碎纸片的拼接复原方法研究

摘　要

本文研究了不同切割方式下中、英文碎纸片的拼接复原方法。文中充分利用不同图片边缘数据信息的匹配度、行间距及字高等文字排版数据建立了一系列拼接复原模型。其中对于纵横切碎片复原问题，按照先找各裁剪行的组成碎片，然后实现行内拼接，最后对裁剪行之间的拼接顺序进行求解。

对仅纵切方式下的中、英文碎纸片的复原问题，本文建立了基于边缘列灰度值向量相似性的图像拼接复原模型：提取图像矩阵的左、右边缘灰度列向量，引入Minkowski距离，来衡量一个图像右边缘和另一个左边缘的匹配度，依据最佳匹配进行拼接。将仅纵切方式下中、英文碎片的复原结果列入表1和表2中，将复原结果图列入附录5和附录8中。

在纵向且横向切割方式下的单面图片复原问题中，引入分治的思想，缩小匹配范围，解决了由于碎片长度减小导致的碎片匹配度低的问题。首先对中、英文分别建立了基于像素点投影和灰度计数直方图的分类方式，通过投影和灰度计数直方图将二维的字体转化为一维的像素数向量，并且定义了图片偏移量和行距的概念。其中，由于英文字母的字体不规则，我们根据像素数直方图的特征对其进行了规则化处理。通过对像素数向量的计数和偏移量代表的行距信息将图片按行分为11类。之后对每一类的碎纸片，借用第一问的复原模型和少量人工干预进行行内拼接复原，然后我们使用偏移量和行间距完成了行与行之间的匹配，得到了碎纸片复原图。我们分别将中、英文碎片的复原结果列入表4，5中，将复原结果图列入附录1，2中。中、英文复原过程中分别人工干预12和15次（每次干预为手工移动一个碎纸片），干预项目见附录4、7。

在纵向和横向两种切割方式下的正反面图片复原问题中，我们首先按照第二问的分类模型完成分类。由于正反面切割导致图片匹配复杂度增加，本文在问题一的复原模型基础上，以碎片拼接时正反两面的Minkowski距离之和来刻画碎片拼接时的匹配度，扩大了不匹配图片之间的差异，进而增加了同类碎片之间的匹配准确度。然后使用与问题二相同的算法完成了行内拼接复原。对于行间拼接，将问题一中边缘灰度值列向量的相似度概念推广，结合问题二的模型综合求解。我们将碎片的复原结果列入表8，9中，将复原结果图列入附录10中，复原过程中人工干预38次，干预项目见附录11。

关键词：Minkowski距离　分治思想　偏移量　像素投影　　灰度计数直方图

一、 问题的重述

1.1 问题的背景

前东德情报机构“斯塔西”官员将大量绝密文件撕成 6 亿多块碎纸片后丢进 16000 个垃圾袋。历史学家认为恢复这些文件意义重大，但如果用人工手段进行恢复，将耗费至少 400 年时间。大量的纸质物证复原工作目前基本上都是 以手工方式完成的。一旦碎纸的数量增大到几百甚至上千块的时候，如果仍然依靠手工完成，不但耗费大量的人力、物力，而且还可能对物证造成一定的损坏。目前在国际上，德国等发达国家对破碎文件的自动修复技术已经进行了相当长时间的研究。但是由于技术封锁的原因，我们所能够搜集的资料非常有限。而在国内，还没有类似的研究成果问世。因此，结合碎纸自动拼接在司法技术鉴定中的应用这一背景，把计算机视觉和模式识别应用于碎片复原，开展对碎纸自动拼接技术的研究具有重要的现实意义。

1.2 问题的提出

破碎文件的拼接在司法物证复原、历史文献修复以及军事情报获取等领域都有着重要的应用。传统上，拼接复原工作需由人工完成，准确率较高，但效率很低。特别是当碎片数量巨大，人工拼接很难在短时间内完成任务。随着计算机技术的发展，人们试图开发碎纸片的自动拼接技术，以提高拼接复原效率。请讨论以下问题：

1.对于给定的来自同一页印刷文字文件的碎纸机破碎纸片（仅纵切），建立碎纸片拼接复原模型和算法，并针对附件 1、附件 2 给出的中、英文各一页文件的碎片数据进行拼接复原。如果复原过程需要人工干预，请写出干预方式及干预的时间节点。

2.对于碎纸机既纵切又横切的情形，请设计碎纸片拼接复原模型和算法，并针对附件 3、附件 4 给出的中、英文各一页文件的碎片数据进行拼接复原。如果复原过程需要人工干预，请写出干预方式及干预的时间节点。复原结果表达要求同上。

3.上述所给碎片数据均为单面打印文件，从现实情形出发，还可能有双面打印文件的碎纸片拼接复原问题需要解决。附件 5 给出的是一页英文印刷文字双面打印文件的碎片数据。请尝试设计相应的碎纸片拼接复原模型与算法，并就附件 5 的碎片数据给出拼接复原结果，结果表达要求同上。

二、 问题的分析

2.1 问题一的分析

问题一要求我们考虑纸片仅在纵切情况下的拼接复原方式，考虑纵切情况下的纸片保留了原文的行序数和行间距等信息，我们主要通过研究每一个纵切纸片与其他纵切纸片在边界处的相似程度，来衡量任意两个纸片是否应该相接，以及是左相接或是右相接。因为本问题的附件内容只有切碎纸片的图片，为了衡量两个纸片之间的相似程度，考虑到附件中的图片都是黑色字体和白色纸面，每一张图片都有对应的灰度值矩阵，我们可以使用软件读取出各图片的灰度值信息，然后通过对灰度值的分析和处理来确定纸片是否相接。

因为两张纸片的拼接与否一方面取决于文字内容的连续性，另一方面取决于文字位置的衔接程度，在碎纸片数量较大情况下是无法考虑内容上的连续性的，所以本文

优先考虑文字在位置上的衔接性，最后通过考虑文字内容上的连续性来修正我们的结果。本文考虑对图片的左、右边界点的灰度值数据进行统计处理，我们首先统计每一张图片在左、右边界点处的灰度值，然后利用 Minkowski 距离中的绝对值距离公式计算各组数据之间的“距离”，我们就通过这个“距离”值来衡量各组灰度值样本之间的相似性，相似性最好的两组数据对应的两个图片的边界就是我们需要拼接的位置。

