Date of acceptance Grade

Instructor

Semi-automatic solving of "Jigsaw puzzles" for material reconstruction of Dead Sea Scrolls

Tommi Jalkanen

Helsinki June 3, 2017

UNIVERSITY OF HELSINKI
Department of Computer Science

HELSINGIN YLIOPISTO — HELSINGFORS UNIVERSITET — UNIVERSITY OF HELSINKI

Tiedekunta — Fakultet — Faculty	Laitos — Institution — Department
Faculty of Science	Department of Computer Science

Tekijä — Författare — Author
Tommi Jalkanen

Työn nimi — Arbetets titel — Title

Semi-automatic solving of "Jigsaw puzzles" for material reconstruction of Dead Sea Scrolls

Oppiaine — Läroämne — Subject
Computer Science

Työn laji — Arbetets art — Level	Aika — Datum — Month and year	Sivumäärä — Sidoantal — Number of pages
	June 3, 2017	28 pages + 0 appendices

Tiivistelmä — Referat — Abstract

Digital solving of jigsaw puzzles have been well researched throughout the years and multiple approaches to solve them have been proposed. But these approaches have not been applied to reconstructing ancient manuscripts out of transient material such as leather or parchment. The literature describes ways to reconstruct ancient artefacts but they describe the process for more durable objects like pottery. In this thesis we explore the usability of the existing state-of-the-art methods for the purpose of aiding reconstructing of the Dead Sea Scrolls, also known as Qumran scrolls. Our experiments show that the existing methods as such do not provide good results in this domain, but with modifications provide help through a semi-automated reconstruction process. We expect these modifications and the software that was created as a by-product of this thesis to ease the researchers' work by automating the previously laborious manual work.

ACM Computing Classification System (CCS):
Computer vision,
Computer vision representations,
Computer vision problems

Avainsanat — Nyckelord — Keywords
Reconstruction, Matching

Säilytyspaikka — Förvaringsställe — Where deposited

Muita tietoja — övriga uppgifter — Additional information

Contents

1 Introduction

The term jigsaw puzzle refers to a puzzle of assembling shaped pieces by tiling them together to form a picture [FG64]. Material reconstruction of discovered ancient artefacts is a special case of a jigsaw puzzle, where some of the pieces might be damaged or missing [WC08]. The reconstruction process of these artefacts is still mainly done manually [Paj13], and as such it is a potent field of research for computer-assisted problem solving.

Fully automated solution to the problem can probably not be achieved from solely pictorial data, because the new findings have to be verified with the help of the original fragments [Paj13]. Therefore, the approach in this thesis will be done in a semi-automated manner, in which the program forms preliminary suggestions that are then validated by human experts. This kind of semi-automated solving process will help the researchers as it is very likely for a human to miss relevant matching pieces in a huge pictorial data set.

2 Working Methods

The thesis will be made in collaboration with Dr. Jutta Jokiranta from the Academy of Finland Centre of Excellence Changes in Sacred Texts and Traditions. The collaborative nature of the thesis is also a major motivation for the thesis to be written in English. There are researchers outside of Finland working on reconstruction of scrolls, and that is why opening up the results of this study might prove useful for them as well.

The thesis consists of two separate parts: algorithms for solving the problem and implementation of a graphical user interface. The main focus will be on the technical side of the thesis such as finding suitable algorithms and optimizing them for the specific task at hand. Likewise, the algorithmic solving process is also split into two separate tasks. First, the program has to sort the fragment pieces into groups based on how well they resemble each other, in other words, how likely the pieces are to be located near each other. A proper similarity measure needs to be defined for the resemblance between fragment pieces, and as such is an important subtask in the thesis project. The data will be obtained from open databases and Dr. Jokiranta. The second task involves making the actual suggestion for a reconstruction.

The GUI (Graphical User Interface) is there to provide means for the researchers to

utilize these tools in their research. This piece of software will be developed using standard agile methodologies that are prevalent in the IT industry. This will help to maximize the usability and usefulness of the software for the scientists that are going to be using it in their research.

3 Reconstruction Process

Thousands of fragments from various artefacts are found from excavation sites worldwide. These objects play a big role in trying to reconstruct history. However, manual assembly of these artefacts is laborious and digital assistance is a rarity at least when it comes to the Dead Sea scrolls. To further complicate things some fragments are found together with other fragments that might not even belong to the same manuscript. Digital processing of these fragments would speed up the process by being able to go through the pieces much faster. Also a program might even be able to suggest pairings that scientists hadn't even considered before. With the help of a computer we could aid the classification process and at the same time get notifications for possible misclassifications in the already manually classified datasets.

