HW2

姓名：蔡育嘉 學號：D0641798

函數： Convert to HSV Space

函數： Morphological Operation：Open

函數： Morphological Operation：Close

將不同背景手的照片的手部區域正確標記出

比較在RGB Domain和HSV Domain上進行顏色分割的結果

# 一、程式碼：

|  |
| --- |
| import cv2 import numpy as np  def CvtColor(img):  img\_height,img\_width,img\_channels=img.shape   HSV\_result=np.ones([img\_height,img\_width,img\_channels],dtype='uint8')   Blue=0  Green=0  Red=0   for h in range(img\_height):  for w in range(img\_width):  for c in range(img\_channels):  if c==0: #依序存放BGR值  Blue=img[h][w][c]  if c==1:  Green=img[h][w][c]  if c==2:  Red=img[h][w][c]   HSV\_result[h][w][0]=H\_calculation(Blue,Green,Red)  HSV\_result[h][w][1]=S\_calculation(Blue,Green,Red)  HSV\_result[h][w][2]=V\_calculation(Blue,Green,Red)   return HSV\_result  #B G R S V介於0到1 #H 0到360 def H\_calculation(B,G,R):  B=B/255  G=G/255  R=R/255  Max=max(B,G,R)  Min=min(B,G,R)    if Max==Min:  H=0  elif Max==R and G>=B:  H=60\* (G-B)/(Max-Min)+0  elif Max==R and G<B:  H=60\* (G-B)/(Max-Min)+360  elif Max==G:  H=60\* (B-R)/(Max-Min)+120  elif Max==B:  H=60\* (R-G)/(Max-Min)+240  return H  def S\_calculation(B,G,R):  B=B/255  G=G/255  R=R/255  Max=max(B,G,R)  Min=min(B,G,R)  if Max==0:  S=0  else:  S=1-(Min/Max)  return S\*255 #因為S出來的值介於0~1 但BGR是0~255 所以把除掉的乘回來 def V\_calculation(B,G,R):  B=B/255  G=G/255  R=R/255  Max=max(B,G,R)  Min=min(B,G,R)  V=Max  return V\*255#因為V出來的值介於0~1 但BGR是0~255 所以把除掉的乘回來  def dilation(img):  img\_height,img\_width=img.shape  result=np.zeros([img\_height,img\_width],dtype='uint8')  for h in range(1,img\_height-1):# 這邊先處理邊界以外的部分  for w in range(1,img\_width-1):  if img[h-1][w]==255 or img[h+1][w]==255 or img[h][w+1]==255 or img[h][w-1]==255:  result[h][w]=255  if img[0][1] ==255 or img[1][0]==255: #左上角點  result[0][0]=255  if img[0][img\_width-1-1]==255 or img[1][img\_width-1]==255: #右上角點  result[0][img\_width-1]=255  if img[img\_height-1-1][0]==255 or img[img\_height-1][1]==255:#左下角  result[img\_height-1][0]==255  if img[img\_height-1][img\_width-1-1]==255 or img[img\_height-1-1][img\_width-1]==255:#右下角  result[img\_height-1][img\_width-1]==255   for w in range(1,img\_width-1-1): #上面的邊  if img[0][w-1]==255 or img[0][w+1]==255 or img[1][w]==255:  result[0][w]=255  for w in range(1,img\_width-1-1): #下面的邊  if img[img\_height-1][w-1]==255 or img[img\_height-1][w+1]==255 or img[img\_height-1-1][w]==255:  result[img\_height-1][w]=255  for h in range(1,img\_height-1-1): #左邊  if img[h-1][0]==255 or img[h+1][0]==255 or img[h][1]==255:  result[h][0]=255  for h in range(1,img\_height-1-1): #右邊  if img[h-1][img\_width-1]==255 or img[h+1][img\_width-1]==255 or img[h][img\_width-1-1]==255:  result[h][img\_width-1]=255  return result def erosion(img):  img\_height,img\_width=img.shape  result=np.zeros([img\_height,img\_width],dtype='uint8')  for h in range(1,img\_height-1):# 這邊先處理邊界以外的部分  for w in range(1,img\_width-1):  