图片拼接复原问题的评价模型

问题重述

一、问题背景

本问题要求开发一个图片自动拼接模型，用以复原来自同一页面的碎片图片。碎片图片可能经过不同方式的切割，包括仅纵向切割和纵横两向切割。具体要求建立模型并提出求解算法，复原的结果需以图片和表格的形式呈现。如果过程中需要人工干预，请明确干预的方式和次数，并尽可能实现拼接过程的自动化与人工干预的交互操作。每个任务中都附有部分图片位置作为已知条件或用于验证拼接结果的参考。需要开发算法根据这些条件实现图片的自动拼接，并在必要时人工干预以优化拼接效果。

二、问题要求

1. 仅纵向切割的中文文字图片拼接：涉及的图片数量为 15 张，每张图片是纵向切割的一部分。
2. 纵横切割的中文文字图片拼接：涉及的图片数量为 96 张，排列方式为 12 行 8 列。
3. 纵横切割的含噪声文字图片拼接：涉及的图片数量为 96 张，含有噪声，排列方式为 12 行 8 列。
4. 纵横切割的彩色图片拼接：涉及的图片数量为 120 张，排列方式为 12 行 10 列。

模型建立

模型和算法建议

特征提取与匹配：

- 利用 **SIFT（尺度不变特征转换）**、SURF（加速稳健特征）或 ORB（Oriented FAST and Rotated BRIEF）等算法来提取图片碎片的关键点，并进行匹配。
- 对于文字图片，可以考虑基于形状和文本内容的特征匹配技术。

SIFT 算法原理

1.检测尺度空间极值

检测尺度空间极值就是搜索所有尺度上的图像位置，通过高斯微分函数来识别对于尺度和旋转不变的兴趣点。其主要步骤可以分为建立高斯金字塔、生成 DOG 高斯差分金字塔和 DOG 局部极值点检测。为了让大家更清楚，我先简单介绍一下尺度空间，再介绍主要步骤。

（1）尺度空间

一个图像的尺度空间，定义为一个变化尺度的高斯函数与原图像的卷积。即：

$$L(x,y,\sigma)=G(x,y,\sigma)*I(x,y)$$

其中，*表示卷积计算。

$$G(x,y,\sigma)=\frac{1}{2\pi\sigma^2}e^{-\frac{(x-m/2)^2+(y-n/2)^2}{2\sigma^2}}$$

m、n 表示高斯模版的维度，(x,y)代表图像像素的位置。为尺度空间因子，值越小表示图像被平滑的越少，相应的尺度就越小。小尺度对应于图像的细节特征，大尺度对应于图像的概貌特征，效果如下图所示，尺度从左到右，从上到下，一次增大。

（2）建立高斯金字塔

尺度空间在实现时，使用高斯金字塔表示，高斯金字塔的构建分为两部分：

1.对图像做不同尺度的高斯模糊

2.对图像做降采样（隔点采样）

（3）建立 DOG 高斯差分金字塔

为了有效提取稳定的关键点，利用不同尺度的高斯差分核与卷积生成。

DOG 函数：

$$L(x,y,\sigma)=G(x,y,\sigma)*I(x,y)$$

$$\begin{aligned} D(x,y,\sigma)&=\left[G(x,y,k\sigma)-G(x,y,\sigma)\right]*I(x,y)\\ &=L(x,y,k\sigma)-L(x,y,\sigma)\end{aligned}$$

