Hindawi Publishing Corporation
Abstract and Applied Analysis
Volume 2014, Article ID 829602, 11 pages
http://dx.doi.org/10.1155/2014/829602

Research Article

A Solution to Reconstruct Cross-Cut Shredded Text Documents Based on Character Recognition and Genetic Algorithm

Hedong Xu, Jing Zheng, Ziwei Zhuang, and Suohai Fan

School of Information Science and Technology, Jinan University, Guangzhou 510632, China

Correspondence should be addressed to Suohai Fan; tfsh@jnu.edu.cn

Received 15 April 2014; Accepted 2 June 2014; Published 30 June 2014

Academic Editor: Fuding Xie

The reconstruction of destroyed paper documents is of more interest during the last years. This topic is relevant to the fields of forensics, investigative sciences, and archeology. Previous research and analysis on the reconstruction of cross-cut shredded text document (RCCSTD) are mainly based on the likelihood and the traditional heuristic algorithm. In this paper, a feature-matching algorithm based on the character recognition via establishing the database of the letters is presented, reconstructing the shredded document by row clustering, intrarow splicing, and interrow splicing. Row clustering is executed through the clustering algorithm according to the clustering vectors of the fragments. Intrarow splicing regarded as the travelling salesman problem is solved by the improved genetic algorithm. Finally, the document is reconstructed by the interrow splicing according to the line spacing and the proximity of the fragments. Computational experiments suggest that the presented algorithm is of high precision and efficiency, and that the algorithm may be useful for the different size of cross-cut shredded text document.

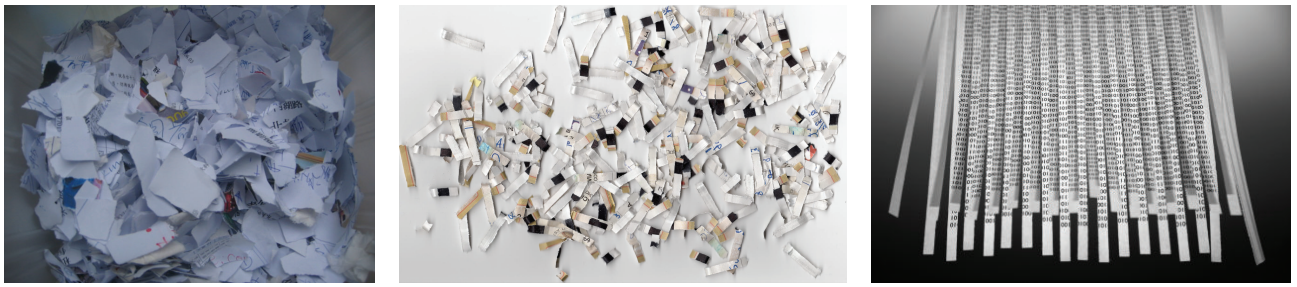
1. Introduction

The reconstruction technology of the shredded document is usually used for obtaining the judicial exhibit, repairing the relics and acquiring military intelligence. It plays an important role in judicial investment to repair the sensitive document damaged on purpose and archaeological research to recognize the cultural relics. Generally, the documents which are single sided or double sided are cut into pieces by hand or paper machine. Schauer et al. [1] thought the shredded document can be considered to be a variation from typical jigsaw puzzles. They define that there were three types of the fragments: the manually torn documents, the cross-cut shredded documents, and the strip shredded documents [1] (see Figure 1).

Recombining the document is tedious and time-consuming due to the tremendous number of the fragments and the missing information of the fragment. The efficient way is to reconstruct the shredded document by the automatic system. The approaches to reconstruct these two kinds of documents are different. The reconstruction of the paper with irregular boundaries shredded manually is based on the similarity of the boundary feature. Nevertheless, the fragments shredded by the paper shredder have smooth boundary, so the reconstruction method for irregular boundaries is not valid. For the smooth boundary, the key to reconstruct the document depends on the characters on the paper edges.

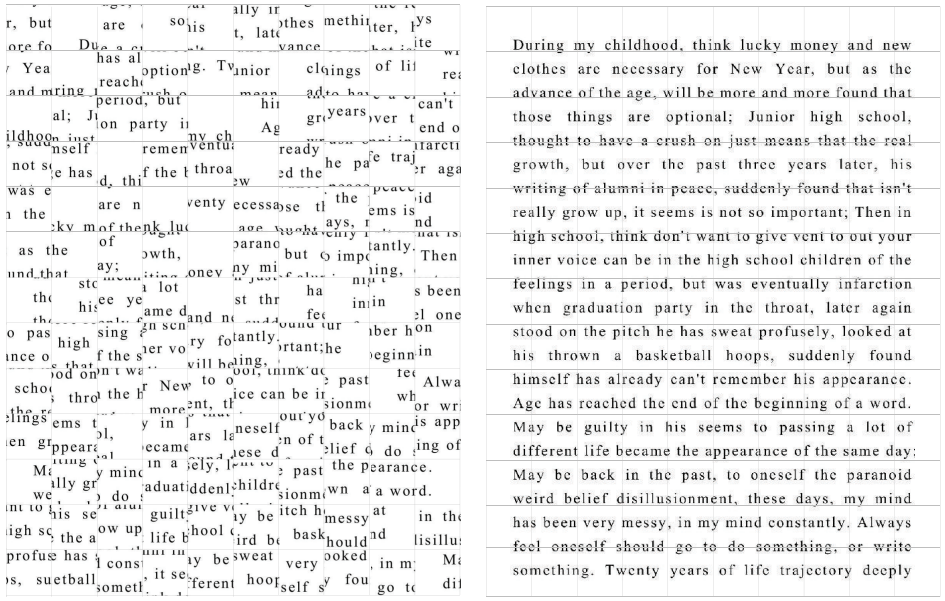
Most of the existing literatures investigate the reconstructing of the fragments shredded manually. Justino et al. [2] used a polyapproximation to simplify the complex contours of fragments and reconstruct the pieces of manually torn document by matching the feature of the polygon. Such method could only reconstruct the document on a small scale. Kesarkar et al. [3] joined the torn pieces of papers with the semiautomatic technique which is comparing edge length and angles, and Richter et al. [4] introduced an algorithmic framework for the automatic assembly of shredded documents based on shape- and content-based information.

