

## 2013 高教社杯全国大学生数学建模竞赛

### 承 诺 书

我们仔细阅读了《全国大学生数学建模竞赛章程》和《全国大学生数学建模竞赛参赛规则》(以下简称为“竞赛章程和参赛规则”，可从全国大学生数学建模竞赛网站下载)。

我们完全明白，在竞赛开始后参赛队员不能以任何方式(包括电话、电子邮件、网上咨询等)与队外的任何人(包括指导教师)研究、讨论与赛题有关的问题。

我们知道，抄袭别人的成果是违反竞赛章程和参赛规则的，如果引用别人的成果或其他公开的资料(包括网上查到的资料)，必须按照规定的参考文献的表述方式在正文引用处和参考文献中明确列出。

我们郑重承诺，严格遵守竞赛章程和参赛规则，以保证竞赛的公正、公平性。如有违反竞赛章程和参赛规则的行为，我们将受到严肃处理。

我们授权全国大学生数学建模竞赛组委会，可将我们的论文以任何形式进行公开展示(包括进行网上公示，在书籍、期刊和其他媒体进行正式或非正式发表等)。

我们参赛选择的题号是(从 A/B/C/D 中选择一项填写)：\_\_\_\_\_ B \_\_\_\_\_

我们的参赛报名号为(如果赛区设置报名号的话)：\_\_\_\_\_ 20001040 \_\_\_\_\_

所属学校(请填写完整的全名)：\_\_\_\_\_ 国防科学技术大学 \_\_\_\_\_

参赛队员(打印并签名)：1. \_\_\_\_\_ 向航 \_\_\_\_\_

2. \_\_\_\_\_ 王帆 \_\_\_\_\_

3. \_\_\_\_\_ 郭树璇 \_\_\_\_\_

指导教师或指导教师组负责人(打印并签名)：\_\_\_\_\_

(论文纸质版与电子版中的以上信息必须一致，只是电子版中无需签名。以上内容请仔细核对，提交后将不再允许做任何修改。如填写错误，论文可能被取消评奖资格。)

日期：\_\_\_\_\_ 2013 年 9 月 16 日 \_\_\_\_\_

---

赛区评阅编号(由赛区组委会评阅前进行编号)：

## 2013 高教社杯全国大学生数学建模竞赛

### 编 号 专 用 页

赛区评阅编号（由赛区组委会评阅前进行编号）：

赛区评阅记录（可供赛区评阅时使用）：

|     |  |  |  |  |  |  |  |  |  |  |
|-----|--|--|--|--|--|--|--|--|--|--|
| 评阅人 |  |  |  |  |  |  |  |  |  |  |
| 评分  |  |  |  |  |  |  |  |  |  |  |
| 备注  |  |  |  |  |  |  |  |  |  |  |

全国统一编号（由赛区组委会送交全国前编号）：

全国评阅编号（由全国组委会评阅前进行编号）：

# 基于旅行商规划模型的碎纸片拼接复原问题研究

## 摘要

本文分别针对 RSSTD(Reconstruction of Strip Shredded Text Document)、RCCSTD(Reconstruction of cross-cut Shredded Text Document)和 Two-Sides RCCSTD 三种类型的碎纸片拼接复原问题进行了建模与求解算法设计。首先我们对于 RSSTD 问题,建立了基于二值匹配度的 TSP 模型,并将其转化为线性规划模型,利用贪心策略复原了该问题的中文和英文碎片;然后对于 RCCSTD 问题,由于中英文文字的差别,我们分别建立了基于改进误差评估的汉字拼接模型和基于文字基线的误差评估的英文字拼接模型,并利用误差评估匹配算法,复原了该问题的中文和英文碎片;随后我们针对正反两面的 RCCSTD 问题,利用基线的概念将正反两面分行,转化为 RCCSTD 问题,并复原了该问题的英文碎片。最后,我们对模型的算法和结果进行了检验和分析。

◎问题一:我们针对仅纵切的情况,首先将图像进行数字化处理,转换为了二值图像,然后得到各图像的边缘,并计算所有碎片与其他碎片边缘的匹配程度。然后,根据两两碎片之间的匹配程度建立了 TSP 模型,并将其划归为线性规划模型。最终,我们根据左边距的信息确定了左边第一碎片,随后设计了基于匹配度的贪心算法从左向右得到了所有碎片的拼接复原结果。结果表明我们的方法对于中英文两种情况适用性均较好,且该过程不需要人工干预。

◎问题二:我们针对既纵切又横切的情况,由于中英文的差异性,我们在进行分行聚类时应采用不同的标准。首先根据左右边距的信息确定了左边和右边的碎片,随后分别利用基于改进误差评估的汉字拼接模型和基于文字基线的误差评估模型,将剩余的碎片进行分行聚类,然后再利用基于误差评估的行内匹配算法对行内进行了拼接,最终利用行间匹配算法对行间的碎片进行了再拼接,最终得到了拼接复原结果。对于拼接过程中可能出现误判的情况,我们利用 GUI 编写了人机交互的人工干预界面,用人的直觉判断提高匹配的成功率和完整性。

◎问题三:我们针对正反两面的情况,首先根据正反基线信息,分别确定了左右两边的碎片,然后利用基线差值将其两两聚类,聚类以后其正反方向也一并确定,随后我们将其与剩余碎片进行分行聚类,最终又利用行内匹配和行间匹配算法得到了最终拼接复原结果。其中,对于可能出现的误判情况,我们同样在匹配算法中使用了基于 GUI 的人机交互干预方式,利用人的直觉提高了结果的可靠性和完整性。

**关键字: 碎片复原、TSP、误差评估匹配、基线误差、人工干预**

## 一、问题重述

破碎文件的拼接复原工作传统上主要需由人工完成，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。复原结果以图片形式及表格形式表达。

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。

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。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。

## 二、问题分析

破碎文件的拼接复原工作，传统人工处理准确率高，但是效率低。随着计算机技术的发展，本问题试图寻找碎纸片的自动拼接方法。本文的碎片均为矩形，且大小一样，这就给碎片的拼接带来了困难，因为难以利用碎片的轮廓信息，从而只能利用碎片边缘的图像像素信息来进行拼接。

对于问题一，给出了纵切的 19 条碎纸片，碎片为  $1980 \times 72$ ，总切线的像素点较多，并且碎纸片的数量较少，从效率出发，可以直接考虑纵切条上的像素统计信息。为了衡量匹配程度，需要建立误差评估函数。本文考虑采用 TSP 模型，将碎纸片作为节点，将误差评估函数的值作为节点之间的权值，寻找最优解。在本问题的情况下，中英文的差别并无太大影响。

对于问题二，由于给出的数据是横纵切的中英文碎片，碎片大小为  $180 \times 72$ ，各 208 张。因为碎纸片的数量较多，直接匹配则为 NP 问题中的无法用多项式解决的问题，所以先将行分组匹配完成后再把各行拼接起来。本问题主要解决：1) 如何正确高效分组；2) 如何准确高效拼接。考虑到中文和英文在字形和印刷结构上的不同，需要定义不同的特征向量去描述中文和英文的碎纸片上的信息。文字各行的位置和间距将作为重要的分组依据。在问题一的基础上，由于每个碎纸片的边缘像素较少，为了充分利用包含信息，误差评估函数的定义将会比问题一中更加细化。行之间的拼接将会利用边缘像素和间距匹配。在本问题中，中英文

的基线选择方法将会有很大不同。

对于问题三，在问题二的基础上，加入了正反面信息，碎纸片的形状并没有改变。因此，需要在问题二的误差评估函数的基础上，综合考虑正反面的匹配误差。分组的方法与行间的拼接方式与问题二基本相同。

综上所述，本问题可以看做是 TSP 问题，碎纸片为节点，误差评估函数值为边的权值，寻求最优解的问题。

### 三、模型假设

1. 假设需要复原的碎片是来自同一张纸，且对于该张纸具有完备性。
2. 假设同一页中，文字的种类、行间距和段落分布情况是相同的。

### 四、符号说明

|              |                              |
|--------------|------------------------------|
| $m_{ijk}$    | 碎片 $i$ 和碎片 $j$ 第 $k$ 行之间的匹配度 |
| $M_{ij}$     | 碎片 $i$ 和碎片 $j$ 之间的匹配度        |
| $d_{tb}$     | 由上边缘向下像素点由黑过度到白的距离           |
| $d_{bb}$     | 由下边缘向上像素点由黑过度到白的距离           |
| $d_{tw}$     | 由上边缘向下像素点由白过度到黑的距离           |
| $d_{bw}$     | 由下边缘向上像素点由白过度到黑的距离           |
| $c_h(i, j)$  | 碎片 $i$ 和碎片 $j$ 之间左右边缘误差评估值   |
| $c'_h(i, j)$ | 碎片 $i$ 和碎片 $j$ 之间左右边缘相对误差评估值 |
| $c_w(p, q)$  | 行片段之间上下边缘的误差评估值              |