可以通过高斯差分图像看出图像上的像素值变化情况。（如果没有变化，也就没有特征。特征必须是变化尽可能多的点。）DOG 图像描绘的是目标的轮廓。

（4）DOG 局部极值检测

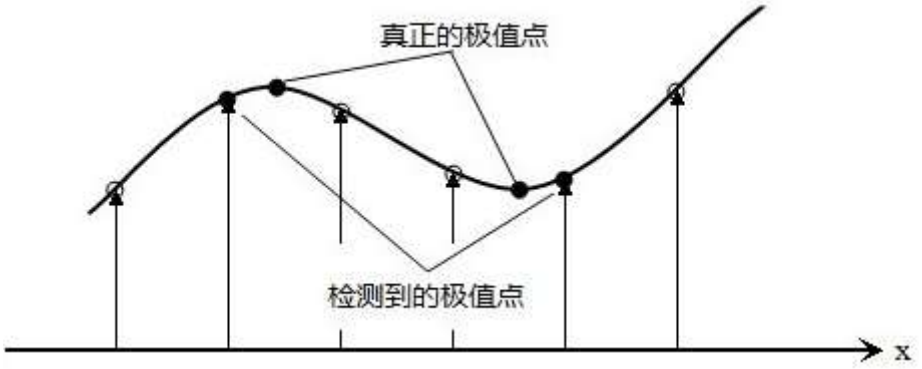
特征点是由 DOG 空间的局部极值点组成的。为了寻找 DOG 函数的极值点，每一个像素点要和它所有的相邻点比较，看其是否比它的图像域和尺度域 的相邻点大或者小。

2.关键点的精确定位

以上方法检测到的极值点是离散空间的极值点，以下通过拟合三维二次函数来精确确定关键点的位置和尺度，同时去除低对比度的关键点和不稳定的边缘响应点(因为 DOG 算子会产生较强的边缘响应)，以增强匹配稳定性、提高抗噪声能力。

（1）关键点的精确定位

利用已知的离散空间点插值得到的连续空间极值点的方法叫做子像素插值（Sub-pixel Interpolation）。



为了提高关键点的稳定性，需要对尺度空间 DOG 函数进行曲线拟合。利用 DOG 函数在尺度空间的 Taylor 展开式(拟合函数)为：

$$D(X) = D + \frac{\partial D^T}{\partial X} X + \frac{1}{2} X^T \frac{\partial^2 D}{\partial X^2} X$$

其中，$X = (x, y, \sigma)^T$。求导并让方程等于零，可以得到极值点的偏移量为：

$$\hat{X} = -\frac{\partial^2 D^{-1}}{\partial X^2} \frac{\partial D}{\partial X}$$

对应极值点，方程的值为：

$$D(\hat{X}) = D + \frac{1}{2} \frac{\partial D^T}{\partial X} \hat{X}$$

其中，$\hat{X} = (x, y, \sigma)^T$ 代表相对插值中心的偏移量，当它在任一维度上的偏移量大于 0.5 时（即 x 或 y 或），意味着插值中心已经偏移到它的邻近点上，所以必须改变当前关键点的位置。同时在新的位置上反复插值直到收敛；也有可能超出所设定的迭代次数或者超出图像边界的范围，此时这样的点应该删除，在 Lowe 中进行了 5 次迭代。

（2）去除边缘响应

由于 DOG 函数在图像边缘有较强的边缘响应，因此需要排除边缘响应 DOG 函数的峰值点在边缘方向有较大的主曲率，而在垂直边缘的方向有较小的主曲率。主曲率可以通过计算在该点位置尺度的 2×2 的 Hessian 矩阵得到，导数由采样点

相邻差来估计：

$$H=\begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix}$$

下标表示 DOG 金字塔中某一尺度方向求导两次。

D 的主曲率和 H 的特征值成正比。令 α ，β 为特征值，则

$$\frac{Tr(H)^2}{Det(H)}=\frac{(\alpha+\beta)^2}{\alpha\beta}$$

$$\det(H)=\alpha\beta$$

该值在两特征值相等时达最小。Lowe 论文中建议阈值 T 为 1.2，即 $\frac{Tr(H)^2}{Det(H)}<T$ 时保留关键点，反之剔除。

3.关键点主方向分配

关键点主方向分配就是基于图像局部的梯度方向，分配给每个关键点位置一个或多个方向。所有后面的对图像数据的操作都相对于关键点的方向、尺度和位置进行变换，使得描述符具有旋转不变性。

