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）：73

所属学校（请填写完整的全名）：上海交通大学

参赛队员（打印并签名）：1. 李斌彬

2. 郝楠

3. 陈颖昭

指导教师或指导教师组负责人（打印并签名）：数模指导组

（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 15 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）：

基于内容相关的碎纸拼接问题的建模与求解

摘要

本文针对文字文件等距横纵切割的碎纸片拼接复原问题，考虑边缘信息有限、双面拼接等实际情况，充分合理利用碎片信息，建立碎纸片拼接复原模型，实现完整高效的碎纸拼接。模型的核心算法为边缘匹配、文字高度匹配和同步验证，应用到了局部拼接模型、数据统计模型和合理优化模型。利用 MATLAB 2013a 编程，对模型进行求解验证，实现碎片的复原，同时对算法进行了一定的讨论优化。

对于问题 1，利用碎片边缘灰度信息提供的约束条件，构建碎纸拼接模型 Ⅰ，由于问题 1 的求解规模很小，所以采用穷举搜索算法，并且只考虑边缘灰度信息一种匹配方法。利用 MATLAB 编程求解验证，得到了附件 1，2 的拼图（表格见表一表二，拼图见附录一）。

对于问题 2，在碎纸拼接模型 Ⅰ 基础上进行优化，构建拼接模型 Ⅱ，先利用文字高度匹配法（行间距匹配）对所有碎片进行分组，再在各组内以边缘灰度信息匹配和同步反向验证进行局部拼接，此间需要极少数次人工干预。局部拼接完成后再利用上述两种匹配方法实现所有拼接。利用 MATLAB 编程求解，得到了附件 3 的拼图和附件 4 的部分拼图（表格见表三表四，拼图见附录一）。

对于问题 3，增加了双面匹配的约束条件，采用碎纸拼接模型 Ⅲ，先考虑所有正面碎片的拼接，在分组和匹配时，以反面信息作为制约检验条件，利用 MATLAB 编程求解，得到了附件 5 的拼图（表格见表五，拼图见附录一）。

最后，我们就模型的结果进行了分析和评价。

关键字：碎纸拼接模型 文字内容拼接 灰度相关匹配算法 边缘匹配

1.问题重述

破碎文件的拼接在司法物证复原、历史文献修复及军事情报获取等领域都有着重要的应用。在碎片形状一致、内容为纯文字的情况下，要求讨论以下问题：

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。

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 的碎片数进行拼接复原。

上述问题复原结果要求以图片形式和规定表格形式表达。

【数据文件说明】

(1) 每一附件为同一页纸的碎片数据。

(2) 附件 1、附件 2 为纵切碎片数据，每页纸被切为 19 条碎片。

(3) 附件 3、附件 4 为纵横切碎片数据，每页纸被切为 11×19 个碎片。

(4) 附件 5 为纵横切碎片数据，每页纸被切为 11×19 个碎片，每个碎片有正反两面。该附件中每一碎片对应两个文件，共有 2×11×19 个文件，例如，第一个碎片的两面分别对应文件 000a、000b。

2.模型的假设与符号说明

2.1 模型的假设

（1）附件中的碎片均满足理想碎片模型，即碎片拼接复原后的图像与未切割前完全一致，不存在任何残缺或误差，也即任何相邻两碎片的边缘连续。

（2）一个印刷文字文件中的每一行和边缘两列均与纸张的边缘平行，所有文字格式（字体大小颜色间距）均一致，横纵切割均匀等间距且与纸张边缘平行，所有碎片大小形状一致，可以保证像素均为 72*180。

（3）文件的两侧空白明显大于文字间距，即所有最左侧碎片可以直接检测到，右侧亦然。基于一定测量和计算，所有文件的边缘空白宽度均大于 10 像素。

（4）基于一定的测量及计算，假设附件三中每个汉字所占的高度为 39 像素，行间距为 30 像素。（具体算法见 4.2.（6）文字高度和行间距的统计模型）

2.2 符号的说明

a=72　表示所有碎片宽度的像素值；

b=180　表示所有碎片高度的像素值。

$00i$　表示按题目所给顺序的第 i 个碎片；

X_i　表示按题目所给顺序的第 i 个碎片在 MATLAB 中读出的矩阵；

R_A　表示矩阵 A 最右侧一列的向量；

$\vec{L}_A$ 表示矩阵 A 最右侧一列的向量；

M_A 表示矩阵 A 第一列到第十一列组成的矩阵；

附件一附件二：

Y_i 表示复原后从左往右的第 i 个碎片对应的矩阵；

附件三附件四附件五：

d 表示文件中文章行间距的所占的像素值的平均数；

h 表示文件中汉字行所占高度的像素值的平均数；

L_k 表示第 k 个碎片对应的矩阵，同时满足第 k 个碎片为原文件左侧边缘碎片；

R_k 表示第 k 个碎片对应的矩阵，同时满足第 k 个碎片为原文件右侧边缘碎片；

$\vec{H}_i$ 表示第 i 个碎片由各行文字的中心高度在对应矩阵中的行数所组成的向量；

m_i 表示第 i 个碎片的第一个完整文字行的上边界在该碎片对应矩阵 X_i 中的行数；

n_i 表示第 i 个碎片的第一个完整文字行的下边界在该碎片对应矩阵 X_i 中的行数；

G_{kj} 表示第 j 个碎片对应的矩阵，k 表示分到 L_k 一组；

G_{kj} 表示且与分到 L_k 一组且又是 R_k 的矩阵；

三，问题分析

半自动碎纸拼接复原技术是根据碎片的轮廓及内容的特征信息，利用相关算法模型进行匹配，从而高效完整复原图像的技术，在许多行业中均有着广泛应用。在实际生活中，大多数碎纸是不规则的图像，因此相应的拼接一般基于轮廓，轮廓不规则的图像碎片虽然信息繁杂处理难度高，要经过选取合适的特征描述、轮

廓提取等预处理、还要实现更加复杂的碎片匹配和拼接算法，但图像中包含的特征信息也相对丰富，可以提取链码、角点、样条等进行特征描述。

问题中被切割的文件为印刷文字文件，文件每一行和边缘两列均与纸张边缘平行，所有文字格式（字体大小颜色间距）均一致，横纵切割均匀等间距且与纸张边缘平行，所有碎片大小形状一致。因此可以排除碎片的轮廓信息，将碎片边缘和文字分布信息作为关键特征。基于内容相关的碎片拼接的主要工作流程可概括如下：

（1）碎片预处理：即将物体碎片数字化，得到碎片的数字图像。

（2）碎片匹配：通过匹配算法找出相互匹配的图像碎片。算法的核心是边缘匹配法、文字高度匹配法（也即行高度匹配法）和同步验证法。

（3）碎片拼合：将相互匹配的图像碎片拼合在一起得到最终正确结果。

碎片匹配算法依题意应满足以下两个要求：

1.算法可将题目所给的图片一次性完整正确拼接复原。

2.算法尽可能减少计算量，人工干预次数尽可能降低。

对于三个问题，具体分析如下：

问题一：

首先，对于附件 1，碎片符合假设（1）的情况，由于仅有纵切，每张碎片的高度均为原文件的高度。可以观察到，碎片的宽度大于一个汉字加两个空格且小于两个汉字加一个空格，则任意碎片两侧边缘都有相当多的被截断的汉字，由汉字笔画的连续性和假设（1）中碎片的连续性，易知这些截断的汉字笔画信息可成为匹配的关键。

在 MATLAB 中读入图片，即可生成图片的灰度矩阵，矩阵的最左侧向量和另一矩阵的最右侧向量间距越小，则说明这两矩阵对应的碎片边缘笔画吻合度高。由于被截断汉字数目多和匹配对象的数目少，正确对象的匹配吻合度远大于其他对象的吻合度，因此可搜索出最佳匹配。

经检验，算法运行结果理想。考虑优化模型，可以将穷举搜索法改为条件搜索法，先将最左侧碎片找出，然后从左往右依次匹配，每次匹配成功的碎片将不参与下一轮匹配。

对于附件 2，文件上的印刷文字均为英文，英文字母较汉字而言笔画明显简单、空白分布增多、横向笔画也更难检测。但与附件一同理，碎片高度大导致边缘截断笔画多且碎片数目少，重复附件一的解法，也能迅速拼出原文件。

