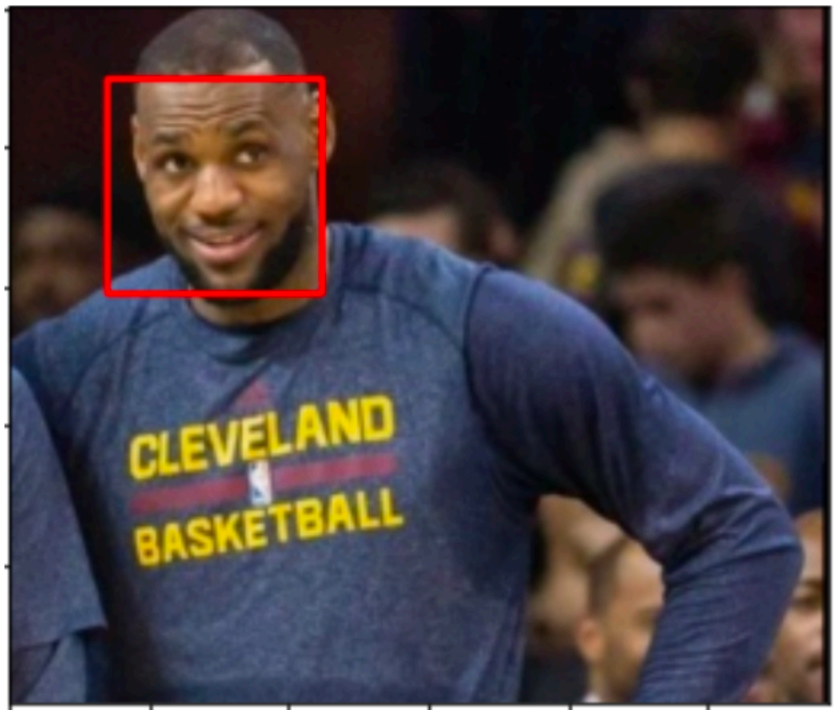
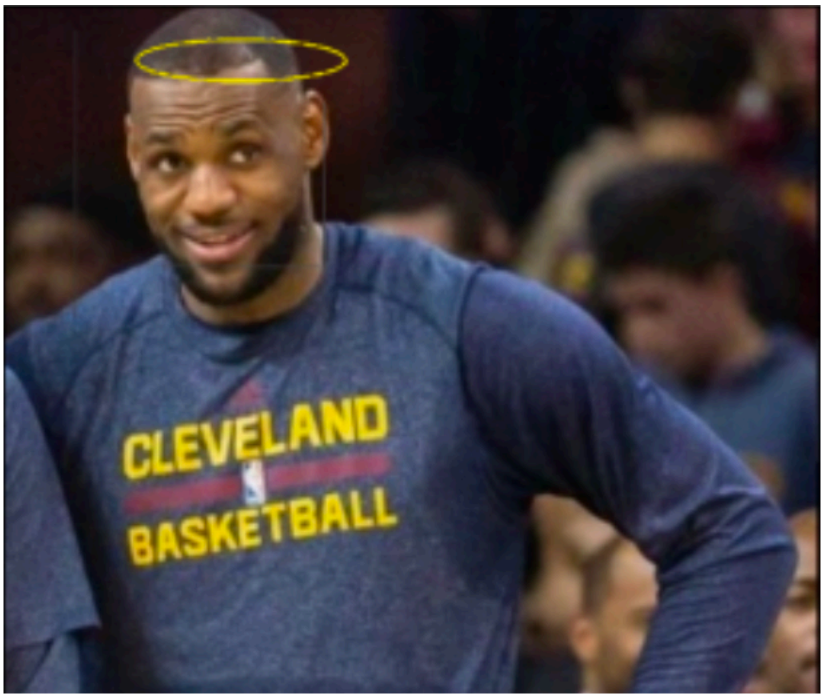
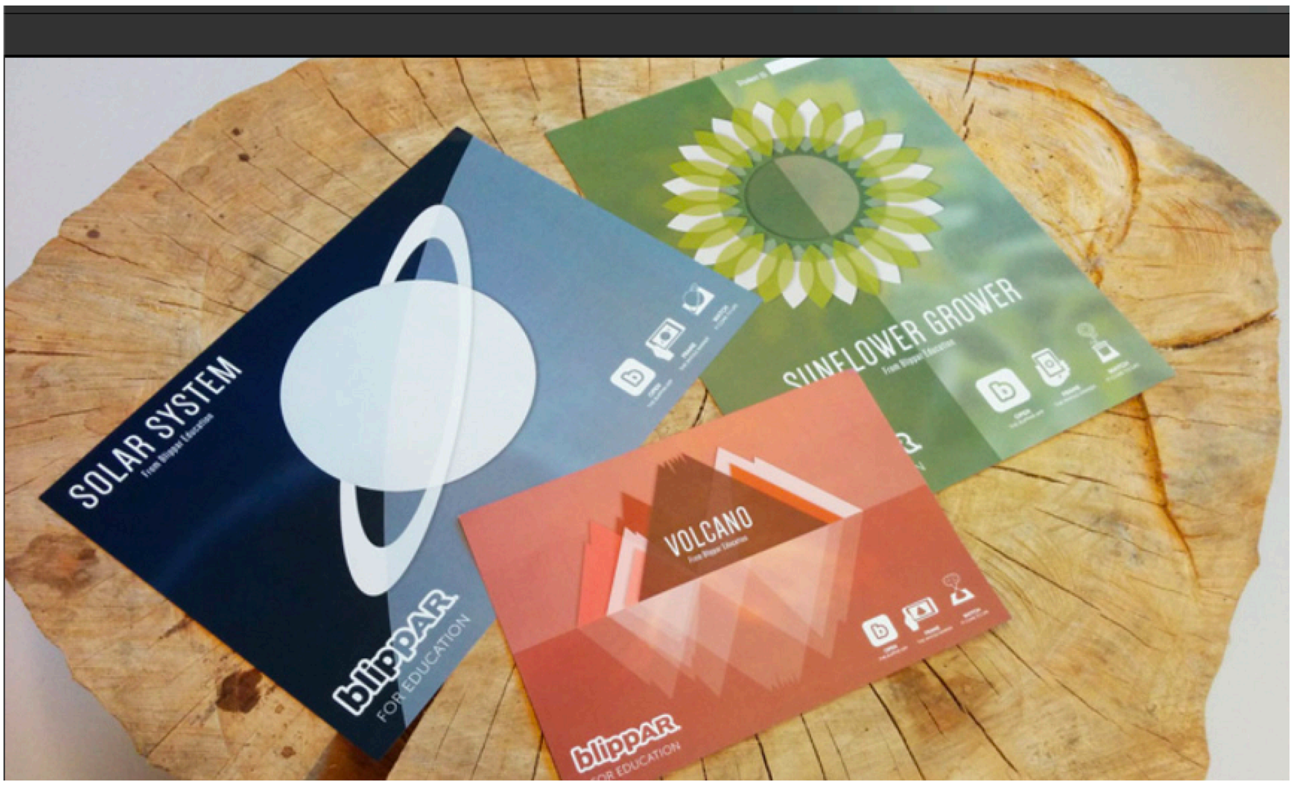
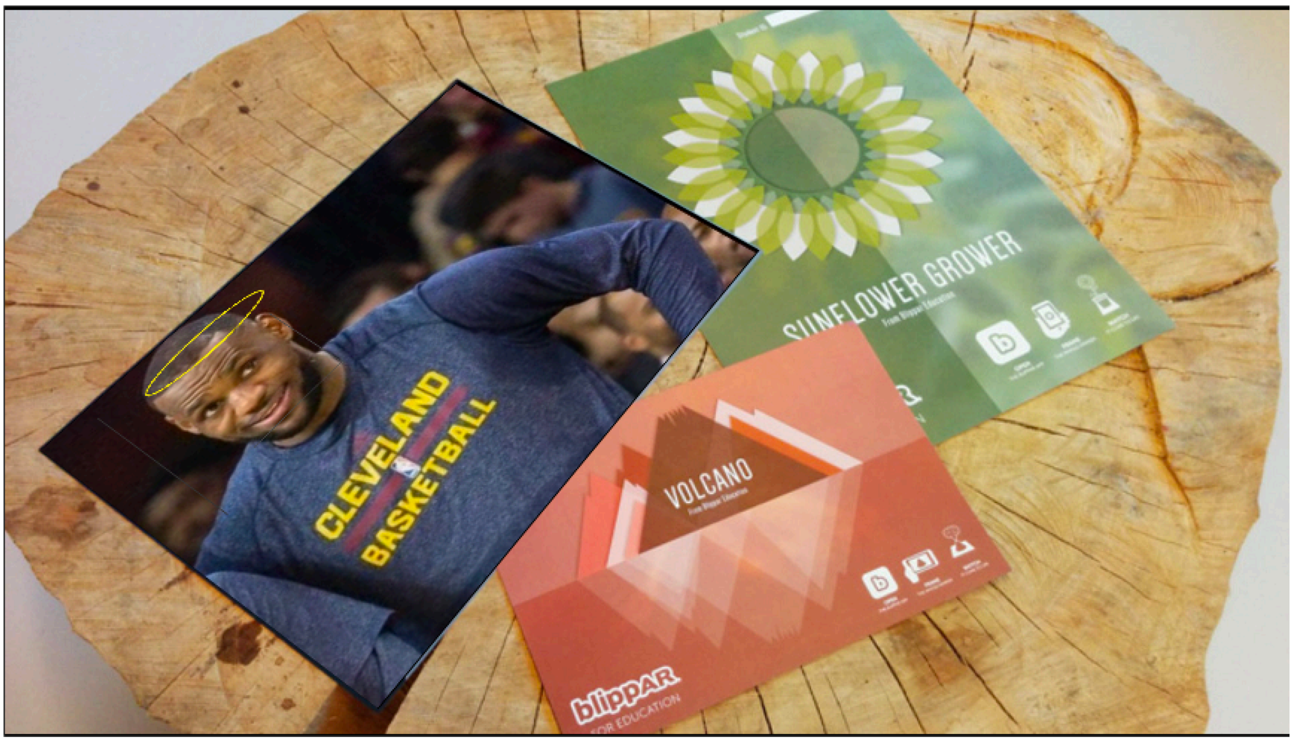
設定滑鼠點擊的callback function收集四個點
```python
def onMouse(event, x, y, flags, pointsList):
    if(event == cv2.EVENT_LBUTTONDOWN):
        pointsList.append([x,y])
#%%
import cv2
import matplotlib.pyplot as plt
import numpy as np
faceCascade = cv2.CascadeClassifier(cv2.data.haarcascades + 
'haarcascade_frontalface_default.xml')

faceImage = cv2.imread('Picture.png')
effectImage = cv2.imread('effect.png')
# print(faceImage.shape)
# print(effectImage.shape)
faces = faceCascade.detectMultiScale(faceImage, 1.3, 5)
```
在找到的臉上放上特效
```python
for (x, y, w, h) in faces:
    faceImageCopy = faceImage.copy()
```
畫出臉的方框
```python
    cv2.rectangle(faceImageCopy, (x,y), (w+x,y+h), (0,0,255), 3)
    # print(faceImageCopy.shape)
    cv2.imshow('Get Face', faceImageCopy)
