Available online at www.sciencedirect.com
SCIENCE DIRECT
Forensic Science International 160 (2006) 140–147
Forensic Science International
www.elsevier.com/locate/forsciint

Reconstructing shredded documents through feature matching

Edson Justino [a], Luiz S. Oliveira [a,b,*], Cinthia Freitas [a]

[a] *Pontifical Catholic University of Parana (PUCPR), Graduate Program in Applied Computer Science (PPGIA), Rua Imaculada Conceição 1155, Prado Velho, 80215-901 Curitiba, PR, Brazil*
[b] *Tuiuti University of Parana (UTP), Faculty of Exact Sciences and Technology, Rua Sidney A.R. Santos, 238-82010-330, Curitiba, PR, Brazil*

Received 27 June 2005; received in revised form 6 September 2005; accepted 6 September 2005
Available online 25 October 2005

Abstract

We describe a procedure for reconstructing documents that have been shredded by hand, a problem that often arises in forensics. The proposed method first applies a polygonal approximation in order to reduce the complexity of the boundaries and then extracts relevant features of the polygon to carry out the local reconstruction. In this way, the overall complexity can be dramatically reduced because few features are used to perform the matching. The ambiguities resulting from the local reconstruction are resolved and the pieces are merged together as we search for a global solution. The preliminary results reported in this paper, which take into account a limited amount of shredded pieces (10–15) demonstrate that feature-matching-based procedure produces interesting results for the problem of document reconstruction.

Keywords: Document reconstruction; Feature matching; Polygonal approximation

1. Introduction

Questioned Document Examination (QDE) is a sub-field of forensic sciences and it is related to the federal, civil, law enforcement, and justice areas. The task of document examination is to compare a questioned document, using a scientific method to a series of known standards, i.e., signature verification, handwriting identification, etc. In order to perform a reliable analysis, forensic document examiner must count on well-preserved documents.

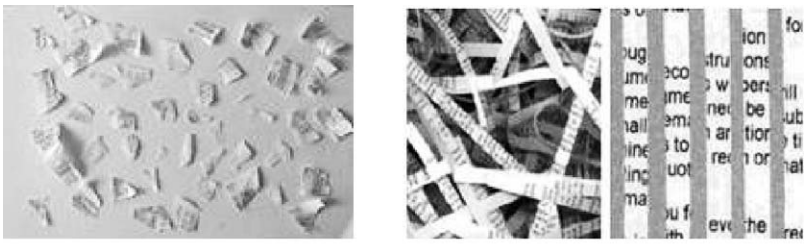
However, very often questioned documents suffer damages at several levels, such as, torn edges, moisture, obliteration, charring, and shredding. In the latter case, shredding can be performed by a machine or by hand (Fig. 1). In both cases, documents need to be reconstructed so that forensic examiners can analyze them. The amount of time necessary to reconstruct a document depends on the size and the number of fragments, and it can be measured in days or even weeks. Sometimes some fragments of the document can be missing, and for this reason, the document can be only partially reconstructed. Even then, the manual effort of the forensic examiner, which is tedious and laborious, can be alleviated.

One problem faced when reconstructing documents by hand lies in its manipulation. The physical reconstruction of a document modifies some aspects of the original document because products like glue and adhesive tape are added into it. This type of manipulation is known as destructive analysis.

In this paper, we focus on the reconstruction of documents shredded by hand, which is similar to the *automatic assembly of jigsaw puzzle.* Puzzle pieces are often represented by their boundary curves and local shape matching is

* Corresponding author. Tel.: +55 41 3271 1361; fax: +55 41 3271 2121.
E-mail address: soares@ppgia.pucpr.br (L.S. Oliveira).

0379-0738/$ – see front matter
doi:10.1016/j.forsciint.2005.09.001



(a) (b)

Fig. 1. Different kinds of shredding.

usually achieved by curve matching. However, matching between two pieces usually occurs over only a fraction of their correspondent boundaries, and for this reason, a partial curve matching is necessary.