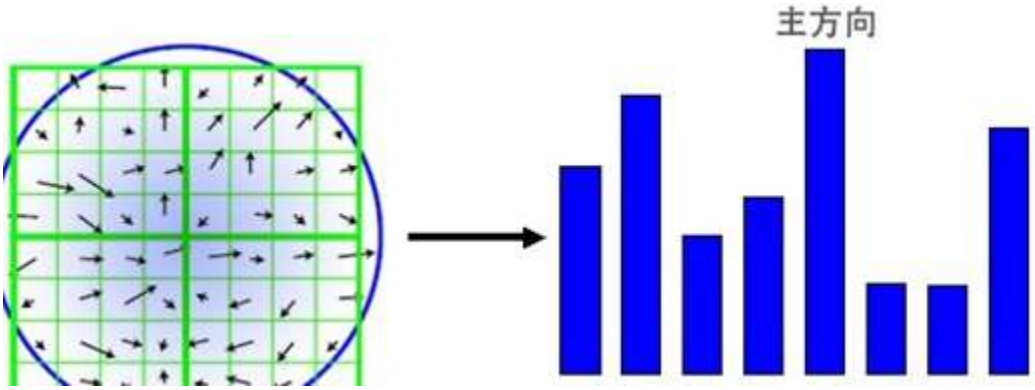
对于在 DOG 金字塔中检测出的关键点，采集其所在高斯金字塔图像 3σ 邻域窗口内像素的梯度和方向分布特征。梯度的模值和方向如下：

梯度幅值：
$$m(x,y)=\sqrt{\left(L(x+1,y)-L(x-1,y)\right)^2+\left(L(x,y+1)-L(x,y-1)\right)^2}$$

梯度方向：
$$\theta(x,y)=\tan^{-1}\left[\frac{L(x,y+1)-L(x,y-1)}{L(x+1,y)-L(x-1,y)}\right]$$

在完成关键点的梯度计算后，使用直方图统计邻域内像素的梯度和方向。梯度直方图将 0~360 度的方向范围分为 36 个柱(bins)，其中每柱 10 度。如下图所示，直方图的峰值方向代表了关键点的主方向，(为简化，图中只画了八个方向的直方

图)。



4.关键点的特征描述

通过以上步骤，对于每一个关键点，拥有三个信息：位置、尺度以及方向。接下来就是为每个关键点建立一个描述符，用一组向量将这个关键点描述出来，使其不随各种变化而改变，比如光照变化、视角变化等。这个描述子不但包括关键点，也包含关键点周围对其有贡献的像素点，并且描述符应该有较高的独特性，以便于提高特征点正确匹配的概率。

SIFT 描述子是关键点邻域高斯图像梯度统计结果的一种表示。通过对关键点周围图像区域分块，计算块内梯度直方图，生成具有独特性的向量，这个向量是该区域图像信息的一种抽象，具有唯一性。

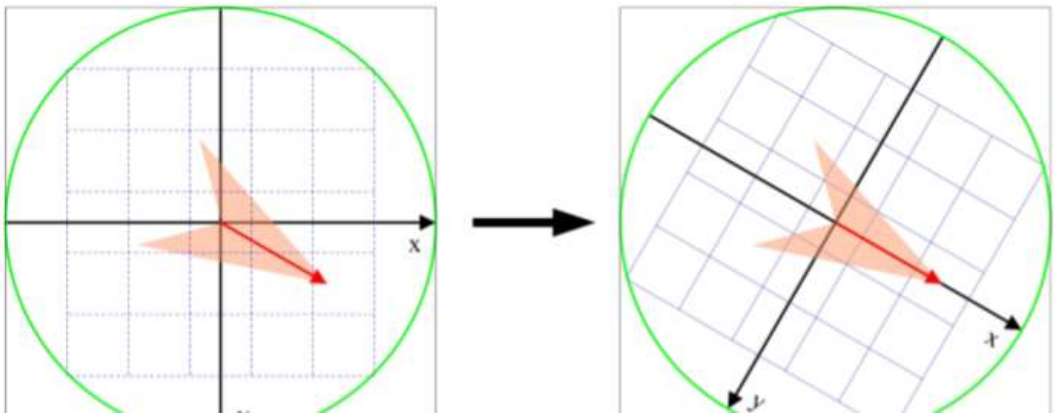
（1）计算描述子所需的图像区域

特征描述子与特征点所在的尺度有关，因此，对梯度的求取应在特征点对应的高斯图像上进行。将关键点附近的邻域划分为 d*d(Lowe 建议 d=4)个子区域，每个子区域做为一个种子点，每个种子点有 8 个方向。每个子区域的大小与关键点方向分配时相同，即每个区域有 $3\sigma\_oct$ 个子像素，为每个子区域分配边长为的矩形区域进行采样(个子像素实际用边长为 $3\sigma\_oct$ 矩形区域进行采样，考虑到实际计算时，需要采用双线性插值，所需图像窗口边长可知。在考虑到旋转因素(方