2.2 问题二的分析

问题二要求我们考虑纸片在除了纵切还有横切情况下的拼接复原问题。对比仅考虑纵切的情况，横切极大地破坏了文本原有的统一的行间距信息，我们除了要考虑各碎片在横向上能否构成一行，还要考虑碎片在纵向上能否构成一列。因为完整文本的各行行距相同，我们首先考虑利用各碎片包含的行距信息将碎片进行分类，每一类中的碎片就在位置上处于同一行中，然后我们对各行的文件碎片进行拼接，我们同样使用绝对值距离来刻画各图片间是否应该拼接。在拼接完各行的碎片后，由于横向切线包含的灰度值信息量很可能不如问题一中纵向切线包含的多，所以我们在将所有的行拼接为完整的图片时，不使用边界“距离”来衡量图片能否拼接，而是通过考虑行间距能否匹配来确定两张图片是否应该拼接。

在衡量图片是否位于同一行位置时，我们首先定义基于像素点计数的字高和行距等概念，通过统计附件3和附件4中文字的字高和行距，得到中文和英文的文字特征。然后我们设定图片偏移量的概念，即根据某张图片首行字体最下端的灰度值非255的像素点距图片上边缘的像素点个数跨度值，来衡量这张图片的偏移量，我们由之前得到的文字特征，分别对中英文设定合理的分类标准，我们就将偏移量在同一标准下的图片分为一类。我们对中、英文建立的不同的分类标准中，因为中文字体大小基本相同，所以偏移量是由首行字体的下端来计算，而英文字母的纵向跨度大，字体大小相差很大，为了减小字体差异带来的误差，偏移量是由首行完整字体的下端来计算。

在将同一类中的图片拼接为一张图片时，我们的算法虽然是建立在问题一的拼接算法基础上的，但是由于同一个算法对不同类型数据的容错率不同，我们还需要对原算法可能带来的错误设计改正算法。

2.3 问题三的分析

问题三要求我们考虑双面打印的纸片在纵切和横切情况下的拼接复原问题。本问题相对于问题二，除了图片总数增加以外，图片正反面的引入也增加了我们在拼接时的约束条件。经过统计我们发现同一碎片正反两面的偏移量相差不超过 2 个像素点，所以在我们提出的分类标准下，同一张纸片的正反两面可以被分入同一类中，这时我们还是可以将所有的图片分为 11 类，使每一类含有 38 张图片。

在对每一类内部进行拼接复原时，我们采用类似于问题一中的基于灰度值距离的拼接复原算法，但是由于图片数量较多，一些不在同一行中的图片可能具有相似的行距信息，导致我们在进行图片分类时，得到某几类的图片数量为 38 的整数倍，从而导致我们在对该类的拼接时出现较多的人工干预。所以我们在原拼接复原算法的基础上提出两点改进的做法：为了提高拼接复原时的精确度，我们除了要考虑图片的正面和其它图片正面拼接时的相似程度，同时还要考虑反面两条边拼接时的相似程度，用两面的灰度值向量距离之和来衡量总的相似程度；对数量过多的类别，修正我们的分类方式，进行更加精细的分类。

三、 模型的假设

1．同一页的印刷文字格式均相同，段间距、行间距均相同；
2．同一个附件中给出的各破碎纸片规格相同；
3．附件图片在读取时不会发生像素丢失；
4．题目附件中给出的碎片数量完整。

四、 符号说明

符号	说明
$H(i)$	第 i 张图片的灰度值矩阵
$H_L(i)$	第 i 张图片灰度值矩阵的首列列向量
$H_R(i)$	第 i 张图片灰度值矩阵的末列列向量
$d(H_1(i), H_n(j))$	第 i 张图片的左边界和第 j 张图片的右边界的匹配度
d_i	第 i 张图片的偏移量

注：其它符号将在下文中给出具体说明。

五、模型的建立与求解

5.1 问题一：纵切碎纸片拼接复原模型

5.1.1 模型的建立

我们首先使用 MATLAB 软件中具有图像读写功能的 imread()函数读取附件 1 中的 19 张破碎纸片图片，得到了所有图片的灰度值矩阵之集 H，我们记矩阵中各个元素的灰度值为 $h_{ij}(i)$，所以我们得到第 i 张图片的灰度值矩阵 $H(i)$ 为

$$H(i)=\begin{bmatrix} h_{11}(i) & \cdots & h_{1n}(i) \\ \vdots & \ddots & \vdots \\ h_{m1}(i) & \cdots & h_{mn}(i) \end{bmatrix}, i=1,2,\cdots,19, \text{其中} m=1980, n=72.$$

我们读取的图片像素点灰度值的取值为[0,255]中的整数，通过查阅资料得知，灰度值为 0 代表黑色，255 代表白色。

由 2.1 中的问题分析我们知道，想要确定两图片能否拼接，我们可以根据图片左、右边界的灰度值情况做出相似性判断。考虑到我们只需要使用矩阵在左、右边界处的数值，我们将 $H(i)$ 的第一列和最后一列的列向量记为

$$H_L(i)=\begin{bmatrix} h_{11}(i) \\ \vdots \\ h_{m1}(i) \end{bmatrix}, H_R(i)=\begin{bmatrix} h_{1n}(i) \\ \vdots \\ h_{mn}(i) \end{bmatrix}, i=1,2,\cdots,19. \quad (1)$$

在得到各图片边界处的灰度值向量之后，为了衡量任何两个图片拼接边界处的相似程度，即为了度量两个已知的样本数据在空间中的相似性，我们引入 Minkowski 距离的一种特殊形式——绝对值距离：

$$d(x,y)=\sum_{k=1}^{q}|x_k-y_k|.$$

参考绝对值距离公式，本文定义第i张图片的左边界和第j张图片的右边界的距离为

$$d(H_1(i),H_n(j))=\sum_{k=1}^{m}\left|h_{k1}(i)-h_{kn}(j)\right|, i,j=1,2,\cdots,19且i\neq j. \quad (2)$$

也即我们是使用拼接后任何两个相邻像素点位置的灰度值的差值之和来衡量任意两张图片在相接时的匹配程度。当距离值越大时，说明当前所取的这两张图片的两条边匹配度低，不应该拼接，当距离值越小时，说明当前的两张图片的边界匹配度高，应该拼接。

在通过(2)式计算出各图片左、右边界的距离后，我们对于某张固定的图片i，分别对其左、右边界寻找与其距离最小的图片$l^*(i)$和$r^*(i)$，其中

$$\begin{cases} l^*(i)=\arg\min\limits_{l\neq i} d(H_L(i),H_R(l)), \\ r^*(i)=\arg\min\limits_{r\neq i} \{d(H_L(r),H_R(i)). \end{cases} \quad (3)$$