Wolfson in [11] describes two curve matching algorithms where the boundaries are represented by shape feature strings which are obtained by polygonal approximation. The matching stage finds the longest common sub-string and it is solved by geometric hashing. The algorithms described by Wolfson are pretty fast, and for this reason are used by most puzzle solving methods. According to Kong and Kimia in [5], these algorithms fail when the number of puzzle pieces become larger, though. Other methods for curve matching have been proposed in the literature [9,3] to perform matching at fine scale, however, their expensive computational cost compromises their application for puzzle solving.

Kong and Kimia [5] propose re-sampling the boundaries by using a polygonal approximation in order to reduce the complexity of the curve matching. They make a coarse alignment using dynamic programming on the reduced version of the boundaries. Thereafter, they apply dynamic programming again into the original boundaries to get a fine-scale alignment. A similar approach is applied by Leitao and Stolfi [8], where they compare curvature of the fragments, at progressively increasing scales of resolution, using an incremental dynamic programming sequence matching algorithm. Dynamic programming for puzzle solving has been used also by Bunke and Kaufmann [2] and Bunke and Buehler [1].

Many times local shape analysis produces ambiguous matches and it gets worse as the number of puzzle pieces increases. In order to eliminate the ambiguity in the global picture, a global search technique is required. Wolfson et al. in [12] report an algorithm to solve large puzzles, but with some constraints regarding the shape of the puzzle pieces. They show good results, but this kind of strategy is not practical in many real applications.

Uçoluk and Toroslu [10] search all pairs of pieces and the best match is selected and merged to form a new piece. The algorithm is repeated until there is only one piece left. Since invalid matching may occur during the merging process, a backtracking procedure is considered. If the backtracking is used very often, then the idea becomes computationally expensive. Alternatively, computers may be used only in the local shape analysis stage and a human can be used to assist the global search [7,8]. This is known as ‘‘human in the loop’’ evaluation, a concept that has been largely applied to critical systems [6].

In this work, we propose a local reconstruction based on two steps. First of all, we apply a polygonal approximation in order to reduce the complexity of the boundaries and overcome specific problems faced in document reconstruction. Most of the works found in the literature exploit the fact that ordinary puzzle pieces have smooth edges and well defined corners. However, we demonstrate that pieces of paper shredded by hand does not follow this pattern. Then, the second step consists in extracting relevant features of the polygon and using them to make the local reconstruction. In this way, the overall complexity can be dramatically reduced because few features are used to perform the matching.

The ambiguities resulting from the local reconstruction are resolved and the pieces are merged together as we search for a global solution. We demonstrated by comprehensive experiments that this feature-matching-based procedure produces interesting results for the problem of document reconstruction. A global search is considered to reconstruct the entire document.

The remaining of this work is organized as follows: Section 2.1 presents an overview of the proposed methodology. Section 2.2 describes the feature set we have used to carry out the local matching. Section 2.3 shows how we compute the similarity between the polygons as well as the global search algorithm. Finally, Section 3 reports the experimental results and Section 4 present some perspectives of future works and concludes this work.

2. The proposed methodology

Our methodology is composed of three major steps as depicted in Fig. 2. Initially, each piece of the document is pre-processed through polygonal approximation in order to reduce complexity of the boundaries. Then, a set of features is extracted from each polygon in order to carry out the matching. In the following sections, we describe in details each component of the methodology.

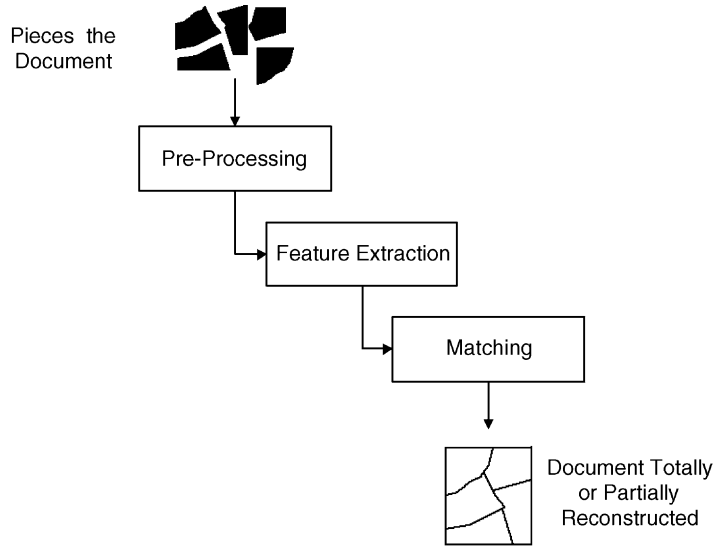


Fig. 2. The block diagram of the proposed methodology.

