

Explain My Code Work

需要使用终端安装的包

原谅我不会打包python的库TAT

语言 python

```
pip install opencv-python  
pip install -U scikit-image  
pip install imutils  
pip install scikit-learn
```

函数解释

- cv2.imread(Path)

opencv2中提供的读取图片函数

- cv2.cvtColor(pic , cv2.COLOR_BGR2GRAY)

opencv2中提供的颜色空间转换函数,第二个参数提供cv2.COLOR_BGR2GRAY时, 完成将 pic 从多通道有色图片转为单通道灰度图像

- structural_similarity是skimage库中提供的结构相似性算法

调用(score, diff) = compare_ssim(grayA, grayB, full=True) 可以将灰度图 grayA 和 grayB 进行结构相似性比对，并返回不同的部分的图像为diff，返回相似性系数score

- diff = (diff * 255).astype("uint8"),

将返回的diff图像每个像素值从[0,1]变为[0,255]

- cv2.threshold(diff, 0, 255, cv2.THRESH_BINARY_INV | cv.THRESH_OTSU)[1]

opencv 库中对 cv2 进行二值化，0,255为像素灰度上下界，cv2.THRESH_BINARY_INV | cv2.THRESH_OTSU代表结合了二进制阈值和大津法进行二值化，cv2.threshold会返回两个值，其中第二个为二值化后的图像

- cnts = cv2.findContours(thresh.copy(),
cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
cnts = imutils.grab_contours(cnts)# 找轮廓

其中cv2.findContours为 opencv2 提供寻找轮廓的函数，RETR_EXTERNAL代表只返回最外层的轮廓，cv2.CHAIN_APPROX_SIMPLE代表只返回角点信息

imutils.grab_contours(cnts)通过判断opencv版本号，返回轮廓

- cv2.boundingRect()

返回轮廓近似的矩形结构(x,y,h,w)

- cv2.rectangle(imageA, (x, y), (x + w, y + h), (0, 0, 255), 2 (代表线宽))

绘制一个矩形框

- feature.hog(blockA, orientations=9, pixels_per_cell=[10,10], cells_per_block=[3,3], visualize=True, feature_vector = True)

该函数会对blockA图像，以9通道，cell大小为 10 * 10，block 大小 3 * 3 * cell，返回可视化图像，将HoG向量压缩为一维向量返回，其中第一维为压缩的向量，第二维为可视化图像

- cosine_similarity([A], [B])

