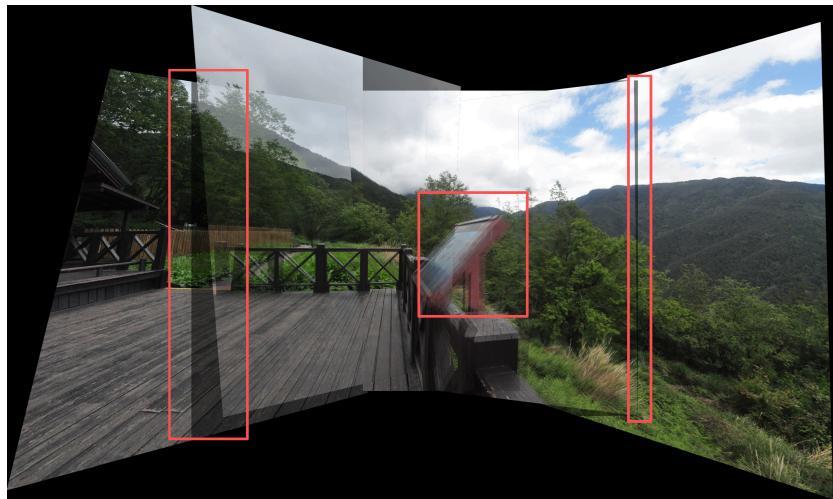


2) Advanced blending techniques: 对于stitch_blend函数的优化

原理简述

给出的函数框架是通过传统的 alpha blending，对重叠区域的像素采用简单平均或固定权重。这方法可以正确拼接融合但容易在图像边界产生较为明显的深色缝隙、内部重影和轻微的色调不一致（见对比图，尤其在panorama_4中明显，如下：）



为了解决这些问题，我进行了简单的优化调整，采用**距离变化权重融合** (feathering)的方法，实现更自然的过渡，避免突兀的边界和重影。

这个方法的核心就是强调重叠区域中，距离图像边缘越远的像素越可靠，因此赋予更高的融合权重；而距离边缘越近的像素权重越低，具体实现流程如下：

- 对每张图像构造一个二值 mask，表示哪些区域是有效像素
- 使用 `cv2.distanceTransform` 计算每个像素到边缘的距离
- 将两个图像的距离图归一化，作为融合权重
- 最终融合图像时，使用这两个权重进行加权平均

实现

在 `stitch_blend` 函数中，替换了原有的 alpha 融合部分，即：

```
alpha1 = cv2.remap(np.ones(np.shape(img_1)), trans_x, trans_y, cv2.INTER_LINEAR)
alpha2 = cv2.remap(np.ones(np.shape(img_2)), x, y, cv2.INTER_LINEAR)
alpha = alpha1 + alpha2
alpha[alpha == 0] = 2
alpha1 = alpha1 / alpha
alpha2 = alpha2 / alpha
est_img = est_img_1 * alpha1 + est_img_2 * alpha2
```

将其替换为距离权重融合的逻辑：

```
#构造有效区域的mask，mask1和mask2是两个二值图 表示每张图像在融合区域中的有效像素位置
mask1 = cv2.remap(np.ones(img_1.shape[:2], dtype=np.uint8), trans_x, trans_y,
cv2.INTER_NEAREST)
mask2 = cv2.remap(np.ones(img_2.shape[:2], dtype=np.uint8), x, y, cv2.INTER_NEAREST)
```

```

#计算距离图, dist1和dist2是两个浮点图, 表示每个像素到各自图像边缘的距离
dist1 = cv2.distanceTransform(mask1, cv2.DIST_L2, 5)
dist2 = cv2.distanceTransform(mask2, cv2.DIST_L2, 5)

#归一化为权重, 保证alpha1+alpha2=1
alpha1 = dist1 / (dist1 + dist2 + 1e-8)
alpha2 = dist2 / (dist1 + dist2 + 1e-8)

#扩展为三通道
alpha1 = np.repeat(alpha1[:, :, np.newaxis], 3, axis=2)
alpha2 = np.repeat(alpha2[:, :, np.newaxis], 3, axis=2)

est_img = est_img_1 * alpha1 + est_img_2 * alpha2 # 融合图像

```

采用距离变化权重融合后，拼接图像的视觉质量有较大提升，明显表现为：边缘处的缝隙和图像内部的重影减弱，且对齐误差更容忍，即使存在轻微错位也能平滑过渡。

整体来看，这个方法可以在沿用原有框架的基础上，既不增加太多计算量的前提下，又显著提升融合的质量，是一个不错的选择。具体效果对比图如下：

panorama_1 优化前/后





panorama_2 优化前/后