2.1. Pre-processing

Traditional puzzle solving algorithms usually take into account smooth edges and well defined corners. However, dealing with shredded documents is quite more complex. The act of shredding a piece of paper by hand often produces some irregularities in the boundaries, which makes it impossible to get a perfect curve matching. Fig. 3 shows an example of this problem.

It can be observed from Fig. 3b, that the fragment has two boundaries: the inner and the outer boundaries. The problem lies in the fact that when acquiring the images of this kind of fragments, the inner boundary is lost, and it is easy to see that the outer boundaries of the fragments (a) and (b) do not match perfectly.

In order to overcome this kind of problem, we have tested different algorithms, and the one that brought the best results was the well-known Douglas–Peucker (DP) algorithm [4]. This algorithm implements a polyline simplification and it is used extensively for both computer graphics and geographic information systems.

The DP algorithm uses the closeness of a vertex to an edge segment. This algorithm works from the top to down by starting with a crude initial guess at a simplified

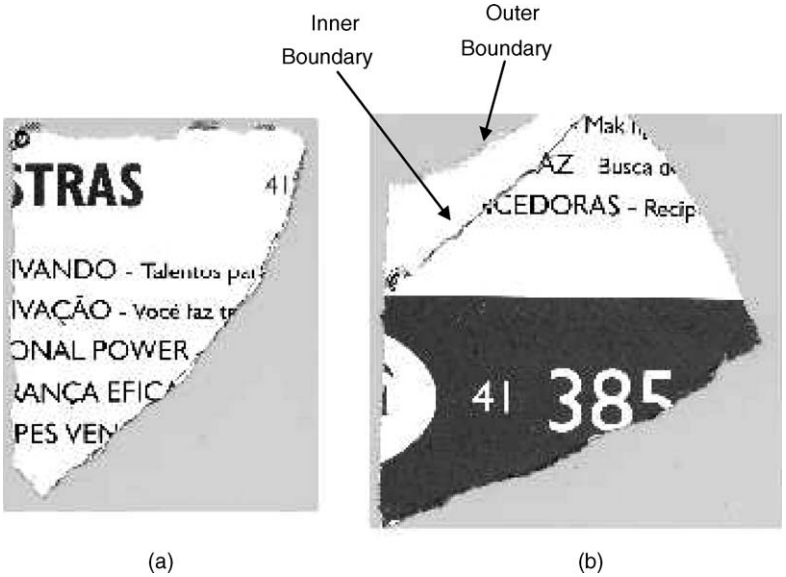


Fig. 3. Inner and outer boundaries produced by shredding.

polyline, namely the single edge joining the first and last vertices of the polyline. Then the remaining vertices are tested for closeness to that edge. If there are vertices further than a specified tolerance, $T > 0$, away from the edge, then the vertex furthest from it is added the simplification. This creates a new guess for the simplified polyline. Using recursion, this process continues for each edge of the current guess until all vertices of the original polyline are within tolerance of the simplification.

More specifically, in the DP algorithm, the two extreme endpoints of a polyline are connected with a straight line as the initial rough approximation of the polyline. Then, how well it approximates the whole polyline is determined by computing the distances from all intermediate polyline vertices to that (finite) line segment. If all these distances are less than the specified tolerance T, then the approximation is good, the endpoints are retained, and the other vertices are eliminated. However, if any of these distances exceeds the T tolerance, then the approximation is not good enough. In this case, we choose the point that is furthest away as a new vertex sub-dividing the original polyline into two (shorter) polylines.