## 五、模型建立与求解

文件碎片拼接主要有两类不同的技术标准：

### (1) Reconstruction of Strip Shredded Text Documents(RSSTD)

碎片的长度和原始文件的长度相等，同时所有碎纸片都是等宽的长方形。

### (2) Reconstruction of Cross-cut Shredded Text Document(RCCSTD)

所有的文件碎片都是长方形且大小相等，但其长度小于原始文件的长度。

问题一为典型的 RSSTD 问题，我们通过分析碎纸片文字的行特征，建立了一种基于匹配度的求解模型，下面就对该模型的建立进行详细的讨论。

## 5.1 问题 1——基于匹配度的 RSSTD 问题研究

为了便于对图像数据进行分析，我们首先需要对其进行数字化处理以及文字行特征的提取，而后再建立基于匹配度的模型。

### 5.1.1 图像数字化处理

图像的数字化处理主要包括以下几个方面：

#### (1) 二值化处理

由于纵切碎纸片的长度特征（碎片长度与原始文件长度相等），其边缘像素点信息量丰富。因此我们采取二值化而非灰度对图像进行了处理，进一步简化了算法的复杂度，得到每条碎片像素大小为  $72 \times 1980$ （长 $\times$ 宽），像素点值取 $\{0,1\}$ ，其中 0 和 1 分别表示颜色的黑与白。

#### (2) edge 矩阵的建立

edge[i,j]为  $19 \times 2$  的矩阵，储存每一条碎片的边缘像素点信息。其中 i 表示碎片的编号，j=0 表示左侧边缘像素点信息，j=1 表示右侧边缘像素点信息。例如，edge[0,0]储存 001 号碎片左侧边缘像素点信息。

#### (3) count 矩阵的建立

根据 edge[i,j]矩阵，我们可以计算得到一个  $19 \times 2$  的 count[i,j]矩阵，该矩阵储存每一条碎片边缘取值为 0 的像素点的数量。例如，count[0,0]=350 表示 001 号碎片左侧边缘共有 350 个黑色像素点。

#### (4) shred with blank left 的选取

根据 count 矩阵，若 edge[i,0]=0，那么我们就认为该碎片在原始文件中处于

最左端，即为选取的 shred with blank left。

### (5) 碎片行特征的提取

由于原始文件文字行特征明显，因此我们需要提取碎片的行特征（其具体的原因见匹配度的定义）。首先确定碎片顶端取值为 0 的像素点的位置，以此作水平线为上边界，依次向下取  $w$  为行宽（这里取  $w=40$  pixels 以保证能容纳每一个文字）直至下边缘，得到每条碎片的行数为  $\{n_1, n_2, \dots, n_{19}\}$ ；然后取  $n=\max\{n_1, n_2, \dots, n_{19}\}$  作为最终确定的行数，并以该条碎片的行化方式为标准对每一条碎片进行行化；最终每一条碎片被划分为 28 行。

每条碎片维护着以下的特征数据表：

表 1 碎片  $i$  特征数据表

| $k$      | 1         | 2         | 3         | ... | 27         | 28         |
|----------|-----------|-----------|-----------|-----|------------|------------|
| $n_{kl}$ | $n_{i1l}$ | $n_{i2l}$ | $n_{i3l}$ | ... | $n_{i27l}$ | $n_{i28l}$ |
| $n_{kr}$ | $n_{i1r}$ | $n_{i2r}$ | $n_{i3r}$ | ... | $n_{i27r}$ | $n_{i28r}$ |

其中， $k$  为行号； $n_{kl}$  为第  $k$  行左侧边缘取值为 0 的像素点的数量， $n_{kr}$  为第  $k$  行右侧边缘取值为 0 的像素点的数量； $n_{ikl}$  为第  $i$  条碎片第  $k$  行左侧边缘取值为 0 的像素点的数量。

### 5.1.2 匹配度的定义

为了衡量两个碎片之间的匹配程度，我们定义匹配度  $M_{ij}$  作为评价的标准，其计算公式如下：

$$m_{ijk} = \frac{\min(n_{ikr}, n_{jkl})}{\max(n_{ikr}, n_{jkl})}$$

$$M_{ij} = \sum_{k=1}^n m_{ijk}$$

其中， $n$  为总的行数（如  $n=28$ ）； $m_{ijk}$  表示碎片  $i$  和碎片  $j$  第  $k$  行之间的匹配度； $M_{ij}$  表示碎片  $i$  和碎片  $j$  之间的匹配度。

这里我们不选择通过统计碎片边缘总黑色像素点的数量来计算两碎片之间的匹配度，而是通过计算两碎片行匹配度来间接计算其匹配度，原因在于：

- ① 若以总的边缘黑色像素点计算匹配度，行与行之间的差异性无法体现，匹配结果误差过大；
- ② 通过行匹配度计算碎片之间的匹配程度，更加充分的挖掘了碎片边缘的信息，更为精确。

## 5.1.3 基于 TSP 问题的模型建立与求解

### 5.1.3.1 模型的建立

根据匹配度的定义，我们可以计算得到 19 个碎片两两之间边缘的匹配度，现可将问题化简为：已知起始碎片和其他碎片之间的匹配度，寻找一序列使得碎片之间总的匹配度达到最大。如右图所示：

(1) 为简化问题，我们引入空白碎片 0，同样可计算得到空白碎片与其他碎片的匹配度；

(2) 其中节点表示碎片，有向线段长度表示费用  $w_{ij}$ ，且  $w_{ij}=1-M_{ij}$ ，箭头所指方向表示前一条碎片右侧边缘到后一条碎片左侧边缘。如①→②表示 1 号的碎片右侧边缘与 2 号碎片左侧边缘的费用；

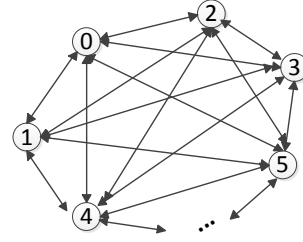


图 1 问题示意图

(3) 现要求寻找一条回路遍历所有的节点使得费用达到最小。

通过分析，可以发现这是一典型的 TSP 问题。为解决上述问题，我们建立模型如下：

假设图中存在 Hamilton 回路，有  $n$  个节点，图中  $(i,j)$  边的权重为  $w_{ij}$ ，设决策变量为  $x_{ij}=1$  说明弧  $(i,j)$  进入到 Hamilton 回路中，则线性规划模型可表达<sup>[6]</sup>为：

$$\begin{aligned} \min \quad & \sum_{(i,j) \in E} w_{ij} x_{ij} \\ \text{s.t.} \quad & \begin{cases} \sum_{(i,j) \in E} x_{ij} = 1 \\ \sum_{(j,i) \in E} x_{ji} = 1 \\ x_{ij} \in \{0,1\}, (i,j) \in E \\ \text{Only One Circle} \end{cases} \end{aligned}$$

求解该问题就是求解该线性规划问题的最优解。

### 5.1.3.2 模型的求解

为了进一步求解上述模型，我们设计的算法如右图所示：

- (1) 将所有碎片放入地址池中；
- (2) 选出左侧全为白色像素点的碎片作为基准碎片，记为  $s_i$ ，并将其从碎片池中剔除；

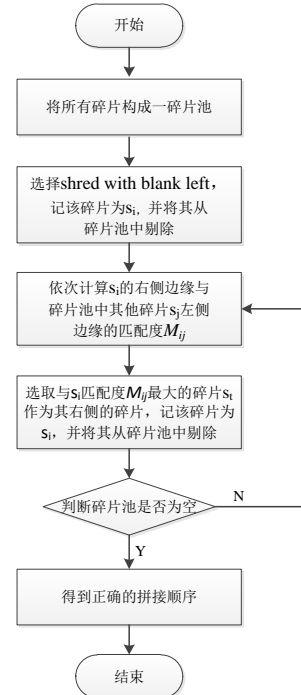


图 2 算法流程图



(3) 依次计算  $s_i$  右侧边缘与地址池中其他碎片  $s_j$  左侧边缘的匹配度  $M_{ij}$ ;

(4) 选择匹配度最大的碎片  $s_i$  作为碎片  $s_j$  的右侧碎片，并将其记为  $s_i$ ，从碎片池中剔除该碎片；

(5) 重复 (3) ~ (4) 步直至碎片池为空。

由于该算法使用统计量构建匹配度，很好的避免了中英文之间的差异性，适用性较好，在实际的应用中都收到了很好的效果。最终可以得到附件 1 与附件 2 中碎片的正确序列如下所示：

表 2 附件 1 排序后碎片序列表

| 序号   | 1 | 2  | 3  | 4  | 5 | 6  | 7 | 8  | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 |
|------|---|----|----|----|---|----|---|----|---|----|----|----|----|----|----|----|----|----|----|
| 碎片编号 | 8 | 14 | 12 | 15 | 3 | 10 | 2 | 16 | 1 | 4  | 5  | 9  | 13 | 18 | 11 | 7  | 17 | 0  | 6  |

表 3 附件 2 排序后碎片序列表

| 序号   | 1 | 2 | 3 | 4 | 5  | 6  | 7  | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 |
|------|---|---|---|---|----|----|----|---|---|----|----|----|----|----|----|----|----|----|----|
| 碎片编号 | 3 | 6 | 2 | 7 | 15 | 18 | 11 | 0 | 5 | 1  | 9  | 13 | 10 | 8  | 12 | 14 | 17 | 16 | 4  |

通过求解结果发现，我们利用贪心策略求解得到了全局最优解，原因在于匹配度的定义较好。同时由于碎片两侧提供的信息量大，很好的避免了中英文之间的差异性。若碎片规格变小，信息量减少，中英文之间差异性的讨论显得十分有必要。