Some literature focuses on the reconstruction of the strip shredded documents. Because the information in the boundary of the strip shredded documents is adequate, it is easy to calculate the correlation of two fragments and splice them if their proximity is high. Lin and Fan-Chiang



(a) The manually torn document (b) The cross-cut shredded document (c) The strip shredded document

Figure 1: The document shredded by hand (a) and by paper shredder ((b), (c)).



(a) (b)

Figure 2: The cross-cut shredded text document (a) and the fully reconstructed document (b).

[5] presented an algorithm based on image feature matching through the graph-based sorting scheme to reassemble the pieces of the shredded document. They merged the fragments with the average word length and the highest correlation of the binary codes. Also, some researchers made use of the character features to match two fragments. Perl et al. [6] proposed an optical character recognition (OCR) algorithm to match two fragments' contours through the probability histograms of the characters in the border region. If the lines of the text decrease, the precision of the paper reconstruction drops. Diem and Sablatnig [7] proposed an optical character recognition (OCR) to recognize the characters in the ancient manuscripts.

For the cross-cut shreds, the reconstruction of cross-cut shredded text documents (RCCSTD) problem proved to be NP-complete by Prandtstetter [8]. Biesinger et al. [9] investigated this problem with an improved genetic algorithm without using any pattern recognition technique. Schauer et al. [1] used a cost function to determine whether two shreds are adjacent according to the likelihood of the gray value of the pixels along the shreds edges through the memetic algorithm. The ant colony optimization and a variable neighborhood search were developed for RCCSTD by Prandtstetter and Raidl [10]. Sleit et al. [11] put up with a different approach for RCCSTD based on iteratively building clusters of shreds. A cross-cut shredded document is shown, for example, in Figure 2, including its correct reconstruction.

The reconstruction of cross-cut shredded text documents (RCCSTD) problem is defined as an injective map $\Pi : S \to \mathbb{D}^2$, where the set of the fragments $S = \{1, \ldots, m \times n\}$ belongs to one single-side printed document and $m \times n$ is the number of grid-shaped images. In this mapping, each fragment $s = (x, y) \in \mathbb{D}^2$, where $x \in \{1, \ldots, n\}$ and $y \in \{1, \ldots, m\}$, is assigned to one position in the two-dimensional (Euclidean) space $\mathbb{D}^2 = \{(x, y) \mid x = 1, \ldots, n; y = 1, \ldots, m\}$, such that the virtual fragments are represented by the white rectangles in Figure 3, and each virtual fragment is allowed to be used once [9, 12]. Furthermore, assume that the orientation of the

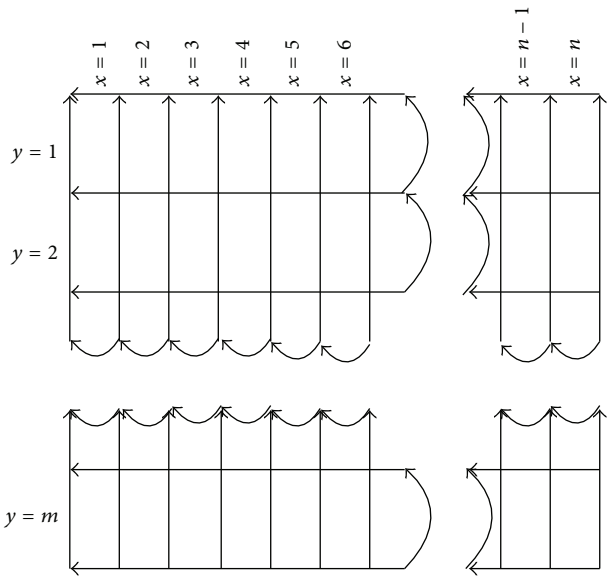


Figure 3: The explanation of RCCSTD problem [9, 12].

fragment is consistent and there is no character adhesion in the text.

The reconstruction of these fragments will be studied in this paper. First of all, statistic features of the English letters (capital and lowercase) including the letter size and the space between the letter and the ruled lines are measured, and we set a database of 52 letters (26 lowercase and 26 capital letters) of different fonts in the numeric way. The letter database is built for the character recognition, which can recognize the characters in the fragments and reconstruct the document in the row clustering and the intrarow splicing. Afterwards, since the ruled line of the characters in the common row is the same, cluster the fragments by the clustering algorithm with the help of the ruled line position determined by the identified characters in the fragments. The ruled lines in this paper are similar to the top lines and the based lines of the text [13]. Later, sort the fragments of each row concurrently. Splice some fragments according to identifiability of the characters on the joint of the boundaries at first. Then the combined fragments are regarded as the new fragments or vertices and their edges are the Euclidean distance between the border matrices of two new fragments. The TSP model and genetic algorithm are adopted to sort the fragments in the second stage. Finally, reconstruct the whole document using the fixed line spacing and the proximity of the fragment strings. Our methodology in detail is shown in Figure 4.

This paper is organized as follows. Section 2 introduces how to establish the letter database for the character recognition system. In Section 3, the procedure of character recognition is introduced. The algorithm of row clustering based on the database is proposed in Section 4 while the process of intrarow splicing is shown in Section 5. Section 6 wraps up the work by interrow splicing. The simulation is given in Section 7. Finally, in Section 8, we conclude this paper and present some ideas in the future work.

2. Letter Database

Initially, the database of the letters is needed. We obtain the gray image matrices of the 26 English capital letters and 26 lowercase letters. And then we get the binary gray image matrix of each letter through setting the threshold at 205, which means the pixels whose gray value is over the threshold are indicated by 0, and other pixels are indicated by 1. The letter database will be used later to recognize the characters of the fragments during the reconstruction.

2.1. Letter Extraction. Character recognition is the key to document reconstruction. Importantly, all the given fragments should be transferred into the binary image matrix the same as the process for the letter mentioned above.

We adopt the approach to extract the letter from Zhang et al. [14].

For a letter binary matrix $L = (l_{ij})_{p\times q}$, where $p \times q$ is the scale of the letter matrix. The leftmost side of the letter is $\min\{j \mid l_{ij} = 1, i = 1, \ldots, p; j = 1, \ldots, q\}$. The rightmost side of the letter is $\max\{j \mid l_{ij} = 1, i = 1, \ldots, p; j = 1, \ldots, q\}$. The top edge of the letter is $\min\{i \mid l_{ij} = 1, i = 1, \ldots, p; j = 1, \ldots, q\}$. The bottom edge of the letter is $\max\{i \mid l_{ij} = 1, i = 1, \ldots, p; j = 1, \ldots, q\}$ (see Figure 5).