3.1 Traditional Reconstruction

In 1990 Stegemann [Ste90] shared his 12-step method for reconstructing scrolls. The first step involved gathering all fragments, that are from the top or bottom of a column, of a scroll. Then all fragments that exhibit signs of a right or left margin of a column, sewing seams or column to column transitions are to be gathered. Third, all the places with uninscribed drylines, vacat-lines or transitional devices in the text should be noted. Next information in the edition about the location of the beginning of the text as well as which other fragments were found in the vicinity and their location compared to the fragment in question are searched. Then all the pieces will be gone through looking for similar features or shapes of recurring damage. At this point a definition for an average width or limit for the columns is attempted. After this also an estimate for the number of lines in each column can be produced. At this point to ease the reconstruction the roll up direction of scroll needs to be confirmed or otherwise both directions have to be tested. By looking at the overall appearance of the pieces we can try to subgroup pieces that seems more similar to each other than the rest. Another way to aid placing pieces to specific

parts of the scroll is to look for differences in distances between lines, heights of letters, traces of the pen and the flow of ink. Then starting from the larger pieces and groups of similar fragments all parts of the scroll are assembled to their places. Finally, a schematic drawing of the scroll is prepared from the evidence supported by the remains.

In the same publications Stegemann also shares his method for finding fragments with similar shapes. First two photocopies are created from a printed edition of the fragments. Then the edges, gaps and breaks are sharpened on those photocopies with a pencil. Finally similar shapes are sought after by placing the copies on top of each other in front of a light source and moving the copies around.

As can be seen from the two introduced methods, the reconstruction process contains many manual steps that could be aided with digital automation. According to Stegemann reconstructing around 12 columns may take several weeks and further publication as well as writing work can additionally take several months. The amount of this process takes can be reduced by digitally looking for similar shapes, clustering the fragments to groups and finding matching parallel texts, the latter of which is not as beneficial when it comes to Qumran scrolls, because about half of the findings represent previously unknown manuscripts and the ordering of the passages cannot be assumed to be the same as in the parallel texts [Ste90].

3.2 Related Work

The process of reconstructing ancient manuscript is related to that of solving a jigsaw puzzle. Freeman and Garder [FG64] in 1964 were the first to introduce the problem of digitally solving a jigsaw puzzle. Their main focus was to propose techniques for solving apictorial puzzle where only the shape of the object counted. All the way to recent years many have followed suite to further research aspects of digitally solving jigsaw puzzles such as in [RB82, WSKL88, CFF98, GMB02, DD07, NDH08]. Wolfson et al. [WSKL88] proposed combinatorial optimization and curve matching to solve the problem. Goldberg et al. [GMB02] used a similar approach but extended the solution to cover pieces with more than four neighbours and to enable solving bigger puzzles. After much research in trying to find an efficient solution to the problem, Erik Demaine and Martin Demaine [DD07] showed that solving a jigsaw puzzle is a NP-complete problem.

In 1998 Chung et al. [CFF98] introduced a pictorial variant for jigsaw puzzles where

the image features of puzzle pieces are also taken into account in addition to shape information. They also described three new algorithms solving this variation on the puzzle, namely TSP&K best-based, TSP&AP-based and AP-based algorithms. Nielsen et al. [NDH08] proved that it is possible to produce a solution solely based on image information by using an edge matching approach. Further studies expanded the image based solutions to include surface texture and picture based reconstructions.

Research in the field has also been applied to reconstruct documents that have been torn or shredded. A global approach to assembling shredded documents was proposed by Zhu et al. [ZZH08]. Curve matching ambiguities were handled in their case by adding a definition for neighbouring match compatibility. The need for putting together hand torn documents have also been raised in the field of forensics, where Justino et al. [JOF06] demonstrated the feasibility of a simple multifeature matching scheme. SVM classifiers were introduced to find corresponding points between fragments by Richter et al. [RRCL13]. Their solution uses both curvature and image features to solve the puzzles while also retaining good results when some pieces were removed randomly from the set. Cao et al. [CLY10] backed up the importance of merging shape and appearance information. Their scheme incorporated the possibility for material loss in the pieces which is highly relevant as well in a scroll reconstruction process.

