The Deshredder: A Visual Analytic Approach to Reconstructing Shredded Documents

Patrick Butler* Prithwish Chakraborty† Naren Ramakrishan‡

Department of Computer Science and Discovery Analytics Center, Virginia Tech, Blacksburg, VA 24061

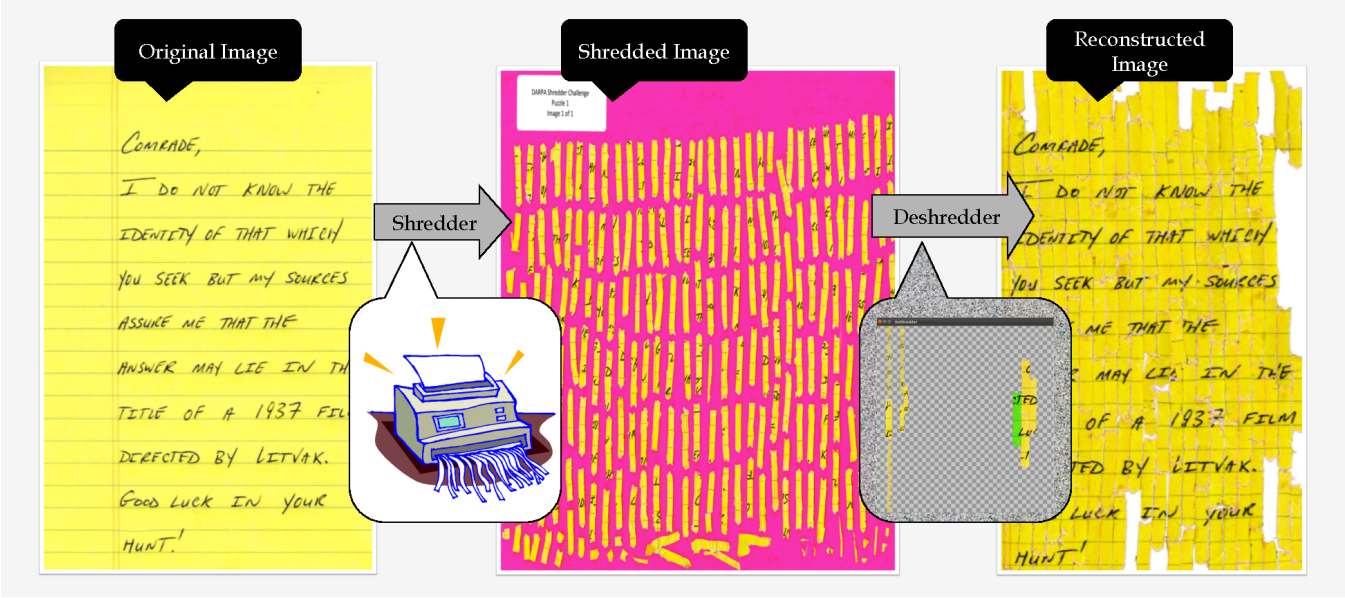


Figure 1: Overview of *Deshredder*. (Left) original document; (Middle) Document shreds to be reconstructed and preview of matchings identified by *Deshredder*; (Right) image reconstructed from the shredded pieces. Some parts of the figure courtesy of [1].

ABSTRACT

Reconstruction of shredded documents remains a significant challenge. Creating a better document reconstruction system enables not just recovery of information accidentally lost but also understanding our limitations against adversaries' attempts to gain access to information. Existing approaches to reconstructing shredded documents adopt either a predominantly manual (e.g., crowd-sourcing) or a near automatic approach. We describe *Deshredder*, a visual analytic approach that scales well and effectively incorporates user input to direct the reconstruction process. *Deshredder* represents shredded pieces as time series and uses nearest neighbor matching techniques that enable matching both the contours of shredded pieces as well as the content of shreds themselves. More importantly, *Deshredder*'s interface support visual analytics through user interaction with similarity matrices as well as higher level assembly through more complex stitching functions. We identify a functional task taxonomy leading to design considerations for constructing deshredding solutions, and describe how *Deshredder* applies to problems from the DARPA Shredder Challenge through expert evaluations.

Index Terms: I.7.5 [Document and Text Processing]: Document Capture—Graphics recognition and interpretation; I.4.7 [Image Processing and Computer Vistion]: Feature Representation—; H.5 [Information Interfaces and Presentation]: User Interfaces—Graphical user interfaces (GUI)

*e-mail: pabutler@vt.edu
†e-mail:prithwi@vt.edu
‡e-mail:naren@cs.vt.edu

1 INTRODUCTION

Putting back shredded pieces of documents into a whole is an important problem in many domains, such as intelligence, security informatics, health informatics, and insurance claim analysis. Engineering a better document reconstruction system enables not just recovery of information accidentally lost but also understanding our limitations against adversaries' attempts to gain access to information. Because significant domain knowledge can be exploited in the reconstruction process, human input is crucial. At the same time, the scale at which shreds have to be processed necessitates automated algorithmic support. It is hence natural to view deshredding as a visual analytics problem where human judgment and automated algorithmic assistance can be fruitfully combined.

However, designing a semi-automatic deshredding system is quite a challenge. A range of research problems manifest, from the need to address imperfections caused by the actual shredding process, to designing best algorithmic strategies for matching shreds, to supporting novel problem solving strategies for human users. The span of backgrounds from which techniques have to be drawn is also broad: image matching, computer vision, and picture reconstruction, to name a few. Here, we present a systematic pipeline of processing stages that address practical issues in analyzing shredded pieces and also provides significant leverage to users in directing the reconstruction process.

We present *Deshredder*, a visual analytic framework for reconstructing shredded documents. Our specific contributions are:

1. A novel time series approach to represent shreds that enable matching both the contours of shredded pieces as well as the content of shreds themselves.

2. Interfaces that support both matching shreds through interaction with similarity matrices as well as higher level assembly through more complex stitching functions.
3. Evaluations with two expert users that successfully illustrate both the capabilities of *Deshredder* and also reveal different problem solving strategies of users.

2 Background and Design Considerations

2.1 DARPA Shredder Challenge

The DARPA Shredder Challenge [1] can be credited with the recent spurt of interest in the deshredding problem. This was a contest where participants were required to re-assemble five sets of shredded documents, with increasing levels of difficulty, and answer specific questions that pertain to the contents of the documents. Different teams adopted widely varying strategies and it is instructive to review these strategies. Interaction with these winning teams enabled us to understand the strategies of those that were placed first, second, third, fifth, and sixth (strategies of the fourth wasn't available.)

The winning solution ('All Your Shreds Are Belong To U.S.') used a custom-coded computer vision algorithm that suggested fragment pairings to human assemblers for verification [1]. Among other techniques, the team employed a scoring mechanism to evaluate their matching algorithm [10]. Also, the team found repeating patterns of yellow dots which, according to the team, were a characteristic of the problem set and used the dots to guide the shred matching stage.

Similar ideas were adopted by the second ('Shroddon'), third ('wasabi') and fifth ('mmbd') placed teams. The 'Shroddon' team also noted that the shredders can shear the papers in depth, and used this information in their final solution [10]. In addition the team searched for the most probable character that can follow the set of characters recognized by the human eye, during the process of shred assembly, through dictionary search. The 'wasabi' team [20] employed a number of computer vision algorithms such as image rotations, trimming, and computed similarity matrices between shreds based on a few simplifying assumptions. These matrices were based on line connections, edge similarity, color similarity, and pen connections. For the initial puzzle sets, the pieces were assembled using picture editing software (e.g., gimp). However, according to [20] a custom assembly tool was used for the latter problems.

