Reconstruction of Shredded Documents in the Absence of Shape Information

Joanna Balme
joanna.balme@yale.edu
Advisor: Avi Silberschatz
Yale University
May 7, 2007

Abstract
Many current image and document reconstruction systems rely primarily on shape data, often to the exclusion of additional color information. However, when given a data set absent of significant shape information, such as the fragments produced by common document shredders, color information becomes a necessary part of reconstruction. In this paper I propose to examine some general techniques for strip-shredded document reconstruction; in particular, using color information to reduce the search complexity of shred matching.

1 Introduction
A significant amount of the previous research in fragmented document reconstruction is based on sample data sets where shape information is paramount. A frequent sample model for such reconstruction algorithms is the jigsaw puzzle: techniques developed for jigsaws can later be applied to archaeological and forensic reconstruction problems, for example. In their paper *An Automatic Jigsaw Puzzle Solver* (1994) Kosiba, Devaux, and Balasubramanian address the disparity between shape and color-based solvers by incorporating image information into their primarily shape-based matching algorithm. First, they use the scanned image of the puzzle’s cover to speed up matching, and secondly, they use border pixels to confirm potential pairings. However, the image information is simply added to a feature set which helps to test a sample match, but not used to narrow the set of potential matches to be tested. Although this paper does seek to take advantage of the image data which had been largely neglected by previous jigsaw solvers, it is merely used as an additional test, as opposed to a tool for reducing search time by classifying fragments into testing groups.

The most relevant recent research on the topic is that done by Patrick De Smet of Ghent University. In his 2005 paper *Semiautomatic reconstruction of strip-shredded documents* he experiments with a method for classifying strips into groups, so that each strip need only be tested against a small subset instead of an expensive brute-force search of all possible pairings. De Smet identified a number of possible grouping methods, and reported his findings on the use of a particular method he referred to as binary text-line determination. This procedure tests each row of pixels on a given strip; if greater than a certain threshold of non-background colored pixels are detected, that row is considered a text line. With the resulting codification of strips in a manner very much resembling a bar code, the strips can be sorted into potential matching groups. Only after this preliminary sorting has occurred is a more precise test run on each of the pairs within a given sorted group. Professor De Smet has identified the need to first use image information to strictly limit the size of potential match sets when
testing for pairings among shreds. Although algorithms which either approve or reject a match given a particular pair of shreds are also important, having a realistic running time requires an excellent preprocessing system to reduce the potential match set to be tested. Although the binary text-line algorithm had interesting results, the set of documents and shred types for which it can be used is fairly specific. Only strip-shreds running perpendicular to the text of a document can be used. Photographs and other documents not containing standard text require a more generalized type of algorithm. Shreds produced from the increasingly popular cross-cut shredders are also not eligible for analysis under this algorithm. My goal was therefore to build upon this research and develop a system for grouping, match set optimization, and precise testing that did not have such limitations; purely based on the color information as opposed to dependent on the particular type or orientation of the document.

2 Image Acquisition

A matching system first requires the construction of a shred database. Shreds were scanned in groups against a blue background at a 200dpi resolution. To detect the shreds a region-grower algorithm traverses the image: if a non-background-colored, previously un-visited pixel is detected, it is first marked as visited, then all adjacent pixels are added to a queue to be considered as members of the same shred. This process repeats until the entire shred has been detected. Shreds of under 1000 pixels in area are discarded as noise (a figure chosen based on the size of sample shreds and the scan resolution). Corner detection algorithms yield a way to define the four edges of the shred, rather than treating it as a single amorphous blob. Once the boundaries of a given shred are defined, information collection begins. During an edge-detection pass down each of the long edges of the shred, the pixels composing the edge are placed in an array for use in future testing. Also, grayscale section averages are computed and shred length is calculated as these edge pixels are accumulated (see Grouping Methods).

3 Grouping Methods

If no search optimization method was used, each shred would have to be tested against \(3n^2\) other shreds, where \(n\) is the number of shreds in the database. The directionality of each shred is not assumed to be correct, so in addition to comparing every left edge against every right edge, one must compare each left edge with every reversed left edge, and each right edge with every reversed right edge to perform comprehensive matches. To improve upon \(3n^2\) calls to the expensive comparison function (see Intra-group Testing), we must find a way to exclude clearly incorrect matches from being tested.

Length: A simple method of exclusion, particularly useful when various types of documents have been shredded together, is to first exclude based on length. In order to be flexible to various scan resolutions, a percentage of length is used as a cutoff instead of a fixed pixel value. This has the added benefit of excluding malformed or excessively torn shreds - since the length is calculated during the accumulation of edge information with edge detection (not the simple length between corners) it is sensitive to meandering edges.

Segment Averages: Ideally, we should avoid doing a full test on shreds with wildly different color information. To that end, during the gathering of edge information, average grayscale values for an arbitrary \(k\) sections along the shred are computed. Later, instead of doing the comprehensive pixel-by-pixel match, these values can first be checked to see if the shreds fall within the same grayscale ranges.

<table>
<thead>
<tr>
<th></th>
<th>Image 1</th>
<th>Image 2</th>
<th>Image 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>no segmentation</td>
<td>81</td>
<td>93</td>
<td>60</td>
</tr>
<tr>
<td>k = 1</td>
<td>31.9</td>
<td>58.2</td>
<td>42.6</td>
</tr>
<tr>
<td>k = 2</td>
<td>11.1</td>
<td>39.9</td>
<td>33.1</td>
</tr>
<tr>
<td>k = 4</td>
<td>9.1</td>
<td>34.0</td>
<td>29.4</td>
</tr>
</tbody>
</table>

Table 1: Number of segments(k) vs. Number of matches done per shred

I selected \(k=2\) for use in my system as a good com-
promise between adding the extra weight to the shred objects and optimizing the number of matches.