2.2. Size Feature Extraction. We obtain the height and width of a letter, which is the so-called size feature of a letter, through the approach in Section 2.1. The space between the top edge and the bottom edge of a letter is defined as the height of a letter. The space between the leftmost side and the rightmost side of a letter is defined as the width of a letter. The size feature of letters is one crucial aspect of the letter database.

2.3. Ruled Line Space Extraction. Lu et al. [13] used the local maxima of the horizontal projection histogram in order to identify the top lines and the base lines of the text lines. Similarly, we locate the ruled lines using the horizontal projection of the binary image of the whole alphabet. If the ruled lines are found, the space between the letter and the overline or underline will be measured (shown as Figure 6). We set the overline as the uppermost line of the horizontal histogram and the underline as the bottom of the horizontal histogram. As for different letters, the space between the letter and the ruled lines (the overline and the underline) is the important feature of letters, playing the vital roles in the letter database. In other words, once certain letter in a fragment is recognizable, the exact ruled lines are found based on the space between the letter and the ruled lines (Figure 6).

3. Character Recognition

Initially, the character to be recognized is extracted in the rectangle as the method mentioned in Section 2.1. If the character is cut by the boundary line, before extracting, the fragment should be merged with another fragment (Figure 7).

After the character extraction, the character recognition begins. Character recognition consists of two steps: height-width ratio judgment and binary matrix judgment (Figure 8).

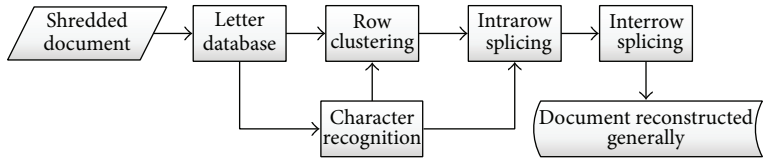


Figure 4: The flowchart of our methodology.

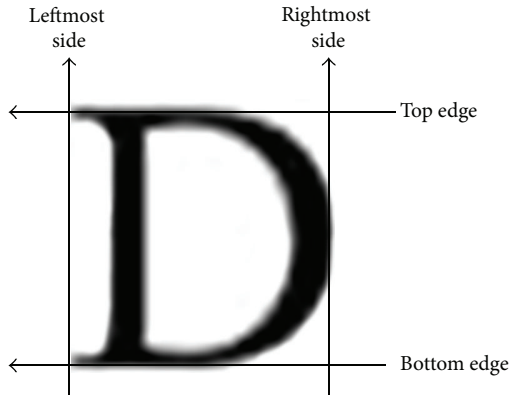


Figure 5: The approach to extract the certain letter.

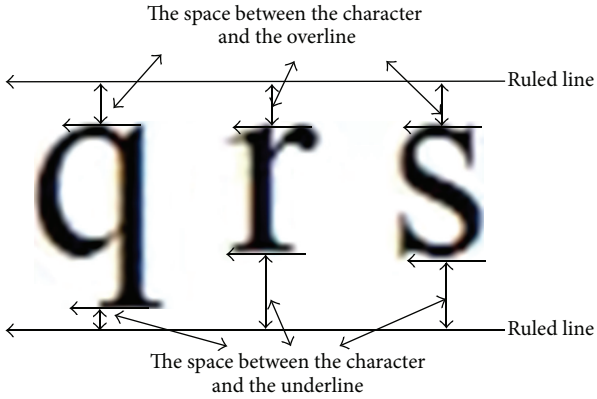


Figure 6: The space between the letter and the ruled lines.

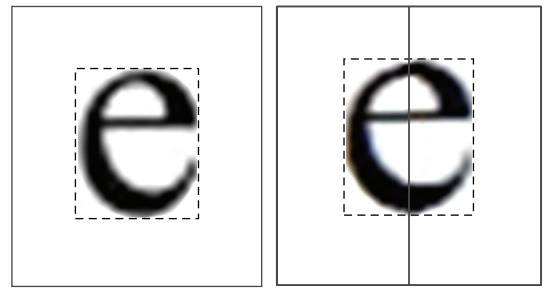


Figure 7: The character to be recognized is extracted in the rectangle.

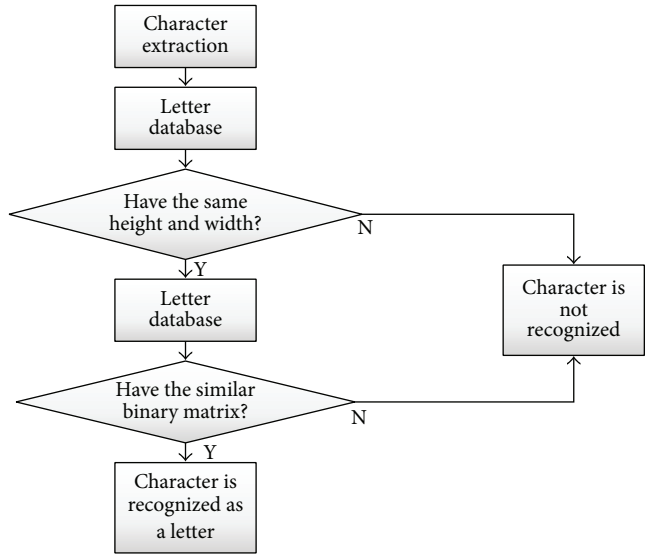


Figure 8: The flowchart of the character recognition.

A character is thought to be recognized as a letter if it has the same height-width ratio and the similar binary matrix to a certain letter in the letter database. In order to measure the similarity of the binary matrix, we set a threshold rate. That is to say, a character is recognizable if it shares the same height-width ratio with the certain letter in the database and its shape is over the threshold rate similar to this letter. Moreover, once a character is identified, the ruled lines of the line containing the identified character can be found according to the space between the letter and the ruled lines in the database.

4. Row Clustering

4.1. The Clustering Vector. We use the ruled line to cluster the fragments; however, the data of the gray image matrix is so enormous that we transform the data into the clustering vector to describe a fragment. The binary gray image matrix of each fragment can be described as a 4×1 clustering vector through feature extraction. The clustering vector is defined as $\mathrm{CV} = (a_1, a_2, a_3, a_4)^T$, where a_1 represents the lower position of the unidentified row on the top of the fragment and a_4 represents the upper position of the unidentified row at the bottommost of the fragment. Meanwhile, a_2 represents the position of the overline of the last identified row, and a_3 represents the position of the underline of the last identified row.