## 5.2 问题 2——基于误差评估的 RCCSTD 问题研究

由于中英文的字存在显著的差异性，对算法的设计有很大影响，因此我们需要对它们进行具体的讨论。

如右图所示，我们可以分析得到如下的差异性特征：

(1) 中英文的字行存在显著的差异性，中文字结构复杂，笔画繁多，无法确定固定的基准线；

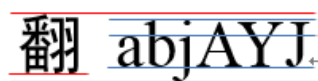


图 3 中英文字对比图

(2) 英文字印刷格式固定，一般为“四线三格”的形式，容易通过基准线来确定具体字母的位置信息。

基于以上分析，我们需要针对中英文设计不同的模型进行碎纸片的拼接复原。下面首先建立文字的拼接复原模型。

### 5.2.1 基于误差评估的汉字拼接模型

#### 5.2.1.1 碎片的逐行分组

### (1) 逐行分组原因分析

首先参考问题一中的算法提取位于原始文件左右边界位置的 22 块碎片，以此为基础将所有碎片分为 22 组，而非按列分组或分为 11 组，主要原因如下：

①若按列分组，由于碎片上下边缘信息量少，分组误差较大，不可取；

②若以原始文件左侧边缘的 11 块碎片为基础将所有碎片分为 11 组，可能存在以下的问题：如右图所示，014 块碎片为原始文件的左侧边缘碎片，128 块碎片为其右侧碎片。但由于 014 块碎片处于一自然段的开始位置，上半部分基本为空白，而 128 块碎片上半部分信息丰富，两者匹配失败率高，很可能导致该组元素个数为 0。因此我们将碎片分为 22 组，同时考虑左侧边缘与右侧边缘的信息，分组效率高且准确率高。

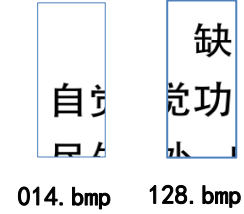


图 4 匹配示意图

### (2) 分组标准的确定

我们确立四个参数  $d_{tb}$ ,  $d_{bb}$ ,  $d_{tw}$ ,  $d_{bw}$  其意义如右图所示。其中  $d_{tb}$ ,  $d_{bb}$  代表由上下边缘至中间像素点由黑过度到白的距离； $d_{tw}$ ,  $d_{bw}$  代表上下边缘至中间像素点由白过度到黑的距离。

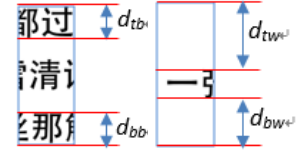


图 5 参数示意图

根据上述定义，对于每一幅图像，首先判断其上下边缘的过度情况，由此可以确定上下边缘的两个参数。那么我们认为若两个碎片具有相同的参数，如同样为  $d_{tb}$  和  $d_{bw}$ ，同时相同参数之间的差值不超过  $4pixels$ ，即：

$$|d_{tb} - d'_{tb}| \leq 4pixels \quad \text{且} \quad |d_{bw} - d'_{bw}| \leq 4pixels$$

那么，我们认为两碎片在同一组内。最终可以得到分组情况如下表所示：

表 4 分组情况表

| 组号 | 组内成员编号   | 组号 | 组内成员编号   |
|----|--|----|--|
| 1  | {7, 68, 175, 0, 126, 174, 56, 53, 208, 158, 138, 137, 45}  | 12 | {18, 195}  |
| 2  | {14}   | 13 | {36}   |
| 3  | {29}   | 14 | {43}   |
| 4  | {38, 189, 88, 103, 35, 24, 148, 167, 161, 46, 122, 130, 193}   | 15 | {59, 37, 98, 104, 10, 75, 48, 171, 207, 92, 111, 55, 180, 5, 172, 64, 44, 201} |
| 5  | {49, 54, 28, 192, 22, 129, 190, 186, 118, 143, 65, 57, 2, 91}  | 16 | {60, 205, 85, 152}   |
| 6  | {61, 116, 67, 162, 79, 99, 63, 20, 72, 6, 139, 136, 135, 134, 133, 132, 131, 130, 129, 128, 127, 126, 125, 124, 123, 122, 121, 120, 119, 118, 117, 116, 115, 114, 113, 112, 111, 110, 109, 108, 107, 106, 105, 104, 103, 102, 101, 100, 99, 98, 97, 96, 95, 94, 93, 92, 91, 90, 89, 88, 87, 86, 85, 84, 83, 82, 81, 80, 79, 78, 77, 76, 75, 74, 73, 72, 71, 70, 69, 68, 67, 66, 65, 64, 63, 62, 61, 60, 59, 58, 57, 56, 55, 54, 53, 52, 51, 50, 49, 48, 47, 46, 45, 44, 43, 42, 41, 40, 39, 38, 37, 36, 35, 34, 33, 32, 31, 30, 29, 28, 27, 26, 25, 24, 23, 22, 21, 20, 19, 18, 17, 16, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1} | 17 | {74, 8, 25, 9}   |

|    |   |    |  |
|----|---|----|--|
|    | 69, 96, 131, 78, 163, 52, 19, 177}  |    |  |
| 7  | {71}  | 18 | {123, 152, 194, 207, 154, 119, 146, 108, 185, 4, 155, 40, 102, 101, 140, 113}          |
| 8  | {89}  | 19 | {141, 188, 12}   |
| 9  | {94, 42, 144, 97, 34, 77, 127, 90, 84, 121, 136, 47, 149, 164, 183, 112, 124} | 20 | {145, 181, 204, 182, 16, 110, 66, 184, 13, 106, 109, 150, 139, 187, 197, 157, 21, 173} |
| 10 | {125}   | 21 | {176, 115, 12, 3, 134, 135, 82, 39, 169, 31, 203, 199, 51, 159, 128, 107, 73, 160}     |
| 11 | {168, 142, 147, 76, 50, 23, 120, 87, 179, 62, 100, 191, 41, 26}               | 22 | {196, 32, 93, 153, 70, 166}  |

由上表可知, 由于某些碎片的特殊性, 仍不能归入任何一组。这些碎片有 1, 15, 17, 27, 30, 33, 58, 80, 81, 83, 86, 95, 105, 114, 117, 132, 133, 156, 165, 170, 178, 198, 200, 202。我们可以将他们放入碎片池中, 这一部分碎片后续将进行单独的处理。

### 5.2.1.2 组内碎片匹配算法的设计

评价两块碎片拼接的好坏程度, 我们用误差评估函数来描述。

由题可知, 每页被分为了  $11 \times 19$  个碎片, 每个碎片具有同样的规格。那么, 就有  $h_i = h_j$ , 表示两碎片的高度相等。我们定义误差评估函数  $c_h(i, j)$  为:

$$c_h(i, j) = \sum_{y=3}^{h-2} e_h(i, j, y)$$

$$e_h(i, j, y) = \begin{cases} 1 & \text{if } e'_h(i, j, y) \geq \tau \\ 0 & \text{otherwise} \end{cases}$$

$$e'_h(i, j, y) = |0.7 \cdot (v_r(i, y) - v_l(j, y)) + 0.1 \cdot (v_r(i, y+1) - v_l(j, y+1)) + 0.1 \cdot (v_r(i, y-1) - v_l(j, y-1)) + 0.05 \cdot (v_r(i, y+2) - v_l(j, y+2)) + 0.05 \cdot (v_r(i, y-2) - v_l(j, y-2))|$$

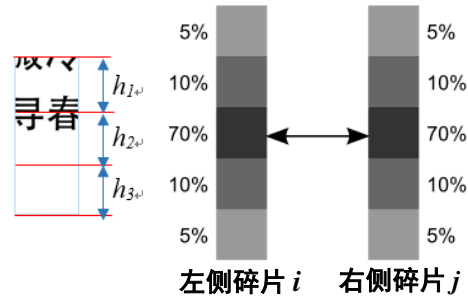


图 6 评估函数参数示意图

其中, 每个像素点以 8 位数值表示, 有:

$$v_l(j, y), v_r(i, y) \in \{0, \dots, 255\} \quad \text{with } 1 \leq y \leq h \text{ for the left and right edge}$$

如上图所示, 我们将一碎片边缘等分为三分。在考虑像素点  $y$  时, 对其邻域  $y+1, y+2, y-1, y-2$  共 5 个像素点进行分析, 并赋予不同的权重, 那么可以分别计算得到三段的误差评估函数为  $c_{h_1}(i, j), c_{h_2}(i, j), c_{h_3}(i, j)$ 。

参考问题一，设计组内匹配算法如下：

- (1) 将组内碎片以及剩余未分组的碎片放入碎片池中；
- (2) 以左侧全为白色像素点的碎片为基准碎片，记为  $s_i$ ，并将其从碎片池中剔除；
- (3) 依次计算  $s_i$  右侧边缘与地址池中其他碎片  $s_j$  左侧边缘的误差评估值  $c_{h_1}(i, j), c_{h_2}(i, j), c_{h_3}(i, j)$ ；
- (4) 依次计算碎片  $s_i$  与池中其他碎片  $s_j$  的相对误差：<sup>[1]</sup>

$$c'_h(i, j) = \sum_{k=1}^3 \frac{|c_{h_k}(i, j) - \min c_{h_k}(i, j')|}{c_{h_k}(i, j)}$$