This procedure is repeated recursively on these two shorter polylines. If at any time, all of the intermediate distances are less than the T threshold, then all the intermediate points are eliminated. The routine continues until all possible points have been eliminated. Fig. 4 shows two different levels of approximation.

2.2. Feature extraction

After the complexity reduction through polygonal approximation, the next step consists in extracting features to carry out the local matching. The feature extraction can be seen also as a complexity reduction process, since it converts the polygon in a sequence of features. Here, we propose a simple feature set that can be used to carry out the local matching.

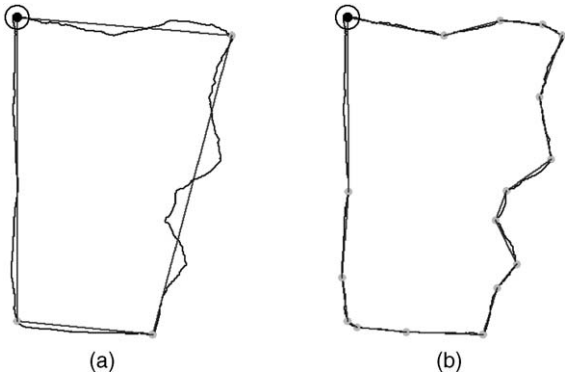


Fig. 4. Inner and outer boundaries produced by shredding.

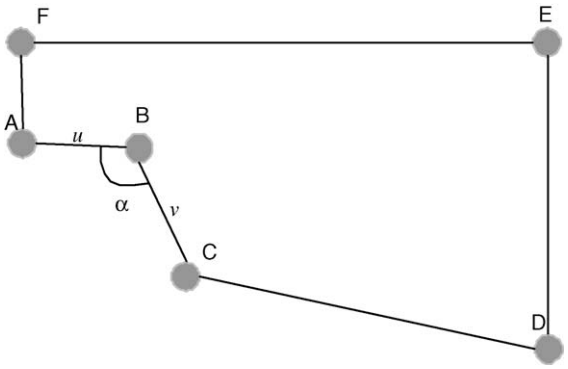


Fig. 5. Angle features extracted from the polygon.

The first feature is the angle of each vertex with respect to its two neighbors. Consider, i.e., the vertices A and B in the polygon depicted in Fig. 5. The angle α is given by

$$\cos\alpha = \frac{uv}{|u||v|} \quad (1)$$

We also verify whether such an angle is convex or concave. For example, in Fig. 5, vertex B has a convex angle while vertex C has a concave one. To complete our feature set, we compute the distances between the vertex and its neighbors (next and previous in a clockwise sense). Such distances are achieved by means of the well-known Euclidean distance. Table 1 describes the feature vector extracted from the polygon depicted in Fig. 5. The last two features are the coordinates of the vertex in the image.

This table can be read as follows: The angle of the vertex B, which is computed by using vertex A and C, is 120°. The Euclidean distances between B and its neighbors A and C are 45.0 and 43.6, respectively. The coordinates of the vertex B in the image are (55,67).

2.3. Matching

2.3.1. Computing the similarity between polygons

The feature vector described so far allows us to compute a degree of similarity, which is used to measure the quality of the matching between two fragments of the document.

Table 1
Description of the feature vector

Vertex	Angle (°)	Distances		X	Y
		Next	Previous		
A	270	40.0	45.0	10	70
B	120	45.0	43.6	55	67
C	200	43.6	115.7	67	25
D	245	115.7	11.0	180	0
E	270	110.0	170.0	180	110
F	270	170.0	40.0	10	110

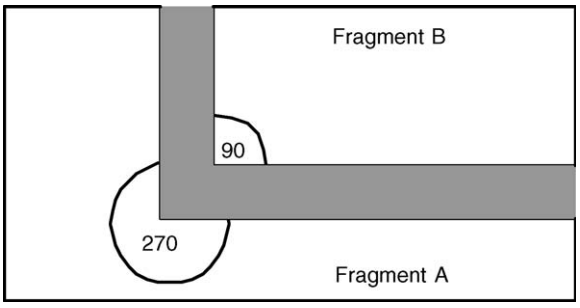


Fig. 6. Similarity between angles.