The steps of the feature extraction are as follows (see Figure 9).

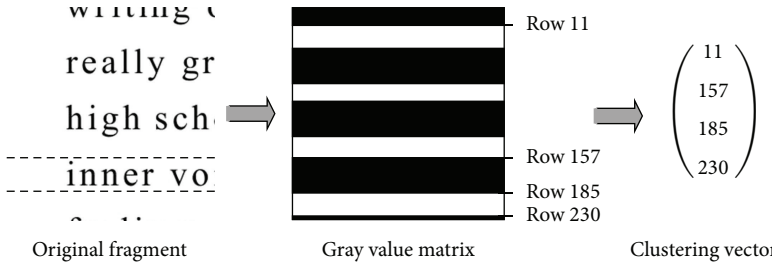


Figure 9: Example of feature extraction.

S1. If the border-top of binary image matrix is equal to 0, there is no unidentified row on the top of the fragment; let $a_1 = 0$; if not, continue to the next step.

S2. If the upper part of the fragment is identified, there is no unidentified row on the top; let $a_1 = 0$; if not, let $a_1 = l_1$, where l_1 is the lower position of the unidentified row.

S3. If the border-bottom of binary image matrix is equal to 0, there is no unidentified row at the bottom of the fragment; let $a_4 = 0$; if not, continue to the fourth step.

S4. If the foot of the fragment is identifiable, let $a_4 = 0$; if not, let $a_4 = l_2$, where l_2 is the upper position of the unidentified row.

S5. If there is any identified row in the fragment, identify one character in the identified row nearest to the border-bottom by the character matching algorithm. Let $a_2 = l_a$ and $a_3 = l_b$, where l_a is the overline position of the identified row and l_b is the underline position of the identified line; if not, let $a_2 = 0, a_3 = 0$.

S6. Finally, we get the clustering vector of the fragment $v = (a_1, a_2, a_3, a_4)^T$.

4.2. The Cluster Center. Define the first fragment in each row of the original document as the cluster center, which has the fixed and larger blank than the other fragments in the leftmost space. Therefore, we find the cluster centers of the fragments. According to the fact that the space before the document has the largest blank, the fragments of the first column as the cluster centers are easily found (Figure 10).

The steps of finding the cluster center are as follows.

S1. Initialize $t = 1$.

S2. Initialize the number of cluster center $p = 0$ and $j = 1$.

S3. If the pixels, which mean the element in the gray image matrix of fragment j, from 1 to tth column are equal to 0, let $p = p + 1$.

S4. If $j < m \times n$, $j = j + 1$, return to S3.

S5. If $p > m$, $t = t + 1$, return to S2. If $p = m$, we obtain m fragments, also called cluster centers. The pixels from 1 to tth column of their gray image matrices are equal to 0. Denote the cluster centers as the cluster centers $C_1, C_2, \ldots, C_m$.

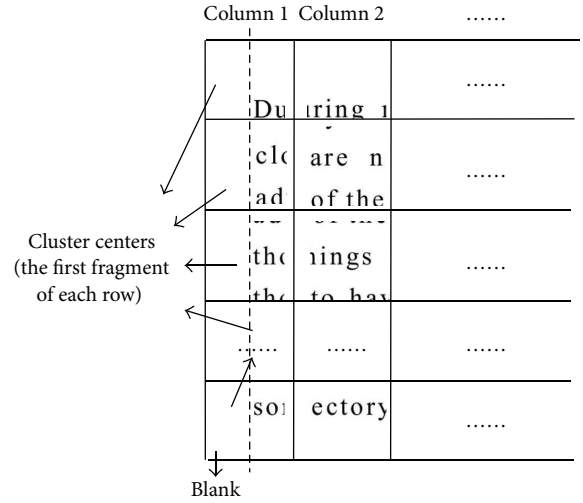


Figure 10: The cluster center of all the fragments.

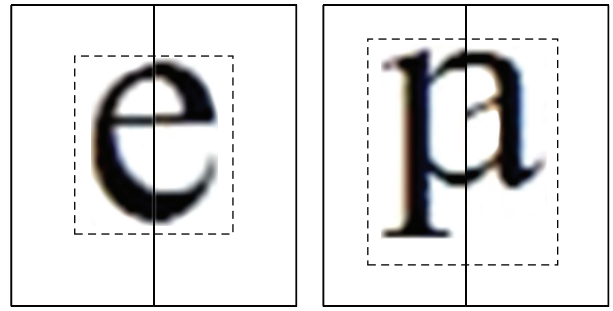
4.3. The Distance between Other Fragments and Cluster Center. All fragments are assigned to their closest cluster centers according to the similarity of their feature vectors. Assume that the feature vector of the cluster center is $\mathrm{CV}' = (a_1', a_2', a_3', a_4')^T$. It is easier to estimate the similarity of two fragments by comparing two clustering vectors. If there is an identified row in both lines and the ruled lines of these two fragments are in the same position, that is, $a_2 = a_2', a_3 = a_3'$, these two fragments are likely to be in the same cluster. If two ruled lines are not in the same position, these two fragments are not adjacent. The distance vector D_{j,C_p} means the distance between the clustering vector CV_j of the fragment j and the clustering vector CV_{C_p} of cluster center C_p, and it is defined as

$$D_{j,C_p} = \begin{cases} (0,0), & a_2 = a_2' \neq 0,\ a_3 = a_3' \neq 0 \\ (\mathrm{Inf}, \mathrm{Inf}), & a_2 \neq a_2' \text{ or } a_3 \neq a_3' \\ \left(\left|a_1 - a_1'\right|, \left|a_4 - a_4'\right|\right), & a_2 = a_2' = a_3 = a_3' = 0. \end{cases} \quad (1)$$

The steps of the row clustering algorithm are as follows.

S1. Compute the distance vector D_{j,C_p}.

S2. If both the components of the vector D_{j,C_p} are no more than the threshold, it shows that the fragment j and the cluster center C_p are in the same cluster.



(a) (b)

Figure 11: The character on the joint boundary is identifiable (a). The character on the joint boundary is unidentifiable (b).

S3. Cluster all the fragments into m clusters, and find the number of the fragments of each cluster.