An interactive query based system was used by the 'mmbd' team [14]. According to their public webpage [14], this team used features such as line connectivity and stroke connectivity as the basis to connect shreds. A Google Drawing powered interactive framework was next used where best possible matches for the pieces were presented for perusal and selection by the user.

The sixth team ('UCSD') adopted a crowdsourcing approach [7] that was effective at solving the first few problems. This team was plagued by player fraudulence and the designers had to implement a new security feature to qualify users.

2.2 Commercial Tools

The field of commercially available tools to address the deshredding problem is surprisingly sparse. To the best of our knowledge, only one system exists, viz. *Unshredder* [18]. Unshredder is primarily targeted at traditional shredding, while our approach is applicable to cross-cut shredding (traditional shredding cuts in one direction, whereas cross-cut shredding yields parallelogram-shaped pieces). From the limited information available online, this approach makes most of its matches using computer vision algorithms, offering limited opportunities for user interaction. However, multiple users have the ability to collaborate in solving a deshredding problem.

2.3 Functional Task Taxonomy

Before we introduce our Deshredder approach, it is helpful to cast the above deshredding solutions along an axis that emphasizes the 'mixing' between automated analysis and human/visual input:

- Near-automated solutions: Unshredder.
- Backend analysis with visual front-ends: 'All Your Shreds Are Belong To U.S.', 'wasabi' and 'mmbd'.
- Visual analytic frameworks: Deshredder.
- Predominantly manual (e.g., crowdsourcing) approaches: 'UCSD'.

In our visual analytic approach, we emphasize automated algorithms for shred matching, the interactivity enabled by users to critique matches, and most importantly the ability to incorporate user feedback to improve the reconstruction process. It is this closed loop framework that sets Deshredder apart from the other solutions above.

A functional taxonomy of tasks that must be supported by a visual analytic solution involves four aspects as shown in Table 1. Shreds can be represented in numerous ways that emphasize the features crucial for effective matching. Deshredder uses a time series representation that aids in identifying matches quickly with low underlying computational complexity. Second, we must decide on the matching algorithm for comparing shreds and identifying nearest neighbors; here we show how the use of the Chamfer distance measure enables us to expressively take multiple factors into account, including brightness, physical shape, and color. More importantly, it enables the ready incorporation of user feedback. Third, the strategy for shred assembly can be automated to different levels. Here we demonstrate how the use of a similarity matrix visualization enables a truly visual analytic approach that can incorporate user feedback continually in the process to identify and discard bad matches. This enables the user to follow different strategies of interest in reconstructing documents, e.g., 'go for the low hanging fruit first' (i.e., identify easy matches and rule out other fragments based on these matches) or a 'toddler approach' (i.e., identify random matches across the board and then piece them together). Finally, higher-level functions are crucial as the number of shreds increases. We demonstrate the use of features such as constraint propagation, and state capture and reuse, to enable the creation of more complex problem solving strategies.

A comparison of Deshredder with other systems alongside some system features is given in Table 2.

Table 1: Functional taxonomy of the deshredding process.

Shred representation	– How are shreds represented?
	image representations
	time series
Matching algorithm	– How are shreds matched?
	OCR tags
	pen connectivity
	stroke continuity
	Chamfer distance
Assembly	– How are shreds assembled?
	Manual assembly
	Similarity matrix interaction
Workflow tools	– Higher-level reconstruction
	Constraint propagation
	State capture and reuse

3 Related Work

The design of *Deshredder* draws upon work from many related backgrounds, which we survey below.

Computer Vision: Stitching images together based on overlapping regions, i.e., panoramic stitching, is a well studied problem in the field of computer vision [21, 8, 3]. One of the seminal works in this field is attributed to Lowe [17]. This paper proposed SIFT (scale invariant feature transform), an algorithm to detect and describe local features of images, which has become one of the mainstays of major image stitching algorithms. SIFT has been used by Brown et al. [6] for automatic panoramic stitching. In 2006, Ward [32] presented a HDR-photography stitching algorithm and in 2010 Xiong et al. [35] presented a mobile and faster panoramic stitching method. A good review of the major stitching algorithms can be found in Woeste [33]. A key lesson from these works is the choice of feature representation and their capabilities for supporting rapid and accurate stitching. In our work, significant knowledge exists in the detailed shapes and content of pieces which are often non-overlapping

Table 2: Comparative analysis of capabilities of various deshredding algorithms

Criteria	Deshredder	Unshredder	'All Your Shreds Are Belong To U.S.'	'wasabi'	'mmbd'	'UCSD'
Cross-Cut Shreds	Yes	No	Yes	Yes	Yes	Yes
User Collaboration	Yes	Yes	No	N/A	N/A	Yes
Algorithmic Support for matching	Yes	Yes	Yes	Yes	Yes	No
Visual Analytics	Yes	No	No	No	No	No
Applicable to Sensitive Data	Yes	Yes	Yes	Yes	Yes	No

and hence we focus on capturing them in a suitable time series representation. There have been several preliminary attempts at academic solutions to document reconstruction, however they either deal with large pieces [9, 25, 29], non-cross cut shreds [19], or they do not provide user guided iterative machine learning techniques [23]. Some other approaches can be found in [15, 22, 26, 28, 30]

Motif Mining and Time Series Modeling: Concurrent to advancements in image processing tools, there have been extensive research in representing images through 1-dimensional time series and mining motifs from these time series in order to match image objects [4, 11]. These methods work with image boundaries in general and hence do not require overlapping regions to compare two images. Keogh et al [16] in 2006 detailed the process of converting a 2-D shape into a time series and matching two shapes based on the motifs discovered in these time series, allowing for rotations. Yankow et al [36] and Xi et al. [34] extend this framework further to more complex scenarios of matching and similarity search. In 2007, Yankow et al. [37] presented a uniform scaling approach to match differently scaled shapes. Rakthanmanon et al. [24] matched near duplicate figures found in historical documents using a time series approach. Our work is motivated by the above techniques but, as we will show, we require significantly specialized pipelines to accommodate shredded documents.

Similarity matrix interaction and visualization: Similarity matrices are the underlying basis for many data mining and visual analytic algorithms, e.g., clustering [13]. There is significant work on matrix visualization and interaction, in general (e.g., [27]) and similarity matrix visualization, in particular (e.g., [31]). Much of these works are focused around problems like social network analysis, bioinformatics, and network traffic. Here, we demonstrate the use of similarity matrix visualization and interaction as a primary mechanism for users to understand the landscape of possible shred matches and how they can systematically prune this space to identify key shreds that can (or cannot) form important segments in the reconstructed image.

4 Overview

We introduce *Deshredder* by identifying three of its salient themes.

4.1 Representing Shapes as Time Series

The first step in *Deshredder* is to process the individual shreds and extract two time series from each of these shreds. Although it might appear unconventional to represent fragments as time series, as introduced in the related work section there is precedence for such a representation and it provides significant benefits in matching as we shall see. The two extracted time series correspond to the left and right side of the shreds, respectively, and closely follows the relevant contours of the raw shreds as shown in Figure 2. Using a perfectly vertical line as reference, the time series captures the distance of the shred boundary from the vertical line. Subsequently, the time series are used to compute different similarity metrics between shreds which in turn form the basis of the assembly process.