便下一步将坐标轴旋转到关键点的方向)

（2）将坐标轴旋转为关键点的方向

将坐标轴旋转为关键点的方向是为了确保旋转不变性，如下图所示：



旋转后邻域内采样点的新坐标为：

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \quad (x, y \in [-radius, radius])$$

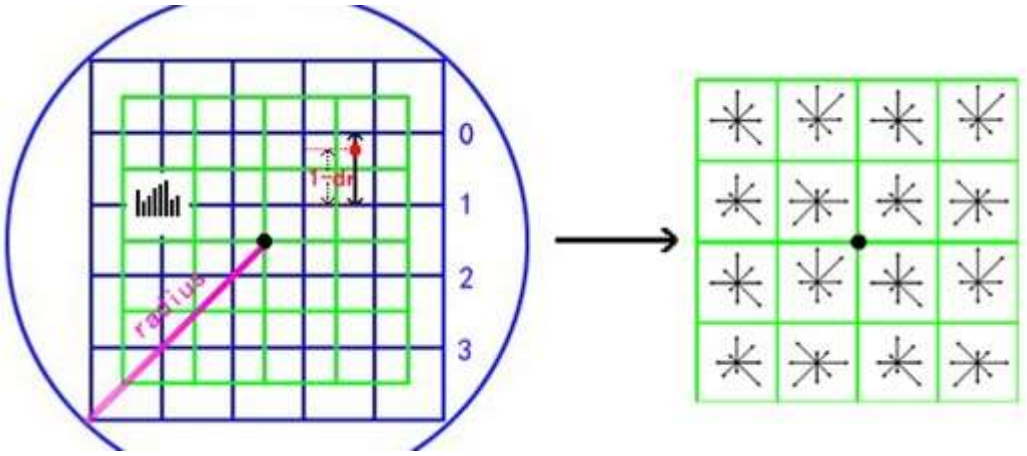
3）将邻域内的采样点分配到对应的子区域内

将子区域内的梯度值分配到 8 个方向上，计算其权值。旋转后的采样点坐标在半径为 radius 的圆内被分配到的子区域，计算影响子区域的采样点的梯度和方向，分配到 8 个方向上。

旋转后的采样点落在子区域的下标为

$$\begin{pmatrix} x'' \\ y'' \end{pmatrix} = \frac{1}{3\sigma\_oct} \begin{pmatrix} x' \\ y' \end{pmatrix} + \frac{d}{2}$$

（4）插值计算每个种子八个方向的梯度



如上图所示，将所得采样点在子区域中的下标(x",y")(图中蓝色窗口内红色点)线性插值，计算其对每个种子点的贡献。如图中的红色点，落在第 0 行和第 1 行之间，对这两行都有贡献。对第 0 行第 3 列种子点的贡献因子为 dr，对第 1 行第 3 列的贡献因子为 1-dr，同理，对邻近两列的贡献因子为 dc 和 1-dc，对邻近两个方向的贡献因子为 do 和 1-do。则最终累加在每个方向上的梯度大小为：

$$weight = w * dr^{k} * (1-dr)^{1-k} * dc^{m} * (1-dc)^{1-m} * do^{n} * (1-do)^{1-n}$$