Some work has already been done to solve the problems researchers face when dealing with reconstruction of ancient artefacts. Projects in [CWA+01, PPE+02, KTN+06] represent real-world applications for restoring pots, wall paintings and marble constructs. In the field of two-dimensional ancient artefact reconstruction, curve matching based works in [dGLS02, MK03] represent the state-of-the-art. Kleber and Sablatnig [KS09] have published a survey summarizing relevant techniques used in the field of object reconstructions. Research on restoration of scrolls, made out of more transient material such as parchment or leather, are under-represented.

4 Jigsaw Puzzles

Jigsaw puzzle refers to a pattern recognition problem, where the end goal is to assemble a single finished puzzle with the help of shape and pictorial information of the puzzle pieces. The core difficulties arising in trying to digitally solve the problem can be summed up into three points: puzzle piece description, rotation and

matching of the pieces and evaluation of the found matches [FG64]. How do we represent the puzzle pieces digitally in order to be able to solve the puzzle? Which features are important enough to keep? Which kept features indicate a good match between puzzle pieces and what do we do if along the way we notice that an incorrect pairing was made in the beginning?

In a traditional jigsaw puzzle we do not get any prior information about the orientation of the pieces and it is also assumed that the pieces can be reconstructed into a single structure without any gaps between the pieces [FG64]. The puzzles don't necessarily have to only have one correct solution, but puzzle uniqueness is generally the norm. Commercially sold jigsaw puzzle are a good example of this. Most of them contain no missing pieces and once opened the pieces could be found in any arbitrary orientation. Also in most of the cases the pieces can be formed into a single unique solution.

In addition to aspects of jigsaw puzzles already mentioned, Freeman and Garder [FG64] explain another key characteristic called *radiality*. Radiality refers to the kinds of interior or exterior junctions possible between the puzzle pieces. A junction of three boundary lines in a piece is called a triradial junction. Whereas in the case of a junction with four boundary lines, we refer to a quadradial junction and so on.

4.1 Puzzle variations

Through modification of the core characteristics of jigsaw puzzles multiple variations of the problem can be created to make it easier or harder. Just by simply removing all pictorial information from the equation the problem becomes harder to solve, because we have to solve it solely by relying on the shape information [FG64]. Jigsaw puzzles with this kind of modification are known as apictorial jigsaw puzzles, which were the main focus of the original introduction by Freeman and Garder as well as other research in the field until Chung et al. [CFF98] added their pictorial solver into the mix. This versatility in breaking some of the rules to create different problems is what allows jigsaw puzzles to be applied to multiple domains such as historical restorations.

For certain jigsaw puzzles we might have prior domain specific knowledge. In those situations it might be worthwhile to exploit the information to make the problem easier than to resort to more general approaches. For example all of our puzzle pieces might contain text that indicates the correct orientation for the said pieces.

Such puzzles are called oriented puzzles and they simplify the work by allowing most of the rotation related aspects to be ignored during matchmaking. Simplicity could as well be obtained from knowing how many neighbours any given piece might have, the radiality distribution of the kinds of junctions that can occur between pieces or their exterior boundary geometry.

4.2 Dead Sea Scrolls as a Jigsaw Puzzle

By observing the types of object we are dealing with the Qumran scrolls, we can define which jigsaw puzzle rules apply to them. According to Stegemann [Ste90] most Qumran fragments are from scrolls, mostly of leather or papyrus, that were rolled without a protective handling stick leaving the inner and outer layers exposed to wear. These scrolls have been mainly damaged by decay over time, by animals and also by human intervention.

Because they were once scrolls written in a certain order, we know that our puzzles have one unique correct solution. However, because of the damage they have suffered, some pieces might be missing and full assembly might thus be unreachable. In fact so many pieces might be missing that the problem essentially becomes one of solving multiple separate jigsaw puzzles. We also cannot make any assumptions about the orientation or shapes of the pieces. Once assembled there is a possibility for the pieces to have gaps between them or even pieces that share no boundary with any of its neighbours.

Pictorial information is particularly beneficial in the reconstruction process. Especially colour changes in the material can give us a clue about the layer and location on the rolled up scroll in which these fragments originally existed [Ste90]. Even though there are inscribed passages written on pieces of fragments, we cannot expand this property to all of the pieces to make assumptions about their orientation solely on the basis of this.