5. Intrarow Splicing

After row clustering, the set of the fragments in the same row is supposed to be $\{j_1, j_2, \ldots, j_n\}$. Intrarow splicing can be modeled as the problem of finding the optimal path of the graph (also called the traveling salesman problem). The traveling salesman problem figures out the shortest path for the salesman to visit each city exactly and only once and finally return to the original starting point.

As intrarow splicing can be regarded to be the travelling salesman problem, the solution to this problem is to find a shortest path in the undirected graph that visits each vertex exactly once. Each fragment is a vertex of the graph, and the adjacent correlations of the fragments are the edges of the vertices. The goal is to figure out the shortest path which connects the start vertex (the leftmost fragment) to the ending vertex (the rightmost fragment).

The intrarow splicing contains three steps: constructing distance matrix, the first stage splicing based on the character recognition, and the second stage splicing.

5.1. Constructing Distance Matrix. There are two boundaries in a fragment (the left edge and the right edge) when splicing the fragments to the others. Calculate the edges correlation according to the ordinary Euclidean distance metric. For n fragments in the set of fragments $\{j_1, j_2, \ldots, j_n\}$, we define an $n \times n$ correlation matrix to show the adjacency among the fragments. Let $X_{j_p} = (x_1, x_2, \ldots, x_n)^T$ be the gray image matrix of the rightmost edge of fragment j_p and let $Y_{j_q} = (y_1, y_2, \ldots, y_n)^T$ be the gray image matrix of the leftmost edge of fragment j_q, where j_p, j_q represents the fragment j_p and the fragment j_q, respectively $(p, q = 1, 2, \ldots, n)$. Define the distance between the fragment j_p and the fragment j_q as

$$D\left(j_p, j_q\right) = \begin{cases} 0, & j_p = j_q \\ -\text{Inf}, & p_{j_p j_q} = 1,\ j_p \neq j_q \\ d_{j_p j_q}, & p_{j_p j_q} = 0,\ j_p \neq j_q, \end{cases} \tag{2}$$

where

$$d_{j_p j_q} = \sum_{i=1}^{n} \left| x_i - y_i \right|. \tag{3}$$

And $p_{j_p j_q}$ represents whether the character on the joint of two adjacent fragments is identifiable or not:

$$p_{j_p j_q} = \begin{cases} 0, & \text{the character is unidentifiable} \\ 1, & \text{the character is identifiable.} \end{cases} \tag{4}$$

For example, in Figure 11(a), the character on the joint boundary is identifiable because the feature of the character is almost the same as that of the letter "e" in the database. In Figure 11(b), the character on the joint boundary is unidentifiable because the feature of the character is different from that of any letters in the database.

Therefore, we obtain the distance matrix of fragments in the same row.

5.2. The First Stage Splicing. The fragments j_p and j_q are possibly adjacent when their correlation $D(j_p, j_q) = -\text{Inf}$. Select the pairs of the adjacent fragments to verify whether they can be merged into each other. Two fragments are adjacent when one character on the joint of the two fragments is identifiable according to the character database. Figure 11 is an example to explain whether the characters on the boundaries are identifiable or not.

In the first stage splicing, according to distance matrix, we denote the set of fragments $\{j_1, j_2, \ldots, j_n\}$ as follows.

(a) For each given k $(k = 1, 2, 3, \ldots, n)$, if $D(j_p, j_q) > -\text{Inf}$, $l = 1, 2, 3, \ldots, n$, the fragment j_k is denoted as (j_k).

(b) If the following equations exist $D(j_{k_1}, j_{k_2}) = D(j_{k_2}, j_{k_3}) = D(j_{k_3}, j_{k_4}) = \cdots = D(j_{k_{t-2}}, j_{k_{t-1}}) = D(j_{k_{t-1}}, j_{k_t}) = -\text{Inf}$, $1 \leq k_1, k_2, \ldots, k_t \leq n$, and $k_1, k_2, \ldots, k_t$ are pairwise different, the frag-ments $j_{k_1}, j_{k_2}, \ldots, j_{k_{t-2}}, j_{k_{t-1}}, j_{k_t}$ are denoted as $(j_{k_1}, j_{k_2}, \ldots, j_{k_{t-2}}, j_{k_{t-1}}, j_{k_t})$.

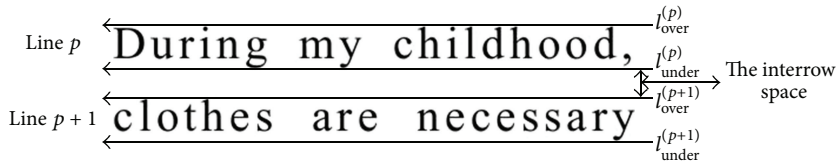


Figure 12: The interrow space.

Table 1: The fragment scale of the shredded document.

Scale (piece)	12	25	35	45	48	54	66	84	96	104	112	120	126	130
Row	4	5	7	9	8	9	11	14	12	13	14	15	14	13
Column	3	5	5	5	6	6	6	6	8	8	8	8	9	10

Finally, the set of fragments $\{j_1, j_2, \ldots, j_n\}$ can be rewritten as follows:

$$\left\{\left(j_{k_1^{(1)}}, j_{k_2^{(1)}}, \ldots, j_{k_{L_1}^{(1)}}\right), \left(j_{k_1^{(2)}}, j_{k_2^{(2)}}, \ldots, j_{k_{L_2}^{(2)}}\right), \ldots, \left(j_{k_1^{(r)}}, j_{k_2^{(r)}}, \ldots, j_{k_{L_r}^{(r)}}\right)\right\}, \tag{5}$$

where $1 \le r, L_r \le n$, $k_s^{(i)} = 1, 2, \ldots, n$, $s = 1, 2, \ldots, L_i$, and $i = 1, 2, \ldots, r$.

Furthermore, $\{(j_{k_1^{(1)}}, j_{k_2^{(1)}}, \ldots, j_{k_{L_1}^{(1)}}), (j_{k_1^{(2)}}, j_{k_2^{(2)}}, \ldots, j_{k_{L_2}^{(2)}}), \ldots, (j_{k_1^{(r)}}, j_{k_2^{(r)}}, \ldots, j_{k_{L_r}^{(r)}})\}$ can be denoted as $\{J_1, J_2, \ldots, J_r\}$, where $(j_{k_1^{(i)}}, j_{k_2^{(i)}}, \ldots, j_{k_{L_i}^{(i)}})$ is represented by J_i ($i = 1, 2, \ldots, r$). After the first stage splicing, the set of fragments $\{j_1, j_2, \ldots, j_n\}$ is represented by $\{J_1, J_2, \ldots, J_r\}$.