其中，k,m,n 为 0 或为 1。

（5）描述符向量元素门限化

即把方向直方图每个方向上梯度幅值限制在一定门限值一下（门限一般取 0.2）

（6）描述符向量元素归一化

特征向量形成后，为了去除光照变化的影响，需要对它们进行归一化处理，对于图像灰度值整体漂移，图像各点的梯度是邻域像素相减得到，所以也能去除。

得到的描述子向量：

$$H = (h_1, h_2, \ldots, h_{128})$$

归一化后的特征向量：

$$L = (l_1, l_2, \ldots, l_{128})$$

则

$$l_i = \frac{h_i}{\sqrt{\sum_{j=1}^{128} h_j}}, \quad j = 1,2,3,\ldots$$

三、关键点匹配

分别对模板图（参考图，reference image）和实时图（观测图， observation image）建立关键点描述子集合。目标的识别是通过两点 集内关键点描述子的比对来完成。具有 128 维的关键点描述子的相似 性度量采用欧式距离。

模板图中关键点描述子：

$$R_i = \left(r_{i1}, r_{i2}, \cdots, r_{i128}\right)$$

实时图中关键点描述子：

$$S_i = \left(s_{i1}, s_{i2}, \cdots, s_{i128}\right)$$

任意两描述子相似性度量：

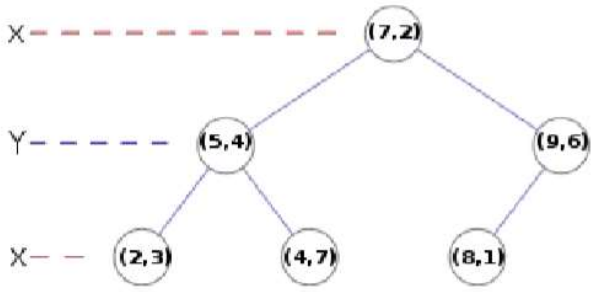
$$d\left(R_i, S_i\right) = \sqrt{\sum_{j=1}^{128}\left(r_{ij} - s_{ij}\right)^2}$$

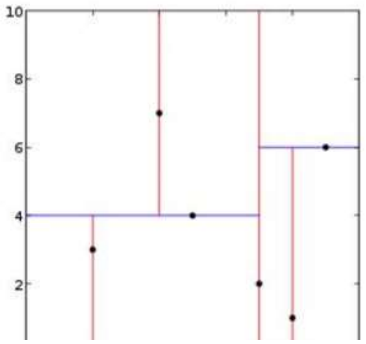
要得到配对的关键点描述子，$d\left(R_i, S_j\right)$ 需满足

$$\frac{\text{实时图中距离 } R_i \text{ 最近的点 } S_j}{\text{实时图中距离 } R_i \text{ 的次最近点 } S_p} < Threshold$$

由于关键点的匹配可以采用穷举法来完成，但是这样耗费的时间太多，一般都采用 kd 树的数据结构来完成搜索。搜索的内容是以目标图像的关键点为基准，搜索与目标图像的特征点最邻近的原图像特征点和次邻近的原图像特征点。

Kd树是一个平衡二叉树





问题求解

1.仅纵切的中文文字：这种情况相对简单，因为碎片只沿一个方向切割。拼接模型可以依赖文本行的连续性和字形的一致性。

实施步骤

1. **数据准备**：导入所有图片碎片，并对其进行预处理，如灰度转换、噪声过滤等。
2. **特征提取**：对每个碎片使用特征提取算法，获取关键点和描述符。
3. **碎片匹配**：通过比较描述符，找出最有可能匹配的碎片对。
4. **拼接优化**：应用优化算法确定所有碎片的最佳排列顺序。

（1）预处理

首先对每张图片进行预处理，包括:

- **灰度化**：将彩色图片转换成灰度图片，以减少计算复杂度。
- **二值化**：通过阈值分割将灰度图片转换为黑白图片，这样可以更清晰地区分文字和背景。

（2）特征提取

使用 OCR（光学字符识别）技术提取图片中的文字内容。这一步骤对于文字图片尤其重要，因为文字本身的形状和排列可以提供强有力的线索来辅助拼接。

（3）碎片匹配

基于提取的文字特征，进行碎片间的匹配。具体可以采用以下方法：

- **基于内容的匹配**：对每个碎片进行 OCR 识别，然后通过比较相邻碎片的文字内容来推断它们之间的关系。
- **边缘匹配**：利用图像处理技术检测每个碎片的边缘特征，如边缘线条和角点，然后根据这些特征对碎片进行匹配。