First of all, we verify the complementarity between the angles of the two vertices being compared. Generally speaking, both angles must sum up 360°, as illustrated in Fig. 6. Of course, we have to consider some degrees of freedom, since they could not sum up 360° due to the estimations made during the polygonal approximation. If the complementarity is verified like in Fig. 6, then $W_{\text{angles}} = 1$.

Thereafter, the distances between the vertex and their neighbors are compared as illustrated in Fig. 7. D_{p1} is the Euclidean distance between the vertex A_1 and its previous neighbor C_1. D_{n1} is the Euclidean distance between the vertex A_1 and its next neighbor B_1. The distances D_{p2} and D_{n2} are computed in the same way. After computing such distances, a measure of similarity W_{matching} is calculated by using Eq. (2).

$$W_{\text{matching}} = \begin{cases} 1 & \text{if } [(D_{\text{p1}} \simeq D_{\text{p2}} \quad \text{or} \quad D_{\text{n1}} \simeq D_{\text{n2}}) \quad \text{and} \quad W_{angle} = 1] \\ 5 & \text{if } [(D_{\text{p1}} \simeq D_{\text{p2}} \quad \text{and} \quad D_{\text{n1}} \simeq D_{\text{n2}}) \quad \text{and} \quad W_{angle} = 1] \end{cases} \tag{2}$$

It is clear from Eq. (2) that the weight is much more relevant when both distances are similar. These values were determined empirically through several experiments.

Finally, we consider the relevance of the matching regarding the perimeter of the fragment using the following rules:

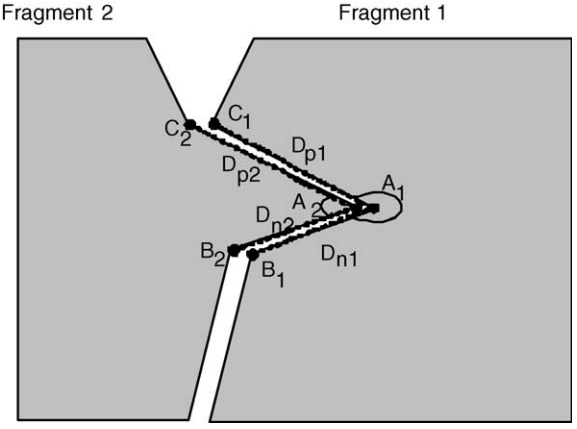


Fig. 7. Distance features extracted from the polygon.

- If the contour matched represents more than 1/5 of the perimeter of the fragment, then $W_{\text{matching}} = W_{\text{matching}} + 2$.
- If the contour matched represents more than 1/10 of the perimeter of the fragment, then $W_{\text{matching}} = W_{\text{matching}} + 1$.
- Otherwise, W_{matching} is not increased.

After several experiments, we realized that these simple rules allow a more reliable identification of the relevant matchings.

2.3.2. Global search

Once the metric to measure a matching has been defined, the next step consists in reconstructing the entire document. As stated somewhere else, this global search also eliminates the ambiguities resulting from the local reconstruction described in the previous section.

The method applied here is based on the algorithm proposed by Leitao and Stolfi [8], which tries to match two pieces at time. Let us consider a shredded document $D = \{F_1, F_2, \ldots, F_n\}$ composed of n fragments. The algorithm compares the fragment F_1 with all the other fragments searching for the best matching, i.e., the match that maximizes the W_{matching} defined previously. Then, the fragments F_i and F_j that maximizes W_{matching} are merged forming a new fragment F_{ij}.

The feature vector of the new fragment F_{ij} is then modified by removing the vertices matched. Fig. 8 shows this merging and the vertices removed as well.