这一类模型的缺点在于，由于我们只以边缘灰度相关匹配作为唯一的判断标准，即在所有向量距离中寻找最小值，若一个边缘向量接近零向量，相应地，其对应碎片边缘被截笔画较少，则其匹配的误差将难以控制。另外，没有其他判断标准保证符合向量距离极小值的碎片一定是正确的匹配。所以本题解法仅适用于碎片边缘信息丰富、特征明显、搜寻对象群体小的拼接问题。

问题二：

先考虑附件三，一张印刷汉字文件被横纵切割成 11*19 张轮廓相同小碎片。若依旧使用第一题的灰度相关匹配法和穷举搜寻法，拼接难以实现，主要原因有：

（1）边缘信息少。本题的碎片在字体大小和横向宽度与附件一相同的情况

下，高度变为了原来的十一分之一。即碎片边缘包含的灰度信息利用率将大大降低，极易出现单侧（图 1）或双侧（图 2）边缘信息缺失的碎片，它们的边缘向量接近零向量，同时碎片数量也成为原先的十一倍，所以利用灰度信息相对匹配将会匹配到大量边缘向量接近零向量的碎片，正确率非常低。可见此时边缘信息的获取量无法满足本题的要求。

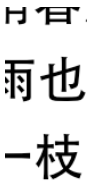


图 1

图 2

（2）原来的拼接模型判断标准只有单方向灰度相关匹配一种，这样既无法保证这种单向搜寻找到的碎片一定为正确匹配。且在目前的碎片边缘灰度信息获取量低的情况下，形如上图右侧的碎片也无法找到其左侧的正确匹配。

（3）碎片数目多，利用问题一的穷举搜寻法将造成巨大的运算量和难以修改的误差。

所以，我们需要在前一题的拼接模型的基础上进行优化。

首先，因为单侧边缘灰度信息缺失极容易造成匹配出错，因此可以优先将最左侧和最右侧的各 11 个碎片挑选出（利用合理假设 3）。另外将其他单侧边缘灰度信息缺失的碎片进行整理观察。

其次，因为文章行距比文字间距明显宽，且文章中还存在段落完结换行的情况，纵向空白明显比横向多，即纵向文字截断率低，所以依旧先考虑横向（即左右）边缘匹配。

然后，仔细观察整理出的单侧边缘信息缺失碎片，发现碎片中文字的分布位置具有较强区分度，即各行文字的中心所在该碎片中的行数有较明显差别。尝试计算选出的最左侧 11 个碎片的文字中心行数（用向量表示，有几个文字中心即有几个向量维度），发现区分度足够明显。则考虑以碎片中各行文字中心所在行的向量为另一判断标准，将所有非文件边缘碎片进行分类，分到各最左侧碎片的组内。

在各组内利用边缘灰度相关匹配法从左至右依次进行搜寻，并且每求出一个匹配对象就对求得的对象应用反向边缘灰度相关匹配法，进行结果验证。对验证不通过的进行二次验证，若验证碎片与剩余碎片不匹配则默认成功继续匹配，否则将验证碎片设为断点，拼接中断并反向匹配。若反向匹配可一直顺利匹配到断点，则该行可以成功拼接，否则将进行人工干预。

再考虑附件 4，一张印刷英文文件被横纵切割成 11*19 张轮廓相同小碎片。仿照附件 3，运用文字高度匹配法对碎片分组，再利用缘碎片优先搜寻法寻找出最左侧及最右侧的 22 个碎片，最后用灰度相关匹配法对碎片进行逐个匹配。但由于英文字母并不像中文汉字那样具有等大的性质，在分组时需要分多种情况（经观察共有 11 种情况）讨论进而将碎片归入合适的分类。

由于附件 4 中若干组的字母高度仅差两至三个像素，根据我们的分类方法很

难区分，因此将碎片分成了组数为19或38的若干组，通过逐个匹配并加以人工干预得到最终结果。

问题三：附件5与附件4的碎片大小相同，但具有更多的约束条件，对于任意两个碎片的比较可通过同时验证其背面对应的两个碎片是否匹配进行筛选。基本模型也和拼接模型II一致。

4.模型的建立和求解

4.1 拼接模型I（适用于附件1、2）

4.1.1 模型建立

（1）.图片处理

首先将文件中的图片$00i$按顺序在MATLAB中转化为第个i矩阵X_i,该矩阵的最左侧向量记为$\vec{L}_{Xi}$,最右侧向量记为$\vec{R}_{Xi}$，$(0 \le i < 19)$。

（2）.边缘碎片优先搜寻法

在$X_i\ (0 \le i < 19)$中找出原文件中最左侧的碎片对应矩阵，方法如下：

首先作如下分析:最左侧碎片具有左侧边缘空白带较多的特征信息,对应于矩阵,其左边缘值均为255的列向量较多,假设最左侧碎片左边最多有x个如此的向量,可以构建向量ones(height,1:x）*255，将每个碎片左边等大的向量与其作差,若差向量全部项的和为0则满足条件。通过从1开始递增x的值，直至所求项数量唯一，此唯一项即为所求。经多次计算，发现x>=7时所求项即可唯一，x=11时所求值依然存在，则x的建议值为11。

对每个矩阵X_i求其第一列到第十一列的矩阵M_{Xi}，若M_{Xi}=ones(180,1:11)*255,则将X_i记为Y_1,Y_1即为复原后从左到右第一张碎片的矩阵。

（3）.全局拼接法求

从左侧向右依次拼接图片，拼接完成第j步时得到Y_j。

$\|\vec{R}_{Yj} - \vec{L}_{X_i}\|\ (0 \le i < 19)$表示已经得到$Y_j$时，$Y_j$的右侧向量与$X_i\ (0 \le i < 19)$的左侧向量的距离。

$\min \| \vec{R}_{Yj} - \vec{L}_{Xi} | (0 \le i < 19)$ 为此距离的最小值，满足此极值条件的 X_i 即可看做和 Y_j 右侧边缘匹配度最高的矩阵，将 X_i 记为 Y_{j+1}，则得出复原后从左至右第 $i+1$ 个碎片，完成了第 $i+1$ 步拼接。然后继续重复上述做法求出 Y_{j+2}、Y_{j+3} ..... Y_{19}。

（4）.图像显示

得出矩阵列 $\{Y_i\}$ 后，按顺序在 MATLAB 中读出图片，即可得到复原后的图片。此模型对附件 1 和附件 2 均复原效果理想。

4.1.2 模型求解

分别在两个附件中运行附录二中的程序 ad_1，ad_2 得到如下结果：（复原拼图见附录一）

附件 1:

008	014	012	015	003	010	002	016	001	004	005	009	013	018	011	007	017	000	006

表一

附件 2:

003	006	002	007	015	018	011	000	005	001	009	013	010	008	012	014	017	016	004

表二

此模型对附件 1 和附件 2 均复原效果理想。

4.2 针对附件 3 的拼接模型 II

4.2.1 模型建立

（1）.图片处理

首先将文件中的图片 $00i$ 按顺序在 MATLAB 中转化为第个 i 矩阵 X_i，该矩阵的最左侧向量记为 $\vec{L}_{Xi}$，最右侧向量记为 $\vec{R}_{Xi}$，$(0 \le i \le 209)$。

(2).边缘碎片优先搜寻法

在 $X_i\ (0 \le i \le 209)$ 中找出原文件中最左侧和最右侧的碎片对应矩阵，方法同模型 I 中的（2）

若 X_k 为原文件中最左侧的碎片对应矩阵，令 $L_k = X_k$ 。

用 MATLAB 编程，运行后得出符合这样条件的 k 有：7、14、29、38、49、61、71、89、94、125、168.

若 X_k 为原文件中最右侧的碎片对应矩阵，令 $R_k = X_k$ 。

用 MATLAB 编程，运行后得出符合这样条件的 k 有：18、36、43、59、60、74、119、141、145、176、196.