（4）优化算法

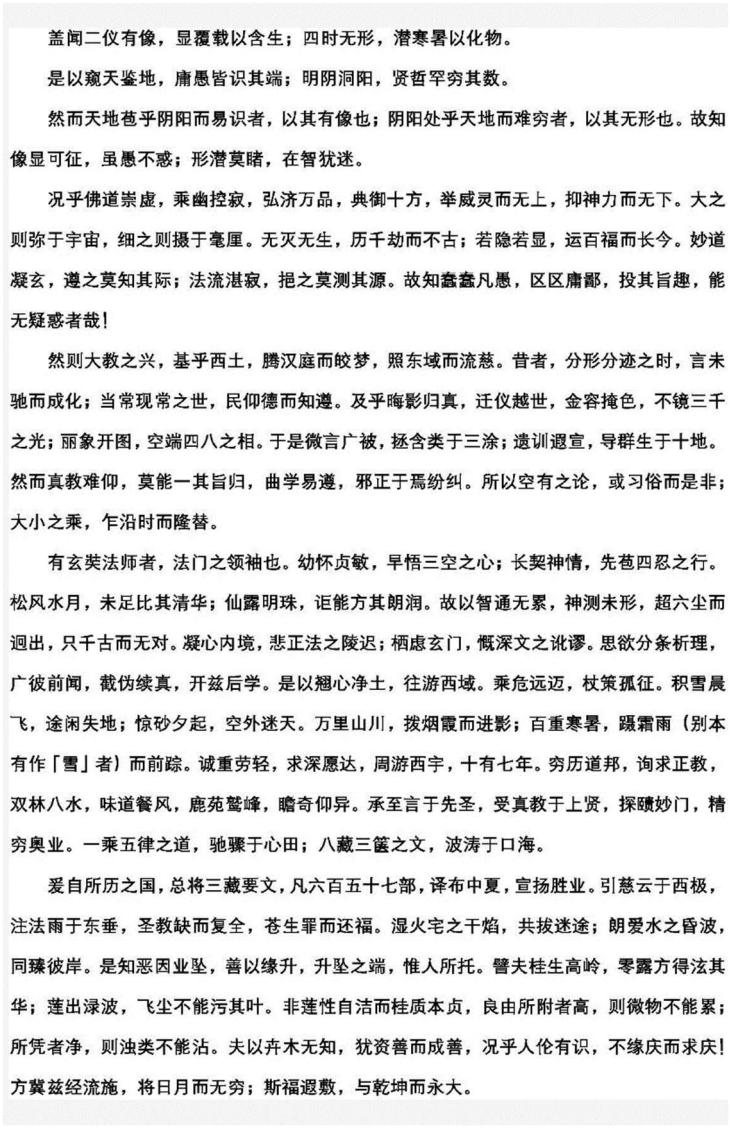
使用图像拼接中常见的优化算法来精细调整碎片位置：

- **全局优化**：可以使用图论中的最小生成树（MST）或最短路径算法来寻找最佳的拼接路径。
- **局部调整**：在全局拼接框架下，对局部区域进行细微调整，确保文字对齐自然，无明显的错位。

（5） 结果验证和调整

最后，需要有一个结果验证的步骤，可能包括人工干预来调整部分可能存在错误的拼接区域。可以设计一个交互式的 GUI 工具，允许用户查看拼接结果，并进行手动调整。

结果如下：



盖闻二仪有像，显覆载以含生；四时无形，潜寒暑以化物。

是以窥天鉴地，庸愚皆识其端；明阴洞阳，贤哲罕穷其数。

然而天地苞乎阴阳而易识者，以其有像也；阴阳处乎天地而难穷者，以其无形也。故知像显可征，虽愚不惑；形潜莫睹，在智犹迷。

况乎佛道崇虚，乘幽控寂，弘济万品，典御十方，举威灵而无上，抑神力而无下。大之则弥于宇宙，细之则摄于毫厘。无灭无生，历千劫而不古；若隐若显，运百福而长今。妙道凝玄，遵之莫知其际；法流湛寂，挹之莫测其源。故知蠢蠢凡愚，区区庸鄙，投其旨趣，能无疑惑者哉！