其中， $j$  表示当前比较的碎片， $j'$  为碎片池中其他的碎片， $k$  表示上下三段，如  $h_1, h_2, h_3$ ；

(5) 选取相对偏差最小的碎片，且  $c'_h(i, j) \leq \xi$  时，将该碎片作为碎片  $s_i$  的右侧碎片，并记该碎片为  $s_i$ ，将其从碎片池中删除；

(6) 重复 (3) ~ (5) 步 18 次直至结束。

12~22 组可以根据同样的算法进行碎片的拼接复原。在此过程中由于最小相对误差的碎片存在多个，计算机很难进行取舍。因此，在程序中，我们设计了人机交互的界面，当遇到此类情况时，通过人工干预的方式来选择最佳的碎片。最后，可以得到 22 个拼接好的片段。

### 5.2.1.3 组间片段的匹配

组间片段的匹配，我们主要依据以下两点原则：

- (1) 若两个片段在同一行，那么两个片段内碎片的数量应当互补(和为 19)；
- (2) 以一个片段为基础，计算其与右侧片段的相对偏差，相对偏差越小，匹配程度越高。

根据以上原则，最终可以得到左右两侧片段按行匹配的结果如下图所示：

表 5 22 组片段按行匹配情况表

| 行号     | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  | 10 | 11 |
|--------|----|----|----|----|----|----|----|----|----|----|----|
| 左侧片段组号 | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  | 10 | 11 |
| 右侧片段组号 | 22 | 21 | 15 | 74 | 19 | 13 | 16 | 18 | 14 | 20 | 12 |

注：这里行号不表示该行在原始文件中的顺序。

#### 5.2.1.4 行间匹配

类似于组内碎片匹配的算法，通过计算行片段之间上下边缘的误差评估值  $c_w(p, q)$ ，最终可以得到行片段的正确序列。行误差评估值<sup>[2]</sup>的计算公式如下：

$$c_w(p, q) = \sum_{x=3}^{w-2} e_w(p, q, x)$$

其中， $p, q$  表示行片段号， $w$  表示行像素点的数量。

根据上述模型及算法，最终可得到  $11 \times 19$  个碎片在原始文件中的序列如下表所示：

表 6 附件 3 排序后碎片序列表

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

#### 5.2.2 基于基线的英文字拼接模型

由上文的分析可以知道，英文与中文在字形、行分布等方面存在着很大的不同。在英文的印刷排版中对应字体均有比较严格的基准线，基线是排成一行的西文字符主体上的一条不可见的假想线。对于英文段落上完整的一行，一定存在至少四条基准线，将整行英文分布在三个格子中，即英文的“四线三格”。本问题中给出的是衬线体（serif 字体）示意图如下：

Tayhblgp

图 7 “三线四格”示意图

根据上述特征，我们建立了基于基线的碎片拼接模型。同中文字体，我们首先需要对  $11 \times 19$  个碎片进行。

##### 5.2.2.1 按行分组

按行分组的重要依据是每一块碎片的基准线，因此首先需要确定每一个碎片基准线的具体位置。通过观察发现，基准线是黑白像素点的分水岭，因此通过统计黑白渐变处黑色像素点的数量，可以确定基准线的位置。具体方法如下所示：

如右图所示，由上到下，由左到右按列统计从白色像素点到第一个黑色像素点的距离  $l_x$ ，那么可以得到 72 个  $l_x$  的数据，取众数即为第一条基准线的位置  $line_1$ ，同样，从下到上统计 2 得到第二条基准线  $line_2$ 。这样，每个碎片可得到至少 2 条基准线以及他们的位置信息。

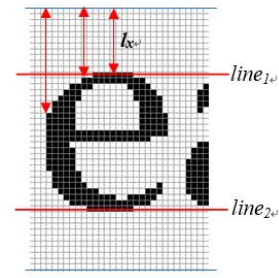


图 8 基准线示意图

同样，按行分组时：

(1) 选取左侧留白的碎片作为标准碎片，以基准线为依据，若碎片  $i$  和碎片  $j$  满足： $|line_{i1} - line_{j1}| < 2pixels$ 。那么，我们认为这两个碎片处于同一行。

(2) 若该碎片的基准线不和任何一行的标准碎片相吻合，那么将该碎片放入碎片池中。

最终可以得到 11 个组以及碎片池中少量的碎片。

#### 5.2.2.2 组内匹配算法设计

由于在组内，英文碎片的边缘信息与汉字类似，因此我们同样采用误差评估函数  $c_h(i, j)$  来衡量两个碎片之间匹配程度。不同的是由于相对误差最小的碎片不一定是该碎片右侧的碎片。因此，我们做了如下的处理：

当两碎片的相对误差  $c'_h(i, j) < 0.5$  时，接受碎片  $j$  作为碎片  $i$  的右侧碎片，相反，则加入人工干预的方式，帮助判断是否为右侧碎片。

#### 5.2.2.3 行间匹配

增加人工干预的方式，最终可以得到 11 组完整的碎片片段。同样利用行碎片上下边缘的信息，计算行片段之间上下边缘的误差评估值  $c_w(p, q)$ ，最终可以得到行片段的正确序列。

根据上述模型及算法，最终可得到  $11 \times 19$  个碎片在原始文件中的序列如下表所示：

表 7 附件 4 排序后碎片序列表

|   | 1   | 2   | 3   | 4   | 5   | 6   | 7   | 8   | 9   | 10  | 11  | 12  | 13  | 14 | 15  | 16  | 17 | 18  | 19  |
|---|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|----|-----|-----|----|-----|-----|
| 1 | 191 | 75  | 11  | 154 | 190 | 184 | 2   | 104 | 180 | 64  | 106 | 4   | 149 | 32 | 204 | 65  | 39 | 67  | 147 |
| 2 | 201 | 148 | 170 | 196 | 198 | 94  | 113 | 164 | 78  | 103 | 91  | 80  | 101 | 26 | 100 | 6   | 17 | 28  | 146 |
| 3 | 86  | 51  | 107 | 29  | 40  | 158 | 186 | 98  | 24  | 117 | 150 | 5   | 59  | 58 | 92  | 30  | 37 | 46  | 127 |
| 4 | 19  | 194 | 93  | 141 | 88  | 121 | 126 | 105 | 155 | 114 | 176 | 182 | 151 | 22 | 57  | 202 | 71 | 165 | 82  |
| 5 | 159 | 139 | 1   | 129 | 63  | 138 | 153 | 53  | 38  | 123 | 120 | 175 | 85  | 50 | 160 | 187 | 97 | 203 | 31  |

|    |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |
|----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 6  | 20  | 41  | 108 | 116 | 136 | 73  | 36  | 207 | 135 | 15  | 76  | 43  | 199 | 45  | 173 | 79  | 161 | 179 | 143 |
| 7  | 208 | 21  | 7   | 49  | 61  | 119 | 33  | 142 | 168 | 62  | 169 | 54  | 192 | 133 | 118 | 189 | 162 | 197 | 112 |
| 8  | 70  | 84  | 60  | 14  | 68  | 174 | 137 | 195 | 8   | 47  | 172 | 156 | 96  | 23  | 99  | 122 | 90  | 185 | 109 |
| 9  | 132 | 181 | 95  | 69  | 167 | 163 | 166 | 188 | 111 | 144 | 206 | 3   | 130 | 34  | 13  | 110 | 25  | 27  | 178 |
| 10 | 171 | 42  | 66  | 205 | 10  | 157 | 74  | 145 | 83  | 134 | 55  | 18  | 56  | 35  | 16  | 9   | 183 | 152 | 44  |
| 11 | 81  | 77  | 128 | 200 | 131 | 52  | 125 | 140 | 193 | 87  | 89  | 48  | 72  | 12  | 177 | 124 | 0   | 102 | 115 |

### 5.3 问题三——Two-Sides RCCSTD 问题研究

通过对问题三的分析，发现问题的解决方案基本与问题二类似，但针对正反碎片的特点，对问题二中的算法做了进一步的改进。

#### 5.3.1 基于问题二算法的改进

(1) **边缘基准碎片的选择。**由于一个碎片存在正反面的信息，因此在选择边缘基准碎片时，需要考虑正反面两侧边缘留白的情况。若某一碎片正反面两侧均有留白，说明该碎片为边缘基准碎片；相反则不是。这样可以选取到 22 组边缘基准碎片对（即一碎片的正反面）。

(2) **分组标准的确定。**在问题二中，我们通过分析不同碎片基准线之间的差异性来判断两块碎片是否在同一行。而对于正反的碎片，可获取的信息更多。因此采取下述标准进行分组：

若  $(a_1, b_1)$  为左侧基准碎片， $(a_2, b_2)$  为右侧比较碎片，那么有右图所示的两种比较方式。同问题二的基准线误差计算方式，可以得到两组数据：

$$((l_{t1}, l_{b1}), (l_{t2}, l_{b2}))$$

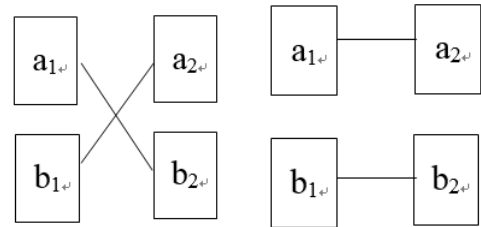


图 9 碎片的两种匹配方式示意图

其中， $(l_{t1}, l_{b1})$  表示在第一种情况下匹配，上下面基准线的差值； $(l_{t2}, l_{b2})$  表示在第二种情况下匹配，上下面基准线的差值。为了便于比较，我们作如下的简化：对于每一次匹配可以得到一组数据  $(l_1, l_2)$ ，其中， $l_1 = l_{t1} + l_{b1}, l_2 = l_{t2} + l_{b2}$ 。