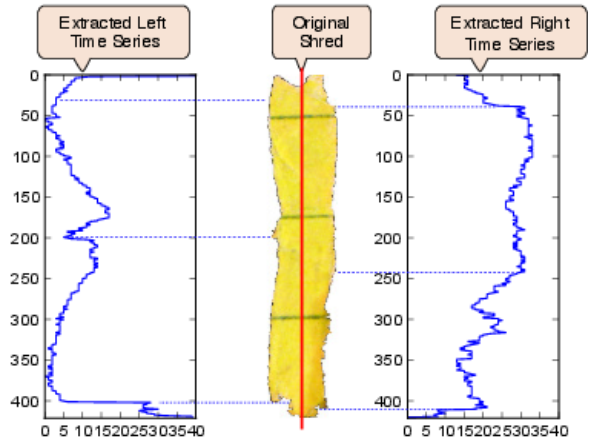


Figure 2: Extraction of left and right time series from a shred.

4.2 Visualizing and Interacting with Similarity Matrices

One of the basic interfaces available in Deshredder is the similarity matrix interaction capability (e.g., see Fig. 4 (A)), which provides a high level overview of the best matches between all pairs of shreds (assuming all possible orientations). It enables the user to spot patterns that occur over large sets. One such useful pattern occurs with blank or nearly blank pieces, which are composed of primarily the background color and therefore yield relatively good matches with most other shreds. This can be seen by prominent rows and columns of blue pixels in Figure 4 . The user can use this view to choose a threshold ('Threshold' slider) of what equates to a good match, and then automatically discount any pieces that have too many good matches ('Max Positives' slider).

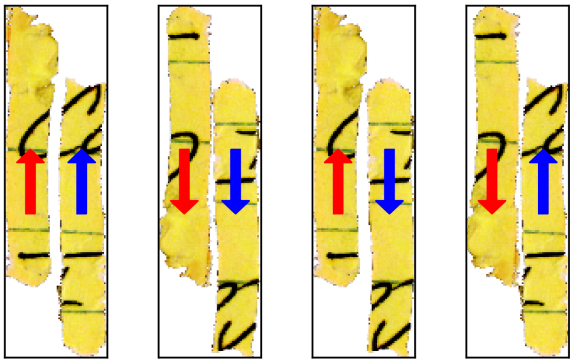


Figure 3: Four possible orientations between two shredded pieces.

4.3 Constraint Propagation via User Interaction

Constraint propagation is an important tool for reducing the number of possible matches to the user. During the process of reconstructing documents in the physical world humans employ constraint propagation in order to reduce the number of choices remaining. Deshredder automatically employs several forms of constraint propagation. The first is enabled when a user matches all possible spaces along a single

piece's side. Deshredder then knows not to show any further matches that include the 'used side' of that piece. Deshredder is also aware of orientation possibilities (see Fig. 3). Once a match is identified, Deshredder will remove from consideration other candidate matches whose orientations do not align with the identified match(es). Furthermore if a user matches an unoriented piece with an oriented piece it propagates the orientation information to the unannotated piece, leading to a cascade of simplifications, that often results in a significant reduction to the number of matches that a user must consider.

4.4 Example workflow

An example of a typical work flow of a user is provided in Figure 4. A user's attention typically focuses on only a few shreds. At the start, it is important to be able to quickly sort through the difficult or uninteresting pieces to find the easiest pieces to match – this is analogous to focusing first on the border pieces in a jigsaw puzzle. In step A of Fig. 4 the user is presented with all possible matches and a visualization of the similarity matrix with known constraints resulting from the chosen match algorithm. As mentioned before, one of the largest sources of false positives is due to the presence of blank pieces that are seemingly good matches to many other pieces. In Deshredder, the user can observe these pieces as creating long rows and columns of blue pixels in the visualization. The user sets the threshold slider and the max positives slider to first decide what makes a good match and then discounts any piece that matches with too many other pieces. In step B, we see the results of discounting these pieces in the similarity matrix, where invalid matches are marked in black and the corresponding matches shown in red. In step C, the user annotates the orientation of several pieces, which has the result of updating the constraints (notice the greater presence of black lines in the similarity matrix view). When the orientations of both pieces in a possible match are known, it allows us to remove two out of four possible matches between the two shreds (see Fig. 3). (When the orientation as well as the match is confirmed, it allows us to remove three out of the four possible matches.) These prunings enable the removal of displayed matches that have incompatible orientations. this removes one of the matches displayed because the match represents two pieces with incompatible orientations. In step D, the user confirms two matches which involve the right side of the shred containing 'V'. This further constrains any matches that might aim to match that side of the shred, but not ones that use the left side of that shred. Furthermore, the orientation annotation is propagated to matched pieces, which again reduces the number of pieces under consideration. In step E, the user confirms two more matches which again propagate their orientations to other pieces, as well as serving to remove matches that overlap the confirmed matches. In step F, the user returns to the overview display to observe the fruit of his interactions.

5 Deshredder in Detail

The overall algorithmic pipeline of *Deshredder* is given in Figure 5. *Deshredder* takes as input a single image containing all the shreds from the original document. Then using computer vision algorithms outlined below we separate each shred into a single image, and apply a straightening algorithm to guarantee that each shred has a known orientation. Next, we apply a feature matching technique to find the best matches between each pairwise set of shreds. The features can be based on shape, brightness, or a particular color. This information is then used in the *Deshredder* interface where a user is presented with the best of matches for a given shred, and visualization of the similarity to see overall progress and to help weed out unhelpful shreds. The user can then choose the correct matches and further eliminate possibilities by annotating the shreds with a correct orientation. *Deshredder* takes the verified matches and orientation information and performs constraint propagation to further eliminate potential matches. If the users wishes, at any time they can change the features to match on, either focusing on a color using the color selector or choosing shape, or brightness. These steps are repeated until the original document is reconstructed. More details are provided below. We now detail each of the stages below.

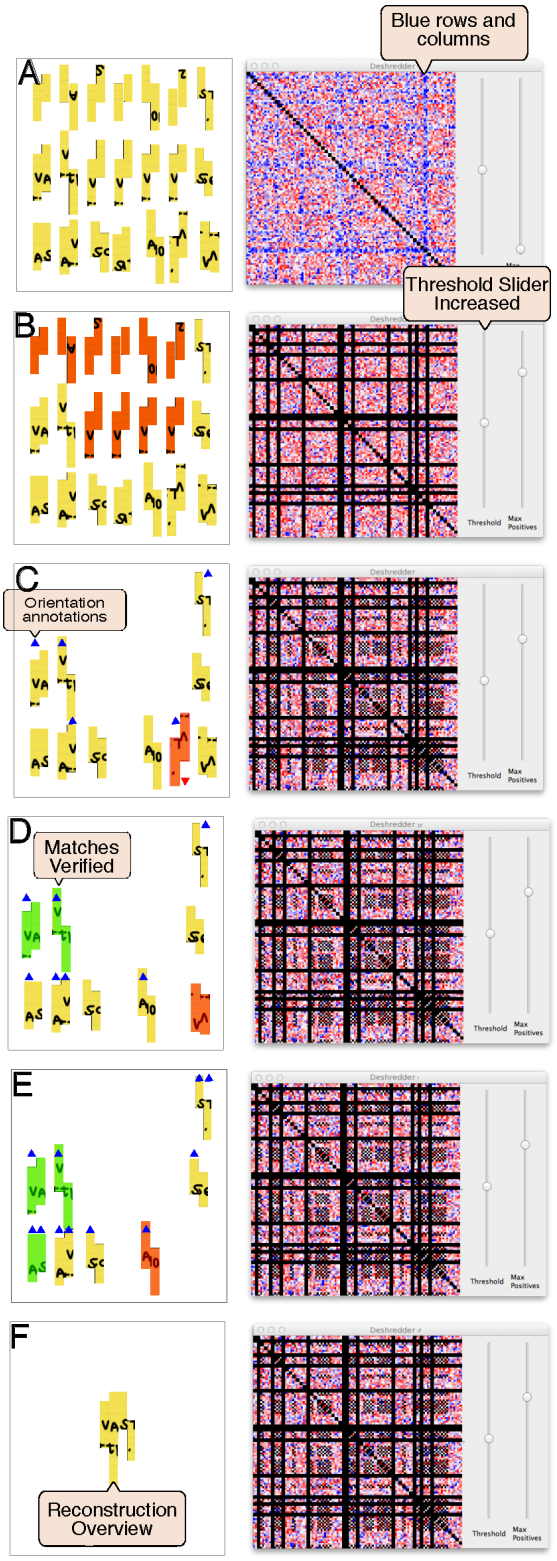


Figure 4: An example Deshredder session of interacting with similarity matrices and propagating identified matches to reconstruct a shredded document.