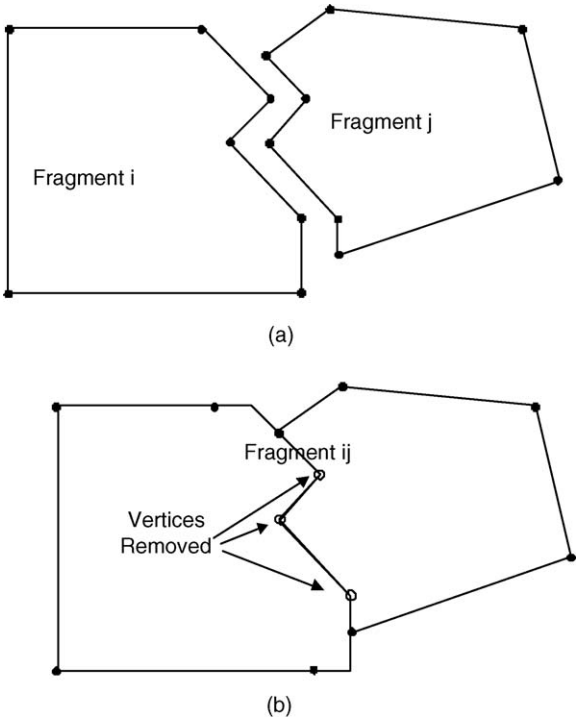


Fig. 8. Best matching (a) fragments i and j and (b) new fragment F_{ij} where three vertices were removed.

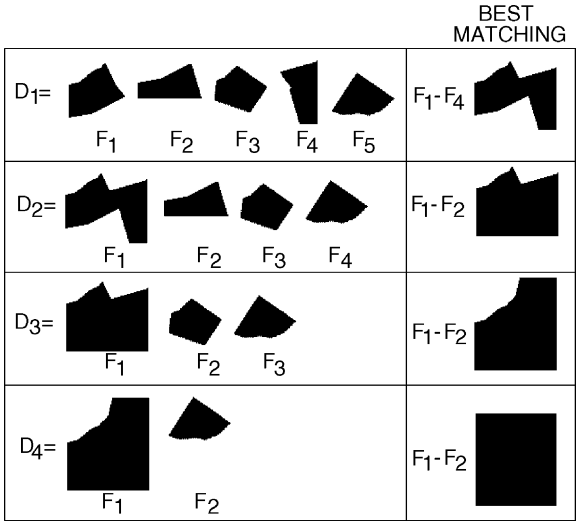


Fig. 9. Steps of the document reconstruction.

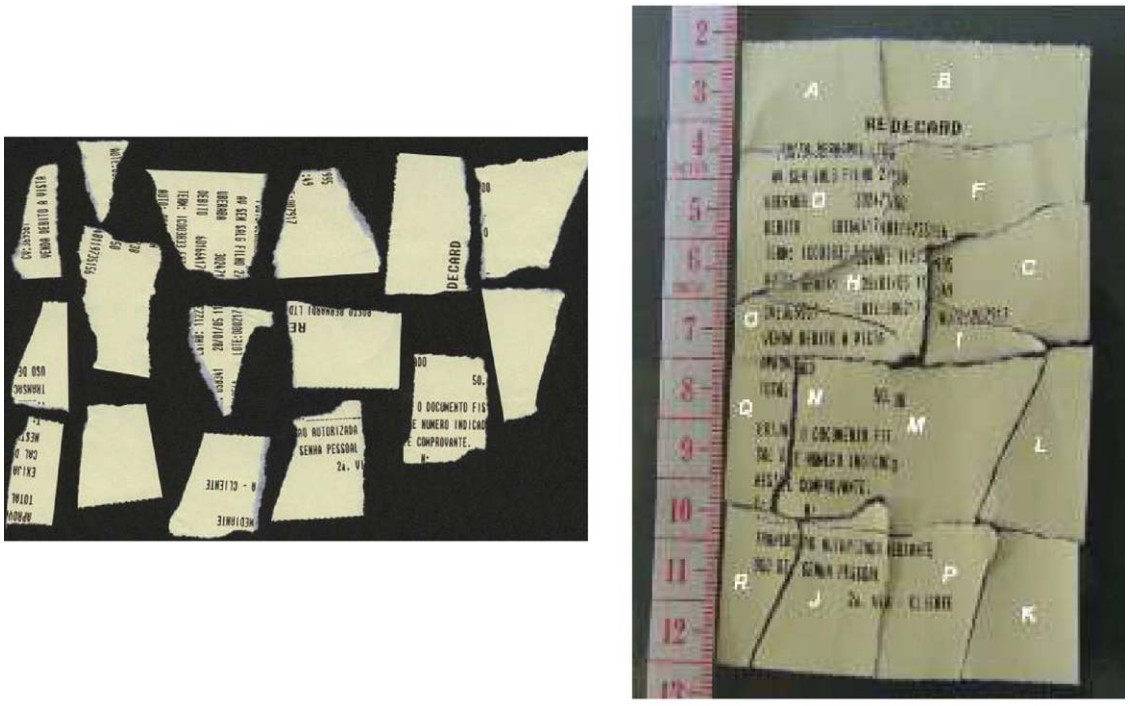
After merging, the process starts again but now the document has $(n-1)$ fragments. It ends when the number of fragments is 1 or none fragments have been merged. This procedure is described in Algorithm 1.