那么，在分组过程中，依次取碎片分别与 22 组边缘基准碎片做匹配，得到 22 组共 44 个数据： $(l_{1,1}, l_{2,1}; l_{1,2}, l_{2,2}; \dots; l_{1,22}, l_{2,22})$ ，取其中的最小值对应的碎片即可将其归入相应的组。同时可判断匹配的方式，即：选取  $l_{1i}$  表示为第一种匹配方式，选取  $l_{2i}$  表示为第二种匹配方式。这样可以将所有碎片分为 22 组，一些未归入任何组的碎片可放入地址池中。

(3) **组内匹配算法的改进。**对于同一组内的碎片，我们同样可以以误差评

估函数  $c_h(i, j)$  来衡量两个碎片之间匹配程度。不同的是在匹配的过程中，我们可以计算得到上下两面的误差值分别为  $c_{h1}(i, j)$  和  $c_{h2}(i, j)$ ，为了便于比较，同样以  $c_h(i, j) = c_{h1}(i, j) + c_{h2}(i, j)$  来作为衡量匹配程度的标准。 $c_h(i, j)$  越小，匹配度越好，即可作为被匹配碎片的右侧碎片。这样，可以得到 22 组碎片片段。

(4) **行间匹配的改进。**利用问题二中行间匹配的算法，计算上下面的误差评估值的综合指标，即  $c_w(i, j) = c_{w1}(i, j) + c_{w2}(i, j)$ 。最终可以拼接复原得到原始文件。

### 5.3.2 模型计算结果

由上述改进后的算法，最终可以计算得到附件 5 排序后碎片的序列如下两表所示：

表 8 附件 5 排序后一面碎片序列表

|    | 1    | 2    | 3    | 4    | 5    | 6    | 7    | 8    | 9    | 10   | 11   | 12   | 13   | 14   | 15   | 16   | 17   | 18   | 19   |
|----|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 1  | 136a | 047b | 020b | 164a | 081a | 189a | 029b | 018a | 108b | 066b | 110b | 174a | 183a | 150b | 155b | 140b | 125b | 111a | 078a |
| 2  | 005b | 152b | 147b | 060a | 059b | 014b | 079b | 144b | 120a | 022b | 124a | 192b | 025a | 044b | 178b | 076a | 036b | 010a | 089b |
| 3  | 143a | 200a | 086a | 187a | 131a | 056a | 138b | 045b | 137a | 061a | 094a | 098b | 121b | 038b | 030b | 042a | 084a | 153b | 186a |
| 4  | 083b | 039a | 097b | 175b | 072a | 093b | 132a | 087b | 198a | 181a | 034b | 156b | 206a | 173a | 194a | 169a | 161b | 011a | 199a |
| 5  | 090b | 203a | 162a | 002b | 139a | 070a | 041b | 170a | 151a | 001a | 166a | 115a | 065a | 191b | 037a | 180b | 149a | 107b | 088a |
| 6  | 013b | 024b | 057b | 142b | 208b | 064a | 102a | 017a | 012b | 028a | 154a | 197b | 158b | 058b | 207b | 116a | 179a | 184a | 114b |
| 7  | 035b | 159b | 073a | 193a | 163b | 130b | 021a | 202b | 053a | 177a | 016a | 019a | 092a | 190a | 050b | 201b | 031b | 171a | 146b |
| 8  | 172b | 122b | 182a | 040b | 127b | 188b | 068a | 008a | 117a | 167b | 075a | 063a | 067b | 046b | 168b | 157b | 128b | 195b | 165a |
| 9  | 105b | 204a | 141b | 135a | 027b | 080a | 000a | 185b | 176b | 126a | 074a | 032b | 069b | 004b | 077b | 148a | 085a | 007a | 003a |
| 10 | 009a | 145b | 082a | 205b | 015a | 101b | 118a | 129a | 062b | 052b | 071a | 033a | 119b | 160a | 095b | 051a | 048b | 133b | 023a |
| 11 | 054a | 196a | 112b | 103b | 055a | 100a | 106a | 091b | 049a | 026a | 113b | 134b | 104b | 006b | 123b | 109b | 096a | 043b | 099b |

表 9 附件 5 排序后另一面碎片序列表

|    | 1    | 2    | 3    | 4    | 5    | 6    | 7    | 8    | 9    | 10   | 11   | 12   | 13   | 14   | 15   | 16   | 17   | 18   | 19   |
|----|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 1  | 078b | 111b | 125a | 140a | 155a | 150a | 183b | 174b | 110a | 066a | 108a | 018b | 029a | 189b | 081b | 164b | 020a | 047a | 136b |
| 2  | 089a | 010b | 036a | 076b | 178a | 044a | 025b | 192a | 124b | 022a | 120b | 144a | 079a | 014a | 059a | 060b | 147a | 152a | 005a |
| 3  | 186b | 153a | 084b | 042b | 030a | 038a | 121a | 098a | 094b | 061b | 137b | 045a | 138a | 056b | 131b | 187b | 086b | 200b | 143b |
| 4  | 199b | 011b | 161a | 169b | 194b | 173b | 206b | 156a | 034a | 181b | 198b | 087a | 132b | 093a | 072b | 175a | 097a | 039b | 083a |
| 5  | 088b | 107a | 149b | 180a | 037b | 191a | 065b | 115b | 166b | 001b | 151b | 170b | 041a | 070b | 139b | 002a | 162b | 203b | 090a |
| 6  | 114a | 184b | 179b | 116b | 207a | 058a | 158a | 197a | 154b | 028b | 012a | 017b | 102b | 064b | 208a | 142a | 057a | 024a | 013a |
| 7  | 146a | 171b | 031a | 201a | 050a | 190b | 092b | 019b | 016b | 177b | 053b | 202a | 021b | 130a | 163a | 193b | 073b | 159a | 035a |
| 8  | 165b | 195a | 128a | 157a | 168a | 046a | 067a | 063b | 075b | 167a | 117b | 008b | 068b | 188a | 127a | 040a | 182b | 122a | 172a |
| 9  | 003b | 007b | 085b | 148b | 077a | 004a | 069a | 032a | 074b | 126b | 176a | 185a | 000b | 080b | 027a | 135b | 141a | 204b | 105a |
| 10 | 023b | 133a | 048a | 051b | 095a | 160b | 119a | 033b | 071b | 052a | 062a | 129b | 118b | 101a | 015b | 205a | 082b | 145a | 009b |
| 11 | 099a | 043a | 096b | 109a | 123a | 006a | 104a | 134a | 113a | 026b | 049b | 091a | 106b | 100b | 055b | 103a | 112a | 196b | 054b |

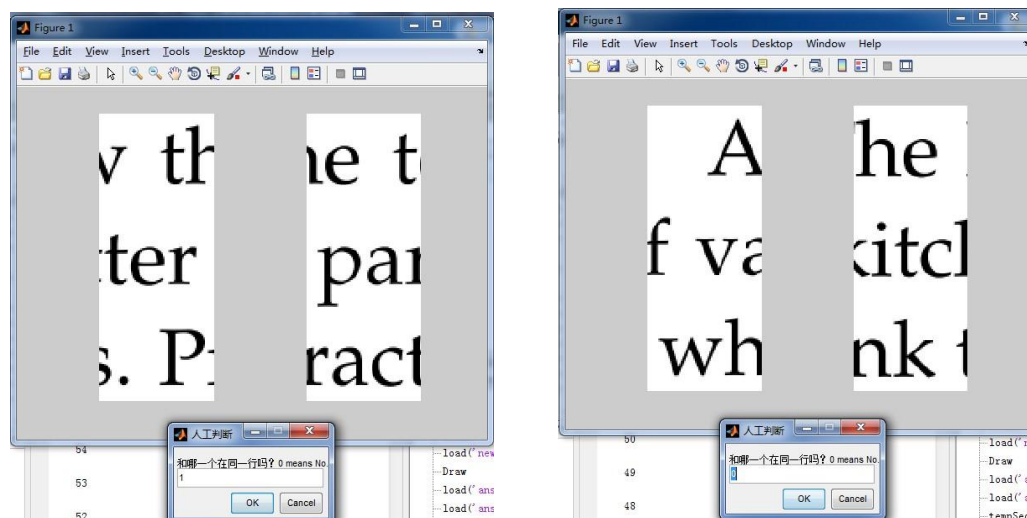
### 5.4 关于人工干预的说明



由于算法的局限性，碎纸片在 *cross-cut* 的情况下需要人工参与决策。在我们的算法中，有两种情况需要进行人工干预。

**情况一：**在组内进行匹配时，我们选取误差值最小的碎片作为匹配的碎片，但在实际中误差值最小的碎片不一定为正确的匹配对象。因此，在问题二中，我们设计为：当最小的误差值小于 0.5 时，该碎片被完全接受；相反，当其大于 0.5 时，该碎片需要人工干预判断其是否被接受。