5.3. The Second Stage Splicing. After the first stage splicing, the fragments in the same row are denoted by $\{J_1, J_2, \ldots, J_r\}$. Calculate the distance $D(J_k, J_l)$ as defined in the first stage splicing, where $k, l = 1, 2, \ldots, r$.

Suppose that the fragment J_1 contains the clustering center of the row, so the goal of TSP in the intrarow splicing is transferred and modeled as

$$\min\left\{D(J_1, J_i) + \sum_{1<u,v\le r, u\ne v} D(J_u, J_v)\right\}. \tag{6}$$

In other words, we find the solution of the TSP problem with J_1 as the starting point. We apply the genetic algorithm to solve this problem.

With the usual coding method, generate an $1 \times r$ array randomly within the interval $[0, 1]$. The r components in the array correspond to $J_1, J_2, \ldots, J_r$, respectively. The sort in ascending order of the $1 \times r$ array represents the position of $J_1, J_2, \ldots, J_r$. In addition, adopting the multiple point crossover method, select two random positions of the crossover point, and exchange the gene segments. Moreover, adopting the multiple point mutation method, select some random position of the mutation point and replace the value on the mutation point. The fitness function of the genetic algorithm is

$$F = D(J_1, J_i) + \sum_{1<u,v\le r, u\ne v} D(J_u, J_v). \tag{7}$$

The new genetic algorithm with multiple point crossover operators and multiple point mutation operators is applied in order to increase the diversity of the individual. In addition, as for the first stage splicing based on the character recognition, we can offer the optimal initial solution to the genetic algorithm. Therefore the algorithm can be converged quickly. The optimal sequence of the fragments in a row can be obtained through the genetic algorithm.

6. Interrow Splicing

In the text document, suppose that the overline and the underline of line p ($p = 1, 2, \ldots, m$) are denoted as $l^{(p)}_{\text{over}}$ and $l^{(p)}_{\text{under}}$, respectively. The interrow space is defined as the space between $l^{(p)}_{\text{under}}$ and $l^{(p+1)}_{\text{over}}$ (Figure 12). Moreover, all of the interrow spaces in the text document are consistent. Finally, we can splice the string of fragments according to the interrow space and the proximity of the fragments.

7. Simulation

The shredded document with New Times Roman and 20 fonts is simulated to test the efficiency of the model, which is shown in Figure 2. As for the parameter, the threshold for character recognition is set to 0.83, and the threshold for row clustering is set to 232. In genetic algorithm, the population size is set to 100, the iteration is set to 50, and mutation rate is set to 0.6. We implemented our approach in MATLAB and performed all tests on a double core of an Intel core CPU with 2.10 GHz and 2 GB RAM.

A document in Figure 2 is shredded into different scales of fragments to test the algorithm. The scales of fragments are as in Table 1.

Now, we cut the documents into 13×10 fragments and use this as an example to illustrate the process of document reconstruction in detail. First of all, the database that fits for the document reconstruction is applied, which is shown in Table 2.

Secondly, with the help of the database and the character recognition, the clustering vector of each fragment is given through the row clustering algorithm (shown in Table 3).

Table 2: The feature information of letters with New Times Roman and 20 fonts in database.

	LW	LH	OLS	ULS		LW	LH	OLS	ULS
A	28	27	1	10	a	16	19	10	9
B	24	26	2	10	b	18	28	1	9
C	24	28	1	9	c	15	19	10	9
D	27	26	2	10	d	19	28	1	9
E	23	26	2	10	e	16	19	10	9
F	20	26	2	10	f	16	27	1	10
G	27	28	1	9	g	18	27	10	1
H	28	26	2	10	h	20	27	1	10
I	11	26	2	10	i	9	27	1	10
J	14	27	2	9	j	11	36	1	1
K	29	26	2	10	k	20	27	1	10
L	23	26	2	10	l	9	27	1	10
M	34	26	2	10	m	30	18	10	10
N	29	27	2	9	n	20	18	10	10
O	26	28	1	9	o	17	19	10	9
P	21	26	2	10	p	18	27	10	1
Q	26	35	1	2	q	19	27	10	1
R	27	26	2	10	r	13	18	10	10
S	18	28	1	9	s	12	19	10	9
T	22	26	2	10	t	11	24	5	9
U	28	27	2	9	u	20	19	10	9
V	28	27	2	9	v	19	19	10	9
W	37	27	2	9	w	28	19	10	9
X	28	26	2	0	x	19	18	10	10
Y	28	26	2	10	y	19	27	10	1
Z	23	26	2	10	z	17	18	10	10

*LW, LH, OLS, ULS are short for the width of letter, the height of letter, the space between the letter and the overline, the space between the letter and the underline.

Table 3: Clustering vectors of 130 fragments.

Fnum	1	2	3	4	5	6	⋯	123	124	125	126	127	128	129	130
CV	0	0	0	0	0	0	⋯	0	0	0	0	0	0	0	0
	89	89	89	89	89	89	⋯	25	25	25	25	25	25	25	25
	126	126	126	126	126	126	⋯	62	62	62	62	62	62	62	62
	0	0	0	0	0	0	⋯	0	0	0	0	0	0	0	0

*Fnum represents the fragment number, and CV represents the clustering vector of each fragment.

Based on the algorithm of row clustering, 130 fragments are distributed into 13 rows according to their clustering vectors (Table 4).

After the row clustering, the intrarow splicing starts. We use Row 13, for example. The distance matrix is used as in Tables 5 and 6.

Row13: {121 122 124 123 125 126 129 128 127 130}

→ {(121, 122, 123, 124), (126, 127), (128, 129, 130), (125)}

→ {(121, 122, 123, 124), (125), (126, 127), (128, 129, 130)}

→ {121, 122, 123, 124, 125, 126, 127, 128, 129, 130}.

Splicing in the other rows is similar to Row 13.

After intrarow splicing and interrow splicing, as the end of the reconstruction algorithm, the document is finally reconstructed. The reconstruction result is shown in Table 7, corresponding to Figure 13(a).

The correct order of the document is shown in Table 8, corresponding to Figure 2(b).

The precision of the reconstruction is calculated by

$$\text{Precision} = 1 - \frac{\text{the number of fragments in wrong position}}{\text{the total number of fragments}} = 1 - \frac{29}{130} = 0.7769. \quad (8)$$

Table 4: The result of row clustering.