由此，本文建立了基于图片灰度值统计量的纵切碎纸片拼接复原模型，我们给出该拼接复原模型的算法如下：

Step1.依取19张图片的灰度值矩阵放入集合α中，建立空集合β；

Step2.计算每张图片右侧边缘与其他图片左侧边缘的 Minkowski 距离；

Step3.找出左侧空白像素最多的图的矩阵记为K_l，并放入集合β中；

Step4.寻找集合α中与K_l的 Minkowski 距离最小的图片矩阵记为K_r，拼在K_l右侧，将K_r从集合α中删除并放入集合β中，将K_r赋给K_l，重复 step4 直到集合α为空。

5.1.2 模型的求解

我们首先对附件 1 中的中文文件碎片使用拼接算法进行拼接求解。我们先统计出各图片灰度值的$H_L(i)$和$H_R(i)$向量，具体结果列入附录 1 中，然后根据(3)式求解得到

了中文文件碎片的拼接复原方式，我们将破片序号按复原后顺序填入下表（表 1），我们同时将拼接结果做成拼接复原示意图（见附录 1），该求解结果是程序的直接输出，并没有手动干预的步骤。

表 1　附件 1 中文碎片拼接复原结果表

[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

我们通过观察复原示意图(附录 1)可以发现,图中的文字不仅具有很高的衔接性，而且拼接后句子的含义也是连续的。

我们使用相同的算法对附件 2 中的英文文件碎片进行了拼接复原。我们将破片序号按复原后顺序填入下表（表 2），同时做出了拼接复原示意图（图片见附录 2），该求解结果也是程序的直接输出，并没有手动干预的步骤。

表 2　附件 2 英文碎片拼接复原结果表

[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

我们通过观察复原示意图（附录 2）可以发现，图中的英文单词间不仅具有很高的衔接性，而且拼接后句子的含义也是连续的。

5.1.3 结果分析

我们通过 5.1.2 中的求解结果可以看到，两种文字的文件碎片在我们给出的拼接复原算法下都得到了极好的还原结果。我们的算法可以在无手动干预的情况下给出最优解的原因，一方面是因为问题一中的两个附件的文件碎片只有 19 个，数量较少，并且都是纵切后的碎片，只需要考虑图片左、右两侧灰度值的相似性，复杂度低。另一方面是因为我们在 5.1.1 中给出的算法对本问题的数据容错率高，因为附件 1 和附件 2 中的图片在纵向上的像素点远多于横向的，而纵向正是我们要拼接的位置，这导致了我们统计了大量的绝对值距离，所以我们在衡量图片边缘的相似性时，(2)式中大量的求和运算“吸收”了极少数相邻像素点的计算距离比实际值大的错误，使得我们的结果并没有因为极少数的像素点位置极端而发生改变。

5.2 问题二：纵、横切碎纸片拼接复原模型

5.2.1 基于像素点投影和计数的中文图片分类方式

考虑到汉字本身的结构方正，大部分文字的尺寸相同，所以我们可以通过各文字在灰度值矩阵中所占像素点的尺寸来衡量文字的大小，也可以用来衡量行的间距信息。

根据 2.2 的问题分析，我们定义第 i 张图片的偏移量 d_i，它的含义是第 i 张图片的首行（不一定是完整行）文字最下端的有色像素点与图片上边缘的距离，这里的距离就是像素点的个数。为了统计这个数，我们将图片横向投影，投影后图像仅有一列像素点，投影的原则是：只要某行的像素点中存在有色像素点，那么这一行在投影后的像素点也为有色。完成投影后，我们在计算偏移量时，就去计算投影区域最下端距离图片上边缘的距离。为了直观说明，我们做出了附件 3 中编号为 000 的图片的中文偏移量示意图（图 1）：

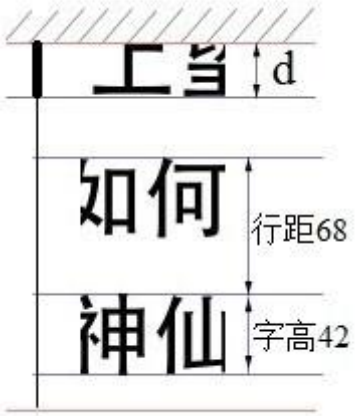


图 1　中文偏移量定义示意图

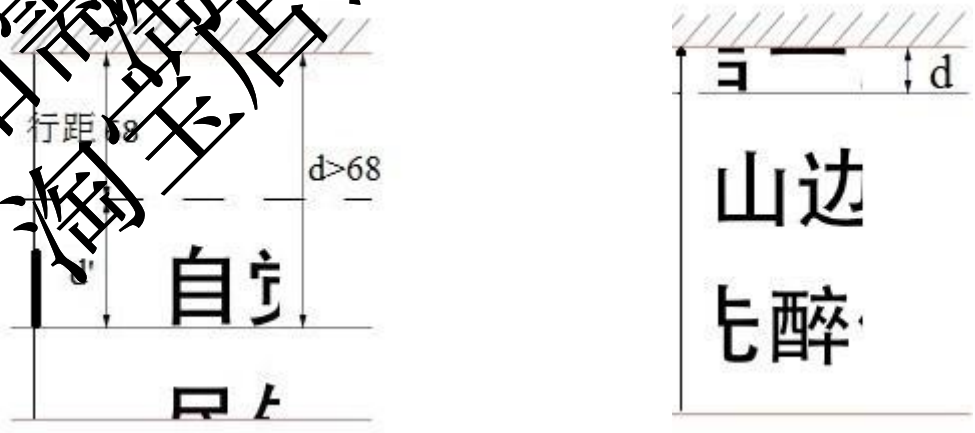
图 1 中的偏移量的定义针对附件 3 中的绝大多数图片是有效的，能成功刻画他们的行距信息。但是经过我们统计发现，附件 3 中还有个别图片因为文字换行或是个别中文的特殊结构导致该定义下的偏移量较为特殊，我们需要对这些图片的偏移量做出相应的修正。为了精确地修正特殊的偏移量，我们定义了图片中文字的字高和行距：字高是某文字的顶端位置水平线到底端位置水平线间的像素点个数；行距是某行文字顶端水平线至下一行文字顶端水平线的像素点个数。为了直观表示，我们也将这两个概念标注在图 1 中。我们对附件 3 中的文字进行了统计，得到大部分汉字的字高为 42 个像素点，行距 N 为 68 个像素点。

在建立相关的概念后，我们给出正文中分析出的两类特殊现象并给出其解决方法：

(1)第一类特殊现象：当文字的偏移量大于行距时，程序实际读取的是第二行的文字的偏移量，这时将这张图片为其读取第一行文字偏移量的图片进行拼接显然是不合理的。所以我们对程序读出的偏移量做差处理，令 $d' = d - 68$；