**干预方式：**如下图所示：



案例一：误差值大于 0.5 但匹配成功

案例二：误差值大于 0.5 但匹配失败

当人工匹配接受时，输入 1；相反，人工匹配拒绝时，输入 0。

**情况二：**当组内匹配成功后，组间匹配失败的碎片片段，通过人工干预的方式完成组间的匹配。

**干预方式：**由于碎片片段数量有限，可通过观察直接获得位于同一行的两片段的序号。

## 六、模型检验与结果分析

### 6.1 碎纸片复原中文算法的检验

为了检验我们的模型是否具有较强的适用性，我们利用现有文档，将其裁剪后打乱顺序，随后利用复原算法对其进行复原，以此来检验算法。其步骤如下：

我们首先利用 PDF XChange Viewer 软件将下列所示 PDF 文档转换为 bmp 图像，然后利用 Photoshop 图像处理软件利用基准线和裁剪工具，将其划分为 196\*2145 像素点的 10 个碎纸片，然后随机编码这些碎纸片，利用碎纸片复原的中文算法对这些随机编码的碎纸片进行复原，以此来检验算法的正确性。



改进和优化。(2) 模型中可以考虑将适当的语料库加入，利用 OCR 工具将局部结果进行文字转化然后与语料库进行匹配，以提高模型的准确性，减少人工干预量。(3) 现有模型的算法对于相关参数比较敏感，为了提高算法的适用性，可以尝试引入模拟退火算法、遗传算法、蚁群算法等智能算法。(4) 模型中针对的碎纸片形状都是规则的矩形，而实际情况的碎纸片往往是不规则的，为了提高模型的适用性，可以考虑将形状的匹配纳入我们的模型。(5) 模型中没有考虑实际情况中可能存在彩色字体，在碎纸片拼接的实际应用中可以考虑将颜色作为匹配的依据，这样能够适当提高模型的准确性。

## 八、模型优缺点

表 10 模型的优缺点

| 优点   | 缺点   |
|--|--|
| (1) 模型中针对不同碎片类型和尺寸，具体问题具体分析，为每种情况设计了不同的算法实现了碎纸片的复原。                        | (1) 模型中仅针对中文和英文的字符特点进行了匹配算法的设计，缺乏对中英文混合以及阿拉伯字符等更一般情况的考虑。 |
| (2) 为了能够结合计算机进行人工干预，我们专门在计算机很难分辨的特殊情况下设计了人工干预 GUI 界面，使得人工干预与机器复原巧妙地结合到了一起。 | (2) 算法的匹配准确率对于相关参数比较敏感，因此应用于不同碎纸片对象时需重新设定。               |
| (3) 模型中对于中文和英文的字符特点进行了充分挖掘，使得模型对于中文和英文的匹配准确度较高。                            | (3) 模型中缺乏针对碎纸片的语义分析，人工干预量还有待进一步减少。                       |

## 九、参考文献

- [1] Christian Schauer, Matthias Prandtstetter, and Günther R.Raidl. A Memetic Algorithm for Reconstructing Cross-Cut Shredded Text Documents. Institute of Computer Graphics and Algorithms Vienna University of Technology, Vienna, Austria.
- [2] Johannes Perl, Markus Diem, Florian Kleber, and Robert Sablatnig. Strip Shredded Document Reconstruction Using Optical Character Recognition. In Imaging for Crime Detection and Prevention 2011 (ICDP 2011), 4th International Conference on, pages 1 –6, nov. 2011.
- [3] 罗智中。基于文字特征的文档碎纸片半自动拼接，计算机工程与应用。2012，48(5)：207---210。
- [4] 刘金根，吴志鹏。一种基于特征区域分割的图像拼接算法[J]。西安电子科技大学学报，2002，29（6）：768-771。
- [5] Prim, R.C.: Shortest Connection Networks and Some Generalizations. The Bell System Technical Journal 3,1389—1401(1957).
- [6] 王剑文，戴光明，谢柏桥，张全元。求解 TSP 问题算法综述[J]。计算机工程与科学.2008(02)。
- [7] 张辉,赵正德,杨立朝,赵郁亮. TSP 问题的算法与应用的研究[J]。计算机应用与软件. 2009(04)。

## 十、附录

### 10.1 程序运行环境

本文题的解决主要在 Mathwork matlab 中完成代码的编写和编译。计算机操作系统：32 位 Win7 操作系统。软件环境：Matlab R2012b。

### 10.2 碎片拼接复原结果

#### (1) 附件 1 复原图

城上層樓盤巖。城下清淮古沐。舉手揖吳云。人與碧天俱遠。憑斷。魂斷。屋梁松月滿。簷簷衣市莎寒花。村里村北晚鐘車。牛衣古樹黃瓜。尚余珠璣一重。清曉近市橋。照脂粉與勾溪。偏向臉邊流。小娘非常蓮記。二南依旧能詩。更有鮑參軍切諫。几平篇數知。自古相从休務日。何姑似唱離吟。天垂云玉作容顏。坐中人半醉。窗外雪傳深。數聲殘照。晴窗柳渡風偏翠。妙舞難駐。雪上青輕露態妍。碧幕輕籠兩風。零煙淡拂雙鴛。為誰流解不歸家。錯認門前過馬。

我功暫張歸去好。从来自已忘情。坐心清盡還心平。江南与奉北。何處不堪行。閑莠陽。誰念雲橫塞王。何曾夢雲南。旧恨前歡。心事兩无展。要知欲見无由。痴心犹自。倩人道。一声传语。风卷珠帘自上钩。萧萧乱叶似新秋。独携纤手上高樓。臨水縱凌回曉碧。归来转覺情懷動。梅笛隨中聞几弄。秋晴重。西山雲淡云漸凍。免為愁遠。見長空万里。云无留迹。雄雉飞来光射处。冷浸一天秋碧。玉宇琼樓。紫雲來去。人在清涼國。江山如畫。望中樓閣歷歷。省可清言揮玉尘。真傳寶鼎金真。風塵何處過家純。不应用烏客。惟要卓文君。自惜风流云雨散。关山有恨清无限。待君空见寻芳伴。方说相思。目断西樓燕。黃根黃花未吐。且教紅粉相扶。酒醒不必看柔莢。惜卿人問今古。玉容那忍薄雲。冰姿自有仙風。尚仙附處探芳丛。倒挂綠毛么鳳。

烟豆庚桑曾過女。凭君流与尚矣。即西吳越報丰登。君王如有問。結珠璣玉生。師母誰家。崇風雨兩。借君拍板与門墻。我也逐逐作戏。笑相疑。翠膠樓枕印。印就嫌厚。閑閒疎妝。幾波麻風雨。可恨相摩能几日。不知重會是何年。華資仔細更重看。午夜風輕暖。三更月似床。雙紋如水玉臥凉。何物与依归去。有残妝。金炉犹睡烟凝曉。惜香更把宝钗鎖。重開处。余薰在。这一番。气味胜从前。菊蕊荷枯一夜霜。新苞綠叶無絲光。竹葉茶香出青窠。霜降水痕收。淡香疎菊露深。酒力漸消风力軟。尚處。破曉多情却恋頭。映影偏風。一枕傷春緒。归不去。風樓何處。芳草迷歸路。湯安云。蘇軾白。香浮花乳輕。人間誰處爭妍。斗取紅塵面。炙手無人便屋頭。蕭蕭晚雨脫輕裘。誰怜李下戴銀鉤。

附图 1

## (2) 附件 2 复原图

fair of face.  
The customer is always right. East, west, home's best.  
Life's not all beer and skittles. The devil looks after his own.  
Manners maketh man. Many a mickle makes a muckle. A man who is his own lawyer has a fool for his client.  
You can't make a silk purse from a sow's ear. As thick as thieves. Clothes make the man. All that glisters is not gold.  
The pen is mightier than sword. Is fair and wise and good and gay. Make love not war. Devil take the hindmost. The female of the species is more deadly than the male. A place for everything and everything in its place. Hell hath no fury like a woman scorned. When in Rome, do as the Romans do. To err is human; to forgive divine. Enough is as good as a feast. People who live in glass houses shouldn't throw stones. Nature abhors a vacuum. Moderation in all things.  
Everything comes to him who waits. Tomorrow is another day. Better to light a candle than to curse the darkness.  
Two is company, but three's a crowd. It's the squeaky wheel that gets the grease. Please enjoy the pain which is unable to avoid. Don't teach your Grandma to suck eggs. He who lives by the sword shall die by the sword. Don't meet troubles half-way. Oil and water don't mix. All work and no play makes Jack a dull boy.  
The best things in life are free. Finders keepers, losers weepers. There's no place like home. Speak softly and carry a big stick. Music has charms to soothe the savage breast. Ne'er cast a clout till May be out. There's no such thing as a free lunch. Nothing venture, nothing gain. He who can does, he who cannot, teaches. A stitch in time saves nine. The child is the father of the man. And a child that's born on the Sab-