Analyzing the algorithm, we can observe that it returns the document either totally or partially reconstructed. Fig. 9 shows the results of the global search for a 5-fragment document. In such a case, the document was totally reconstructed.

Algorithm 1. Global search

```
D = {F_1, F_2, ..., F_n}
repeat
  best = NULL
  for i = 2 to n do
    Compute all possible W_matching for F_1 and F_i
    if there is a W_matching > 0 then
      best = i that maximizes W_matching
    end if
  end for
  if best ≠ NULL then
    F_new = F_1 ∪ F_best
    Remove F_1 and F_best from D.
    Insert F_new into D
    n = n−1
  end if
until n = 1 or best ≠ NULL
return F_new
```

Fig. 10. Examples of the documents in the database.



(a) (b)

Fig. 11. Examples of a document totally reconstructed: (a) fragments and (b) document reconstructed.

3. Experiments

In order to validate the proposed methodology, we have built a database of shredded documents. Firstly, we have collected 100 diversified documents containing handwriting, machine printed, images, and graphs. Then, the documents were shredded into 3–16 fragments and their size range from 1 cm × 1 cm to 5 cm × 5 cm. It is worth of remark that the documents were randomly shredded, in other words, we have not used any criteria to perform this task. The idea was to create a database as close as possible of a real database. After shredding, the fragments of the documents were labelled and digitalized in 150 dpi, gray-level. It is important to mention that the fragments have been scanned grouped belonging to the same document. Fig. 10 shows some examples of the documents in the database, while Fig. 11 shows an example of a document totally reconstructed.

We have used 10% of the database to train the system (fine-tuning the parameters) and the remaining were used to performance evaluation. Fig. 12 reports the average performance of the system as the number of fragments increases. As we can see, the performance drops as the number of fragments gets bigger. This is a drawback of the polygonal approximation. It allows us to reduce considerably the complexity of the matching process, but on the other hand, such an approximation makes it difficult to keep the same level of performance for document with large number of fragments.

A possible solution for that lies in moving from coarse to fine-scale alignment so that the effects of the approximation could be minimized. In the long run, it is a question of choosing the best trade-off between complexity and performance.

As we have mentioned before, the smallest fragment size we have in our database is about 1 cm × 1 cm. We have not performed tests with smaller fragment sizes. In spite of the

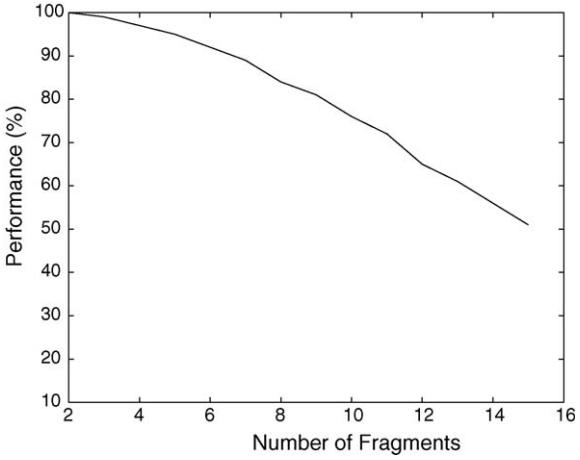


Fig. 12. Performance of the proposed methodology in reconstructing documents shredded by hand.

fact that the proposed approach has no constraints regarding the fragment size, the system will perform worse as the fragment sizes get smaller (less features to match).