(2)第二类特殊现象：有些文字的独特结构使得一行文字被误识为两行，所以我们需要根据文字的实际内容，手动修改程序读取偏移量的位置。

为了直观表示对这两类特殊的处理，我们做出了附件 3 中编号为 014 和 058 的偏移量示意图如下图 2)：



a.第一类特殊现象示意图　　**b.第二类特殊现象示意图**

图 2　存在特殊现象下的中文偏移量定义示意图

通过这样的偏移量定义，我们定义了所有中文图片的偏移量，并且在一些定义错误的情况下作了适当的修正，保证了我们使用偏移量来衡量图片行距信息时得到的都是每一行图片的正确信息。在分类时，我们将所有照片的偏移量按照从小到大的顺序排序，依序按每一组需要的照片数量选取就完成了对照片的分类。

5.2.2 基于像素点投影和计数的英文图片分类方式

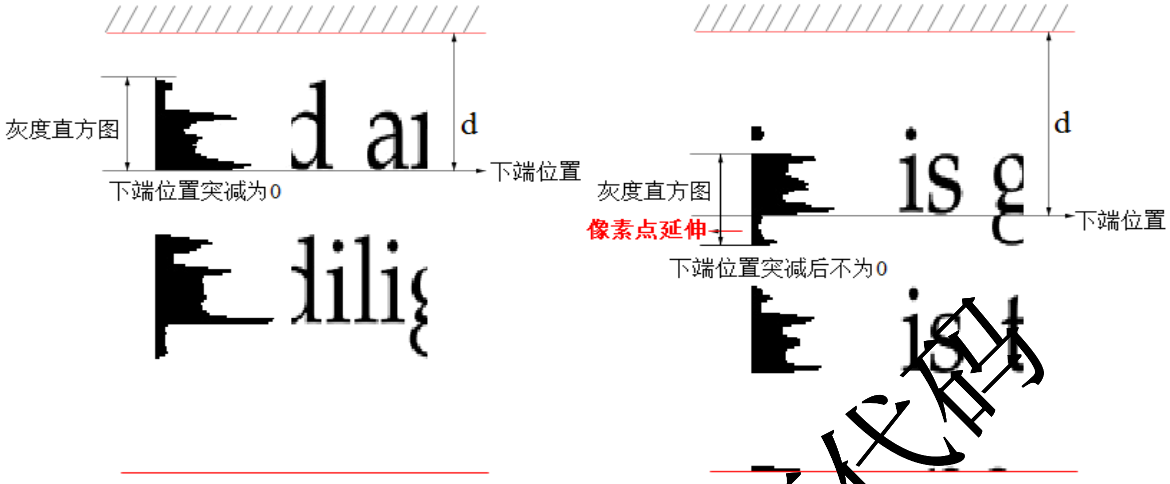
考虑到英文单词笔画数较少，英文的字体结构多变，不同字母组合出的单词尺寸也不相同，这导致中文碎片的拼接模型不能很好的应用在本问题中，所以我们在定义第i张图片的偏移量d_i时，需要对英文字母不同的特殊结构，做出对应的统计处理。英文图片偏移量的含义是第i张图片中出现首行完整文字最下端的有色像素点距离图片上边缘的像素点个数。为了对英文字母划定相同的规格，我们在这里定义的文字下端，是类似于 “a”、“b”、“c”这样在四线格中占上两格的字母的下端，所以我们需要对类似于“y”、“g”、“j”等使用到第三格的字母的下端部分做删除处理。为了直观说明，我们做出了附件 4 中编号为 000 和 011 的图片的英文偏移量示意图，同时标出了我们定义的英文文字下端位置（图 3）。

(a)图片 000 的偏移量示意图　　(b)图片 011 的偏移量示意图

图 3　英文字母偏移量定义示意图

为了实现程序对字母下端位置的识别，我们仍然将图片横向投影，但是因为英文字母左拐向处的像素点位置分布不统一，字母字体形状也不统一，仅衡量投影区域的位置不能正确反映字母的位置，所以我们还要衡量投影区域的“厚度”，所以我们改变了投影的规则，将照片每一行的所有像素点横向投影，我们统计每一行投影时的有色像素点的个数，做出灰度值直方图。通过对英文字母的字体结构进行分析，我们可以根据直方图在字母下端发生突变后是否还有一段延续的像素点来判断该行字母中是否含有类似于“y”、“g”和“j”的字母。不同的字母组合会带来不同的直方图结构，所以我们在得到不同的直方图后，就可以判定是哪一类的字母组合带来的这一结果，对于不同类的字母组合，我们分别提出对应的偏移量修正方法。

我们以附件 4 中标号为 000,002,007 和 169 的图片为例，分别做出他们的像素点投影示意图，并且画出灰度直方图如下（图 4）：



(a)附件 4 中标号为 000 的图片投影示意图　**(b)附件 4 中标号为 002 的图片投影示意图**

(c)附件 4 中标号为 007 的图片投影示意图　**(d)附件 4 中标号为 169 的图片投影示意图**

图 4　附件 4 的英文字母投影示意图

通过观察图 4 我们可以看到，当图片的完整首行中含有“y”、“g”、“j”等下端较长、干扰到我们读取偏移量的字母时，会在首行投影直方图的突减位置之后还有一段有色像素点的延伸，我们读取到的这段延伸的像素点，就是我们在计算偏移量时需要删去的部分。

在我们对所有照片进行是否发生特殊突减的判断以及删去所有发生这种突减的位置之后，就保证了我们的图片处于一个相同的规格之下，这时就可以准确的计算出各张照片的偏移量，然后就可以根据偏移量代表的行距信息对照片进行分类。我们同时注意到，英文也会出现类似于我们在 5.2.1 中提到过的两种特殊情况，我们对附件 4 中的英文统计得到英文行间距为 64 个像素点。对于第一类特殊情况，字母至图片上边缘的间距大于行间距时，我们对偏移量做差处理，即令 $d'' = d - 64$；对于第二类特殊情况，因为引起该类特殊情况的原因是个别英文标点或者字母“i”、“j”上端的点，并且他们在尺寸上有明显的特征，所以我们可以控制程序识别读取偏移量的位置。

