ImageSeamCarving

Course Project for Introduction to Computation

Proposed by TA Sizhe Li

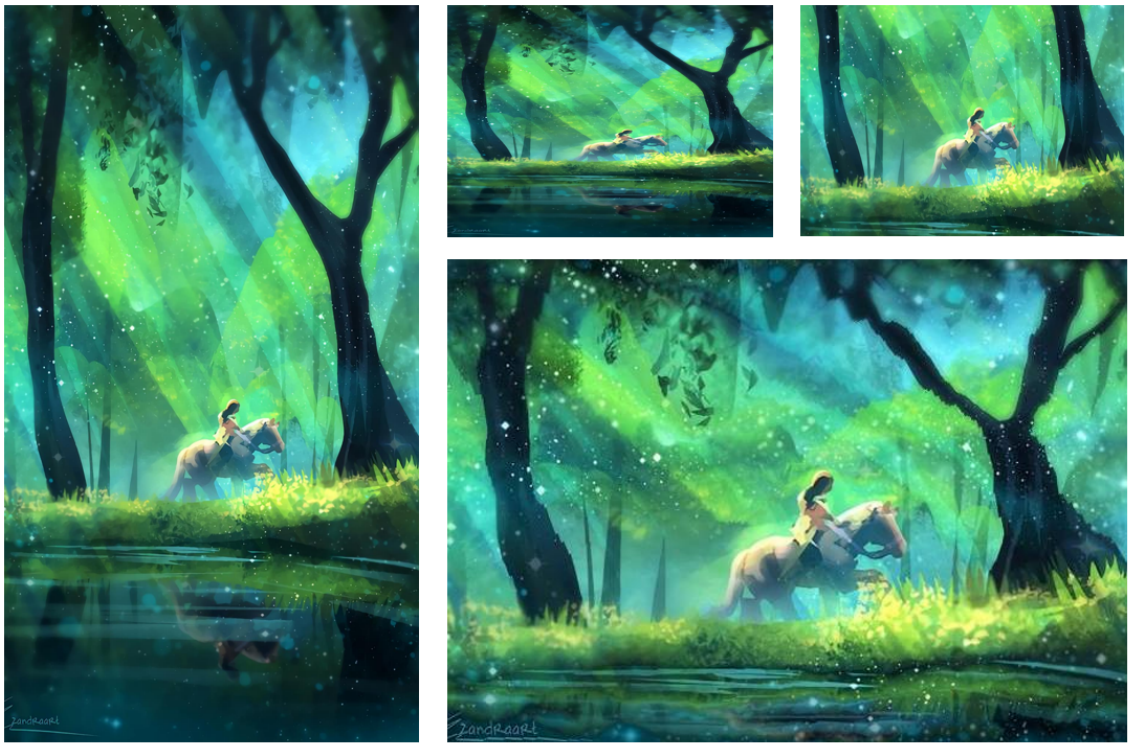
0. 准备

本项目核心为**动态规划算法**，建议对动态规划理解程度较好的同学选择，也欢迎想要了解传统**计算机视觉(Computer Vision)** 的同学阅读或选择。项目要求实现三个基本函数，所需代码量约百行。

1. 引言

生活中我们时常遇到需要调整图片比例的场景，例如将横板图片调整为竖版。两种常用的方法分别为直接进行裁剪，以及进行图片的拉伸。但在实际应用中，前者往往会丢失图片中的部分内容，而后者则容易使得图像失真。

在本次项目中，你将复现一个经典的图像"接缝剪裁"（Seam Carving）算法。Seam Carving算法是由以色列的两位教授Shai Avidan和Ariel Shamir于2007年提出，算法本身基于动态规划处理图像，实用且轻便，也可作为日常的小工具使用。算法实现了在随意改变图像比例的同时，保证图像中重要部分大小不变。简单来说，图片缩放后仍然能够维持整体的完整性。



如上图所示，左侧为原始图片，算法的目标为更改图片的比例至横板，右侧上半部分展示了直接拉伸和剪裁之后的效果图，右下图为采用Seam Carving算法处理后的效果图。在内容完整性和真实性方面，Seam Carving算法明显优于前述两种方法。

2. 算法

以下步骤均以将横板图片压缩为竖版图片为例。

首先我们定义缝隙（seam），即从左到右或从上到下的连续像素，它们横向或纵向穿过整个图像。而在“接缝剪裁”算法中，核心目标即为找到图像中“最不关键”的区域进行删除，而压缩过程可看作每次删除一条缝隙，重复若干次，直到所得图片比例符合要求。

我们将整个算法分为两个步骤：

1. 计算原始图片的能量图
2. 找到并删除能量最低的接缝，不断重复

关于这些概念及过程的具体解释可参考文件`tutorial.pdf`

3. 拓展

上一章节我们仅讲述了Seam Carving算法的最基本应用，即横向压缩图片，同理，纵向压缩图片也可用类似的方法实现。

除此之外，拉伸图片也可以使用类似方法，即找到原图中能量最低的路径并复制该路径。值得注意的是，若每次循环中直接寻找能量最低路径并复制，则多次循环中很可能找到相同路径，从而导致图像失真。一种解决方案是依次找到能量最低的若干条路径并存储，在合适时间同时复制，实现图片的拉伸。

Seam Carving算法的另一个应用为同时压缩图像长宽，只保留最关键部分。这个功能的实现可以视为同时删除横向及纵向接缝，但在实现的过程中依然需要考虑接缝删除的顺序，以尽可能保证图片不失真。

在Seam Carving算法的实现过程中，每次循环导致了大量重复的计算，而若修改目标图片的比例，整个算法又要重新运行，费时费力。而Seam Carving的实时算法只需在输入图片时对图片进行预处理，储存所有的接缝，就可以在极短时间内将图片变换到任意指定尺寸，从而达到实时的目的。

Seam Carving算法同样可以拓展到视频中，结合光流算法或网络流算法，从而实现视频的尺寸更改。

4. 环境配置

项目所需python库包括`numpy`，`matplotlib`，`yaml`等。

为更方便项目的实现，建议安装`tqdm`，`scipy`等库。

5. 文档结构

```
ImageSeamCarving
        ├──────── sc.py          # 项目主体代码
        ├──────── util.py        # 可视化及参数解析（不需修改）
        ├──────── config.yml     # 参数文件（可根据需要修改）
        ├──────── README.pdf     # 项目介绍
        └──────── tutorial.pdf # 作业指导
```

6. 作业要求

详见`tutorial.pdf`