（3）.文字行高度与行间距的统计模型

因为纸张印刷用同一字体，汉字行的高度是个定值，除少数汉字如汉字“一”外，多说汉字的高度与汉字行高度一致。用像素测量方式，测量记录多张碎片的相关数据。

基于实验测量数据，可求得汉字行的平均高度 h 为 39 个像素值，行间距的平均值 d 为 30 个像素值。方差值大小完全在可控范围内，所求平均值具有高度代表性，可以使用。

（4）.文字高度测量的优化模型

在本题的文字高度匹配算法中，我们选取第 i 个碎片中每行文字的高度中心所在的行数所组成的向量 $\vec{H}_i$ 作为特征描述。需特别注意，此中心为文件格式下文字行的高度中点，非被截文字部分的高度中点，即 $\vec{H}_i$ 反映了一个碎片中能完整出现的所有文字行的中心高度。

由假设（2），任意 X_i 行数为 180，又由上一步骤得知汉字平均高度为 39，行间距 30，（39*3+30*2）=177，易知任意碎片中最多只有三行完整文字（如下行图三），可能存在一碎片有两行不完整文字和两行完整文字可查出四行文字的情况（图四）。则任意 $\vec{H}_i$ ，其维数只可能是 2 或 3。

了，

司出

解系

图 3　　图 4

对任意一个 X_i，求其 $\overrightarrow{H_i}$ 的方法如下：

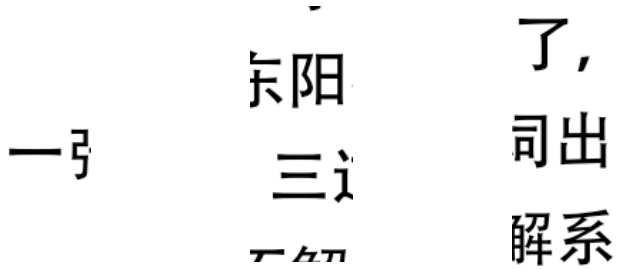
① 求出 X_i 的第一个完整行的上边界的行数记为 m_i，求出 X_i 的最后一个完整行的下边界的行数记为 n_i，计算 $n_i - m_i$ 的值。

下表为前 100 个碎片的 m_i、n_i 和 $n_i - m_i$。

numbe r	0	1	2	3	4	5	6	7	8	9
mi	57	23	38	26	13	33	63	57	45	44
ni	165	127	147	134	121	141	171	165	84	84
mi-mi	108	104	109	108	108	108	108	108	39	40
numbe r	10	11	12	13	14	15	16	17	18	19
mi	32	40	26	9	94	2	7	5	20	63
ni	140	146	134	115	134	177	47	177	127	171
mi-mi	108	106	108	106	40	175	40	172	107	108
numbe r	20	21	22	23	24	25	26	27	28	29
mi	63	7	38	20	44	44	20	69	40	33
ni	171	46	146	128	153	84	128	109	146	139
mi-mi	108	39	108	108	109	40	108	40	106	106
numbe r	30	31	32	33	34	35	36	37	38	39
mi	25	27	57	72	51	44	64	32	45	26
ni	128	134	97	177	158	153	171	140	152	134
mi-mi	103	107	40	105	107	109	107	108	107	108
numbe r	40	41	42	43	44	45	46	47	48	49
mi	14	20	51	51	34	57	45	51	32	38
ni	121	128	159	159	139	166	153	158	141	147
mi-mi	107	108	108	108	105	109	108	107	109	109
numbe r	50	51	52	53	54	55	56	57	58	59
mi	19	27	64	56	40	32	59	39	7	33
ni	128	133	171	165	147	141	164	146	159	140

mi-mi	109	106	107	109	107	109	105	107	152	107
numbe r	**60**	**61**	**62**	**63**	**64**	**65**	**66**	**67**	**68**	**69**
mi	4	63	19	63	32	39	9	63	57	64
ni	109	171	128	171	139	146	46	172	166	171
mi-mi	105	108	109	108	107	107	37	109	109	107
numbe r	**70**	**71**	**72**	**73**	**74**	**75**	**76**	**77**	**78**	**79**
mi	57	69	63	26	45	32	20	50	65	63
ni	97	176	171	134	83	139	128	158	172	170
mi-mi	40	107	108	108	38	107	108	108	107	107
numbe r	**80**	**81**	**82**	**83**	**84**	**85**	**86**	**87**	**88**	**89**
mi	69	45	26	69	50	2	22	20	45	84
ni	177	152	133	178	159	109	128	127	152	121
mi-mi	108	107	107	109	109	107	106	107	107	37
numbe r	**90**	**91**	**92**	**93**	**94**	**95**	**96**	**97**	**98**	**99**
mi	50	39	32	56	51	42	64	51	32	63
ni	159	147	140	96	158	146	172	158	139	171
mi-mi	109	108	108	40	107	104	108	107	107	108

② 已知文字行高度h和文章行距d，通过分类讨论可知，$n_i - m_i$的值一定接近$h = 39$、$2*h + d = 108$、$3*h + 2*d = 177$这三个数值中的一个。对应图像分别如下行展示。



$n_i - m_i$的值接近 39 时，可知此时一个文字中心高度为$m_i + n_i$,考察$m_i + n_i \pm (h + d)$、$m_i + n_i \pm 2(h + d)$是否大于 0，小于 180，若存在，则该值也为X_i的一个文字中心高度,将X_i的所有文字中心高度从小到大排列即得到向量$\overrightarrow{H_i}$。

$n_i - m_i$ 的值接近108或177时同理可以利用文字行高度 h 和行距 d 计算得到 $\vec{H}_i$ 。

下表为利用上述模型由MATLAB编程求解得到的前100个碎片的文字中心高度向量 $\vec{H}_i$

number	0	1	2	3	4	5	6	7	8	9
low	8	41	59	46	33	53	14	8	65	64
medium	77	109	127	114	101	121	83	77	133	133
high	145	176	0	0	170	0	151	145	0	0
number	10	11	12	13	14	15	16	17	18	19
low	52	59	46	28	45	21	27	24	40	14
medium	120	127	114	96	114	90	96	91	108	83
high	0	0	0	164	0	158	165	158	176	151
number	20	21	22	23	24	25	26	27	28	29
low	14	27	58	40	65	64	40	20	59	52
medium	83	95	126	108	133	133	108	89	127	120
high	151	164	0	177	0	0	177	158	0	0
number	30	31	32	33	34	35	36	37	38	39
low	43	47	8	23	2	65	15	52	65	46
medium	111	115	77	91	71	133	84	120	133	114
high	177	0	146	159	139	0	152	0	0	0
number	40	41	42	43	44	45	46	47	48	49
low	34	40	2	2	53	8	65	2	53	59
medium	102	108	71	71	121	78	133	71	121	127
high	170	177	139	139	0	146	0	139	0	0
number	50	51	52	53	54	55	56	57	58	59
low	40	46	15	7	60	53	10	59	26	53
medium	108	114	84	77	128	121	78	127	83	121
high	177	0	152	145	0	0	146	0	140	0
number	60	61	62	63	64	65	66	67	68	69
low	23	14	40	14	52	59	28	14	8	15
medium	91	83	108	83	120	127	95	84	78	84
high	158	151	177	151	0	0	164	152	146	152
number	70	71	72	73	74	75	76	77	78	79

low	8	20	14	46	64	52	40	1	16	14
medium	77	89	83	114	132	120	108	70	85	83
high	146	157	151	0	0	0	177	138	153	151
numbe										
r	**80**	**81**	**82**	**83**	**84**	**85**	**86**	**87**	**88**	**89**
low	20	65	46	20	1	22	41	40	65	35
medium	89	133	114	90	71	90	109	108	133	103
high	157	0	0	158	139	158	177	176	0	170
numbe										
r	**90**	**91**	**92**	**93**	**94**	**95**	**96**	**97**	**98**	**99**
low	1	59	52	7	2	60	15	2	52	14
number	71	127	120	76	71	128	84	71	120	83
low	139	0	0	145	139	0	152	139	0	151

可观察出，上表格中的数据具有分段集中性。

（5）.按文字高度匹配分组

考察选取出的 11 个 L_k 所对应的 $\vec{H}_k$，数据如下表。

numbe										
r	**7**	**14**	**29**	**38**	**49**	**61**	**71**	**89**	**94**	**125**
low	8	45	52	65	59	14	20	35	2	28
mediun	77	114	120	133	127	83	89	103	71	96
high	145	0	0	0	0	151	157	170	139	164

