

ORB、AKAZE





利用最近邻比率匹配之后

for m, n in matches:  
 if m.distance < 0.75 \* n.distance:  
  
 good.append([m])#good.append(m)

，估算出变换矩阵，使用RANSAC剔除外点。

if len(good) > MIN\_MATCH\_COUNT:  
 src\_pts = np.float32([kp1[m.queryIdx].pt for m in good  
 ]).reshape(-1, 1, 2)  
 dst\_pts = np.float32([kp2[m.trainIdx].pt for m in good  
 ]).reshape(-1, 1, 2)  
  
 M, mask = cv2.findHomography(src\_pts, dst\_pts, cv2.RANSAC, 5.0)

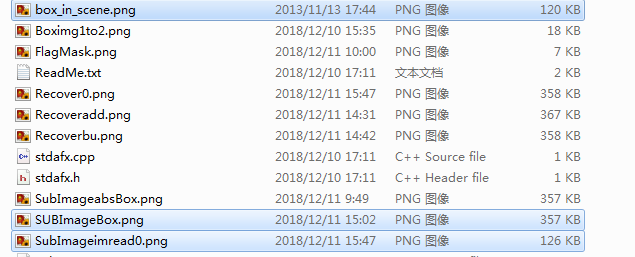






复原图像，根据差值再反向求出原数据集中的图像。

第一个复原图像是怎么得到的。。忘记了。但是看我的命名，一个说add，一个没有add，猜测应该是一个复原方法是保留配准之后的部分，对于全黑填充的部分



图像压缩

图像融合

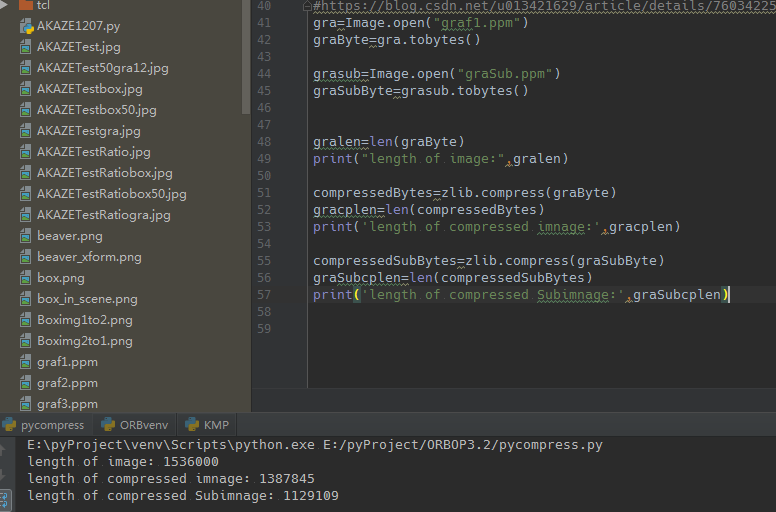
图像集

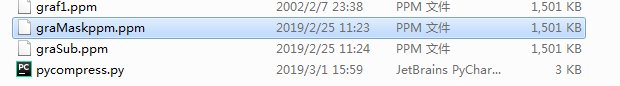
按道理图像大小没有改变，存储空间应该是一样的，但是box中首先是从原始png读入再保存，从120KB到了358KB，差值后的图像357KB，掩膜8KB。在gra中，ppm是1501KB，转成PNG是1003KB，掩膜184KB，差值图像921KB

原来使用opencv的imwrite保存时第三个参数是一个vector，分别保存了图像格式和压缩级别，格式是JPEG时默认级别是95，格式是PNG时默认级别是3。级别越高，图像占用大小越小，压缩时间越长。

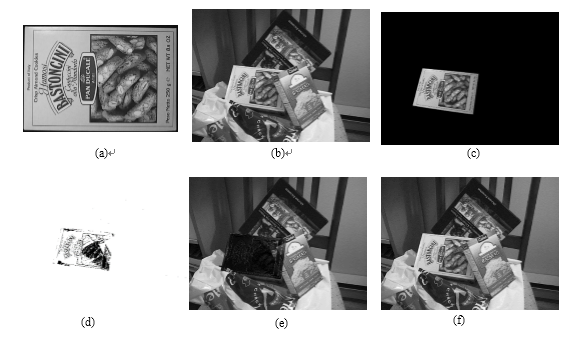
于是直接使用ppm格式，这种格式和bmp格式都不会被压缩。果然，在gra中，差值图像和掩膜图像大小都是1501KB。https://www.cnblogs.com/skyfsm/p/7136709.html

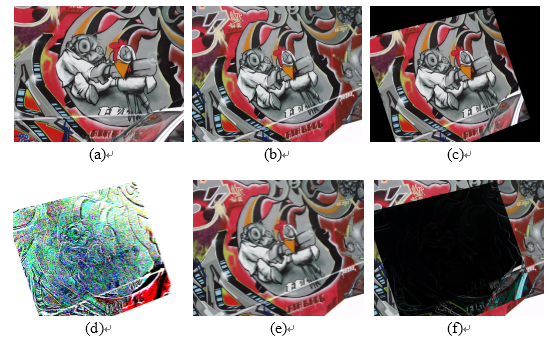
对png，读入再保存，不意味着一定会压缩。





在计算汉明距离时，两幅graf图像的距离是15，但是beaver和graf的距离也有13.将哈希算法中的缩放图像变大到17\*16，还是没有好转。将缩放图像使用时间戳保存下来，可以看到马赛克图像。





两幅图像的哈希值之间的汉明距离是16和15，而哈希值是长度为64的二进制字符串，说明二者的相似性较高，可以进行图像配准。在实验中，使用OpenCV函数库中的imwrite函数可以保存指定格式和压缩质量的图像，这里压缩级别采用了缺省值。当图像格式为jpg时，参考图的原图大小为242KB，差值图像大小可压缩为191KB。