Row 1	1	2	5	4	3	6	8	7	9	10
Row 2	**11**	14	13	12	15	17	16	18	19	20
Row 3	**21**	24	23	22	26	29	28	27	30	25
Row 4	**31**	35	36	32	33	37	38	39	40	34
Row 5	**41**	42	46	44	45	43	47	48	49	50
Row 6	**51**	52	54	55	57	58	59	56	60	53
Row 7	**61**	62	63	64	65	66	67	68	69	70
Row 8	**71**	74	73	72	75	76	77	79	80	78
Row 9	**81**	82	85	83	84	86	88	87	89	90
Row 10	**91**	93	92	94	95	97	96	99	98	100
Row 11	**101**	106	103	109	105	102	107	108	104	110
Row 12	**111**	114	113	115	112	116	119	118	117	120
Row 13	**121**	122	124	123	125	126	129	128	127	130

*The bold numbers are the clustering centers.

Table 5: The distance matrix in first stage.

	121	122	123	124	125	126	127	128	129	130
121	0	−Inf	3050	2234	3656	3657	2777	3647	2831	1681
122	833	0	−Inf	2584	834	833	5591	855	1801	3647
123	2467	2041	0	−Inf	2470	2467	4003	2447	2077	1773
124	0	4428	1031	0	3	0	6416	42	1154	3064
125	0	4428	1031	2451	0	0	6416	42	1154	3064
126	6086	1806	5063	3685	6083	0	−Inf	6072	4938	3090
127	65	4455	1068	2478	62	65	0	101	1187	3107
128	1125	3345	2100	2110	1126	1125	5315	0	−Inf	1993
129	3720	818	3875	1701	3721	3720	2744	3702	0	−Inf
130	0	4428	1031	2451	3	0	6416	42	1154	0

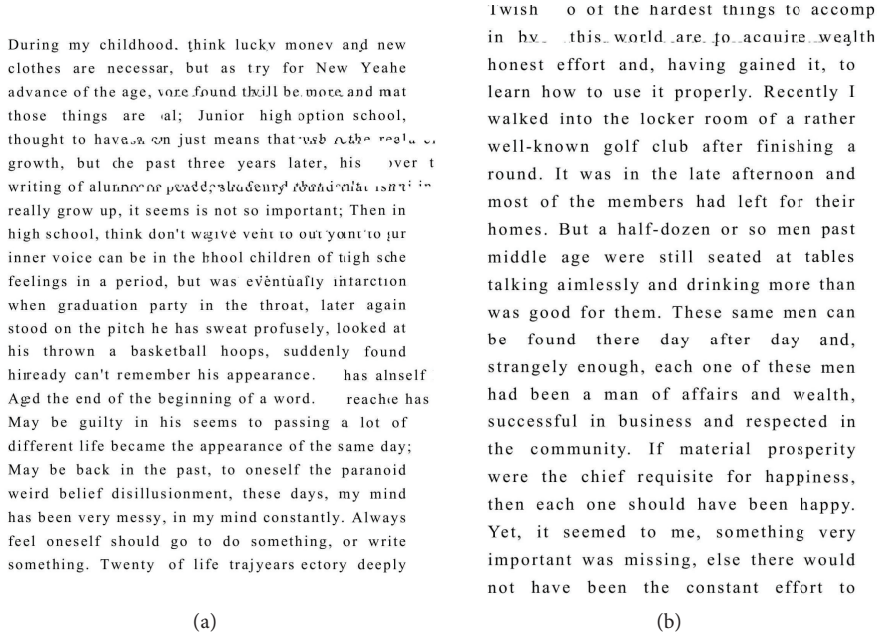


Figure 13: The first reconstructed document (a) and the second reconstructed document (b).

Table 6: The distance matrix in second stage.

	(121, 122, 123, 124)	125	(126, 127)	(128, 129, 130)
(121, 122, 123, 124)	0	3	0	42
125	0	3	0	42
(126, 127)	65	62	65	101
(128, 129, 130)	0	3	0	42

*$(i, j, \ldots)$ means fragment i and fragment j are splicing in the first stage.

Table 7: The final result of the algorithm.

1	2	3	4	**5**	**6**	**7**	**8**	**9**	10
11	12	13	14	**18**	**19**	**15**	**16**	**17**	20
21	22	23	24	**26**	**27**	**28**	**25**	29	30
31	32	33	**35**	**36**	**37**	**38**	**39**	**40**	**34**
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	**57**	**58**	**59**	**56**	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	**84**	**85**	**86**	**87**	**88**	**89**	**90**	**83**	**82**
91	92	93	94	95	96	97	98	99	100
101	102	103	104	105	106	107	108	109	110
111	112	113	114	115	116	117	118	119	120
121	122	123	124	125	126	127	128	129	130

*The bold numbers are in wrong position.

Table 8: The correct order.

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100
101	102	103	104	105	106	107	108	109	110
111	112	113	114	115	116	117	118	119	120
121	122	123	124	125	126	127	128	129	130

In our simulation, it is found that the effective information of the fragment and the precision of document reconstruction descend globally as the number of the fragments increases. But in the local, the precision of the reconstruction has three stages (see Figure 14). When the number of the fragments is less than 55, the precision of this algorithm is 100%; when the number is from 55 to 100, the precision is above 90%; when the fragments are more than 100 pieces, the reconstruction precision fluctuates around 75% to 85%.

Because the initial solution is optimized through combining the adjacent fragments in the first stage splicing, the complexity of genetic algorithm to find the optimal path is simplified. In addition, the fragment splicing algorithm for each cluster can operate in the same time which improves efficiency. Therefore, the total operating time of reconstructing the documents cut into 13×10 fragments is 0.8276 seconds, which is much faster than restoring the fragments by naked eye.

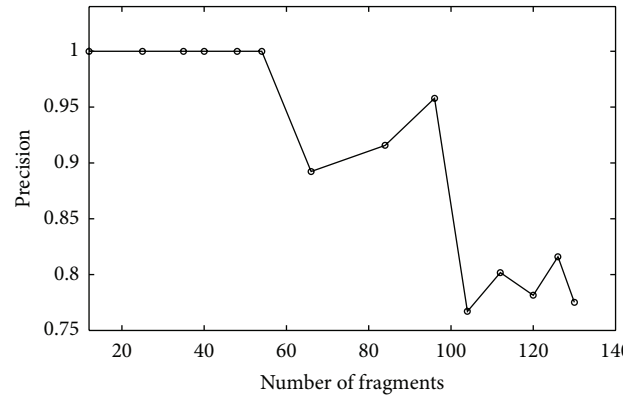


Figure 14: The precision of reconstructing the document (Figure 2) shredded into different pieces.