容易观察到，11 个 L_k 所对应的 $\vec{H}_k$ 的区别度明显，考虑将碎片按文字高度分成 11 个组。

下面利用 $\vec{H}_j$ 作为特征描述按文字高度匹配将碎片分组。

k 的可取值在（2）边缘碎片优先搜寻法中已经找出，分别有 7、14、29、38、49、61、71、89、94、125、168. j 的可取值为 0 到 208 中除去上述数值的整数。

对于任意合理的 j,计算

$$\| \vec{H}_j - \vec{H}_k \| (k = 7、14、29、38、49、61、71、89、94、125、168)$$

满足 $\min \| \vec{H}_j - \vec{H}_k \| (k = 7、14、29、38、49、61、71、89、94、125、168)$ 的 $\vec{H}_k$ 与 $\vec{H}_j$ 向量距离最小,即可把第 j 个碎片分到第 k 个碎片所属的组中，X_j 也可相应记为 G_{kj}。若 X_j 还是文件右侧边缘碎片的矩阵，则 X_j 还可以相应记成 G_{kr}

用 MATLAB 计算分类，效果理想，正好可以将所有 209 个碎片分成 11 组。

（6）.局部拼接法

在每个 L_k 为代表的组内进行拼接。

L_k,G_{kj1},G_{kj2},.....,G_{kj17},G_{kp} 为组内的所有碎片矩阵。

应用 4.1（3）的边缘匹配法求得 G_{kjn} 右侧的配对值 G_{kjm}，再对 G_{kjm} 用左侧的边缘匹配法验证 G_{kjn} 是否为其配对值。若是则匹配继续向右进行，若不是，则对本组剩余的 G_{kjq} 左侧进行边缘匹配，若所有剩余的 G_{kjq} 与 G_{kjn} 不匹配，则默认 G_{kjn} 右侧的配对值依然是 G_{kjm}，否则既中段右侧边缘匹配，从 G_{kr} 开始左侧边缘匹配。若左右两侧匹配在 G_{kjn} 点重合前又中断，则需要人工干预。

在用 MATLAB 实际求解中，我们的确遇到了组内局部拼接出现两个断点需要人工干预的情况，两个断点分别是第 46 和第 8 号碎片。我们人工干预的手段是事先输入正确的匹配值，令拼接进行到第 46 号或第 8 号碎片时，直接匹配出正确的碎片。干预代码在 ad.m 的第 80—86 行。

（7）.纵向匹配法

将上一步骤横向拼接好的每一行记为 Gk,其中 k 为该行左侧碎片的题设编号。

利用 $\overset{\text{ucw}}{H}_k$ 的第一个分量和最后一个分量,求出第 k 个碎片第一行文字高度中心到碎片上边缘的距离 L_k 和最后一行文字高度中心到碎片下边缘的距离 P_k。若 $\min|\mathrm{P}_k+L_{k0}-(h+d)|$ 在 k=ki 时取极值,则 Gki 为 Gk0 的相邻上一行碎片。运用循环依次求出相邻上一行，共循环 18 次。其中 Gk0 为通过 4.1（2）的方法求出的原文件最后一行碎片。

用 MATLAB 通过此方法拼接，可以拼接成功。

4.2.2 模型求解

通过在附件三中运行程序 ad,得到如下结果（复原拼图见附录一）：

049	054	065	143	186	002	057	192	178	118	190	095	011	022	129	028	091	188	141
061	019	078	067	069	099	162	096	131	079	063	116	163	072	006	177	020	052	036
168	100	076	062	142	030	041	023	147	191	050	179	120	086	195	026	001	087	018
038	148	046	161	024	035	081	139	122	103	130	193	088	157	025	008	009	105	074
014	128	003	159	082	199	035	012	073	160	203	169	134	039	031	051	107	115	176
094	034	084	183	090	047	121	042	124	144	077	112	149	097	136	164	127	058	043
125	013	182	109	197	016	184	110	187	066	106	150	021	173	157	181	234	139	145
029	064	111	192	005	092	180	048	037	075	055	044	206	010	104	098	178	171	059
007	200	138	158	126	068	175	045	174	000	137	053	056	093	153	070	166	032	196
071	156	083	132	200	017	098	033	202	198	015	133	170	205	085	152	165	027	060
089	146	102	154	114	040	151	207	155	144	185	108	117	004	101	113	194	179	123

表三

4.3 针对附件 4 的改进拼接模型 II

4.3.1 模型建立

此模型构建与 4.2 中大致相同，因此这里只对要改进的地方进行描述。

由于英文字母不像中文汉字具有等高的性质，因此根据不同的字母组合情形判断其中心位置时需要分多种情况讨论，并做不同的统计处理。根据碎片中不同高度字母组合，大致将碎片分为 11 类。英文 26 个字母中，有高度较低的字母如 a, e, o;有相对于此类字母向上延伸一倍的字母，如 h, b；也有相对于第一种字母向下延伸一倍的字母，如 p, q；还有只比第一类高出一点的特例，如 t。

舍去与上下边缘相交的字母，当余下字母数为 1 行，有如下三种情况：

“a 型”

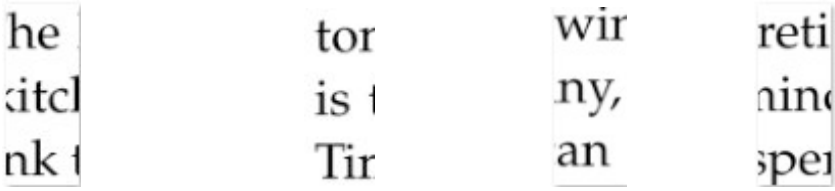
“bf 型”

“gh”型

当余下字母数为 2 行，有如下四种情况：

“oh”型　　“ee 型”　　“bg 型”　　“ta 型”

当余下字母数为 3 行，有如下四种情况：



“hn 型”　　“tn 型”　　“aa 型”　　“tp 型”

根据这 11 种情况运用不同算法求出中心位置，再进行分类。但由于有些不是一组的碎片具有类似的高度特征值，最终将碎片分为长度为 38 的五组，长度为 19 的一组。

同时由于英文边缘信息量较少，单一匹配成功率低.因此可以设定阈值，结合人工干预进行筛选匹配。

4.3.2 模型求解

通过在附件三中运行程序 ad_en,并进行人工干预得到如下结果（复原拼图见附录一）：

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

081	077	128	200	131	052	125	140	193	087	089	048	072	012	177	124	000	102	115

4.4 拼接模型 III

4.4.1 模型建立

拼接模型 III 与 4.2 中的模型基本相同，只是增加了用于约束的验证条件，任意两片碎片在匹配时，只有其反面也同时匹配才可通过验证。

首先需要找到最左侧和最右侧的碎片，通过观察发现，有一侧边缘值均为 255 且行数为碎片高度的列向量最多为 17，其反面为另一面的另一侧边缘碎片，其该列向量数最多也为 17；然而，相应的另一侧“白色列向量”数最多为 12，反面亦然，可以据此分别筛选两侧的起始碎片。

其余的步骤基本与 4.2 中相同，需要添加碎片反面是否匹配的验证条件，在同时匹配出多个结果时进行人工干预。

4.4.2 模型求解

通过在附件五中运行程序 ad5,在同时匹配出多个结果时进行人工干预，得到如下结果（复原拼图见附录一）：

07 8b	11 1b	12 5a	14 0a	15 5a	15 0a	18 3b	17 4b	11 0a	06 6a	10 8a	01 8b	02 9a	18 9b	08 1b	16 4b	02 0a	04 7a	136 b
08 9a	01 0b	03 6a	07 6b	17 8a	04 4a	02 5b	19 2a	12 4b	02 2a	12 0b	14 4a	07 9a	01 4a	05 9a	06 0b	14 7a	15 2a	005 a
18 6b	15 3a	08 4b	04 2b	03 0a	03 8a	12 1a	09 8a	09 4b	06 1b	13 7b	04 5a	13 8a	05 6b	13 1b	18 7b	08 6b	20 0b	143 b
19 9b	01 1b	16 1a	16 9b	19 4b	17 3b	20 6b	15 6a	03 4a	18 1b	19 8b	08 7a	13 2b	09 3a	07 2b	17 5a	09 7a	03 9b	083 a
08 8b	10 7a	14 9b	18 0a	03 7b	19 1a	06 5b	11 5b	16 6b	00 1b	15 1b	17 0b	04 1a	07 0b	12 9b	00 2a	16 2b	20 3b	090 a
11 4a	18 4b	17 9b	11 6b	20 7a	05 8a	15 8a	19 7a	15 4b	02 8b	01 2a	01 7b	10 2b	06 4b	20 8a	14 2a	05 7a	02 4a	013 a
14 6a	17 1b	03 1a	20 1a	05 0a	19 0b	09 2b	01 9b	01 6b	17 7b	05 3b	20 2a	02 1b	13 0a	16 3a	19 3b	07 3b	15 9a	035 a
16 5b	19 5a	12 8a	15 7a	16 8a	04 6a	06 7a	06 3b	07 5b	16 7a	11 7b	00 8b	06 8b	18 8a	12 7a	04 0a	18 2b	12 2a	172 a