if img[h-1][w]==255 and img[h+1][w]==255 and img[h][w+1]==255 and img[h][w-1]==255:  result[h][w]=255  if img[0][1] ==255 and img[1][0]==255: #左上角點  result[0][0]=255  if img[0][img\_width-1-1]==255 and img[1][img\_width-1]==255: #右上角點  result[0][img\_width-1]=255  if img[img\_height-1-1][0]==255 and img[img\_height-1][1]==255:#左下角  result[img\_height-1][0]==255  if img[img\_height-1][img\_width-1-1]==255 and img[img\_height-1-1][img\_width-1]==255:#右下角  result[img\_height-1][img\_width-1]==255   for w in range(1,img\_width-1-1): #上面的邊  if img[0][w-1]==255 and img[0][w+1]==255 and img[1][w]==255:  result[0][w]=255  for w in range(1,img\_width-1-1): #下面的邊  if img[img\_height-1][w-1]==255 and img[img\_height-1][w+1]==255 and img[img\_height-1-1][w]==255:  result[img\_height-1][w]=255  for h in range(1,img\_height-1-1): #左邊  if img[h-1][0]==255 and img[h+1][0]==255 and img[h][1]==255:  result[h][0]=255  for h in range(1,img\_height-1-1): #右邊  if img[h-1][img\_width-1]==255 and img[h+1][img\_width-1]==255 and img[h][img\_width-1-1]==255:  result[h][img\_width-1]=255  return result def MorOpen(img): #先erosion 在dilation  temp=erosion(img)  result=dilation(temp)  return result def MorClose(img): #先dilation 在erosion  temp=dilation(img)  result=erosion(temp)  return result def HSV2Bin(img):  img\_height,img\_width,img\_channels=img.shape  binary=np.zeros([img\_height,img\_width],dtype='uint8')  skin\_upper=[40,150,255]  skin\_lower=[0,30,60]  for h in range(img\_height):  for w in range(img\_width):  for c in range(img\_channels): #如果BGR值 在上下限之間 則設為白色  if img[h][w][c]>skin\_lower[c] and img[h][w][c]<skin\_upper[c]:  binary[h][w]=255  else:  binary[h][w]=0 #其中一個不在範圍內則設黑色 並跳開  break  return binary  def BGR2Bin(img):  img\_height,img\_width,img\_channels=img.shape  binary=np.zeros([img\_height,img\_width],dtype='uint8')  skin\_upper=[215,240,236]  skin\_lower=[82,150,177]  for h in range(img\_height):  for w in range(img\_width):  for c in range(img\_channels): #如果HSV值 在上下限之間 則設為白色  if img[h][w][c]>skin\_lower[c] and img[h][w][c]<skin\_upper[c]:  binary[h][w]=255  else:  binary[h][w]=0#其中一個不在範圍內則設黑色 並跳開  break  return binary img=cv2.imread("C:/Users/user/Desktop/imageprocess/HW2/hand\_3.jpg")  cv2.imshow("Original",img) cv2.waitKey()  result\_BGR=BGR2Bin(img) result\_BGR=MorOpen(result\_BGR) result\_BGR=MorClose(result\_BGR) cv2.imshow("bin\_BGR",result\_BGR)   self\_hsv=CvtColor(img) result\_hsv=HSV2Bin(self\_hsv) result\_hsv=MorOpen(result\_hsv) result\_hsv=MorClose(result\_hsv) cv2.imshow("bin\_HSV",result\_hsv) cv2.waitKey() |

# 二、執行結果：

 

原圖 bin\_BGR bin\_HSV



原圖 bin\_BGR bin\_HSV



原圖 bin\_BGR bin\_HSV

三、比較在RGB Domain和HSV Domain上進行顏色分割的結果：

在第三張手的照片，在RGB的color space裡面，我認為可能是我拍照的時候光亮度不足的原因，就超過我設定的膚色範圍，反而背景落在範圍，可能是我膚色範圍開太大了，在第三張照片來看，HSV的框選的效果最好，而手部照片內的黑點範圍太大所以MorOpen的去噪效果沒有相當理想，至於MorClose的效果在手部區域以外的去噪效果很好。