附图 2

## (3) 附件 3 复原图

傾城，溫香熟。懷雲含翠兩眉。多情看，不是花紅是玉紅。一種嬌  
情清淨，不離香骨，只要人知。玉面飛紅幾處。願乞春華共惜，莫  
誤消息。惜春惜惜。願人向。前時猶見楚宮嬌。春華共惜芳草  
歇。望風光，又況清明。小玉飛紅人別。落紅處。時時春。玉蕊  
須知。要尋花，趁芳心未老。佳處遇時。落紅處。時時春。玉蕊  
飛。惟惜芳心。一任芳心。相對約約。記取春風正。正芳心好  
處。空翠滿。空翠神相得。如與約約。約約。東海遙。雲霞冷。松  
葉寒枝。春情轉。不應留。芳心未老。料得芳心未老。雲霞冷。松  
葉寒枝。春情轉。不應留。芳心未老。料得芳心未老。雲霞冷。松  
葉寒枝。春情轉。不應留。芳心未老。料得芳心未老。雲霞冷。松  
葉寒枝。春情轉。不應留。芳心未老。料得芳心未老。雲霞冷。松

九十日春都过了，贪忙何处追游。三分春色一分愁。雨飘榆荚阵，风转柳花球。白雪清词出坐间。爱君才谔两俱全。异乡风景却依然。问厨只留剩馥，新丝那觫系行人。酒阑滋味似残春。

缺月向人舒窈窕，三星当户照徘徊。香生数畝闻环珮，猿背颠狂归去。山负功名嫌累赘，何如使君才与水，何如。占行人闲一味道，向东头，山尽处。自古空来谁去，搔首何如，秋欲别，使君行不归。别语劝君君一醉，泪满罗带，又是何郎醉，记取秋夜凉如水。莫待付与东流水，西窗山边白鸟飞。落花溪外片帆微。桃花流水逐春归。主人醉小，欲向东风先醉倒，已属春家。且要从容等佳处，恩私已无当世望，似君须向古人求。岁暮松柏犹青青。

水涵空，山横市。西汉二疏乡里。新白发，旧黄金。故人黑头深。谁道东阳都瘦损。凝然点漆精神。瑶林终自属风尘。试看拔葵菜，仍是谪仙人。三过平山堂下，半生弹指声中。十年不见老仙翁。壁上龙蛇飞动。赋句不离芭蕉叶，片片著人无数。楼上望归去，芳草迷归路。犀钱玉果，利市平分沾四座。多少过客。此事何如到渔父。元像是欢喜好。何况公庭民讼少。万室皆来听上寿。十千神酒浇海岛。

虽抱文章，开口谁亲。且陶陶、乐尽天真。几时归去，作个闲人。对一张琴，一壶酒，一溪云。相如未老，梁苑犹能俊。草草闲愁，目断

附圖 3

#### (4) 复原 4 复原图

bath day. No news is good news.

Procrastination is the thief of time. Genius is an infinite capacity for taking pains. Nothing succeeds like success. If you can't beat 'em, join 'em. After a storm comes a calm. A good beginning makes a good ending.

One hand washes the other. Talk of the Devil, and he is bound to appear. Tuesday's child is full of grace. You can't judge a book by its cover. Now drips the saliva, will become tomorrow the tear. All that glitters is not gold. Discretion is the better part of valour. Little things please little minds. Time flies. Practice what you preach. Cheats never prosper.

The early bird catches the worm. It's the early bird that catches the worm. Don't count your chickens before they are hatched. One swallow does not make a summer. Every picture tells a story. Softly, softly, catchee monkey. Thought is already is late, exactly is the earliest time. Less is more.

A picture paints a thousand words. There's a time and a place for everything. History repeats itself. The more the merrier. Fair exchange is no robbery. A woman's work is never done. Time is money.

Nobody can casually succeed, it comes from the thorough self-control and the will. Not matter of the today will drag tomorrow. They that sow the wind, shall reap the whirlwind. Rob Peter to pay Paul. Every little helps. In for a penny, in for a pound. Never put off until tomorrow what you can do today. There's many a slip twixt cup and lip. The law is an ass. If you can't stand the heat get out of the kitchen. The boy is father to the man. A nod's as good as a wink to a blind horse. Practice makes perfect. Hard work never did anyone any harm. Only has compared to the others early, diligently

附图 4

### (5) 复原 5 复原图

### ①其中一面

What can't be cured must be endured. Bad money drives out good. Hard cases make bad law. Talk is cheap. See a pin and pick it up, all the day you'll have good luck; see a pin and let it lie, bad luck you'll have all day. If you pay peanuts, you get monkeys. If you can't be good, be careful. Share and share alike. All's well that ends well. Better late than never. Fish always stink from the head down. A new broom sweeps clean. April showers bring forth May flowers. It never rains but it pours. Never let the sun go down on your anger.

Pearls of wisdom. The proof of the pudding is in the eating. Parsley seed goes nine times to the Devil. Judge not, that ye be not judged. The longest journey starts with a single step. Big fish eat little fish. Great minds think alike. The end justifies the means. Cowards may die many times before their death. You can't win them all. Do as I say, not as I do. Don't upset the apple cart. Behind every great man there's a great woman. Pride goes before a fall.

You can lead a horse to water, but you can't make it drink. Two heads are better than one. March winds and April showers bring forth May flowers. A swarm in May is worth a load of hay; a swarm in June is worth a silver spoon; but a swarm in July is not worth a fly. Might is right. Let bygones be bygones. It takes all sorts to make a world. A change is as good as a rest. Into every life a little rain must fall. A chain is only as strong as its weakest link.

Don't look a gift horse in the mouth. Old soldiers never die, they just fade away. Seeing is believing. The opera ain't over till the fat lady sings. Silence is golden. Variety is the spice of life. Tomorrow never comes. If it ain't broke, don't fix it. Look before you leap. The road to hell is paved with good

附图 5

## ②另外一面

He who laughs last laughs longest. Red sky at night shepherd's delight; red sky in the morning, shepherd's warning. Don't burn your bridges behind you. Don't cross the bridge till you come to it. I hindsight is always twenty-twenty.

Never go to bed on an argument. The course of true love never did run smooth. When the oak is before the ash, then you will only get a splash; when the ash is before the oak, then you may expect a soak. What you lose on the swings you gain on the roundabouts.

Love thy neighbour as thyself. Worrying never did anyone any good. There's nowt so queer as folk. Don't try to walk before you can crawl. Tell the truth and shame the Devil. From the sublime to the ridiculous is only one step. Don't wash your dirty linen in public. Beware of Greeks bearing gifts. Horses for courses. Saturday's child works hard for its living.

Life begins at forty. An apple a day keeps the doctor away. Thursday's child has far to go. Take care of the pence and the pounds will take care of themselves. The husband is always the last to know. It's all grist to the mill. Let the dead bury the dead. Count your blessings. Revenge is a dish best served cold. All's for the best in the best of all possible worlds. It's the empty can that makes the most noise. Never tell tales out of school. Little pitchers have big ears. Love is blind. The price of liberty is eternal vigilance. Let the punishment fit the crime.

The more things change, the more they stay the same. The bread always falls buttered side down. Blood is thicker than water. He who fights and runs away, may live to fight another day. Eat, drink and be merry, for tomorrow we die.

附图 6

## 10.3 运行程序

```
10.3.1 main.m
tic
n=19;
[Edge,row]=match(n);
rowNumber=length(row);
```

```

%% Find the number of pixal of each line left and right edge in picture.
% count is a cell for storage of each line left and right edge pixal
% number.
for i=1:n
    temp=zeros;
    tempEdge=Edge{i};
    for j=1:rowNumber
        temp(j,1)=length(find(tempEdge(row(j):row(j)+40,1)==0));
        temp(j,2)=length(find(tempEdge(row(j):row(j)+40,2)==0));
    end
    count{i}=temp;
end

%% First step is to find the shred with blank left.
result=zeros(19);
for j=1:n
    if sum(count{1,j}(:,1))==0
        result(1)=j;
    end
end

%% Second step is to find the proper picture for the shred with blank left.

start=count{1,result(1)};
picturePool=dropPool([1:19],result(1));
pairA=result(1);
number=2;
tempSimilarity=zeros;
while (~isempty(picturePool))

    tempSimilarity=[];
    for k=1:length(picturePool)
        pairB=picturePool(k);
        tempPic=count{1,picturePool(k)};
        similarity=similar(start,tempPic);
        tempSimilarity=[tempSimilarity;pairA,pairB,similarity];
    end

result(number)=tempSimilarity(find(tempSimilarity(:,3)==max(tempSimil
arity(:,3))),2);
start=count{1,result(number)};
picturePool=dropPool(picturePool,result(number));
pairA=result(number);
number=number+1;

```



```

end
toc

10.3.2 main.m
clc,clear
tic
n=209;
[image,I,Edge,Distance,grayEdge]=match(n);
countLeft=1;
countRight=1;
for i=1:n
    tempEdge=Edge{i};
    if length(find(tempEdge(:,1))==1)==180 && Distance(i,1)>5
        leftHemp(countLeft)=i;
        countLeft=countLeft+1;
        continue;
    end

    if length(find(tempEdge(:,2))==1)==180 && Distance(i,2)>5
        rightHemp(countRight)=i;
        countRight=countRight+1;
        continue;
    end
end

%% Find the number of pixal of each line left and right edge in picture.
% count is a cell for storage of each line left and right edge pixal
% number.

picType=[];
picDistance=[];
for i=1:n

    [A,B]=pictureAttribute(I{i});
    picType=[picType;A];
    picDistance=[picDistance;B];

end

%%