在分类时，我们将所有照片的偏移量按照从小到大的顺序排序，依序按每一组需要的照片数量选取就完成了对照片的分类。

5.2.3 模型的建立

我们首先根据对某类文字的偏移量定义和出现特殊情况的处理方式，求出第i张图片的偏移量$d_i(i=1,2,\cdots,209)$，记偏移量矩阵为

$$D=(d_1,d_2,\cdots,d_{209}).$$

我们将D中的元素按照由小到大的顺序重新排列，记排序后的偏移量向量为

$$D'=(a_1,a_2,\cdots,a_{19},b_1,\cdots,k_{19}), \tag{4}$$

若存在大于 0 的正整数M_1,M_2，使得D'中的元素满足

$$\begin{cases} |a_{19}-a_1|\le M_1,|b_{19}-b_1|\le M_1,\cdots,|k_{19}-k_1|\le M_1, \\ |b_1-a_{19}|\le M_2,|c_1-b_{19}|\le M_2,\cdots,|k_1-j_{19}|\le M_2 \\ M_1\le M_2. \end{cases} \tag{5}$$

成立，则$a_i,b_i,\cdots,k_i(i=1,2,\cdots,11)$分别为一类图片。这样我们就把得到的数值接近（小于 2 个像素点）的偏移量对应的图片归为了一类。

对所有图片进行分类之后，我们对于每一类中的所有图片使用 5.1.1 中建立的拼接复原算法即可完成该类的拼接。由 5.1.3 知，原来的算法对于问题一中的附件数据容错率高，在本问题中，考虑到每一张图片的纵向像素点个数较少，而且图片总数较多，原算法针对本问题的容错率会有所下降，所以我们需要在考虑原算法的基础上，提出对可能出现的错误的改正算法。

原算法在求解时可能会出现两种错误，一种是因为文字笔画恰好在某张图片的边缘停止，导致根据(2)式求得的某两张不应拼接的图片的边缘相似度甚至比应该拼接的大，另一种是因为两张不应该拼接的图片边缘都是空白，导致根据(2)式得出这两张图片应该拼接。由于这两种错误的本质是因为我们设计的算法无法识别图片中存在的特殊信息，所以我们必须在出现错误的时候通过适当的人工干预来改正算法。假设我们计算得出第i张图片右边拼接第j张图片，我们提出改正算法如下：

Step1，按照第一问的模型通过计算机根据$d(H_L(k),H_R(i))$找出最优匹配情况，进入下一步；

Step2，从形象上观察，是否具有明显的不连贯处，有则手工调整，将不连贯图片i和k处的$d(H_L(k),H_R(i))$改成无穷大，或者找到i实际的下一张图片j，将$d(H_L(k),H_R(j))$改成 0，重复 **step1** 直到改行匹配正确；

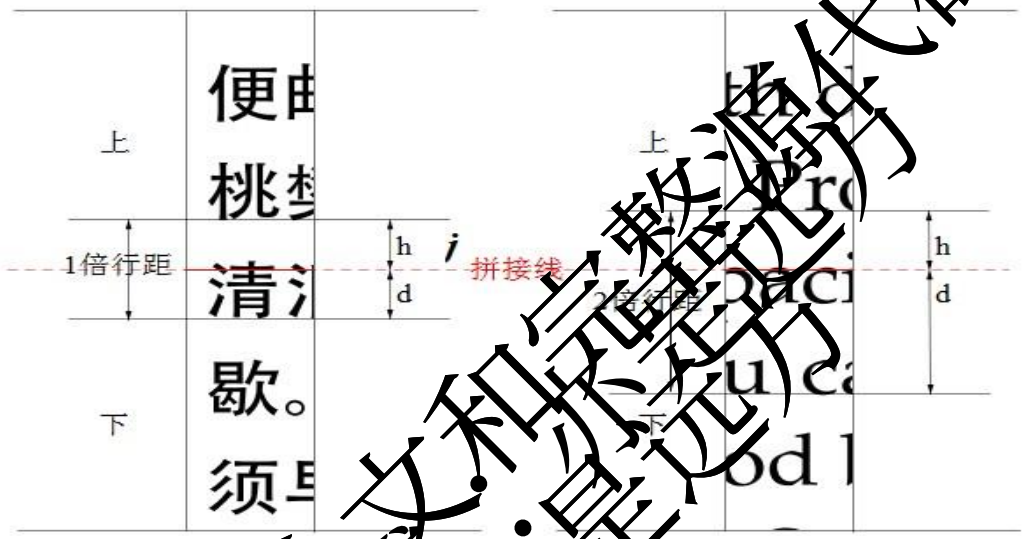
Step3，从语义上观察，是否具有明显的不通顺处，有则手工调整，将不通顺图片i和k处的$d(H_L(k),H_R(i))$改成无穷大，或者找到i实际的下一张图片j，将

$d(H_L(k), H_R(j))$改成 0，重复 **step1** 直到该行拼接匹配正确。

在使用本文提出的分类算法、复原拼接算法和修正算法后，我们就将每一类图片分别复原成了一张图片。根据 2.2 的分析，我们需要使用各行的行距来判断两行图片能否纵向拼接。

对于统一排版下的文字段落，其行距是固定的，所以将一段完整的段落横向切开时，与切开位置相邻的两行文字之间的行距关系是固定的，在总行数较少时，这样的关系甚至是唯一对应的。所以我们建立了以下基于行距关系判断的拼接复原模型。

我们首先做出横切位置的相邻两行文字行距关系示意图（图 5）：



(a)中文拼接行距关系示意图　　(b)英文拼接行距关系示意图

图 5 横切位置相邻文字行距关系示意图

其中上方图片的h是图片中最后一行完整文字（或字母）的下端距图片下边缘的像素点个数，下方图片的变量d就是本文定义的图片偏移量，通过图示我们看出，这两个变量与行距N满足

$$n \cdot N = h + d.(n = 1,2,\cdots) \tag{6}$$

我们在拼接各行的照片时，就是通过判断各行与其他行之间是否满足(6)式，如果满足，就说明这两张照片应该拼接，反之不应该拼接。这样我们就建立了基于行距判断的拼接复原模型。

5.2.4 模型的求解

我们首先计算了附件 3 中的中文文件碎片图片的偏移量，并按照 5.2.1 中的分类方式和(4)式完成了对图片的分类，我们将图片的初始分类结果列入附录 3 中。然后我们使用 5.1.1 中的拼接复原算法以及我们在 5.2.2 中提出的改正算法对每一类图片完成拼接。

在拼接中文图片的第一类的过程中，我们检验程序直接给出的拼接结果，发现文字拼接复原结果正确，没有拼接错误的发生，也没有人工干预。在拼接第二类的过程中，我们得到程序的初始输出结果见下图（图 6）：

