误的，所以有些资料我还是决定自己上网上去查。

下面是思路。首先肯定要先读取两个想让他们相互混合的图片img1=cv2.imread('f.png')【含足球】img2=cv2.imread('j.png')【QQ音乐的图片】

然后得到图片2后，自然想得到关于图片的行数和列数，为后面的在图1中抠出来相应的一部分做准备，rows,cols,channels=img2.shape ，roi=img1[0:rows,0:cols]#roi为图一的相同大小的图片，然后我们要实现的是抠图的操作，先将图片变为灰度图，以便后续操作img2\_Grey=cv2.cvtColor(img2,cv2.COLOR\_BGR2GRAY)

然后我们要做的是一个掩码操作，，随后将其二值化处理，黑白颠倒，扣出图案的大致样式ret,mask=cv2.threshold(img2\_Grey,200,250,cv2.THRESH\_BINARY)mask\_inv=cv2.bitwise\_not(mask)#将其黑白颠倒

然后用opencv中的函数bitwise\_and对相应的图像进行与的操作

img1\_BG=cv2.bitwise\_and(roi,roi,mask=mask)

img2\_FG=cv2.bitwise\_and(img2,img2,mask=mask\_inv)随后用add函数将处理的图像整合到一块，就几乎将所需要的图案抠出来了

dst=cv2.add(img1\_BG,img2\_FG)

下面要进行的是，用红色的矩形框将篮球圈住，由于我不知道篮球的位置在哪，所以我就另外设置了一个py文件，逐步的减小放大范围，逐渐的就找到了篮球的位置，那么如何实现画框的操作呢，大致思路就是用一个较大的红色完全填充的矩形覆盖矩形，然后在此位置上贴一个较小的篮球图片，注意，这里要保存图像的副本，不会对原始图像造成修改

img9=img3.copy()football=img1[385:450,150:220]img5=football.copy()# cv2.imshow("windows",football)img3[375:460,140:230]=(0,0,255)# img4=img1.copy()，最后是实现两图像重合了，现在我已经有了一个图像包含QQ音乐，和圈住篮球的图像，还差一个只含有圈住篮球的没有QQ音乐的，蓑衣我要再创建一个新的图像img11

img3[385:450,150:220]=img5# cv2.imshow("windows",img3)football=img6[385:450,150:220]img11=football.copy()img6[375:460,140:230]=(0,0,255)img6[385:450,150:220]=img11

然后有了图像后还差一个滑动条，我们都知道cv2.createTrackbar是创建滑动条的函数，用cv2.setTrackbarPos设置滑动后返回的值在此处要定义一下nothing为pass,让它继续进行

cv2.createTrackbar(MyTrackBarName,window\_Name,0,100,nothing)cv2.setTrackbarPos(MyTrackBarName,window\_Name,0)

接下来要进行自己定义调节权重的函数了

def img\_blending(wd\_Name,trackBarName, image1, image2): weight\_x: float = (100 - cv2.getTrackbarPos(trackBarName, wd\_Name)) / 100 # 动态获取滑动条的值并转换成小数 weight\_y: float = 1 - weight\_x # 另一个权值为 1 - weight\_x return cv2.addWeighted(image1,weight\_x,image2,weight\_y,0) # 返回对象然后用while使图像一直显示，再弄一个结束的语句，当你按下Esc后，进程结束，不过别忘将窗口与你设置的函数关联起来

while 1: cv2.imshow(window\_Name,img\_blending(window\_Name,MyTrackBarName,img6,img3)) k = cv2.waitKey(1) & 0xFF # 按 “ESC” 退出 if k == 27: break

最后也就是代码的结束，等待几秒后，关闭所有的窗口

第一题的整体演示图片：电脑萤幕的截图

描述已自动生成

第二题的显示效果如下：

电脑屏幕截图

描述已自动生成

这道题，还是相对于第一题来说比较容易，因为我没想太多哈；

学长给出一个长方块的图片要求是追踪颜色的HSV,pdf中给的绿色的阈值，然后根据图片中长方体是红色的，所以我就去网上查了查红色相对准确的的最大值和最小值

lower\_red=np.array([156,43,46])  
upper\_red=np.array([180,255,255])接下来根据设定的阈值进行成掩码，然后安位运算，就可追踪到物体了，画面中自然也就只剩下长方体

mask=cv2.inRange(hsv,lower\_red,upper\_red)  
res=cv2.bitwise\_and(img1,img1,mask=mask)

到了这步，画面中只剩下了长方体，那么如何让长方体进行旋转呢，阅读pdf后发现了一个旋转的函数 warpAffine，

getRotationMatrix2D函数主要用来获得图像围绕某点旋转的矩阵，三个参数分别是旋转中心，旋转角度，旋转后图像的缩放比例，最后在窗口上展示即可