然则大教之兴，基乎西土，腾汉庭而皎梦，照东域而流慈。昔者，分形分迹之时，言未驰而成化；当常现常之世，民仰德而知遵。及乎晦影归真，迁仪越世，金容掩色，不镜三千之光；丽象开图，空端四八之相。于是微言广被，拯含类于三涂；遗训遐宣，导群生于十地。然而真教难仰，莫能一其旨归，曲学易遵，邪正于焉纷纠。所以空有之论，或习俗而是非；大小之乘，乍沿时而隆替。

有玄奘法师者，法门之领袖也。幼怀贞敏，早悟三空之心；长契神情，先苞四忍之行。松风水月，未足比其清华；仙露明珠，讵能方其朗润。故以智通无累，神测未形，超六尘而迥出，只千古而无对。凝心内境，悲正法之陵迟；栖虑玄门，慨深文之讹谬。思欲分条析理，广彼前闻，截伪续真，开兹后学。是以翘心净土，往游西域。乘危远迈，杖策孤征。积雪晨飞，途闲失地；惊砂夕起，空外迷天。万里山川，拨烟霞而进影；百重寒暑，蹑霜雨（别本有作「雪」者）而前踪。诚重劳轻，求深愿达，周游西宇，十有七年。穷历道邦，询求正教，双林八水，味道餐风，鹿苑鹫峰，瞻奇仰异。承至言于先圣，受真教于上贤，探赜妙门，精穷奥业。一乘五律之道，驰骤于心田；八藏三箧之文，波涛于口海。

爰自所历之国，总将三藏要文，凡六百五十七部，译布中夏，宣扬胜业。引慈云于西极，注法雨于东垂，圣教缺而复全，苍生罪而还福。湿火宅之干焰，共拔迷途；朗爱水之昏波，同臻彼岸。是知恶因业坠，善以缘升，升坠之端，惟人所托。譬夫桂生高岭，零露方得泫其华；莲出渌波，飞尘不能污其叶。非莲性自洁而桂质本贞，良由所附者高，则微物不能累；所凭者净，则浊类不能沾。夫以卉木无知，犹资善而成善，况乎人伦有识，不缘庆而求庆！方冀兹经流施，将日月而无穷；斯福遐敷，与乾坤而永大。

可以看出，使用构建的模型完美解决了问题。

2.纵横切的中文文字图片拼接：这里涉及更复杂的拼接，因为图片被纵向和横向切割。模型需要同时考虑两个方向的对齐和匹配。

但是思路没有变化，相对于第一问没有什么特别需要注意的地方。只是多了两个方向上的匹配。

3.含噪声的纵横切文字图片拼接：在有噪声的情况下，正确拼接图片更具挑战性。这需要开发鲁棒的算法来抵抗噪声干扰，并准确匹配图片碎片。

（1） 噪声去除

在处理任何 OCR 任务或图像拼接前，首先需要尽可能去除图片中的噪声。这可以通过各种图像处理技术实现：

- **高斯模糊**：可以帮助减少图像噪声，但需谨慎使用以避免过度模糊。
- **中值滤波**：非常适合去除椒盐噪声，同时保持边缘信息。
- **双边滤波**：保留边缘的同时减少噪声，适合在需要保持边缘清晰度时使用。

（2） 文字识别与 OCR 的增强

噪声会影响 OCR 的准确性，因此可能需要采用一些技术来增强 OCR 过程：

- **使用更加鲁棒的 OCR 配置**：例如，调整 Tesseract 的参数，使用 oem 和 psm 模式，可以改善噪声环境下的识别效果。
- **字符修正算法**：对 OCR 结果进行后处理，根据上下文纠正识别错误。

（3） 特征提取与匹配

在噪声环境下，基于内容的匹配可能不够准确，此时可以利用图像本身的结构特征来辅助匹配：

- **边缘检测和匹配**：通过检测图像碎片的边缘，可以找到可能的拼接线。
- **关键点和描述子匹配**：使用 SIFT、SURF 或 ORB 等算法提取关键点和其描述子，然后进行匹配。