```

```

%Create the remaining pool
picturePool=[1:209];
nleft=length(leftHemp);
nright=length(rightHemp);
totalHemp=nleft+nright;
for i=1:totalHemp
    if i<=length(leftHemp)
        picturePool=dropPool(picturePool,leftHemp(i));
    else
        picturePool=dropPool(picturePool,rightHemp(i-nleft));
    end
end

%
totalPool=length(picturePool);
pictureHemp=zeros(22,19);
pictureCount=zeros(22,1)+1;
count=totalPool*30;
while (count~=0)
    tic
    i=ceil(length(picturePool)*rand());
    for j=1:totalHemp
        if j<=length(leftHemp)
            if picType(leftHemp(j),1)==picType(picturePool(i),1) &&
picType(leftHemp(j),2)==picType(picturePool(i),2)
                if
abs(picDistance(leftHemp(j),1)-picDistance(picturePool(i),1))<3    &&
abs(picDistance(leftHemp(j),2)-picDistance(picturePool(i),2))<3
                    pictureHemp(j,pictureCount(j,1))=picturePool(i);
                    pictureCount(j,1)=pictureCount(j,1)+1;
                    picturePool=dropPool(picturePool,picturePool(i));
                    break;
                elseif
abs(picDistance(leftHemp(j),1)-picDistance(picturePool(i),1))<8    &&
abs(picDistance(leftHemp(j),2)-picDistance(picturePool(i),2))<8
                    subplot(1,2,1),imshow(image{leftHemp(j)});
                    subplot(1,2,2),imshow(image{picturePool(i)});
                    prompt = {'Á»ÖßÔÚÍ-Ò»ÐÐÂð£¿0 is ²»ÔÚ£-1 is ÔÚ££'};
                    dlg_title = 'ÈÈ¹¼ÃÐ¶Í';
                    num_lines = 1;
                    def = {'0'};
                    answer = inputdlg(prompt,dlg_title,num_lines,def);
                    flag=str2num(answer{1,1});
                    if flag==1

```

```

        pictureHemp(j,pictureCount(j,1))=picturePool(i);
        pictureCount(j,1)=pictureCount(j,1)+1;
        picturePool=dropPool(picturePool,picturePool(i));
        break;
    end
end

end

else

    if picType(rightHemp(j-nleft),1)==picType(picturePool(i),1) &&
picType(rightHemp(j-nleft),2)==picType(picturePool(i),2)
        if
abs(picDistance(rightHemp(j-nleft),1)-picDistance(picturePool(i),1))<
3    &&
abs(picDistance(rightHemp(j-nleft),2)-picDistance(picturePool(i),2))<
3
            pictureHemp(j,pictureCount(j,1))=picturePool(i);
            pictureCount(j,1)=pictureCount(j,1)+1;
            picturePool=dropPool(picturePool,picturePool(i));
            break;
        elseif
abs(picDistance(rightHemp(j-nleft),1)-picDistance(picturePool(i),1))<
3    &&
abs(picDistance(rightHemp(j-nleft),2)-picDistance(picturePool(i),2))<
3
            subplot(1,2,1),imshow(image{picturePool(i)});
            subplot(1,2,2),imshow(image{rightHemp(j-nleft)});
            prompt = {'Á¸ÖßÔÚÍ-Ò»ÐÐÂð£¿0 is ²»ÔÚ£-1 is ÔÚ;£'};
            dlg_title = 'ÈÈ¹¼ÅÐ¶Í';
            num_lines = 1;
            def = {'0'};
            answer = inputdlg(prompt,dlg_title,num_lines,def);
            flag=str2num(answer{1,1});
            if flag==1
                pictureHemp(j,pictureCount(j,1))=picturePool(i);
                pictureCount(j,1)=pictureCount(j,1)+1;
                picturePool=dropPool(picturePool,picturePool(i));
                break;
            end
        end
    end
end
end

```

```

        end

    end

    toc

    count=count-1;

end

%% First step is to find the shred with blank left.
leftHeadPool=leftHemp;
RightHeadPool=leftHemp;

for i=1:totalHemp
    if i<=nleft
        Head=leftHemp(i);
        sequence=pictureHemp(i,:);

        [result,leftHeadPool,RightHeadPool,picturePool] =
LeftToRightMatch( Head,grayEdge,sequence,leftHeadPool,RightHeadPool,p
icturePool,pictureHemp,rightHemp,picType,picDistance);
        leftResult{i}=result;
    else
        Head=rightHemp(i-nleft);
        sequence=pictureHemp(i,:);

        [result,leftHeadPool,RightHeadPool,picturePool] =
RightToLeftMatch( Head,grayEdge,sequence,leftHeadPool,RightHeadPool,p
icturePool,pictureHemp,leftHemp,picType,picDistance);
        rightResult{i-nleft}=result;
    end
end

%% Second step is to find the proper picture for the shred with blank left.

start=count{1,result(1)};
picturePool=dropPool([1:19],result(1));
pairA=result(1);
number=2;
tempSimilarity=zeros;
while (~isempty(picturePool))

    tempSimilarity=[];
    for k=1:length(picturePool)

```

```

        pairB=picturePool(k);
        tempPic=count{1,picturePool(k)};
        similarity=similar(start,tempPic);
        tempSimilarity=[tempSimilarity;pairA,pairB,similarity];
    end

result(number)=tempSimilarity(find(tempSimilarity(:,3)==max(tempSimilarity(:,3))),2);
    start=count{1,result(number)};
    picturePool=dropPool(picturePool,result(number));
    pairA=result(number);
    number=number+1;
end
toc

10.3.3 main.m
clc,clear
tic
n=209;
[image,I,Edge,Distance,grayEdge]=match(n);
%% Find the picture with left and right blank.
%
countLeft=1;

for i=1:n
    flag=0;

        tempEdge=Edge{1,i};
        if (length(find(tempEdge(:,1)==1))==180 && Distance(i,1)>4) ||
            (length(find(tempEdge(:,2)==1))==180 && Distance(i,2)>4)

            tempEdge=Edge{2,i};
            if (length(find(tempEdge(:,1)==1))==180 &&
                Distance(209+i,1)>4) || (length(find(tempEdge(:,2)==1))==180 &&
                Distance(209+i,2)>4)

                leftHemp(countLeft)=i;
                countLeft=countLeft+1;
                continue;
            end
        end
end
end

```

```

end

%% Find the number of pixal of each line left and right edge in picture.
% count is a cell for storage of each line left and right edge pixal
% number.

picType=[];
picDistance=[];
for i=1:n
    tempA=[];
    tempB=[];
    for row=1:2
        [A,B]=pictureAttribute(I{row,i});
        tempA=[tempA A];
        tempB=[tempB B];
    end
    picType=[picType;tempA];
    picDistance=[picDistance;tempB];
end

picturePool=[1:209];
picturePool=[picturePool;picturePool];
nleft=length(leftHemp);
for i=1:nleft
    tempPool1=dropPool(picturePool(1,:),leftHemp(i));
    tempPool2=dropPool(picturePool(2,:),leftHemp(i));
    picturePool=[tempPool1;tempPool2];
end

%
totalPool=length(picturePool);
pictureHemp=zeros(11,19);
pictureCount=zeros(11,1)+1;
count=totalPool*60;
while (count~=0)
    tic
    i=ceil(length(picturePool)*rand());
    error=[];
    for j=1:nleft
        error=[error;leftHemp(j) picturePool(i)

```

```

(picType(leftHemp(j),1)==picType(picturePool(i),1) &&
picType(leftHemp(j),2)==picType(picturePool(i),2))
abs(picDistance(leftHemp(j),1)-picDistance(picturePool(i),1))+abs(pic
Distance(leftHemp(j),2)-picDistance(picturePool(i),2))];
    end

    temp=(find(error(:,4)==min(error(:,4))));
    ti=length(temp);

    for k=1:ti
        % for j=1:nleft
            if error(temp(k,1),3)==1

pictureHemp(temp(k,1),pictureCount(temp(k,1),1))=picturePool(i);
                pictureCount(temp(k,1),1)=pictureCount(temp(k,1),1)+1;
                picturePool=dropPool(picturePool,picturePool(i));
                break;
            end
        %end
    end
    toc
    count=count-1;
end

%% First step is to find the shred with blank left.
leftHeadPool=leftHemp;

for i=1:nleft

    Head=leftHemp(i);
    sequence=pictureHemp(i,:);

    [result,leftHeadPool,picturePool] =
LeftToRightMatch( Head,image,grayEdge,sequence,leftHeadPool,picturePo
ol,picType,picDistance);
    leftResult{i}=result;

end

%% Second step is to find the proper picture for the shred with blank left.

start=count{1,result(1)};
picturePool=dropPool([1:19],result(1));
pairA=result(1);

```

```

number=2;
tempSimilarity=zeros;
while (~isempty(picturePool))

    tempSimilarity=[];
    for k=1:length(picturePool)
        pairB=picturePool(k);
        tempPic=count{1,picturePool(k)};
        similarity=similar(start,tempPic);
        tempSimilarity=[tempSimilarity;pairA,pairB,similarity];
    end

    result(number)=tempSimilarity(find(tempSimilarity(:,3)==max(tempSimilarity(:,3))),2);
    start=count{1,result(number)};
    picturePool=dropPool(picturePool,result(number));
    pairA=result(number);
    number=number+1;
end
toc

```