00 3b	00 7b	08 5b	14 8b	07 7a	00 4a	06 9a	03 2a	07 4b	12 6b	17 6a	18 5a	00 0b	08 0b	02 7a	13 5b	14 1a	20 4b	105 a
02 3b	13 3a	04 8a	05 1b	09 5a	16 0b	11 9a	03 3b	07 1b	05 2a	06 2a	12 9b	11 8b	10 1a	01 5b	20 5a	08 2b	14 5a	009 b
09 9a	04 3a	09 6b	10 9a	12 3a	00 6a	10 4a	13 4a	11 3a	02 6b	04 9b	09 1a	10 6b	10 0b	05 5b	10 3a	11 2a	19 6b	054 b

这是其中一面的拼接情况，另一面的拼接表相当于将数字部分左右对称，字母部分“a”，“b”互换。

5.模型的结果分析与讨论

对于模型一，就附件 1 和附件 2 的题目难度要求下，模型一算法简单且能保证正确率。但就附件 1 附件 2 的图片数目而言、模型一运算量大耗时长，有极大的优化空间。并且由于我们只以边缘灰度相关匹配作为唯一的判断标准，当碎片边缘被截笔画较少时，其匹配的误差将难以控制。另外，没有其他判断标准保证符合向量距离极小值的碎片一定是正确的匹配。所以本题解法仅适用于碎片边缘信息丰富、特征明显、搜寻对象群体小的拼接问题。

对于模型二，应用到了局部拼接模型、数据统计模型和合理优化模型，能基本顺利解决中文文章碎片的附件 3 问题，且经过文字高度匹配分组后，局部拼接大幅度节省了运行时间，但是拼接过程依旧存在断点、需要人工干预。因为比赛时间限制，我们只进行了文字高度匹配。设想如果能对文字的中心进行定位，在文字高度匹配分组后，匹配文字中心的间距，可以提高正确率并且较大程度减少人工干预的可能性。

对于附件四，因为英文字母不如中文汉字笔画复杂、字形端正，边缘匹配法的正确率极低，文字高度中心的确定也相当复杂，需在模型二的基础上进行优化。英文字母的中心位置有偏上、正中、偏下三种情况需考虑，根据这三类字母和碎片中完整字母行的个数的区别，我们需要将所有碎片分为 11 种情况进行讨论，这 11 种情况运用不同算法求出中心位置，再进行分类。运算过程相当复杂，且分组效果不甚理想。由于有些不是一组的碎片具有类似的高度特征值，最终将碎片分为长度为 38 的五组，长度为 19 的一组。使得组内局部拼接难以顺利进行，需要多次人工干预。

对于附件五，在模型二的基础上添加反面匹配验证条件，在所有匹配过程中都加以制约，一定程度上避免了附件四中 38 个碎片一组的情况。局部拼接时断点数目影响不大，通过少数人工干预即可实现拼接复原。

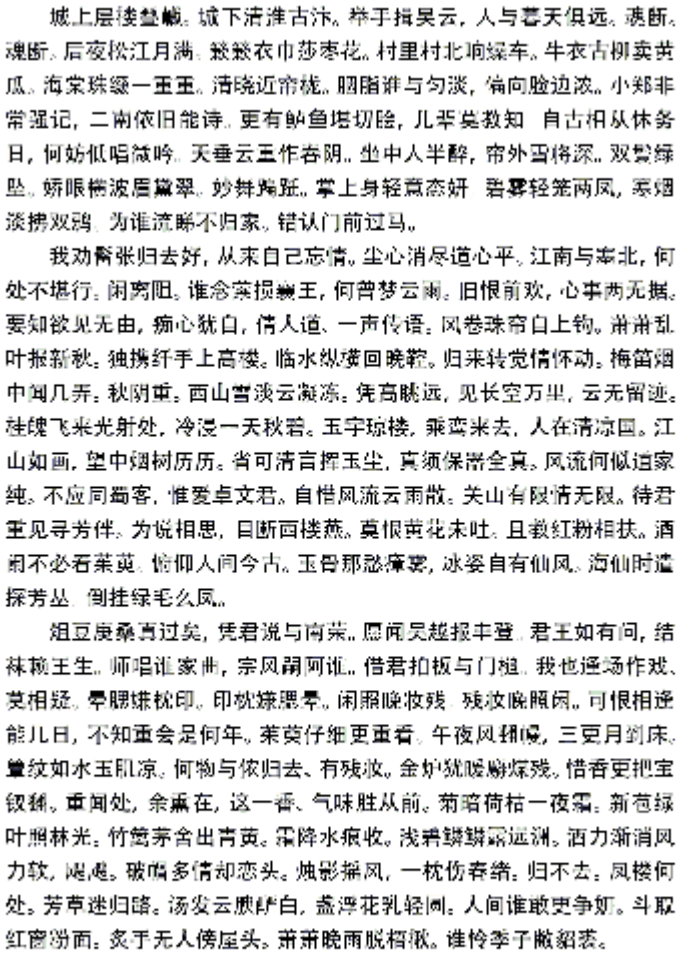
参考文献：

[1] 姜启源，谢金星，叶俊，数学模型，北京：高等教育出版社，2006

[2] 赵东方，数学模型与计算，北京：科学出版社，2007

附录一：

附件 1 复原图：



城上层楼叠巘。城下清淮古沛。举手揖吴云，人与暮天俱远。魂断。魂断。后夜松江月满。簌簌衣巾莎枣花。村里村北响绿车。牛衣古柳卖黄瓜。海棠珠缀一重重。清晓近帘栊。胭脂谁与匀淡，偏向脸边浓。小郑非常强记，二南依旧能诗。更有鲈鱼堪切脍，儿辈莫教知　自古相从休务日，何妨低唱微吟。天垂云三作春阴。坐中人半醉，帘外雪将深。双鬟绿坠。娇眼横波眉黛翠。妙舞翔蹁。掌上身轻意态妍　碧霄轻笼两凤，彩烟淡拂双鸦　为谁流睇不归家。错认门前过马。

我劝髯张归去好，从来自己忘情。尘心消尽道心平。江南与塞北，何处不堪行。闲离阻。谁念菜损巍王，何曾梦云阆。旧恨前欢，心事两无据。要知欲见无由，痴心犹自，倩人道、一声传语。风卷珠帘白上钩。萧萧乱叶报新秋。独携纤手上高楼。临水纵横回晚辔。归来转觉情怀动。梅笛烟中闻几弄。秋阴重。画山雪浅云凝冻。凭高眺远，见长空万里，云无留迹。桂魄飞来光射处，冷浸一天秋碧。玉宇琼楼，乘鸾来去，人在清凉国。江山如画，望中烟树历历。省可清言挥玉尘，真须保啬全真。风流何似逍家纯。不应同蜀客，惟爱卓文君。自惜风流云雨散。关山有限情无限。待君重见寻芳伴。为说相思，目断回楼燕。莫恨黄花未吐。且教红粉相扶。酒阑不必看茱萸。愉仰人间今古。玉骨那愁瘴雾，冰姿自有仙风。海仙时遣探芳丛　倒挂绿毛么凤。