5.1 Segmentation, Filtering, and Straightening

To begin the reassembly, the first step is to typically place all pieces face down on a scanner and then algorithmically separate them from

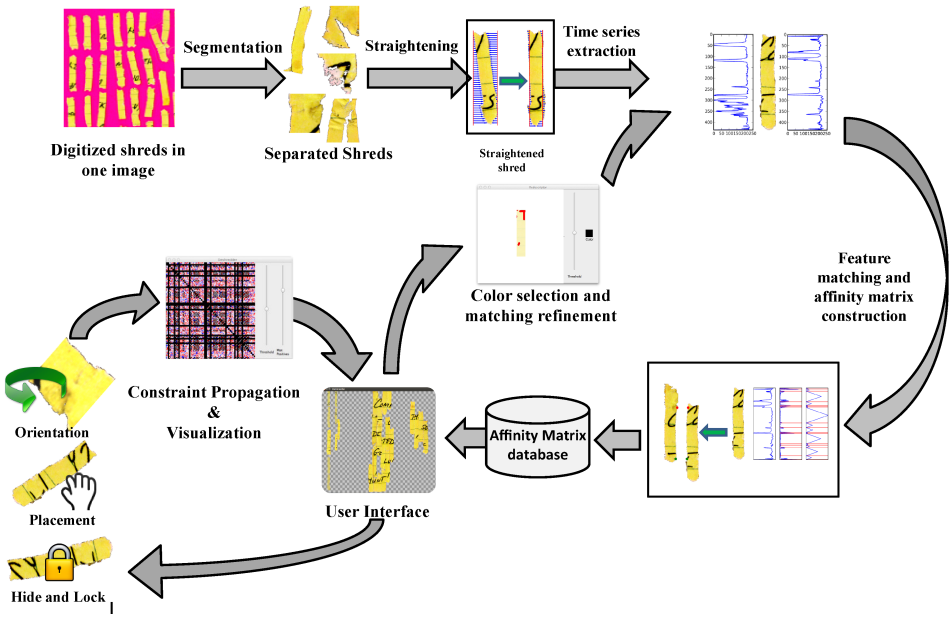
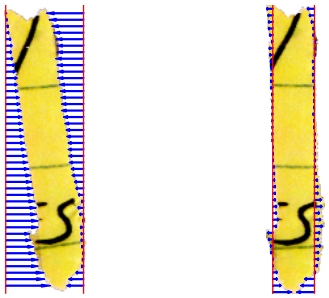


Figure 5: The pipeline of the *Deshredder* algorithm and user interface.

the background. A noise filtering step accounts for many shred defects such as non-straight cuts, bleaching, tearing, and crumpling. This step involves discounting areas near the top and bottom of the edges, and considering an average value of colors near the edges to account for degradations.



(a) (b)

Figure 6: Two methods for determining the correct shred orientation. (a) Maximizing the number of edge pixels. (b) Minimizing the least squares deviation of the edges of the shred. The arrows represent the deviation from vertical.

If it is assumed that the shreds are cut into strips which are taller than they are wider and relatively straight, pieces, there are several simplifying assumptions to help determine the correct orientation. One team ('wasabi'; [20]) in the DARPA Challenge noted that in a perfect rectangle, a correctly oriented shred will have a maximum number of edge pixels oriented along a single vertical line. In practice, however, this is not the case because each piece is not perfectly cut, some cuts can be slightly curved, some may have mangled cuts, and these could lead to errors in using this metric. Instead we build a vertical edge time series as mentioned earlier, in Fig. 2.

Definition 1. *A **Vertical Edge Time Series** is a time series $V(y)$ indexed by row and gives the distance from the left-most or right-most pixel from a perfectly vertical line. For each shred there exists a vertical edge time series for the left and for the right-hand sides ($V_l(y)$ and $V_r(y)$ respectively).*

An image that is well oriented should seek to minimize the average distance between each edge pixel and a vertical line. Therefore in finding the correct orientation for each shred we seek to find the optimum θ^* such that:

$$\theta^* = \arg\min_{\theta} (V(y) - E)^2$$

where E is the average location of the vertical edge that we are trying to orient (Note that $V(y)$ is implicitly a function of θ).

In Figure 7, we study the performance of this approach for Puzzle 1 of the DARPA Shredder Challenge. We can see that our orientation approach performs with $\leq 2\%$ error for the vast majority of shreds; the greater errors are predominantly confined to shreds that are not completely separated from their neighbors. The visual analytics approach described later brings in crucial user input that alleviates some of these outlier cases.

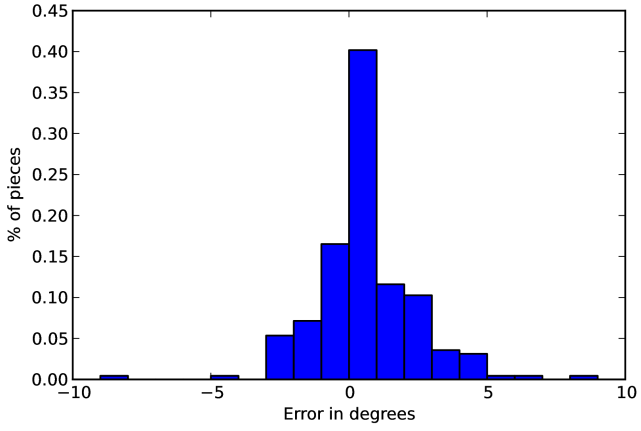


Figure 7: Distribution of straightening errors for Puzzle 1 of the DARPA Shredder Challenge.

5.2 Image Analysis

In matching two pieces together, there are several pieces of information which are available in order to find the optimum matches between them. The first has already been discussed as a tool for finding the correct orientations, viz. the shape of the vertical edges. This information is encoded as the vertical edge time series and used further below. The second piece of information is the content of the shreds themselves. This information is encoded in the Luma (a measure of brightness) time series:

Definition 2. *A **Luma Time Series** is a time series $L(y)$ indexed by row and gives the value of the Luma of the left-most or right-most pixel of that row. For each shred there exists a Luma time series for the left and for the right-hand sides ($L_l(y)$ and $L_r(y)$ respectively).*

Although the datasets we use here contain full color information (and thus three channels of data, one for each primary color), we have found that using Luma [12], defined as $L = .3R + .59G + .11B$ was sufficient.

While both Luma and edge shape information exist for the top and bottom edges, it is important to note that our algorithm focuses on primarily the left and right edges. This is because the cross cut shredding action creates large numbers of deformities on the cross-cuts and often mangle the top and bottom edges. Furthermore, because the top and bottom edges are much smaller, there is less information to match on and consequentially much harder to match effectively.

