作业二：Seam Carving实验报告
计76 沈诣博 2017011427

1.代码项目说明：

1.1 采分点

1. 完成了基础算法
2. 通过接缝插入实现了图像扩展
3. 实现了前向能量公式
4. 实现了移除物体

1.2 目录结构

`seam_carving.py`为我的全部代码

`data/in`为示例原图

`data/out`为示例对应的输出

1.3 依赖和运行方式

本项目依赖`ndimage`,`cv2`,`numpy`和`numba`包，其中`numba`包用于加速dp算法。

使用时，无论是前向能量还是后向能量、图像拓展还是图像删减，以`python3 seam_carving.py <image_path> <output_path> <new_height> <new_weight>`的形式运行，而对于物体移除，以`python3 seam_carving.py <image_path> <output_path> <mask_path>`的形式进行。

对于前向能量和后向能量的转换，只需要调整`seam_carving.py`中的`use_Forward`全局变量即可。True为使用前向能量，False为使用后向能量。

2. 代码思路

2.0 能量函数

后向能量思路很简单，对于x方向和y方向求一下梯度（即是做一个核为[1,0,-1]的一维卷积）之后绝对值相加即可。

前向能量的思路也很简单，对于u,l,r三个方向进行向量级别的操作，从而快速完成每一行能量的计算。

2.1 缩放函数

我简要的实现了一个先增删宽度，再增删高度的缩放函数。因为高度可以试做图片旋转90度后的宽度增删，所以我们只需要讨论宽度的删除和增加。

2.1.1 删除缝隙：

删除缝隙的方法很简单，每次通过动态规划算法找到能量最低的缝删掉就可以了。

2.1.2 增加缝隙：

为了避免如论文图8(b)所示的重复增加同一个缝隙导致的不自然情况，我使用了删除——增加两步走的方式，即通过x次删除，找到数量等同于需要增加宽度x的最低x个能量的缝隙的对应位置，然后在这些位置上对原图依次进行插入，并且依据每一次插入的图像相对变化来更新原有缝隙的绝对位置，从而避免拓展同一个地方。

2.2 最佳缝隙查找

我们采用论文中所示的dp算法，来找到最佳缝隙，因为一个位置(x,y)的"上延"只可能是(x-1,y-1)(x-1,y)(x-1,y+1)中的三个之一，因此我们只需要从这三个中间进行状态转移。

2.3 物体移除

首先，对于要移除的物体的mask，找到原图对应其中的部分，然后将mask内部的像素能量设为一个负的极大数，从而每一次尽可能的移除已有的mask内物体部分。当物体被移除完毕之后，记录移除的接缝个数x，然后进行x个增加缝隙操作，然后就得到尺寸不变的物体移除后的图像了。