组豆庶桑真过矣，凭君说与南荣。愿闻灵越报丰登　君王如有问，结袜挑王生。师唱谁家曲，宗风嗣阿谁。借君拍板与门生。我也逢场作戏、莫相疑。攀腮嫌枕印。印枕嫌腮攀。闲照晚妆残　残妆晚照闲。可恨相逢能几日，不知重会是何年。茱萸仔细更重看　午夜风翻幔，三更月到床。鲛纹如水玉肌凉。何物与侬归去、有残妆。金炉犹暖麝煤残。惜香更把宝钗翻。重闻处，余熏在，这一番、气味胜从前。菊暗荷枯一夜霜。新苞绿叶照林光。竹篱茅舍出青黄。霜降水痕收。浅碧鳞鳞露远洲。酒力渐消风力软，飒飒。破晴多情却恋头。烛影摇风，一枕伤春绪。归不去。凤楼何处。芳草迷归路。汤发云腴庐白，盏浮花乳轻圆。人间谁敢更争妍。斗取红窗粉面。炙于无人傍屋头。萧萧晚雨脱褐歌。谁怜季子敝貂裘。

附件二复原图：

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-

附件 3 复原图：

便邮。温香熟美。醉懒云鬟垂两耳。多谢春工。不是花红是玉红。一颗樱桃突素口。不爱黄金，只爱人长久。学画鸦儿犹未就。眉尖已作伤春皱。清泪斑斑，挥断柔肠寸。啼人问。背灯偷揾拭尽残妆粉　春事阑珊芳草歇。客里风光，又过清明节。小院黄昏人忆别。落红处处闻啼鴂。岁云暮，须早计，要褐裘。故乡归去千里，佳处辄迟留　我醉歌时君和，醉倒须君扶我，惟酒可忘忧。一任刘玄德，相对卧高楼　记取西湖西畔，正暮山好处，空翠烟霏。算诗人相得，如我与君稀。约仲年、东还海道，愿谢公、雅志莫相违。西州路，不应回首，为我沾衣。料峭春风吹酒醒，微冷。山头斜照却相迎。回首向来萧瑟处。归去。也无风雨也无晴。旅陌寻春去，红尘拂面来。无人不道看花回。惟见石榴新蕊、一枝开。

九十日春都过了，贪忙何处追游。三分春色一分愁。雨翻梅叶阵，风转柳花球。白雪清词出坐间。爱君才调两俱全。异乡风景却依然。团扇只堪题往事，新丝那解系行人。酒阑滋味似残春。

缺月向人舒窈窕。三星当户照绸缪。香生浅黛见纤柔。搔首颦归软。自觉功名懒更疏。若问使君才与术，何如。占得人间一味愚。海东头，山尽处。自古空槎来去。槎有信，赴秋期。使君行不归。别酒劝君君一醉。清润潘郎，又是何郎婿。记取钗头新利市。莫将分付东邻子。画塞山边白鸟飞。散花洲外片帆微。挑花流水鳜鱼肥　主人顺小　欲向东风先挽倒。已属君家　且更从容等待他。愿我已无当世望，似君须向古人求　岁寒松柏肯惊秋

水涵空，山照市。西汉二疏乡里　新白发，旧黄金。故人恩义深。谁道东阳都瘦损，凝然点漆精神。瑶林终自隔风尘。试看披鹤氅，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。暖风不解留花住。片片著人无数。楼上望春归去。芳草迷归路。犀钱玉果。利市平分沾四坐。多谢无功。此事如何到得侬。元宵似是欢游好。何况公庭民讼少。万家游赏上春台，十里神仙迷海岛。

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。梁苑犹能陪俊少。莫惹闲愁。且沂

附件四复原图：

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

附件五复原图：

其中一面：

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. 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.

另一面：

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

附录二：

附件 1MATLAB 代码：

主程序ad1：

```
%initialize
array_ini = [];
for i = 0:18
    if i < 10
        name = strcat('00', int2str(i));
    elseif i < 100
        name = strcat('0', int2str(i));
    elseif i > 99
        name = int2str(i);
    end
    name = strcat('C:/Users/lenovo/Desktop/cumcm2013problems/B/附件
1/', name, '.bmp');
    im = imread(name, 'bmp');
    array_ini = [array_ini;im];
end
```

```
width = 72;
height = 1980;

array = int32(array_ini);
first = get_left_margin1(array,height);
columns = [first];
%a = array(29*height+1:30*height,width);
%match the edge

for i = 1:18
    a = array(columns(i)*height+1:(1+columns(i))*height,width);
    index = get_index_1(a, array, height) - 1;
    columns = [columns, index];
end
array_final = [];
for i = 1:19
    array_final                    =                    [array_final,
array_ini(columns(i)*height+1:(1+columns(i))*height,:)];
end
imshow(array_final);
%row

y=get_left_margin1(array,height);
disp(y);
```

函数：

```
function x=get_index_1(a, array, height)
    sums = zeros(1, 19, 'int32');
    for i = 0:18
        b = array(1+height*i:height*(i+1),1);
        c = (a-b) .* (a-b);
        sum = 0;
        for j = 1:height
            sum = sum + c(j);
        end
        sums(i+1) = sum;
    end
    minimum = min(sums);
    x = find(sums == minimum);
end
```

```
function y=get_left_margin1(array,height)
  margin=[];
  tmp = ones(height, 11, 'int32')*255;
  for i = 0:18
     a = array(i*height+1:(1+i)*height,1:11);
     b = (a - tmp);
     if sum(sum(b))==0
        c=[i];
        margin=[margin,c];
     end
  end
  y=margin;
end
```

附件 2MATLAB 代码：

主程序 ad_2：

```
%initialize
array_ini = [];
for i = 0:18
    if i < 10
        name = strcat('00', int2str(i));
    elseif i < 100
        name = strcat('0', int2str(i));
    elseif i > 99
        name = int2str(i);
    end
    name = strcat('C:/Users/lenovo/Desktop/cumcm2013problems/B/附件
2/', name, '.bmp');
    im = imread(name, 'bmp');
    array_ini = [array_ini;im];
end

width = 72;
height = 1980;

array = int32(array_ini);
columns = [get_left_margin2(array, height)];
%a = array(29*height+1:30*height,width);
%match the edge
c=get_left_margin2(array,height);
disp(c);
for i = 1:18
    a = array(columns(i)*height+1:(1+columns(i))*height,width);
```

```
    index = get_index_1(a, array, height) - 1;
    columns = [columns, index];
end
array_final = [];
for i = 1:19
    array_final                    =                    [array_final,
array_ini(columns(i)*height+1:(1+columns(i))*height,:)];
end
imshow(array_final);
%row
```

函数：

```
function x=get_index_1(a, array, height)
    sums = zeros(1, 19, 'int32');
    for i = 0:18
        b = array(1+height*i:height*(i+1),1);
        c = (a-b) .* (a-b);
        sum = 0;
        for j = 1:height
            sum = sum + c(j);
        end
        sums(i+1) = sum;
    end
    minimum = min(sums);
    x = find(sums == minimum);
end

function y=get_left_margin2(array,height)
  margin=[];
  tmp = ones(height, 10, 'int32')*255;
  for i = 0:18
     a = array(i*height+1:(1+i)*height,1:10);
     b = (a - tmp);
     if sum(sum(b))==0
        c=[i];
        margin=[margin,c];
     end
  end
  y=margin;
end
```