Each of the two encoded data sources pose different advantages and disadvantages. The vertical edge time series, while extremely useful for jigsaw style pieces that had exaggerated features in the edges, did not by itself provide enough information to identify good matches. It resulted in false negatives when pieces were mangled on one side of a cut but not on the other, or when both pieces were mangled in different ways. Furthermore, false positives were created as a result of cuts being too straight and thus matching every other straight cut. The Luma time series, on the other hand, provides a better source of data, but brought its own problems. The Luma channel allows for matching content but is susceptible to false positives such as regularly spaced features e.g., the background lines on ruled paper, as can be seen in Figure 6. Furthermore, the data provided by the Luma channel is mostly discontinuous, as the features contained in the shredded documents tend to be sharp and distinct. This means that, in comparing two time series, near misses will generate as much error as complete misses.

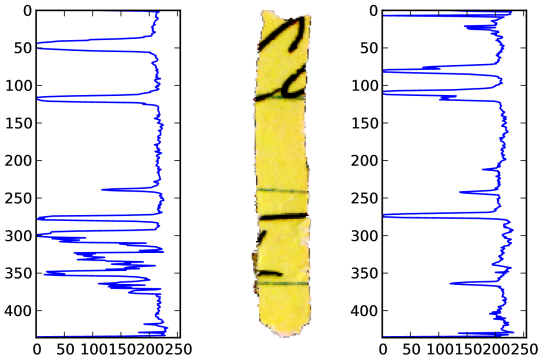
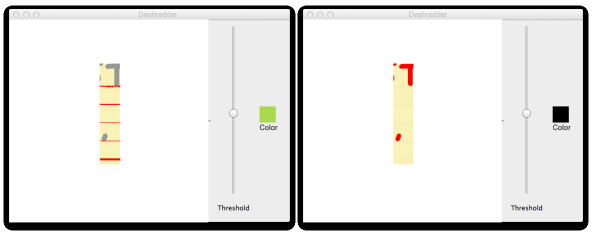


Figure 8: A sample Luma time series for the left- and right-hand sides of the example shred.

5.3 Color Targeting

While by default the Deshredder feature matching algorithm observes the luminosity of the shreds for points of interest to match with, Deshredder also allows the user to filter out a specific color to match pieces against. For instance, in Figure 9a the user desires to begin to match by the colored ruling on the page. The user then uses an eye-dropper tool to select that color from an example piece and



(a) (b)

Figure 9: Deshredder allows a user to guide the matching algorithm to focus on a particular color in comparing shreds.

can use a threshold slider to set the variability of colors to consider. Deshredder responds by highlighting the colors of interest and then rerunning the matching algorithm in the background focusing on the specific colors chosen. In Figure 9b, conversely, the user wishes to match against the black pen color; in response Deshredder highlights the black pen strokes and focuses on matching the pen strokes as it reruns the matching algorithm.

5.4 Matching Shreds

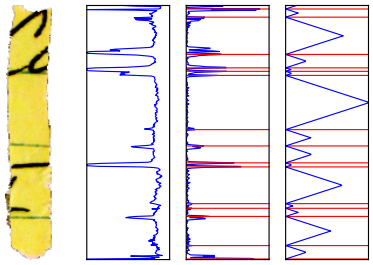


Figure 10: The process for converting an edge, first to a Luma time series, then finding the edges, and finally creating a Chamfer distance distribution. The red lines denote the locations of the features.

While being able to compare shreds based on their content and their shape is useful it is still not sufficient to make good matches. In order to make the best use of the information in the Luma channel, we developed a nearest neighbor matching algorithm based upon the notion of Chamfer similarity, described next.

For each side of each image we build a Luma time series, and from this time series we use a simple one dimensional convolution kernel with weights $[-1, 0, 1]$ to find the edges along the Luma time series. Next, we filter the Luma time series and note the largest peaks in the graph. to find the largest such edges in the image and mark these as our features. Finally, we build a Chamfer distance distribution which denotes the distance of a pixel to the nearest feature. Figure 10 shows each step of the process of building the Chamfer distance distribution [5]. After building a Chamfer distance distribution for each vertical edge of the shred, we can then use these distributions to find the most suitable match between two edges. Suitable matches are found using the Chamfer similarity of two Chamfer distance distributions c_1 and c_2 defined as:

$$\text{ChamferSim}(c_1, c_2) = \frac{c_1 \cdot c_2}{\max(c_1 \cdot c_1, c_2 \cdot c_2)}$$

Here, c_1 and c_2 are distributions defined over the common boundary of the shreds. The entries of these distributions denote distances from the nearest feature in the corresponding shred. $c_1 \cdot c_2$ denotes the scalar (dot) product of the vectors.

We compute the Chamfer similarity four times for every pair of shreds, once for every possible orientation (see Fig. 3). These sim-

(a) (b)

Figure 11: Examples of (a) good and (b) bad matches. The good match has a Chamfer similarity value of .880, while the bad match as a similarity value of .743.

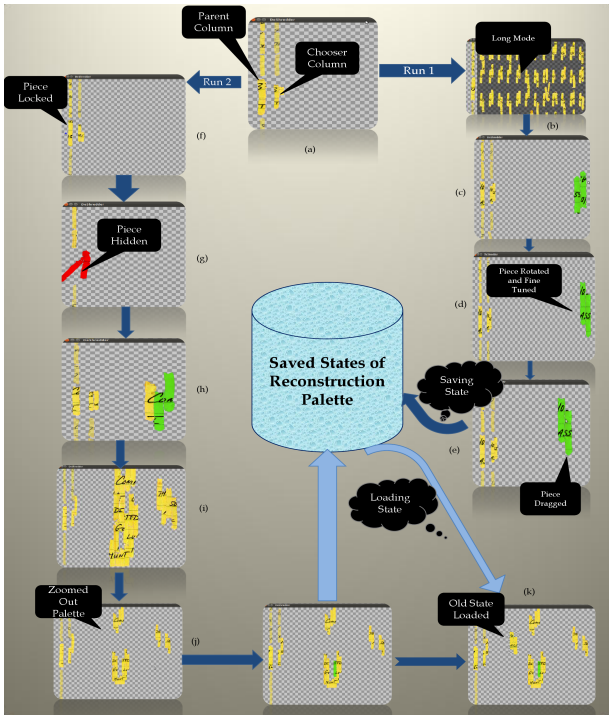


Figure 12: Schematic of a stitching work flow using *Deshredder*. (a) Basic layout of UI; (b)-(e) One run in long mode and demonstrating *reconstruction palette* features;(f)-(j) parallel run using more advanced features of the *reconstruction palette* and use of the *parent column*; (k) merging of two runs. The similarity matrix is a pop-up feature and not shown in the work flow.

ilarity values and the locations for the best similarity in each match are recorded and compiled into the similarity matrix for performing a nearest neighbor search and presenting the best matches to the user. Figure 11 gives examples of good and bad matches as determined by our approach.