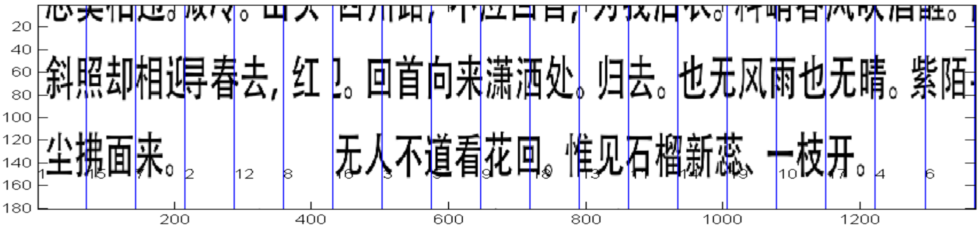


图 6　附件 3 第二类图片初始拼接结果示意图

图 6 中图片中的数字代表该图片在本类中的初始分类序号，观察图 6 我们发现，本类的第 7、2 号图片的拼接结果造成文字位置上的衔接性极差，并且内容上也没有连续性，除此之外还有几处拼接错误。根据本文 5.2.3 中的改正算法，我们需要在此时人为干预图片拼接，我们观察发现第 7 号图片右边应该与第 16 号图片拼接，在查询该类图片的原始标号后，我们令 $d(H_L(161),H_R(46))=0$，重新用软件进行拼接求解，得到一次人为干扰后的拼接结果示意图如下（图 7）：

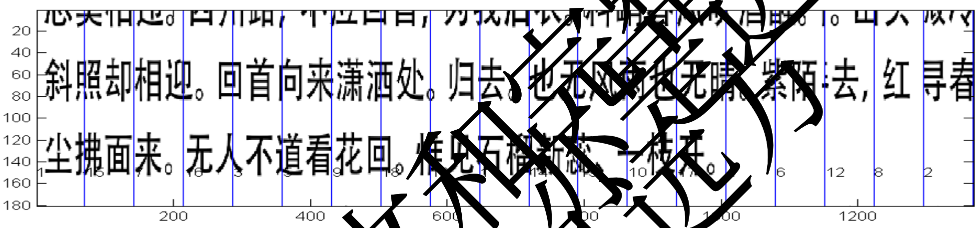


图 7　附件 3 第二类图片一次人为干预后的拼接结果示意图

观察图 7 我们发现，原来的错误已经改正，但是本类的第 6、12 号图片出现了拼接错误，我们进行第二次人为干预，令 $d(H_L(9),H_R(8))=0$。重新用软件进行拼接求解，得到两次人为干扰后的拼接结果示意图如下（图 8）：

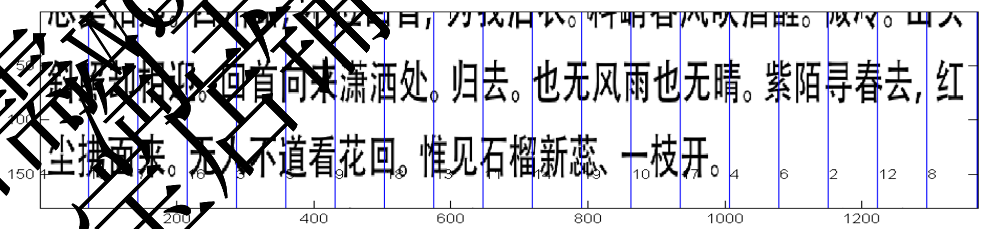


图 8　附件 3 第二类图片两次人为干预后的拼接结果示意图

观察图 8 我们发现，本类的拼接复原结果已经正确，一共有 2 次人为干预。对于其他几类，我们利用相同的算法进行拼接复原，我们将各类的拼接复原结果列入了表 3。其中我们根据 5.2.3 中的改进算法一共进行了 12 次的人工干预，我们将干预方式和时间列入了附录 4。

表 3　附件 3 各类中文碎片的拼接复原结果表

类	图片拼接序号																		
1																			

2																			
3																			
4																			
5																			
6																			
7																			
8																			
9																			
1																			
1																			

在完成各类图片的拼接后，相当于我们得到了原文件在仅横切方式下的 11 块碎片，这时我们根据 5.2.3 中提出的拼接复原算法，求解得到了原文件的拼接复原结果，在拼接过程中没有人为干预的步骤。我们将破片序号按复原后顺序填入下表（表 4），将拼接复原示意图列入附录 5.

表 4　附件 3 中文碎片拼接复原结果表

我们通过观察复原示意图（附录 5）可以发现，虽然问题二中破碎纸片的尺寸变化增加了难度，但是在我们设计的复原模型和算法的求解下，复原图中的文字不仅具有很高的衔接性，而且句子的含义也是连续的。

然后我们计算了附件 4 中的英文文件碎片图片的偏移量，并按照 5.2.2 中的分类方式和(4)式完成了对图片的分类，我们将图片的初始分类结果列入附录 6 中。

在拼接英文图片的第三类时，我们得到程序直接给出的拼接结果见下图（图 9）：

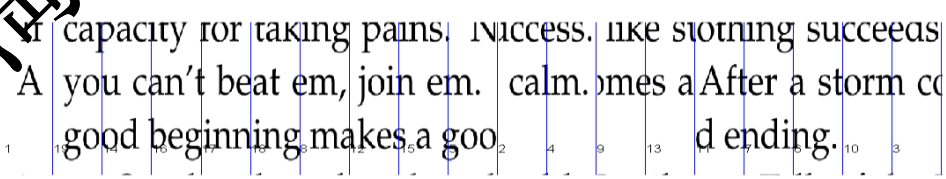


图 9　附件 4 第三类图片初始拼接结果示意图

对于图 9 中的拼接错误，我们进行第一次人为干预，令 $d(H_L(103), H_R(78)) = 0$，重新用软件进行拼接求解，得到一次人为干预后的拼接结果示意图如下（图 10）：

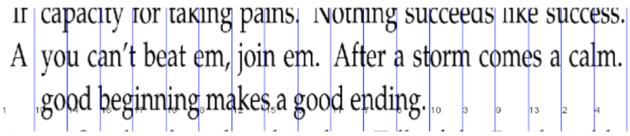


图 10　附件 4 第三类图片一次人为干预后的拼接结果示意图

观察图 10 我们发现，本类的拼接复原结果已经正确，有一次人为干预。对于附件四的其他几类，我们利用相同的算法进行拼接复原，我们将各类的拼接复原结果列入了附录 6。其中我们根据 5.2.3 中的改进算法一共进行了 15 次的人工干预，我们将干预方式和时间列入了附录 7。然后根据 5.2.3 中建立的拼接复原模型求解得到英文文件的拼接复原结果，我们将图片序号按复原后顺序填入下表（表 5），同时将拼接结果示意图列入附录 8。