To summarize, the Qumran scrolls can be reduced to a problem of solving an unique two-dimensional pictorial multiple-connected disjoint-area jigsaw puzzle with an irregular and unknown exterior boundary, and missing pieces. A puzzle the pieces of which are arbitrarily shaped, sized and oriented. During an assembly process the puzzle might divert into a problem of assembling multiple puzzles. From here on whenever the terms jigsaw puzzle or puzzle are mentioned they refer to this specific variant unless otherwise mentioned.

5 Preprocessing of Puzzle Pieces

Images to computers are just a stream of ones and zeroes that don't carry any meaning. We have to provide the computer necessary information about the data in order to be able to complete the task to be computed. To digitally solve a jigsaw puzzle we too have to refine our data into a suitable input for further computing.

The preprocessing stage consists of three parts. The first step involves extracting edge information from the image that will be used as an input for the next step. The second step uses the input given to detect contours in the image. Lastly we will prune the contour points to simplify our dataset while also trying to avoid losing any important curve features.

The first two steps are needed to turn image data into a form that can be used to infer the relationships between the puzzle pieces. Provided that we have infinite amount of time or arbitrarily fast computers the last step is not necessary. As those assumptions never apply in reality we need to be able to reduce the problem space to cut down on the time complexity of the solution.

5.1 Edge Detection

Given an image we would like to find the contours in it, but before we can do that some preprocessing is needed so that we can give the result of that preprocessing as an input to the next step in the pipeline. For simple images performing a binary threshold is sufficient. For more demanding images, where binary threshold is not applicable, we will describe a method for extracting edges with Canny edge detection.

John Canny [Can86] describes three criteria relevant to an edge detectors performance: 1) a low error rate, 2) good localization and 3) single response to one edge. In the first criterion we seek to find a low probability for not marking real edge points and likewise a low probability of accidentally marking false edge points. With the localization criterion we are ensuring that the responses of the operator be in the vicinity of the real edge's center. Multiple responses to a single edge might occur and they are not handled properly by the first two criteria so we need to add a third that considers these extra edges erroneous.

Canny edge detection operates in the following manner. First a Gaussian filter, the width of which varies given the situation, is applied to smooth the image in order to

remove noise. Then we obtain intensity gradients of the image with an operator such as the one described by Prewitt [Pre70]. This is done by convolving an image given as input with a symmetric two-dimensional Gaussian operator. By differentiating the result normal to the edge direction, we are able to obtain a horizontal and a vertical mask. There is no need to differentiate in any additional directions, because the direction of the gradient can be computed solely from these two masks. To provide better results sampling can be done along the edge direction at multiple orientations, the output of which will be joined to form a single mask. Noisy output from these directional masks will be suppressed if the variance is above a squared-error measure. Lastly hysteresis thresholding is used to decide which edges to keep. It is difficult to choose a single threshold with a high chance of properly marking real edges while at the same time having a low probability of marking noise as edges. Instead of a single threshold two separate thresholds, a low and a high, will be used. Any parts of a contour that fall under the low threshold we immediately discard and those that rise above the high threshold we automatically mark as edges. Points that belong to the same connected contour segment as those marked also get included provided that they are also located above the low threshold.

Canny edge doesn't guarantee any ordering for the outputted edge map, but later parts of our preprocessing pipeline require that the input is ordered. This can be solved by either ordering the outputted edge map or converting it to a binary threshold such as in [RRCL13].

5.2 Contour Extraction

To extract contours from the binary segmentation mask obtained from the process described in the previous section a border following scheme by Suzuki and Abe[SK85] will be used. The algorithm they present is able to extract a topological structure from the binary input by following found border starting points during a raster scan. In case we are only interested about the overall shape of an object and want to ignore the internal structure of it, Suzuki and Abe also explain a sligthly modified version of the algorithm which only follows the outermost borders of an image. Contours extracted from the image will be represented by a set of pixels P_i.

5.3 Pruning the Data

Having extracted the contour features, we are still not quite ready to proceed to the actual stage of piece assembly. The data that we obtained in the previous step contains redundant data and thus requires some pruning. Finding a solution for a jigsaw puzzle in general is a NP-complete problem as shown by Demain et al. [DD07] and as such it is justifiable to spend some computing power upfront to reduce the running time in the later steps.