6 Stitching Workflows in Deshredder

The *Deshredder* interface, at its core, provides an interactive approach for the user to explore the matches extracted from the algorithmic phase and helps organize them into a composite image. We used a checkered background to provide contrast to the gaps in the shreds. The framework can be broken down into a number of interdependent processes which are explained below:

Match Selector: The matching facility is organized along two columns. The first column (called the *parent column*) helps the user select a specific shred to be matched. The set of best matches identified for this shred are presented as a stream of shreds in the second column (called the *chooser column*). Shreds other than the one(s) being currently matched are grayed out. In the simplest mode a user can select the most appropriate match from the *chooser column*. Figure 12(a) gives an example of this mode of operation. In contrast, the *Deshredder* also provides an advanced mode, called the *long mode*, where the user can choose to view all the matches as an ordered matrix . Example of such an operation is shown in Figure 12(b). On one hand this mode enables the user to quickly scan through all the possible matches to focus on an active shred in the parent column; at the same time it aids the user to be efficient about the match making process by providing semi non-serial access to the matches.Once the user is satisfied with a suggested match, *Deshredder* provides the capability to store this match in a palette to the right (as in Figure 12(c)) referred to as the *reconstruction palette*

Reconstruction Palette: The reconstruction palette provides a number of functionalities to the user, including (1) the ability to rotate the complete match, (2) drag either shred participating in the match for fine tuning of the placement of the match, (3) drag the entire match to another region of the reconstruction palette, (4) delete the match, (5) zoom in/out of the reconstruction palette. We now explain how each of these capabilities are organized into the workflow of a deshredding session. Functionality (1) is required because the alignment of the shreds can be inverted with respect to other matches in the reconstruction palette. Regarding functionality (2), giving the user the ability to fine tune the placement of the match is an important design requirement as this places less stringency on the alignment matching part of the algorithm. It also helps to better incorporate user experience directly into the interface so that future matches for a participating shred can be displayed from this modified placement. An example of the user activating functionalities (1) and (2) can be seen in Figure 12(d). As the user is traversing through the stitching process, he/she may begin to develop some idea about the relative placements of disconnected regions of matches developed till that point of the time. Functionality (3) enables the user to organize the matches in the reconstruction palette to represent these ideas and accelerate the stitching process and an example can be seen in Figure 12(e). Finally, any match chosen by the user need not be the best when placed relative to other pieces and hence functionality (4) enables the user to explore a number of matches from the *chooser column* without forcing the user to adopt stringent requirements on the allowed number of trials. Functionality (5) is perhaps one of the most important feature as the zoom in/out feature allows the user to closely inspect a match by zooming in and also view the entire palette in a modestly-sized screen by zooming out (refer Figure 12(i,j).

Other Features: There are a few additional features we implemented with respect to the *parent column*. Similar to the simple mode of the *chooser*, the *parent column* can also be traversed one-at-a-time in either direction. For each traversal, the active element is changed and the *chooser* reflects the matches corresponding to this element. An advanced feature of this column is a 'hide' mode (Figure 12(g)) in which the user can decide on a piece to be trivial and hide it from all the suggested matches in the *chooser column* (for all the remaining matches in the current run). Further, in this column one can rotate a piece and this state can be locked. This feature was implemented as it took away the burden of trying to suggest orientations from the user and instead enables the user recognize the orientation most suitable to him/her (Figure 12(f)). Also, when the active element in either the *parent* or the *chooser* column matches a

piece, already matched and saved in the *reconstruction palette*, the pieces in the palette are highlighted(Figure 12(h)). Finally, we implemented a state save-loader mechanism, whereby a user can save his current state of the reconstruction palette and load it for future runs. This is a very important feature as it enables the user to safely keep track of the progress made and at the same time allows collaborative stitching with other users. This feature also opens up the possibility of parallelism with respect to the user as a team can divide up the pieces to be matched among themselves and which can be merged at a later time (Figure 12(k).)

Aside from these features, *Deshredder* also provides tools like color selector (Figure 9) and similarity matrix visualizer (Figure 4) which are designed to pop-out on user command.

All of the above features can be used by a user any number of times in a stitching session. One such sample run where the user is able to reconstruct part of the document is shown in Figure 12 which depicts all the functionalities being used to make the match.

7 Results

We describe results of using *Deshredder* on Puzzle 1 of the DARPA Shredder Challenge. From the image containing the shreds (Figure 1), 225 pieces were extracted for reconstruction. In the challenge some questions about the document were asked which could be answered only after proper reconstruction [1].

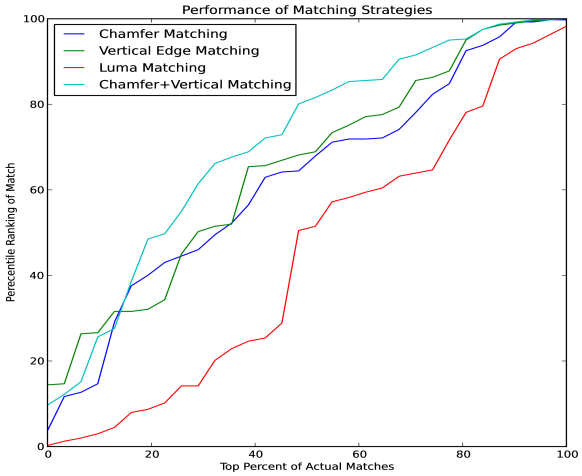
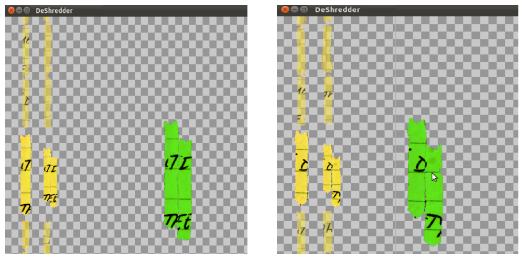


Figure 13: Performance of matching strategies in DARPA Shredder Challenge Puzzle 1 [1].

First, we outline some quantitative results to assess the effectiveness of our matching algorithms. For each edge, we considered all possible matches, and then compared where each correct match occurred in the ordered ranking for each edge. We used four different evaluations for the matches: 1) Chamfer similarity, 2) the RMSE of the vertical edge time series of the match, 3) the RMSE of the Luma time series of the match, and 4) a combination of 1) and 2). Of these, the combination metric worked the best. The graphical results of this exercise are depicted in Figure 13.

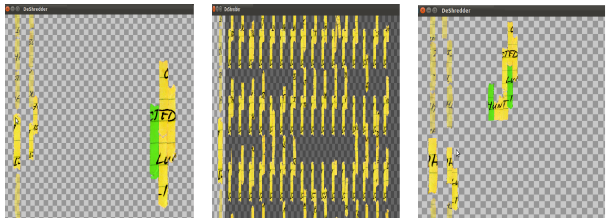
The Luma evaluation performed the worst because as stated above, the discontinuous nature of the Luma time series function decreases the accuracy of the evaluation. Observing the behavior of the Chamfer+Vertical line we see that 50% of the correct matches will show up in the first 20% of the recommendations, ideal for quick visual inspection by analysts.

Deshredder was then used by two experts to reconstruct the first puzzle from scratch. One expert was a trained knowledge discovery professional. The second expert was a practicing professional in the national security industry. We outline the overall strategies employed by them respectively.



(a) (b)

Figure 14: Expert user rejecting apparently good matches. (a) The words formed are meaningless, or (b) the words seem out of context.



(a) (b) (c)

Figure 15: Stitching strategy used by the Expert. (a) Stitching shreds to a single region, (b) using the long mode while stitching the image (c) avoiding matching an already matched piece

7.1 Expert 1 Strategy

The knowledge discovery expert usually favored shreds that have traces of handwriting in them rather than empty shreds of legal pad. Nevertheless, apparently good matches (with respect to the continuation of boundaries) were rejected by this user because the words formed either did not make sense or appeared out of context. An example run is shown in Figure 14. In (a), the words are evidently meaningless and are rejected by the user although the boundaries match quite nicely. In (b), a single 'D' appears in the text and hence was rejected by the user as being an out-of-context match. In this respect, an OCR subsystem to suggest words could be useful but may add increased lag due to dictionary search.