表 5　附件 4 英文碎片拼接复原结果表

191	75	11	154	190	184	2	104	180	64	106	4	149	32	204	65	39	67	147
201	148	170	196	198	94	113	164	78	103	91	80	101	26	100	6	17	28	146
86	51	107	29	40	158	186	98	24	117	150	5	59	58	92	30	37	46	127
19	194	93	141	88	121	126	105	155	114	176	182	151	22	57	202	71	165	82
159	139	1	129	63	138	153	53	38	123	120	175	85	50	160	187	97	203	31
20	41	108	116	136	73	36	207	135	15	76	43	199	45	173	79	161	179	143
208	21	7	49	61	119	33	142	168	62	169	54	192	133	118	189	162	197	112
70	84	60	14	68	174	137	195	8	47	172	156	96	23	99	122	90	185	109
132	181	95	69	167	163	166	188	111	144	206	3	130	34	13	110	25	27	178
171	42	66	205	10	157	74	145	83	134	55	18	56	35	16	9	183	152	44
81	77	128	200	131	52	125	140	193	87	89	48	72	12	177	124	0	102	115

我们通过观察复原示意图（附录 8）可以发现，英文文件碎片复原图中的文字也具有很高的衔接性和句子的含义的连续行。

5.2.5 结果分析

通过 5.2.4 中的求解结果来看，我们的模型都求出了正确的解。在我们事先对照片进行分类处理后，在每一类的拼接过程中，人工干预的次数所占比例很小，在将各类拼接为一张图片时，没有人工干预。这一方面说明本文的基于像素点个数的分类方法是正确的，且极大地简化了后续的拼接难度，另一方面说明了本文的基于行距判断的拼接复原模型是正确的和高效的。

为了进一步说明我们的分类算法的有效性，我们总结了求解中、英文拼接时的偏移量矩阵 D'，得出分类时的偏移量标准，我们列出了偏移量标准表如下（表 6，表 7）：

表 6　附件 3 中文图片偏移量分类标准表

类	1	2	3	4	5	6	7	8	9	1	1

偏移量标准	3-4	16-17	22-23	27-29	34-35	40-42	46-48	53-54	59-60	65-66	77-79
个数	1	1	1	1	1	1	1	1	1	1	1

表 7　附件 4 英文图片偏移量分类标准表

类别	1	2	3	4	5	6	7
偏移量标准	12-	21-	33-	44-	55-	65-	74-
个数	19	38	38	38	38	19	19

我们结合(5)式，得出对于中文图片偏移量，$M_1 = 2$，$M_2 = 4$，满足 $M_1 < M_2$；对于英文图片偏移量，$M_1 = 4$，$M_2 = 8$，同样满足 $M_1 < M_2$。这说明我们在问题二中对中、英文分别提出的两种分类方法的类别区分度明显。正是由于我们考虑通过文字的偏移量来刻画每一张图片的行距信息，根据行距信息将图片按照行的所在位置分类，所以我们的分类结果能够精确地拼接复原原文件。

我们在使用问题一中的拼接复原算法对每一类的图片进行拼接时，出现了问题一中没有出现过的问题，这是因为附件 3 和附件 4 中的图片纵向像素点个数较少，所以在(2)式的求和运算后，并不能完全"吸收"那些相邻像素点的计算距离比实际值大的错误，使得我们的结果因为少数像素点的位置极端而出现错误，也就是说我们在 5.1.1 中给出的算法对问题二的数据类型容错率低，这也就是我们在问题二的求解中使用了人工干预的原因。

5.3 问题三：双面碎纸片拼接复原模型

5.3.1 模型的建立

我们对文字偏移量的定义与上文相同，我们对 5.1.2 中建立的拼接复原模型做出如下修正：

我们记某类图片的第 i 张照片某一面的左、右两侧灰度值矩阵为 $H_L(i), H_R(i)$，记该照片另一面的左、右两侧灰度值矩阵为 $H_L'(i), H_R'(i)$。我们使用第 i 张和第 j 张图片在拼接时正、反两面的距离之和作为这两面是否应该拼接的依据，我们定义第 i 张拼接在第 j 张碎片的右边时的距离为

$$d(i,j) = d(H_L(i), H_R(j)) + d(H_L'(j), H_R'(i)) \tag{7}$$

其中

$$\begin{cases} d(H_L(i),H_R(j))=\sum_{k=1}^{m}\left|h_{k1}(i)-h_{kn}(j)\right|, i,j=1,2,\cdots,38\text{且}i\neq j. \\ d(H_L'(j),H_R'(i))=\sum_{k=1}^{m}\left|h_{k1}'(j)-h_{kn}'(i)\right|, i,j=1,2,\cdots,38\text{且}i\neq j. \end{cases} \quad (8)$$

为了避免出现每一张图片的正面和反面相拼接的情况，我们规定

$$d(H_L'(i),H_R(i))=+\infty,\ (i=1,2,\cdots,209)$$

在以上公式的计算结果下，对于某张固定的图片i，我们分别对其左、右边界寻找与其距离最小的图片l^*和r^*，其中

$$\begin{cases} l^*=\arg\min\limits_{l\neq i} d(l,i), \\ r^*=\arg\min\limits_{r\neq i} d(i,r). \end{cases} \quad (9)$$

在使用(9)式对每一类的图片完成拼接后，我们需要将每一类图片拼接为一张图片。考虑到问题三的数据量较大，而且正反面具有相似行距关系，如果我们只采用问题二中的基于行距的拼接复原模型，在完成各行拼接的同时，很可能出现正面与反面拼接的情况，所以我们在采取基于行距的拼接复原模型的同时，还要使用问题一中的边缘相似性来进一步实现拼接。这样我们就对问题三建立了改进后的拼接复原模型。

5.3.2 模型的求解

我们首先计算了附件 5 中的所有英文图片的偏移量，并按照 5.2.2 中的分类方式完成了对图片的分类，我们将图片的初始分类结果列入附录 3 中。然后我们使用 5.3.1 中改进后的拼接复原算法以及我们在 5.2.2 中提出的改正算法对每一类图片完成拼接。我们将分类后的每一类图片拼接结果列入附录 9 中。

然后我们同时使用了基于行距和基于相似性的拼接复原模型，对已经完成各类拼接的图片实现了横向拼接，同时得到了正反两面的拼接结果。我们将图片序号按复原后顺序填入以下两表（表 8，表 9）：