2.4 加速

因为dp算法每一轮耗时均为$O(xh)$，大大影响了图像编辑的效率，我使用numba包的jit功能加速了这部分。

3. 实验结果和分析

3.1 缩小

图1 (a) 城堡原图，(b)城堡后向，(c)城堡前向，可以看出两种算法都完美的保留了图片重要信息，但是前向能量的细节处理的更好

3.2 放大

图2 (a) 海豚原图，(b)海豚放大后向，(c)海豚放大前向，可以看出两种算法都完美的保留了图片重要信息，但是前向能量的比例更加均匀

3.3 物体去除

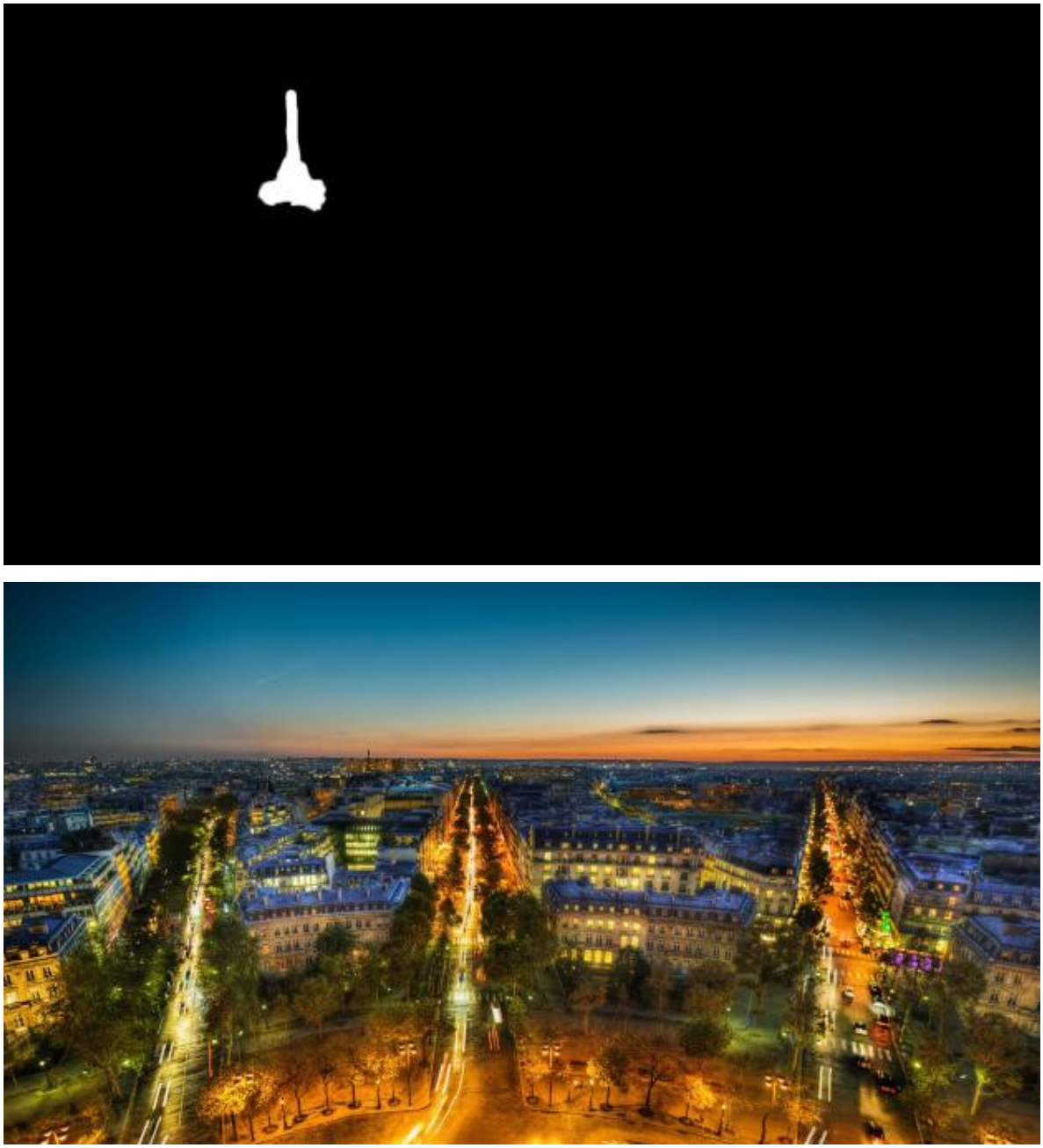

图3 (a) 巴黎夜景原图，(b)埃菲尔铁塔物体mask，(c)后向能量移除的巴黎夜景，可以看出即使是后向能量也能够将物体移除但是并不改变其他部分的纹路。

3.4 不好的图片

图4 上一个大作业草莓的等比例放大

图5 海豚图片的过度压扁

上图中给出了两类不好的结果，两类图片均按照第一张原图第二章修改后图片的顺序等比例排列。

第一类以草莓为例子，在等比例扩大150%的时候，可以看到草莓光滑有几何特征的表面被修改的面目全非。究其原因，是因为单纯梯度的能量函数并不能反应复杂曲线的几何特征，从而导致不同位置的像素被扩大的比例不一，再加上这张照片本来就小(184x240)，从而导致了原有的纹理被毁掉。

第二类以海豚为例子，展示了在不成比例的横向缩小时候，对于海豚和天空信息的严重破坏。这个破坏一方面是因为上述的几何纹理，另一方面也是由于图像上空显示的天空和海豚内部的像素横向更加平均，导致了一味删除上部保留下部，致使图片最终的信息失调。

4. 致谢

感谢徐老师的精彩讲解。

感谢助教的仔细解释和细心批改。