**Useless Shreds:** As discussed in de Smet’s paper, all-white shreds serve only to confuse the matching system, since each all white shred will perfectly match all others. Shreds with an extremely low amount of color information along the edges are excluded from the matching process and printed at the end of the document reconstruction process, in case information on the center of the strip may have significance to the user.

**Margin Runlength:** Another useful technique for separating shreds from different documents is the information contained in the margins. The span of the margins (of lack thereof) at the ends of the strip can distinguish between different document types even where length tests fail, in case all the documents are printed on a standard size of paper. Because this method begins to stray into the realm of content-dependent grouping algorithms, a command-line flag disables this option in case of a horizontally-shredded text document (which may have irregular right-hand margins, confusing the test).

Clearly, a false positive in an exclusionary test would be disastrous here, so the parameters must be tuned to allow for maximum exclusion without error.

4 **Intra-group Testing**

Once the number of potential pairings has been reduced, some sort of precise matching system is needed to give a definitive answer. My algorithm for this was very simple - walk down both edges in question, testing the pairs of pixels for similarity, and producing an overall similarity score. In the sample data sets used, I noticed frequent noise at the edges of the shreds produced by shredder damage and the shadows added by the scanner. To overcome this, the first two layers of pixels are ignored, and the successive 3 inner rows are averaged to produce a reliable value. This significantly reduced the error produced by shredder edge noise. In addition, since slight misalignment of the shreds can cause fine detail to also become misaligned, and thus not contribute to the similarity score, I used a weighted average of the pixels in the area: 70% for the target pixel, 10% for pixels \((\text{target} + 1)\) and \((\text{target} - 1)\), and 5% for pixels \((\text{target} + 2)\) and \((\text{target} - 2)\). This ensures that even in cases of misalignment there is still a slight impact on the similarity score.

After each pair has been assigned a score, the problem still remains of resolving match conflicts. It is possible, and indeed highly likely, that 2 different shreds will both desire the same partner as their highest-score match. To generate a mapping in the face of this conflict, I insert every score into a priority queue, and begin assigning pairings in order of highest similarity score. Once a shred has a pairing assigned to it, it cannot be claimed by a lower-score shred. This continues until either the mapping is complete, or the scores fall below a given threshold. It is common for certain shreds to have pairless edges if they constitute the edges of the document.
5 Reconstruction

Given this mapping, a reconstructed image can be generated, or at least a series of contiguous chunks in the case of an incomplete deshredding. First, the mapping table is examined for unmatched edges, since these usually coincide with the natural edge of the document. If no unmatched edge is found, it simply begins at shred 0 and prints as many contiguous shreds as possible, then moves on to the next chunk of shreds. To print out a selected shred, I compute the smallest-fit bounding box for the shred (since many shreds are actually slightly curved) and rotate as necessary so that the bounding box is horizontal. If a shred has been assigned as “reversed” by the mapping this is also accounted for during the reconstruction process.

6 Results and Future Work

This method of reconstruction proved very effective for documents with certain properties. Naturally, high-contrast images were highly preferable, since they were more conducive to sharply differentiated similarity scores and more definitive matchings. As shown in Figure 4 (Penguin, Cat) the average shred similarity score over all shreds is 75.84 for the penguin, and 81.97 for the cat, indicating that fewer shred comparisons were optimized out of the cat matching process since it was a more homogenous image.

In addition, images with large features spanning several shreds were much easier to reconstruct, as they proved more resistant to noise than images with copious fine detail. As an example of this, see Figure 2 (Go board) and Figure 3 (Circuit). The large features of the go board result in a complete, contiguous reconstruction, whereas the small, precise features of the circuit photograph result in several discrete chunks (printed in arbitrary order by the reconstruction system).

A further aspect that I discovered while working on text processing was that the distinctiveness of the features, as opposed to the number of features, was extremely important. For example, a document covered in many similar-size circles is feature rich, but may generate many false matches since the set of edge patterns is limited by the uniqueness of the features. In the case of text, many characters are of similar height and generate identical edge patterns when cut in half by the shredding process: for an example of this problem, see the red-text shreds marked with arrows in Figure 5. Despite being an obviously incorrect match, this pairing generated a near-perfect similarity score. Even when tested using computer-generated (noise-free) data sets, small text proved very resistant to these matching techniques because they constitute both fine detail and non-distinct features. When either the size of the character increased, or the size of the shred decreased, such that a character spanned across several shreds, reconstruction became more successful (see green text image of Figure 5).

Due to the existence of the length and margin grouping methods, using the system on multi-document shred groups required no adaptation, and in sets with very distinct documents few additional matches were tested compared to running the shred sets separately. Even in the case of similar document sizes and margins the segment averaging algorithm successfully separated the documents.

Although the purpose of my research was to develop content-independent methods for shred matching, this system could certainly benefit from optional content-specific components. Particularly in the realm of text processing, the addition of character recognition techniques could both reduce search complexity by determining the correct orientation of the shred, and remedy the uniformity of text edge patterns by classifying letter fragments and matching based on that criteria. However, even without the addition of content-specific functionality, this reconstruction system speeds up the painstaking process of reconstructing shredded documents by hand by suggesting a small subset of potential matches for a human to evaluate.
7 References


Figure 2: Go Board: Input files and output image
Figure 3: Circuit: Input files and output image
Figure 4: Low contrast vs. high contrast: Average shred similarity over all penguin strips: 75.84, cat strips 81.97
Figure 5: Coarsely shredded characters vs. finely shredded characters