表 8 附件 5 图片正面拼接结果表

078b	111b	125a	140a	155a	150a	183b	174b	110a	066a	108a	018b	029a	189b	081b	164b	020a	047a	136b
089a	010b	036a	076b	178a	044a	025b	192a	124b	022a	120b	144a	079a	014a	059a	060b	147a	152a	005a
186b	153a	084b	042b	030a	038a	121a	098a	094b	061b	137b	045a	138a	056b	131b	187b	086b	200b	143b
190b	011b	161a	169b	194b	173b	206b	156a	034a	181b	198b	087a	132b	093a	072b	175a	097a	039b	083a
088b	107a	149b	180a	037b	191a	065b	115b	166b	001b	151b	170b	041a	070b	139b	002a	162b	203b	090a
114a	184b	179b	116b	207a	058a	158a	197a	154b	028b	012a	017b	102b	064b	208a	142a	057a	024a	013a
146a	171b	031a	201a	050a	190b	092b	019b	016b	177b	053b	202a	021b	130a	163a	193b	073b	159a	035a
165b	195a	128a	157a	168a	046a	067a	063b	075b	167a	117b	008b	068b	188a	127a	040a	182b	122a	172a
003b	007b	085b	148b	077a	004a	069a	032a	074b	126b	176a	185a	000b	080b	027a	135b	141a	204b	105a
023b	133a	048a	051b	095a	160b	119a	033b	071b	052a	062a	129b	118b	101a	015b	205a	082b	145a	009b
099a	043a	096b	109a	123a	006a	104a	134a	113a	026b	049b	091a	106b	100b	055b	103a	112a	196b	054b

表 9　附件 5 图片反面拼接结果表

136a	047b	020b	164a	081a	189a	029b	018a	108b	066b	110b	174a	183a	150b	155b	140b	125b	111a	078a
005b	152b	147b	060a	059b	014b	079b	144b	120a	022b	124a	192b	025a	044b	178b	076a	036b	010a	089b
143a	200a	086a	187a	131a	056a	138b	045b	137a	061a	094a	098b	121b	038b	030b	042a	084a	153b	186a
083b	039a	097b	175b	072a	093b	132a	087b	198a	181a	034b	156b	206a	173a	194a	169a	161b	011a	199a
090b	203a	162a	002b	139a	070a	041b	170a	151a	001a	166a	115a	065a	191b	037a	180b	149a	107b	088a
013b	024b	057b	142b	208b	064a	102a	017a	012b	028a	154a	197b	158b	058b	207b	116a	179a	184a	114b
035b	159b	073a	193a	163b	130b	021a	202b	053a	177a	016a	019a	092a	190a	050b	201b	031b	171a	146b
172b	122b	182a	040b	127b	188b	068a	008a	117a	167b	075a	063a	067b	046b	168b	157b	128b	195b	165a
105b	204a	141b	135a	027b	080a	000a	185b	176b	126a	074a	032b	069b	004b	077b	148a	085a	007a	003a
009a	145b	082a	205b	015a	101b	118a	129a	062b	052b	071a	033a	119b	160a	095b	051a	048b	133b	023a
054a	196a	112b	103b	055a	100a	106a	091b	049a	026a	113b	134b	104b	006b	123b	109b	096a	043b	099b

我们通过观察复原示意图（附录 10）可以发现，虽然问题三中纸片是双面打印的，并且数量很多，但是在我们设计的多种复原模型和算法的求解下，复原图中的文字不仅具有很高的衔接性，而且句子的含义也是连续的。我们统计得到，在问题三中一共发生了 38 次人工干预，这个数量比问题二中有了较大的增加，说明我们的算法在解决数据量较大的复原问题时，有效性会有所降低。

5.3.3 结果分析

我们同样求出分类时的偏移量标准，我们列出了偏移量标准表如下（表 10）：

表 10　附件 5 英文图片偏移量分类标准表

类别	1	2	3	4	5	6
偏移量标	10-14	20-23	29-34	40-46	53-56	63-65
个数	76	75	77	76	76	38

通过观察表 10 我们看到，在一些类别中，图片个数已经不再是该类别应该含有个数的整数倍，说明在数据量增加的情况下，会增加出现特殊图片的可能，这时我们的分类标准就可能无法准确的识别这一些特殊图片。对于这种图片，我们是通过人工干预的方式将其调整至正确的分组内。虽然我们计算得到 $M_1 = 6$，$M_2 = 7$，满足 $M_1 < M_2$，但是我们只分出了 6 类，在增加分类数量、将各类进行细分之后，会发生 $M_1 > M_2$，这就说明了我们的分类方式在数据量增大的情况下有效性和准确性会降低。我们同时看到有很多类别的图片个数为 76，是 38 的两倍，这说明有属于两行的图片因为偏移量相近被分为一类，所以在对这一类进行拼接时，由于图片数量多，必然会造成人工干预次数的增加，这也是题目数据类型带来的结果，也是我们的算法有待改进的地方。

六、 模型的评价

优点：

（1）根据文字的结构形状特征建立的模型对于任何标准排版的文章的拼接复原问题都适用；

（2）模型中定义的边缘匹配度适用于任何形状的数据碎片的匹配；

（3）采用文字的投影将二维文字转化为一维向量，将纵向的差异累计，可以直观区分文字的类型；

（4）模型经过不断的改进，适用性得到很大的提升；

（5）模型结合聚类、匹配度等方法从两个维度进行碎片拼接，大大提高了精确度和效率。

缺点：

（1）对文字的分类方式依赖于字体结构等具体的内容，同一种分类方式不适用于其它字体；

（2）人为干预不可避免，匹配的方向只是文字结构的衔接上，无法上升到语意的分析；

（3）对于尺寸小和数量多的碎片匹配度较低。

七、 模型的改进与推广

本文提出的拼接复原算法可以继续在算法上进行优化，提升算法容错性，实现更多地由程序自动识别字体类型、自动规范文字结构以及自动完成图片的分类与拼接。也可以进一步优化分类算法，一方面细化分类准则，提高分类精度，另一方面挖掘更多同类图片间的共同信息，实现同类间的高效聚类。

我们还可以将本文的碎片拼接算法推广到拼接具有不同碎片形状、不同文字方向以及各种字体混杂出现的复杂碎片情况。本文在读取图片信息时是通过软件读取的灰度值信息，如果碎片图像是彩色图片，我们还可以使用 MATLAB 软件读取图片的色彩数据，剔除从色彩的连续性角度进行文字或图像拼接的模型及算法。

八、参考文献

[1]姜启源，谢金星.数学模型（第三版）[M].北京：高等教育出版社，2003：82-130.

[2]刘卫国.《MATLAB 程序设计与应用》,高等教育出版社.2012.

[3]贾海燕. 碎纸自动拼接关键技术研究[D]. 国防科学技术大学, 2005.

[4]Henri Maitre．现代数字图像处理[M]．孙洪译．北京：电子工业出版社，2006．

[5]马建华, 李本星, 董静, 等. 基于 Minkowski 距离最小化的多模态图像配准[N]. 电路与系统学报, 2008(13).