As stated somewhere else, the results produced by this kind of system are very useful even when the documents are partially reconstructed. In such cases, a human will dispend considerably less efforts to finish reconstructing the document than starting from scratch.

4. Conclusion and future works

In this paper, we have proposed a methodology for document reconstruction based on feature matching. It takes two steps where the former makes an approximation in order to reduce the complexity of the boundaries and overcome specific problems faced in document reconstruction and the latter extracts relevant features of the polygon and uses them to make the local reconstruction.

The results we have shown demonstrates that the methodology, in spite of the fact of using few features, is able to reconstructed documents shredded by hand. As discussed previously, the performance drops as the number of fragments gets bigger due to the scale used during the polygonal approximation. This issue can be addressed by choosing the most important aspects for the application, i.e., reducing complexity or improving performance. It is worth of remark that both views are important. A less complex system, like the one presented here, could be applied initially and then a more complex one, hence more time consuming, could be applied to resolve final confusions.

As future works, we plan to make some experiments in this sense and also to analyze the performance of the system for other kinds of shredding as well. We are also investigating some optimization techniques such as genetic algorithms and particle swarm optimization so that larger amounts of shredded pieces (up to 1000) can be handled by the algorithm.

References

[1] H. Bunke, U. Buehler, Applications of approximate string matching to 2-D shape recognition, Pattern Recognit. 26 (1993) 1797–1824.
[2] H. Bunke, G. Kaufmann, Jigsaw puzzle solving using approximate string matching and best-first search, in: Proceedings of the Fifth International Conference on Computer Analysis of Images and Patterns, 1993, pp. 299–308.
[3] I. Cohen, N. Ayache, P. Sulger, Tracking points on deformable objects using curvature information, in: Proceedings of the Second European Conference on Computer Vision, Santa Margherita Ligure, Italy, 1992), pp. 458–466.
[4] D. Douglas, T. Peucker, Algorithms for the reduction of the number of points required to represent a digitized line or its caricature, Can. Cartogr. 10 (1973) 112–122.
[5] W. Kong, B. Kimia, On solving 2d and 3d puzzles under curve matching, in: Proceedings of the Conference on Computer Vision and Pattern Recognition, Kauai, Hawaii, USA, 2001), pp. 583–590.
[6] A. Korzyk, W. Yurcik, On integrating human-in-the-loop supervision into critical infrastructure process control systems, in: Proceedings of the Advanced Simulation Technologies Conference, San Diego, USA, 2002.
[7] H.C.G. Leitao, J. Stolfi, Digitalization and reconstruction of archaeological artifacts, in: Proceedings of the 14th Brazilian Symposium on Computer Graphics and Image Processing, Florianopolis, SC, Brazil, 2001), p. 382.
[8] H.C.G. Leitao, J. Stolfi, A multiscale method for the reassembly of two-dimensional fragmented objects, IEEE Trans. Pattern Anal. Mach. Intel. 24 (2002) 1239–1251.
[9] T.B. Sebastian, J.J. Crisco, P.N. Klein, B. Kimia, Constructing 2d curve atlases, in: Proceedings of the Mathematical Methods in Biomedical Image Analysis, 2000, pp. 70–77.
[10] G. Uçoluk, I. Toroslu, Automatic reconstruction of broken 3-d surface objects, Comp. Graph. 23 (1999) 573–582.
[11] H. Wolfson, On curve matching, IEEE Trans. Pattern Anal. Mach. Intel. 12 (1990) 483–489.
[12] H. Wolfson, E. Schonberg, A. Kalvin, Y. Lamdan, Solving jigsaw puzzles by computer vision, Ann. Operat. Res. 12 (1988) 51–64.