The main goal of the pruning is to select a subset of the contour points to form a caricature or simplification of the original line. In case of a straight line, this is relatively easy. For example, let's say we are given three points of a line for reduction. We could choose the first and last point of the line to represent it without losing any representative accuracy. Things get trickier once we introduce curves and irregular line shapes, which points should we choose? We don't want to use too simplistic approaches like deleting every nth point on the line, because we might lose interesting line features. To solve this problem we have opted to use the Douglas-Peucker line simplification algorithm [DP73].

Pseudocode for the Douglas-Peucker algorithm is shown in Algorithm 1 and as an input it takes a list of points, P, to be reduced and a threshold ϵ. The algorithm works by selecting points from original line to be included in the caricature and discards the redundant points. The first and the last point of the line are automatically included and for the rest of the points residing between these end points, we calculate their perpendicular distance to the line and find the point with the maximum distance from the line. We compare this maximum distance to the ϵ threshold and in case the distance is greater we know that the line contains points that can't be discarded so we recursively call to simplify lines from the starting point to $max_d(P)$ and $max_d(P)$ to the last point, where $max_d(P)$ is the point with the maximum perpendicular distance to the line. In the case where the threshold is bigger we know that we can safely discard all points expect the first and the last one. As output we will receive a new list of points containing only selected points which are within interesting line features according to the given threshold.

Algorithm 1 Douglas-Peucker Algorithm (P, ϵ) [DP73]

```
Initialize d_max, i_max = 0
end = P.length
for 2 to P.length - 1 do
    distance = perpendicularDistance(P[i], Line(P[1], P[end])
    if distance > d_max then
        i_max = i
        d_max = distance
    end if
end for
if d_max > ε then
    RetList_1[ ] = Douglas-Peucker(P[1 to i_max], ε)
    RetList_2[ ] = Douglas-Peucker[P[i_max to end], ε)
    RetList[ ] = RetList_1[1 to RetList_1.length-1] + RetList_2[ ]
else
    RetList[ ] = new List[P[i], P[end]]
end if
return RetList[ ]
```

6 Assembling the Pieces

This section seeks to solve the three points mentioned in section 4, namely defining digital description for puzzle pieces, problems of matching and rotating fragments as well as evaluation for the resulting matches. First a general approach for digitally solving jigsaw puzzles is introduced. After which feature-based and elastic model based curve matching strategies are explained. Usage of feature-based curve matching is illustrated by going through how it can be utilized to assemble shredded documents in [RRCL13]. Regarding the elastic model based curve matching, two different approaches to reconstructing ancient artefacts are shown.

6.1 General Approach

Generally approaches to solving 2d jigsaw puzzles can be divided into two parts: defining local pairwise similarity and global arrangement of pieces. In the first part we have a set of fragments F with each fragment being represented as some feature vector F_i. In case of a pictorial puzzle we could denote a fragment as a

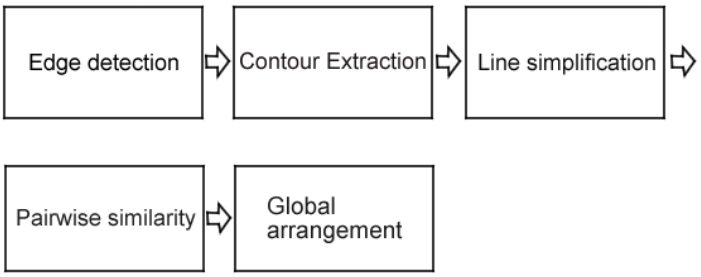

Figure 1: General pipeline for an assembly process.

support vector $F_i = \{S_i, I_i\}$, where S_i is set of support points and I_i is image of the fragment given as an input. For a pair of fragment F_j and F_k we seek to find some similarity measure, attained through comparing their feature vectors, to find potential matches. Fragments which are found to have similar matching features will be marked as mating candidates. Based on the matings we can attempt a global reconstruction by joining the pieces together and appending more pieces to the joined object before we either run out of pieces or possible matches. It is important to detect errorneous matings early to reduce the assembly time as the cost increases with each step required to correct them [FG64]. These steps combined with the previously mentioned image preprocessing form a full pipeline for a general digital solver which is visualized in figure 1.