In terms of 'positive' strategies, once confident of a particular match, this expert usually chose to extend the matches to regions by choosing the shred from the *chooser column* as the new *parent column*, rather than finding disconnected groups of matches. Further, while finding the matches, this expert usually chose to view the best suggested matches for a shred by scrolling rapidly via keyboard options. Based on the observations, the expert decided on whether to look into such matches or move to the next shred. Some of the aspects of the *Deshredder*, such as the highlight feature, helped the expert to keep track of the pieces and avoid spurious matches. Illustrations of these behaviors are shown in Figure 15. The suggested matching regions between two shreds were also in general acceptable and only 17 percent of time dragging a suggested match for a more favorable boundary match was required.

7.2 Expert 2 Strategy

In contrast to the strategy of expert 1, the security expert employed some subtly different ones. This expert really preferred certain features of the *Deshredder* such as the hiding mode by which he was able to hide blank shreds. Also the linear search seemed more acceptable to this expert than the *long mode*. This expert felt more comfortable organizing his thoughts and looking into the details via the *simple mode*. The strategy employed was to match up via lines on pages and then via words. The expert was also seeking distinguishing features of the shreds to make the match. This expert suggested some features such as matching more than two pieces at a time and reorganizing the lines of the strip to retain every piece in the same orientation. The concept of the 'long mode' was appealing in principle, but practically rarely useful, to this expert.

8 Discussions

We outline below some of the lessons learned from our overall project leading to possibilities for future work.

8.1 Lessons Learned

The working model of the *Deshredder* described here was arrived at after a number of missteps-steps which led to the current design considerations. As described in the paper, the *Deshredder* works by comparing the boundaries of two image fragments and inferring best possible matches based on their similarities. Recall that this matching process was implemented by modeling the boundaries as time series (Section 5) and then finding the best possible matches based on a modified Euclidean distance.

Choice of fragment encoding: There are a number of ways described in the literature [16, 37, 24] for converting a image boundary into a time series. One approach that we initially attempted was to encode the image boundary as a time series based on the radial distance of the boundary from the image center. However, a number of the image shreds were rotated [1] and hence rotation invariant strategies [16] were necessary. To understand the performance of this strategy in comparison to our model of time-series extraction we describe the results of the said strategy on a sample image in Figure 16. The blue image matches the red image exactly along a boundary as shown in the Figure 16 (left). Figure 16 (right) shows the best placement of the time series for the matching part of the two images, computed as described in [16].The dissimilarity of these time-series for perfectly aligned pieces as seen in the figure, indicates these methods as unsuitable for use as a similarity measure.

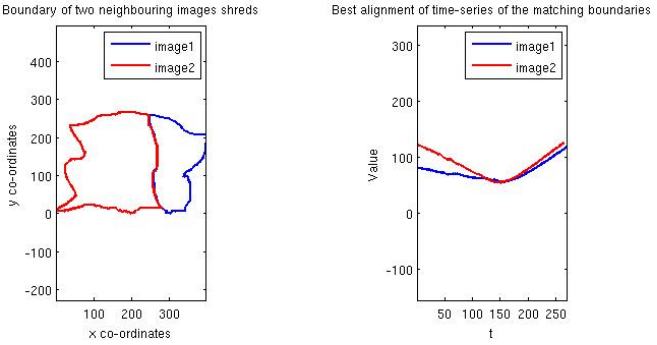


Figure 16: Matching edges using the time series modeling approach of [16].

This lack of accuracy in similarity detection could be attributed to the difference in scales of the two images. Scale dissimilarity leads to radial distortions and if such differences in scales are appreciable, the increased dissimilarity between two such time series will lead the nearest-neighbor algorithm to reject the shred in favor of a possible non-matching but similarly scaled shred. A scale invariant approach was outlined in [37]. However, this method incorporates additional computational complexities. Our proposed method of abstracting out the vertical edges is scale- and rotation- invariant in the sense that the figures in the DARPA Shredder Challenge appeared to be shredded in average in a rectangular manner. Furthermore, comparison of such vertical edges is relatively faster as the possible orientation space has been reduced to 4 combinations for a pair of shreds.

Fragment matching strategy: Reconstruction of original images from the shred was found to be sub-optimal when the time-series was encoded based on the distance of the boundary from the linear edge alone. For example, image 1 of [1] appears to be a handwritten note on a legal pad. Hence, two lowest distance matched edges may result in a reconstruction where the edges are non-continuous with respect to the colors. As such, the distance metric was combined with the color information obtained via a variation of neighboring pixel intensity, to produce a color-aware time-series.

Visual analytics vs. fully automated strategies: Before adopting a visual analytic approach, we evaluated a fully automated method of stitching. A string-based physics model, where the similarity between the time-series representation of two images shreds were used as the attraction measure, was fitted on the shredded images. The aim was to automatically reduce the distance between neighbor-shreds and increase the distance between non-neighbor shreds. This induces a multi-level interaction between image shreds and to properly define such interactions, similarities should be recomputed as we combine image shreds into subunits. As the number of initial shreds is quite large, the number of comparisons required at each level quickly grows exponentially. This method further necessitates the similarity measures to have high fidelity. The visual analytic approach of *Deshredder* however can tolerate a high degree of distortion in matches and the matching process can be done only for the pair of shreds, thus being computationally more tractable.

Interface Design: For the interface we started with the 'long mode' as the default view to an user. However, we realized quickly that 'simple mode' helps a user to focus on the current piece more intently than the 'long mode'. But at the same time the 'long mode' offered the possibility of glossing over multiple recommendations very quickly. After debating a number of possibilities for integrating these two modes such as a separate window, a pop out, or a window overlay, we decided on the final design based on expert user suggestions. In the similarity matrix and the color chooser we decided on an integrated approach to have a consistent look and feel of the software.

8.2 Future Work

Deshredder brings together the fields of time series representation, computer vision, and visual analytics to assist users in reconstructing shredded documents. It is our intent to further develop *Deshredder* into a complete problem solving environment for reconstructing shredded documents. One of the most important aspects of the system is creating good metrics to choose and evaluate matches. We would like to evaluate and develop metrics to further improve the suggestions that the algorithm makes. A second area which we desire to address is the need for a rotationally invariant matching. Our current techniques focus on the vertical edges only, and developing a rotationally invariant matching algorithm would enable us to provide more effective matching for differently shaped shreds as well as avoid the straightening step. Also, akin to Digistrips [2], a gesture and touch-screen driven interface for assisting air traffic controllers in schedule management, we aim to evaluate *Deshredder* on a large high definition touch screen display and evaluate touch-based and gesture-based techniques for user input. Finally, we would like to conduct a complete user study and articulate usability concerns.

9 Conclusion

Automatic deshredding of documents makes a powerful argument for a visual analytics strategy, as we have shown here. *Deshredder* combines the advantages of automated methods (in identifying possible matches) and enables user input (to drive the reconstruction process). Our goals in this paper were to articulate the many design decisions that we have made along the way and catalog them so that others working in this space can build upon these strategies. As we explore larger shredding puzzles using *Deshredder*, we believe the need for interesting problem solving strategies would become paramount, which will hopefully spur more research into visual analytic methods.

Acknowledgements

This work is supported in part by US NSF grant CCF-0937133 and the Institute for Critical Technology and Applied Science (ICTAS), Virginia Tech.

REFERENCES

[1] DARPA Shredder Challenge. http://archive.darpa.mil/shredderchallenge/, Last Visited: March 31, 2012.

[2] Digistrips : humaniser les interfaces. http://pii.tls.cena.fr/index.php?Itemid=7&id=10&lang=en&option=com_content&view=article, Last Visited: June 23, 2012.

[3] A. Agarwala, M. Dontcheva, M. Agrawala, S. Drucker, A. Colburn, B. Curless, D. Salesin, and M. Cohen. Interactive Digital Photomontage. *ACM Transactions on Graphics*, 23(3):294–302, 2004.