    cv2.waitKey()
    # plt.imshow(cv2.cvtColor(faceImageCopy, cv2.COLOR_BGR2RGB))
    # plt.show()
    print(“w,h",w,h)
```
找出特效的方框與臉方框的比例及resize
```python
    ratioHeight = h/100
    ratioWidth = w/100
    W = effectImage.shape[0]
    H = effectImage.shape[1]
    print(effectImage.shape)
    effectImage = cv2.resize(effectImage, (int(w*ratioWidth), 
int(h*ratioHeight)))
    print(effectImage.shape)
    effectImageGray = cv2.cvtColor(effectImage, 
cv2.COLOR_BGR2GRAY)
    th1 = 50
    th2 = 220
    print("x,y",x,y)
    x -= int(w*ratioWidth) - w
    # y -= h
    print("x,y",x, y)
```
不是背景的加入圖中
```
    for i in range(int(w*ratioWidth)):
        for j in range(int(h*ratioHeight)):
            if(effectImageGray[i][j] >= th1 and effectImageGray[i]
[j] <= th2 ):
                faceImage[x+i][y+j] = effectImage[i][j]

    # plt.imshow(cv2.cvtColor(faceImage, cv2.COLOR_BGR2RGB))
    # plt.show()
    cv2.imshow('image', faceImage)
    cv2.waitKey()


pointsList = []
backgroundImage = cv2.imread('background.png')
```
收集四個點
```
cv2.setMouseCallback('image',onMouse, pointsList)
cv2.imshow('image', backgroundImage)
cv2.waitKey()
cv2.destroyAllWindows()
# print(pointsList)

points1 = np.float32([[0,0], [faceImage.shape[1],0],
[faceImage.shape[1], faceImage.shape[0]],
[0,faceImage.shape[0]] ])
pointsList = np.float32(pointsList)
# print(points1)
```
找出轉移矩陣
```
M, status = cv2.findHomography(points1, pointsList)
```
轉移特效圖
```
result = cv2.warpPerspective(src=faceImage, M=M,
dsize=(backgroundImage.shape[1], backgroundImage.shape[0]))
cv2.imshow('image', result)
cv2.waitKey()
```
挖空背景圖再填入準備好的圖
```
cv2.fillConvexPoly(backgroundImage, pointsList.astype(int), 0, 16)
```

```
backgroundImage = cv2.add(backgroundImage, result)
cv2.imshow('image', backgroundImage)
cv2.waitKey()
```









心得：學到了除了圖像處理的技巧十分有趣