（4） 优化与全局拼接策略

由于噪声和可能的识别错误，可能需要一个全局优化策略来确定最佳拼接方式：

- **图论算法**：如最小生成树或最短路径，帮助找到最可能的拼接顺序。
- **能量最小化**：通过定义一个能量函数（例如，拼接边界的不连续性最小化），并使用优化算法（如模拟退火或遗传算法）求解。

（5） 交互式修正与 GUI

由于噪声带来的不确定性，允许用户交互式地检查和修正拼接结果非常重要。可以开发一个简单的 GUI 工具，让用户可以：

- **手动调整拼接**：用户可以拖动碎片，手动调整它们的位置。
- **结果验证**：提供工具让用户可以轻松标记和修正错误的拼接。

4.彩色图片的纵横切拼接：彩色图片拼接不仅要处理多个颜色通道，还要解决因色彩差异引起的匹配问题

(1) 图像对齐和特征匹配

首先，需要在图像之间找到相匹配的特征点。对于彩色图片，我们通常使用以下特征检测和匹配算法：

- **SIFT（Scale-Invariant Feature Transform）**：这是一种检测和描述本地图像特征的算法。它在图像缩放和旋转下保持不变性，非常适用于从不同角度或距离拍摄的图像之间的匹配。
- **SURF（Speeded-Up Robust Features）**：这是 SIFT 的加速版本，它更快但也非常鲁棒。
- **ORB（Oriented FAST and Rotated BRIEF）**：这是一个在计算效率和内存使用方面优化的算法，适合于实时应用。

这些算法将帮助我们识别图像之间的关键点，并找到相应的匹配对。

(2) 图像变换和拼接

一旦找到匹配点，接下来的步骤是估计图像之间的几何变换（例如仿射变换或透视变换），这可以通过 RANSAC（Random Sample Consensus）算法来鲁棒地估计：

- **RANSAC**：这个算法通过迭代方式选择一组随机样本来估计模型参数，并

计算所有数据点的拟合度，选择最佳拟合模型。

⑶ 混合和平滑处理

为了使拼接的图像看起来更自然，需要使用图像混合技术来平滑过渡区域，常见的技术包括：

- **多波段混合**：这种技术在多个频率层次上混合图像，可以减少拼接边界处的视觉伪影。
- **泊松融合**：这种方法可以很好地处理图像颜色和光照的变化，以实现无缝拼接。

应用以上思路，结果图如下

很好的匹配了彩色图片的拼接问题。

模型评价与优化

模型评价

优点

1. **实用性**：
    - 代码解决了图像显示过大的问题，使图像尺寸适应显示屏幕或用户指定的尺寸。

- 通过调整图像尺寸，用户可以更清楚地查看图像的细节，不必担心图像超出屏幕边界。

2. **易用性**：
   - 此模型所用方法简单直观，即使是不具备深厚编程背景的用户也能轻松理解和使用。
   - 只需要修改几个参数，即可适应不同的应用场景，如调整图像的宽度和高度比例。
3. **灵活性**：
   - 用户可以根据需要自由设定目标尺寸或缩放比例，这为处理各种不同尺寸的图像提供了便利。
   - 代码可以轻松集成到更大的图像处理或视觉系统中，为其他功能如图像分析、编辑或批量处理提供基础。

优化建议

1. **自动尺寸调整功能**：
   - 开发一种机制，根据用户的显示设备自动调整图像尺寸。例如，可以检测屏幕分辨率，并据此自动设定最佳的图像尺寸。
2. **用户界面开发**：
   - 创建一个简单的 GUI，允许用户通过拖动滑块来调整图像尺寸，并实时预览调整结果。这将使程序更易于使用，并扩大其潜在用户基础。
3. 改进图像插值方法：
   - 使用 cv2.INTER_AREA 对于缩小图像、cv2.INTER_CUBIC 或

cv2.INTER_LANCZOS4 对于放大图像，以优化图像质量。

附：源代码在同名压缩包下