该函数返回 $n * n$ 矩阵, 对A中每一个向量和B中每一个向量求余弦相似性, 其中第 i, j 个元素为 A_i 和 B_j 的余弦相似度

- `img0_h_B = cv2.calcHist([img0], [0], None, [256], [0, 255])`

该函数返回256柱的像素范围为[0,255]的蓝色直方图, 其中[0]代表B,[1]代表G,[2]代表R

- `similarity_b= cv2.compareHist(img0_h_B, img1_h_B, cv2.HISTCMP_CORREL)`

该函数对两个直方图进行比较, `cv.HISTCMP_CORREL`代表使用相关性比较

作业解释

pic

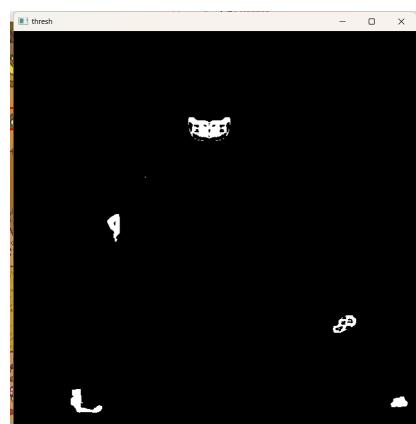
为两张找不同的图片素材

work1

为第一份代码, 其中`work.py`使用了`ssim`整图直接比较

`ssim`的整体比较对图像颜色的变化较不敏感, 无法检测出草丛的颜色变化

效果如下





work2

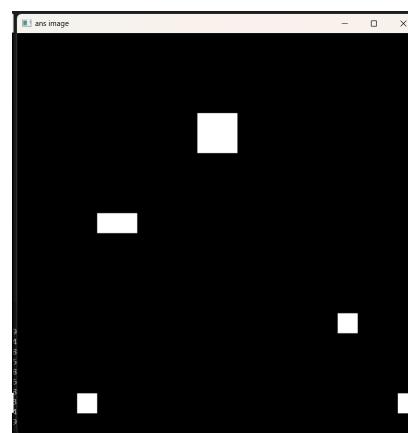
这份作业将图片切割成 20×20 块，并依次进行HoG比对

其中work2.py使用HoG的结果图像素比较后，使用ssim进行进一步检验

其中work_ex.py使用HoG的结果图像素比较后，通过HoG的特征向量进行余弦相关性检验

使用HoG算法的检验可以清楚的检验出图像轮廓方面的变化，但和ssim相同，使用灰度图的检验对图像颜色敏感性低

work_ex.py的效果如下





work3

基于上述两种方法的对图像颜色不敏感性

这份作业将图像分割成 60×60 的小块

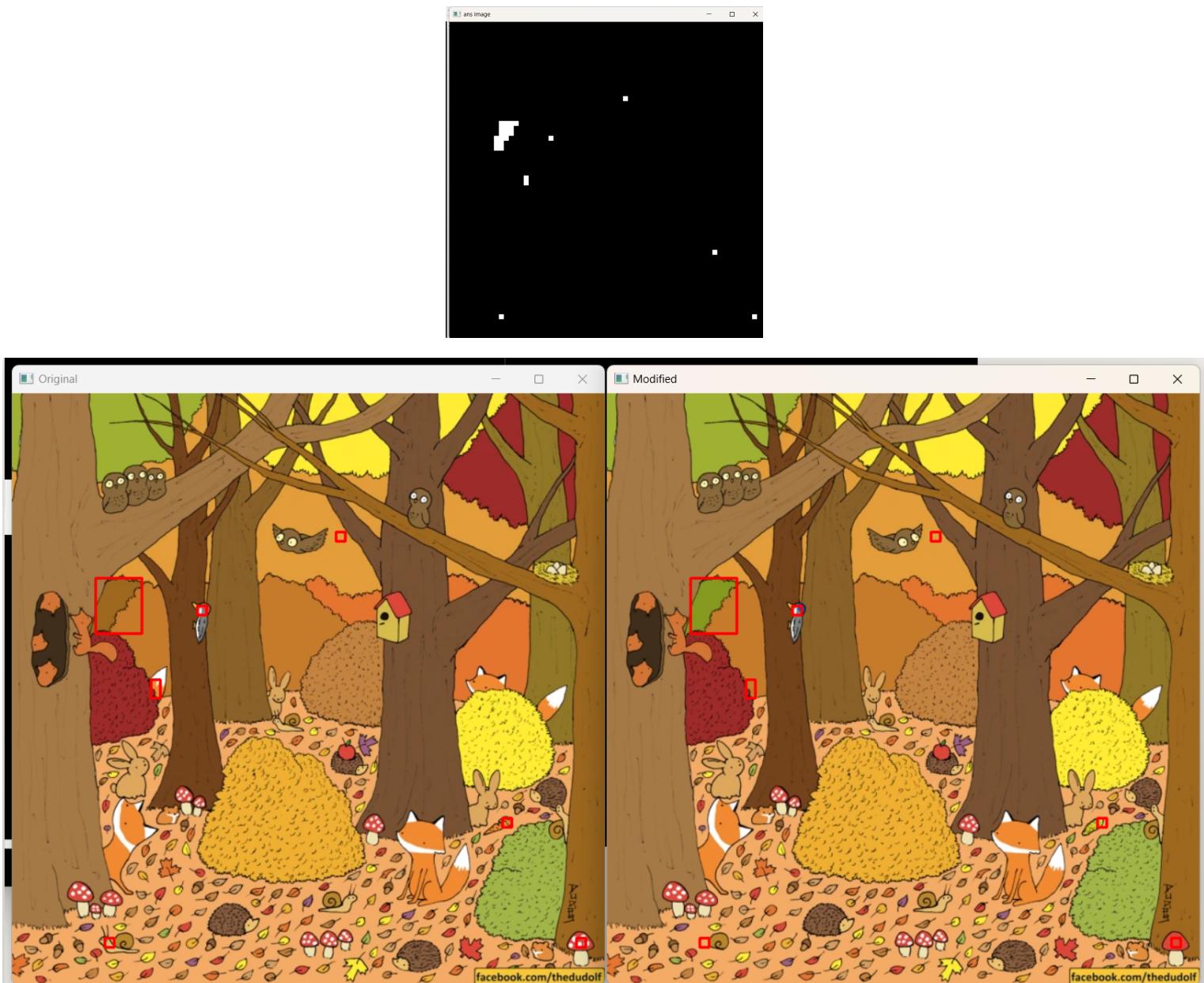
对每小块的三个颜色通道检验颜色直方图相关性

其中 work3.py 通过颜色直方图检验

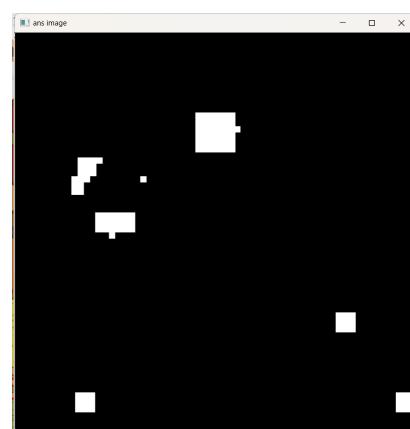
颜色直方图检验的效果对于颜色的更改和不同颜色物品的有无较为敏感，但对于检验图片中的类似于小鸟这种转换了形态但颜色变化不大的较不敏感，且较容易收到噪声的干扰，对阈值限定要求高

所以更新了一份 work3_ex.py 通过颜色直方图和 work2 的 HoG work_ex.py 的结合来检验

work3.py 效果如下



work3_ex.py效果





work4

使用了差异hash代替hog，和颜色直方图结合

误判概率较高