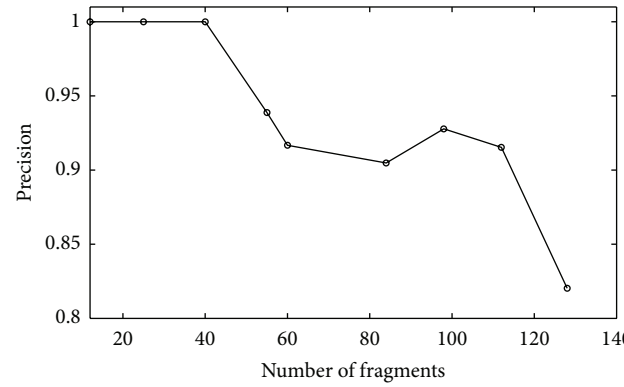


Figure 15: The precision of reconstructing another document shredded into different pieces.

To test the adaptability of the algorithm, we choose another document to compute the precision of document reconstruction (Figure 13(b)). The document is shredded in the same way as the one in the previous experiment, with New Times Roman and 20 fonts. On the one hand, compared with the result in Figure 15, we could find that they have the similar trend, indicating the robustness of the algorithm. On the other hand, the precision in Figure 14 is different from the one in Figure 15, indicating that the result of this algorithm will be influenced by the difference of documents.

Document is reconstructed by the algorithm proposed in this paper without human intervention. As for the unsatisfied result, that is to say, the precision cannot reach 100%, human intervention is necessary. After all, the complete document is our goal.

8. Conclusion and Future Work

This paper studies the reconstruction of the shredded text document cross-cut by the paper shredder. With the construction of the letter database, the character recognition helps find out the accurate ruled line in row clustering. In addition, the character recognition shortens the convergence time by offering the better initial solution to genetic algorithm in intrarow splicing. Meanwhile from the simulation results, the document is reconstructed precisely in that short convergence time. Hence, the feature-matching algorithm based on the character recognition can splice the fragments through row clustering, intrarow splicing, and interrow splicing with high efficiency and high precision.

As a solution for the reconstruction of cross-cut shredded text documents (RCCSTD) problem, the algorithm proposed in this paper can be improved in the future. More feature information of the characters in the same row would be considered to be extracted and used to improving the precision in the row clustering. Also, more advanced OCR technology may help solve the RCCSTD problem a lot. We will use it to splice a larger number of fragments from the cross-cut shredded text documents and keep the high precision which is the unique goal for RCCSTD problem.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

This work was partly supported by the National NSF of China (no. 11071089).

References

[1] C. Schauer, M. Prandtstetter, and G. R. Raidl, "A memetic algorithm for reconstructing cross-cut shredded text documents," in *Hybrid Metaheuristics*, vol. 6373 of *Lecture Notes in Computer Science*, pp. 103–117, Springer, Berlin, Germany, 2010.

[2] E. Justino, L. S. Oliveira, and C. Freitas, "Reconstructing shredded documents through feature matching," *Forensic Science International*, vol. 160, no. 2-3, pp. 140–147, 2006.

[3] P. D. Kesarkar, M. RC. Prasad, and S. L. Tade, "Reconstruction of torn page using corner and segment matching," *Reconstruction*, vol. 2, no. 6, 2013.

[4] F. Richter, C. X. Ries, N. Cebron, and R. Lienhart, "Learning to reassemble shredded documents," *IEEE Transactions on Multimedia*, vol. 15, no. 3, pp. 582–593, 2013.

[5] H. Y. Lin and W. C. Fan-Chiang, "Reconstruction of shredded document based on image feature matching," *Expert Systems with Applications*, vol. 39, no. 3, pp. 3324–3332, 2012.

[6] J. Perl, M. Diem, F. Kleber, and R. Sablatnig, "Strip shredded document reconstruction using optical character recognition," in *Proceedings of the 4th International Conference on IET Imaging for Crime Detection and Prevention (ICDP '11)*, pp. 1–6, London, UK, 2011.

[7] M. Diem and R. Sablatnig, "Recognizing characters of ancient manuscripts," in *Computer Vision and Image Analysis of Art, 753106*, vol. 7531 of *Proceedings of SPIE*, p. 12, IST/SPIE Electronic Imaging. International Society for Optics and Photonics, San Jose, Calif, USA, January 2010.

[8] M. Prandtstetter, *Hybrid optimization methods for warehouse logistics and the re-construction of destroyed paper documents [Ph. D. thesis]*, Vienna University of Technology, 2009.

[9] B. Biesinger, C. Schauer, B. Hu, and G. R. Raidl, "Enhancing a genetic algorithm with a solution archive to reconstruct cross cut shredded text document," in *Computer Aided Systems Theory-EUROCAST*, pp. 380–387, Springer, Berlin, Germany, 2013.

[10] M. Prandtstetter and G. R. Raidl, "Meta-heuristics for reconstructing cross cut shredded text documents," in *Proceedings of the ACM 11th Annual Genetic and Evolutionary Computation Conference (GECCO '09)*, pp. 349–356, Montreal, Canada, July 2009.

[11] A. Sleit, Y. Massad, and M. Musaddaq, "An alternative clustering approach for reconstructing cross cut shredded text documents," *Telecommunication Systems*, vol. 52, no. 3, pp. 1491–1501, 2013.

[12] C. Schauer, *Reconstructing cross-cut shredded documents by means of evolutionary algorithms, [M.S. thesis]*, Vienna University of Technology, Institute of Computer Graphics and Algorithms, Vienna, Austria, 2010.

[13] S. Lu, B. M. Chen, and C. C. Ko, "Perspective rectification of document images using fuzzy set and morphological operations," *Image and Vision Computing*, vol. 23, no. 5, pp. 541–553, 2005.

[14] Z. Zhang, S. Yu, Z. Fang, and Y. Qiao, "Real-time recognition algorithm of Arabic numerals and English letters based on field-programmable gate array design," *Journal of Shanghai Jiaotong University*, vol. 40, no. 1, pp. 12–15, 2006 (Chinese).