附件 3MATLAB 代码：

主程序 ad:

```
%initialize
array_ini = [];
for i = 0:208
    if i < 10
        name = strcat('00', int2str(i));
    elseif i < 100
        name = strcat('0', int2str(i));
    elseif i > 99
        name = int2str(i);
    end
    name = strcat('C:/Users/lenovo/Desktop/cumcm2013problems/B/附件
3/', name, '.bmp');
    im = imread(name, 'bmp');
    array_ini = [array_ini;im];
end

width = 72;
height = 180;
right = width;
left = 1;
array = int32(array_ini);

columns = get_left_margin(array, height);
presence = get_left_margin(array, height);
left_edge = get_left_margin(array, height);
%a = array(8*height+1:9*height,right);

% get the center of each row
center = [];
for i = 0:208
    range = row_interval(array(i*height+1:(1+i)*height,:));
    center = [center; get_center(range)];
end

elements = [];
except_ele = [];

% get all elements of each row
for i = 1:11
    [x, y] = match_center(center, center(columns(i)+1,:), presence,
columns(i));
    if (length(x)==19)
```

```
        elements = [elements; x];
    else
        except_ele = [x];
    end
    presence = y;
end
for i = 0:208
    if (~ismember(i, presence))
        except = i;
        break
    end
end
except_ele = [except_ele, i];
elements = [elements; except_ele];

% set the right edge

right_edge = get_right_margin(array, height);
for i = 1:11
    right_position = int32(find(right_edge(i) == elements));
    result = right_position / 11;
    if (result*11<right_position)
        col_index = result + 1;
        row_index = right_position - 11*result;
    else
        col_index = result;
        row_index = right_position - 11*(result - 1);
    end
    tmp = elements(row_index, 19);
    elements(row_index, 19) = elements(row_index, col_index);
    elements(row_index, col_index) = tmp;
end

% match each row

final_answer = cell(1, 11);
for j = 1:11
    row = elements(j,:);
    final_row = [row(1)];
    for i = 1:18
        % human interfere
        if (final_row(i) == 46)
            final_row = [final_row, 161];
            continue
```

```
        elseif final_row(i) == 8
            final_row = [final_row, 9];
            continue
        end

        a                                                                    =
array(final_row(i)*height+1:(1+final_row(i))*height,right);
        index = get_row_index(a, array, height, final_row, left, row)
- 1;
        a = array(index*height+1:(index+1)*height,left);
        tmp_row = final_row(1:length(final_row)-1);
        index_2 = get_row_index(a, array, height, tmp_row, right, row)
- 1;
        if index_2 == final_row(i);
            final_row = [final_row, index];
        else
            tp = 0;
            for k = 1:19
                if (ismember(row(k), [final_row,index]))
                    continue
                else
                    a = array(row(k)*height+1:(row(k)+1)*height,left);

                    index_3   =   get_row_index(a,   array,   height,
[final_row(1:length(final_row)-1),index], right, row) - 1;
                    if index_3 == final_row(i)
                        tp = 1;
                        break
                    end
                end
            end
            if (tp ~= 0)
                break
            else
                final_row = [final_row, index];
            end
        end
    end
    final_answer{j} = final_row;
    %disp(final_row);
end

final_answer_right = cell(1, 11);
for j = 1:11
```

```
    row = elements(j,:);
    final_row = [row(19)];

    for i = 1:(19-length(final_answer{j})-1)
        a                                                                    =
array(final_row(1)*height+1:(1+final_row(1))*height,left);
        index      =      get_row_index(a,      array,      height,
[final_answer{j},final_row], right, row) - 1;
        a = array(index*height+1:(index+1)*height,right);
        tmp_row = final_row(2:length(final_row));
        index_2      =      get_row_index(a,      array,      height,
[final_answer{j},tmp_row], left, row) - 1;
        if index_2 == final_row(1);
            final_row = [ index,final_row];
        else
            tp = 0;
            for k = 1:19
                if (ismember(row(k), [final_row,index]))
                    continue
                else
                    a                                                        =
array(row(k)*height+1:(row(k)+1)*height,right);
                    index_3    =    get_row_index(a,    array,    height,
[final_row(2:length(final_row)),index, final_answer{j}], left, row)
- 1;
                    if index_3 == final_row(1)
                        tp = 1;
                        break
                    end
                end
            end
            if (tp ~= 0)
                break
            else
                final_row = [index,final_row];
            end
        end
    end
    final_answer_right{j} = final_row;
    %disp(final_row);
end

final_answers = cell(1, 11);
for i = 1:11
```

```
    sumation          =          length(final_answer{i})          +
length(final_answer_right{i});
    final_answers{i} = [final_answer{i}, final_answer_right{i}];
    if sumation >= 20
        final_answers{i} = [final_answer{i}(1:19)];
    end
end

% get the first row(by manual)
first = 49;
tmp = 0;
for i = 1:11
    tmp = ismember(first, final_answer{i});
    if (tmp == 1)
        poi = i;
        break
    end
end
sequence = [];
suquence = int32(sequence);
center_sequence = [];
for i = 1:11
    center_sequence               =               [center_sequence;
center(final_answers{i}(1)+1, :)];
end
sequence = [sequence, poi];

% match as columns(get the order of rows)
for i = 1:10
    if center_sequence(sequence(i),3) == 0
        low = center_sequence(sequence(i),2);
    else
        low = center_sequence(sequence(i),3);
    end
    for j = 1:11
        if (~ismember(j, sequence))
            high = center_sequence(j,1) + 180;
            if abs(high-low-68) < 3
                sequence = [sequence, j];
                break
            end
        end
    end
end
```

```
picture = [];
% construct the picture with right order
for i = 1:11
    picture = [picture; final_answers{sequence(i)}];
end

% joint the whole picture
array_finals = [];
for j = 1:11
    array_final = [];
    for i = 1:19
        array_final = [array_final,
array_ini(picture(j,i)*height+1:(1+picture(j,i))*height,:)];
    end
    array_finals = [array_finals; array_final];
end
imshow(array_finals);
```

函数：

```
function x = get_center(range)
    up = int32(range(1));
    down = int32(range(2));
    difference = down - up;
    center = [];
    if difference < 50
        center = [center, (down + up)/2];
        while up >= 50
            center = [up - 49, center];
            up = up - 69;
        end
        while down <= 131
            center = [center, down + 49];
            down = down + 69;
        end
    elseif difference > 100 && difference < 140
        mean = (up + down)/2;
        center = [center, mean - 34, mean + 34];
        if up >= 50
            center = [up - 49, center];
        elseif down <= 131
            center = [center, down + 49];
        end
    else
```

```
        center = [center, up + 19, (up + down)/2, down - 19];
    end
    if length(center) < 3
        center = [center, 0];
    end
    x = center;
end

function x=get_index_1(a, array, height)
    sums = zeros(1, 19, 'int32');
    for i = 0:18
        b = array(1+height*i:height*(i+1),1);
        c = (a-b) .* (a-b);
        sum = 0;
        for j = 1:height
            sum = sum + c(j);
        end
        sums(i+1) = sum;
    end
    minimum = min(sums);
    x = find(sums == minimum);
end

function y=get_left_margin(array,height)
  margin=[];
  tmp = ones(height, 11, 'int32')*255;
  for i = 0:208
     a = array(i*height+1:(1+i)*height,1:11);
     b = (a - tmp);
     if sum(sum(b))==0
        c=[i];
        margin=[margin,c];
     end
  end
  y=margin;
end

function y=get_right_margin(array,height)
  margin=[];
  tmp = ones(height, 11, 'int32')*255;
  for i = 0:208
     a = array(i*height+1:(1+i)*height,62:72);
     b = (a - tmp);
     if sum(sum(b))==0
```

```
        c=[i];
        margin=[margin,c];
     end
  end
  y=margin;
end

function x=get_row_index(a, array, height, columns, left, row)
    sums = zeros(1, 19, 'int32');
    for i = 1:19
        k = row(i);
        if ismember(k, columns)
            sums(i) = 100000000000;
        else
            b = array(1+height*k:height*(k+1),left);
            c = (a-b) .* (a-b);
            sumk = sum(c);
            sums(i) = sumk;
        end
    end
    minimum = min(sums);
    x = row(find(sums == minimum))+1;
end


function [x,y] = match_center(center, aim, column, ini)
    answer = [ini];
    for i = 1:209
        if ismember(i-1, column)
            continue
        end
        tmp = (aim - center(i, :)) .* (aim - center(i, :));
        sums = sum(tmp);
        if sums < 28
            answer = [answer, i-1];
            column = [column, i-1];
        end
    end
    x = answer;
    y = column;
end
```

```
function x=row_interval(image)
    index = zeros(1, 180, 'int8');
    for i = 1:180
        if min(image(i, :)) <230
            index(i) = 1;
        end
    end

    position = find(index==1);
    len = length(position);
    if (position(1) ~= 1)
        start = position(1);
    else
        tmp = position(1);
        i = 2;
        while 1
            tmp = tmp + 1;
            if (position(i) ~= tmp)
                start = position(i);
                break
            else
                i = i+1;
            end
        end
    end

    if (position(len) ~= 180)
        ends = position(len);
    else
        tmp = position(len);
        i = len - 1;
        while 1
            tmp = tmp - 1;
            if (position(i) ~= tmp)
                ends = position(i);
                break
            else
                i = i - 1;
            end
        end
    end
    %disp(position);
    x = [start,ends];
```

```
end
```