[4] Z. A. Aghbari. Effective Image Mining by Representing Color Histograms as Time Series. *Journal of Advanced Computational Intelligence and Intelligent Informatics*, 13(2):109–114, 2009.

[5] H. G. Barrow, J. M. Tenenbaum, R. C. Bolles, and H. C. Wolf. Parametric Correspondence and Chamfer Matching: Two New Techniques for Image Matching. In *Proceedings of the 5th International Joint Conference on Artificial intelligence*, volume 2 of *IJCAI'77*, pages 659–663, 1977.

[6] M. Brown and D. G. Lowe. Automatic Panoramic Image Stitching using Invariant Features. *International Journal of Computer Vision*, 74(1):59–73, 2006.

[7] M. Cebrin. UCSD DARPA Shredder Challenge Team. http://shredder-challenge.ucsd.edu/login.php, Last visited: June 23, 2012.

[8] S. Dasgupta and A. Banerjee. A Real-time Panoramic Vision System for Autonomous Navigation. In *Proceedings of the 36th Conference on Winter Simulation*, WSC '04, pages 1706–1712, 2004.

[9] M. Diem, F. Kleber, and R. Sablatnig. Document Analysis applied to Fragments: Feature Set for the Reconstruction of Torn Documents. In *Proceedings of the 9th IAPR International Workshop on Document Analysis Systems*, DAS '10, pages 393–400, 2010.

[10] T. Geller. DARPA Shredder Challenge Solved. http://cacm.acm.org/magazines/2012/8/153812-darpa-shredder-challenge-solved/fulltext, Last visited: August 1, 2012.

[11] M. H. A. Hijazi, F. Coenen, and Y. Zheng. Image Classification using Histograms and Time Series Analysis: a Study of Age-related Macular Degeneration Screening in Retinal Image Data. In *Proceedings of the 10th Industrial Conference on Advances in Data Mining: Applications and Theoretical Aspects*, ICDM '10, pages 197–209, 2010.

[12] International Telecommunications Union. *Studio Encoding Parameters of Digital Television for Standard 4:3 and Wide-Screen 16:9 Aspect Ratios*, 2011.

[13] A. K. Jain, M. N. Murty, and P. J. Flynn. Data Clustering: A Review. *ACM Computing Surveys*, 31(3):264–323, Sept. 1999.

[14] B. Jou, H. Li, M.-H. Tsai, D. Pei, F. Marques, and S.-F. Chang. DARPA Shredder Challenge 2011. http://www.ee.columbia.edu/ln/dvmm/shredder/, Last Visited: June 23, 2012.

[15] E. Justino, L. S. Oliveira, and C. Freitas. Reconstructing Shredded Documents through Feature Matching. *Forensic Science International*, 160(2-3):140 – 147, jul 2006.

[16] E. Keogh, L. Wei, X. Xi, S.-H. Lee, and M. Vlachos. LB_Keogh Supports Exact Indexing of Shapes Under Rotation Invariance with Arbitrary Representations and Distance Measures. In *Proceedings of the 32nd International Conference on Very Large Data Bases*, VLDB '06, pages 882–893, Sept. 2006.

[17] D. Lowe. Object Recognition from Local Scale-Invariant Features. In *Proceedings of the 7th IEEE International Conference on Computer Vision*, volume 2 of *ICCV '99*, pages 1150–1157, 1999.

[18] D. Lowe. Unshredder: Shredded Document Reconstruction System. http://www.unshredder.com/, Last vsited: March 31, 2012, 2007.

[19] M. A. O. Marques and C. O. A. Freitas. Reconstructing Strip-shredded Documents using Color as Feature Matching. In *Proceedings of the 2009 ACM Symposium on Applied Computing*, SAC '09, pages 893–894, 2009.

[20] M. Newlin. No Snow, No Pants.: You should Probably Start Burning your Mail: What I Learned from the DARPA Shredder Challenge. http://www.marcnewlin.com/2011/12/you-should-probably-start-burning-your_02.html, Last Visited: June 23, 2012.

[21] T. Ozawa, K. M. Kitani, and H. Koike. Human-centric Panoramic Imaging Stitching. In *Proceedings of the 3rd Augmented Human International Conference*, AH '12, pages 20:1–20:6, 2012.

[22] M. Prandtstetter and G. R. Raidl. Combining Forces to Reconstruct Strip Shredded Text Documents. In *Proceedings of the 5th International Workshop on Hybrid Metaheuristics*, HM '08, pages 175–189, 2008.

[23] M. Prandtstetter and G. R. Raidl. Meta-heuristics for Reconstructing Cross Cut Shredded Text Documents. In *Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation*, GECCO '09, pages 349–356, 2009.

[24] T. Rakthanmanon, Q. Zhu, and E. J. Keogh. Mining Historical Documents for Near-Duplicate Figures. In *Proceedings of the 2011 IEEE 11th International Conference on Data Mining*, ICDM '11, pages 557–566, 2011.

[25] S. A. SantoshKumar and B. K. ShreyamshaKumar. Edge Envelope based Reconstruction of Torn Document. In *Proceedings of the Seventh Indian Conference on Computer Vision, Graphics and Image Processing*, ICVGIP '10, pages 391–397, 2010.

[26] C. Schauer, M. Prandtstetter, and G. R. Raidl. A Memetic Algorithm for Reconstructing Cross-cut Shredded Text Documents. In *Proceedings of the 7th international conference on Hybrid metaheuristics*, HM'10, pages 103–117, 2010.

[27] Z. Shen and K.-L. Ma. Path Visualization for Adjacency Matrices. In *EuroVis07 Joint Eurographics - IEEE VGTC Symposium on Visualization*, pages 83–90. Eurographics Association, 2007.

[28] A. Skeoch. *An Investigation into Automated Shredded Document Reconstruction using Heuristic Search Algorithms*. PhD thesis, University of Bath, 2006.

[29] A. Stieber, J. Schneider, B. Nickolay, and J. Krüger. A Contour Matching Algorithm to Reconstruct Ruptured Documents. In *DAGM Conference on Pattern Recognition*, pages 121–130, 2010.

[30] A. Ukovich, G. Ramponi, H. Doulaverakis, Y. Kompatsiaris, and M. Strintzis. Shredded Document Reconstruction using MPEG-7 Standard Descriptors. In *Signal Processing and Information Technology, 2004. Proceedings of the Fourth IEEE International Symposium on*, pages 334 – 337, dec. 2004.

[31] J. Wang, B. Yu, and L. Gasser. Classification Visualization with Shaded Similarity Matrix. Technical report, GSLIS University of Illinois at Urbana-Champaign, 2002.

[32] G. Ward. Hiding Seams in High Dynamic Range Panoramas. In *Proceedings of the 3rd Symposium on Applied Perception in Graphics and Visualization*, APGV '06, pages 150–150, 2006.

[33] H. Woeste. *Mastering Digital Panoramic Photography*. Rocky Nook Series. 2009.

[34] X. Xi, E. Keogh, L. Wei, and A. Mafra-Neto. Finding Motifs in Database of Shapes. *SDM '07*, 2007.

[35] Y. Xiong and K. Pulli. Fast Panorama Stitching for High-Quality Panoramic Images on Mobile Phones. *IEEE Transactions on Consumer Electronics*, 56(2):298–306, 2010.

[36] D. Yankov and E. Keogh. Manifold Clustering of Shapes. In *Proceedings of the Sixth International Conference on Data Mining*, ICDM '06, pages 1167–1171, 2006.

[37] D. Yankov, E. Keogh, J. Medina, B. Chiu, and V. Zordan. Detecting Motifs Under Uniform Scaling. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '07, pages 844–853, 2007.