附录 4MATLAB 代码:

主程序 ad_en:

```
%initialize
array_ini = [];
for i = 0:208
    if i < 10
        name = strcat('00', int2str(i));
    elseif i < 100
        name = strcat('0', int2str(i));
    elseif i > 99
        name = int2str(i);
    end
    name = strcat('C:/Users/lenovo/Desktop/cumcm2013problems/B/附件
4/', name, '.bmp');
    im = imread(name, 'bmp');
    array_ini = [array_ini;im];
end

width = 72;
height = 180;
right = width;
left = 1;
array = int32(array_ini);
columns = get_right_margin(array, height);
presence = get_right_margin(array, height);
left_edge = get_left_margin(array, height);


% get the center of each row
center = [];
index_poi = [];
for i = 0:208
    [range,                        index_tmp]                        =
row_interval_en(array(i*height+1:(1+i)*height,:));
    index_poi = [index_poi;index_tmp];
    if i == 92
        center = [center; [22,86,150]];
    elseif i == 136
        center = [center; [53,117,0]];
    else
        center        =        [center;        get_center_en(range,
array(i*height+1:(1+i)*height,:))];
    end
```

```
end


pp = cell(1,11);
presences = [];

% get all similar rows
presences = [];
for i = 1:11
    for j = 0:208
        if (~ismember(j, presences))
            presences = [presences,j];
            break
        end
    end
    [x, y] = match_center(center, center(j+1,:), presences, j);
    pp{i} = x;
    presences = y;
    if length(presences) == 209
        break
    end
end

% human interfere
pp{4} = [pp{4},pp{5}];
pp{5} = pp{6};
pp{6} = pp{7};

for i = 1:6
    disp(pp{i});
end
% now the pp includes 5 vectors with 38 elements each and 1 vector
with 19 elements
```

附件 5MATLAB 代码：

主程序 ad_test:

```
%initialize
array_ini = [];
for i = 0:208
    if i < 10
        name = strcat('00', int2str(i));
    elseif i < 100
        name = strcat('0', int2str(i));
```

```
    elseif i > 99
        name = int2str(i);
    end
    name = strcat('C:/Users/lenovo/Desktop/cumcm2013problems/B/附件
1/', name, 'a.bmp');
    im = imread(name, 'bmp');
    array_ini = [array_ini;im];
end

array_ini2 = [];
for i = 0:208
    if i < 10
        name = strcat('00', int2str(i));
    elseif i < 100
        name = strcat('0', int2str(i));
    elseif i > 99
        name = int2str(i);
    end
    name = strcat('/Users/Thamos/Desktop/math_model/B/ap5/', name,
'b.bmp');
    im = imread(name, 'bmp');
    array_ini2 = [array_ini2;im];
end

width = 72;
height = 180;
right = width;
left = 1;
array = int32(array_ini);
array2 = int32(array_ini2);
columns = [7,14,29,38,49,61,71,89,94,125,168];
presence = [7,14,29,38,49,61,71,89,94,125,168];
left_edge = [7,14,29,38,49,61,71,89,94,125,168];
%a = array(8*height+1:9*height,right);

% get the center of each row
center1 = [];
ranges1 = [];
for i = 0:208
    range = row_interval(array(i*height+1:(1+i)*height,:));
    ranges1 = [ranges1;range];
    center1         =         [center1;         get_center_en(range,
array(i*height+1:(1+i)*height,:))];
end
```

```
center2 = [];
ranges2 = [];
for i = 0:208
    range = row_interval(array2(i*height+1:(1+i)*height,:));
    ranges2 = [ranges1;range];
    center2 = [center1; get_center_en(range,
array2(i*height+1:(1+i)*height,:))];
end
array3 = [array;array2];

% final_row = [47];
% a = array(final_row(1)*height+1:(1+final_row(1))*height,right);
% [index,page] = get_row_index_en(a, array3, height, final_row,
left);
% disp(index);
%
% final_row = [136];
% a = array2(final_row(1)*height+1:(1+final_row(1))*height,left);
% [index,page] = get_row_index_en(a, array3, height, final_row,
right);
% disp(index);

left_edges = [78,1; 89,0; 186,1; 199,1; 88,1;
114,0;146,0;165,1;3,1;23,1;99,0];
right_edges =
[136,1;5,0;143,1;83,0;90,0;13,0;35,0;172,0;105,0;9,1;54,1];
%match from left to right

old_row = [];
new_row = [];
number = 0;
for i = 1:11
    row = [];
    row = [row;left_edges(i,:)];
    for k = 1:18
        index_a = row(k,1);
        page_a = row(k,2);
        if page_a == 0
            a = array(index_a*height+1:(1+index_a)*height,right);
            [index,page] = get_row_index_en(a, array3, height,
[old_row;row], left);
        else
            a = array2(index_a*height+1:(1+index_a)*height,right);
```

```
            [index,page]   =   get_row_index_en(a,   array3,   height,
[old_row;row], left);
        end
        if (page == 0)
            a = array(index*height+1:(1+index)*height,left);
            [index2,page2]   =   get_row_index_en(a,   array3,   height,
[old_row;row(1:k-1,:)], right);
        else
            a = array2(index*height+1:(1+index)*height,left);
            [index2,page2]   =   get_row_index_en(a,   array3,   height,
[old_row;row(1:k-1,:)], right);
        end
        if index2 == index_a
            row = [row;[index,page]];
        else
            page_a = 1-page_a;
            if (page_a == 0)
                a = array(index_a*height+1:(1+index_a)*height,left);
                [index,page]  =  get_row_index_en(a,  array3,  height,
[old_row;row], right);
            else
                a = array2(index_a*height+1:(1+index_a)*height,left);
                [index,page]  =  get_row_index_en(a,  array3,  height,
[old_row;row], right);
            end
            if page == 0
                a = array(index*height+1:(1+index)*height,right);
                [index2,page2] = get_row_index_en(a, array3, height,
[old_row;row(1:k-1,:)], left);
            else
                a = array2(index*height+1:(1+index)*height,right);
                [index2,page2] = get_row_index_en(a, array3, height,
[old_row;row(1:k-1,:)], left);
            end
            if index2 == index_a
                row = [row;[index,1-page]];
            else
                break
            end
        end
    end
    new_row = [new_row; row];
end
% disp(new_row);
```

```
%match from right to right

old_row = new_row;
new_rows = [];
number = 0;
for i = 1:1
    row = [];
    row = [row;right_edges(i,:)];
    for k = 1:4
        index_a = row(1,1);
        page_a = row(1,2);
        if (page_a == 0)
            a = array(index_a*height+1:(1+index_a)*height,left);
            [index,page]   =   get_row_index_en(a,   array3,   height,
[old_row;row], right);
        else
            a = array2(index_a*height+1:(1+index_a)*height,left);
            [index,page]   =   get_row_index_en(a,   array3,   height,
[old_row;row], right);
        end
        if page == 0
            a = array(index*height+1:(1+index)*height,right);
            [index2,page2]  =  get_row_index_en(a,  array3,  height,
[old_row;row(1:k-1,:)], left);
        else
            a = array2(index*height+1:(1+index)*height,right);
            [index2,page2]  =  get_row_index_en(a,  array3,  height,
[old_row;row(1:k-1,:)], left);
        end

        if index2 == index_a
            row = [[index,page];row];
        else
            page_a = 1-page_a;
            if page_a == 0
                a = array(index_a*height+1:(1+index_a)*height,right);
                [index,page]  =  get_row_index_en(a,  array3,  height,
[old_row;row], left);
            else
                a                                                    =
array2(index_a*height+1:(1+index_a)*height,right);
                [index,page]  =  get_row_index_en(a,  array3,  height,
```

```
[old_row;row], left);
          end
          if (page == 0)
              a = array(index*height+1:(1+index)*height,left);
              [index2,page2] = get_row_index_en(a, array3, height,
[old_row;row(1:k-1,:)], right);
          else
              a = array2(index*height+1:(1+index)*height,left);
              [index2,page2] = get_row_index_en(a, array3, height,
[old_row;row(1:k-1,:)], right);
          end
          if index2 == index_a
              row = [[index,1-page];row];
          else
              break
          end
       end
   end
   disp(row);
   disp('------');
   new_rows = [new_rows; row];
end
disp(length(new_row));
```