The problem at hand doesn't scale well when the number of puzzle pieces to match increases. While smaller puzzles might be brute forced, the number of orientations the pieces can be in combined with the number of matchings to be checked makes an exhaustive search method an infeasible solution for the problem. Most solutions therefore opt to use orientation-invariant features to compare matchings. In such a case, finding the best match between two fragments for a collection of N fragments requires $\theta(L^2N^2)$ operations, where L is the number of sample points in a fragment [dGLS02].

6.2 Shredded Documents

Richter et. al. [RRCL13] make use of a feature curve based method for piecing back together documents that have been shredded by hand. While often matching is done based on outer borders, in some cases we cannot resort to these kinds of

strategies, because torn documents may contain fragments with partially overlapping parts [KS09]. This could occur for example during ripping of a paper that causes diagonal tear between pieces instead of a straight clean separation. In such a case the front- and backside of a particular area would be split into two or more separate fragments. The fragment description for the method described by Richter et. al. is defined as a support point s with 5 feature descriptions: line segment length, inner angle, shape of the content, colour histogram around the point's circular region and colour layering. With these features, instead of trying to match each support points of a fragment against those of another, a subset of support points that are likely to fit together are consider in the pairing process.

Pairwise affinity between support points is determined separately for each of the features. After a dissimilarity measure is calculated between each of the support point pairs for each feature, these measurements are concatenated into a single dissimilarity vector $d^{s_p^i, s_q^j} = [d_k(s_p^i, s_q^j)], k = 1...5 \in \Re^5$. These features could be modified or additional ones introduces, but the resulting features should respect the rotation invariance property. However, it should be noted that number of combinations for pairing each support point grows quadratically in the number of support points of the respective fragments. The authors had an annotated dataset, containing true and false pairings, which they used to train a support vector machine to root out false matches from the list of possible candidates. This is done by assigning alignment scores, which correspond to our believed correctness, for the pairings with respect to coincident border, relative intersection, geometric and content-based information constraints.

The global puzzle solving algorithm works by creating a graph for the document and outputting a set of alignments for each fragment and a spanning tree for the graph described by an edge set $\hat{E}$. In the document graph, vertices correspond to fragments and the edges are defined as the alignments between those fragments. At the start of the algorithm each fragment is considered a cluster on its own. Iteratively these clusters are joined by choosing the best alignment between any pair of clusters given by $\hat{a} = arg \max_{i<j} \hat{a_{ij}}$. After each joining edge weights are updated. This process is repeated until only a single cluster remains. A more detailed look on the different steps can be found in algorithm 2. The authors mention that outliers in the approach come mainly from pages that are of an uniform colour and contain little to no information in their content. The process doesn't work when too many pieces are removed or if the problem becomes disconnected and as such stopping criteria need to be defined for these kinds of occasions. While some prior good candidate

pairs might turn out to be false pairing during the iterative weight recalculation, the presented algorithm does not contain a backtracking procedure in the case of early occurring errors.

6.3 Ancient Documents

We will now switch our focus on two state-of-the-art methods for solving the two-dimensional ancient artefact reconstruction problem. These solutions also differ from the previously shown feature based curve matching by using a method referred by McBride and Kimia [MK03] as elastic based curve matching. According to them feature based methods suffer from a drawback of having hard time dealing with fragment overlapping and distinguishing between close ambiguous matches. Elastic curve matching seeks to resolve this issue by allowing dynamic one to many mappings between support points of a pair of fragments instead of being limited to one to one correspondences. Both of these methods solve the problem in an apictorial way. However, as multi-scale matching in itself is agnostic about the nature of the samples, features like colour and thickness of the material can also be augmented with the boundary information [dGLS02]. Differences in the two approaches are mainly found in their feature description, performance evaluation for pairings, correct match detection and global puzzle assembly process.

6.3.1 Generic Solver

We will first take a look at the method by Leitão and Stolfi [dGLS02] whose multi-scale matching method reduces the operations, needed for finding the best match between two fragments for a collection of N fragments, from $\theta(L^2N^2)$ to $\theta(N^2LlogL)$. As the piece description they use a curvature graph with points representing local curvature features denoted by $\kappa(t)$. Uniform width between these points is determined by a sampling step δ depending on the level of detail we are interested in.

Pairwise affinity in their approach is computed by summing together $Y^2(a, b; r, s)$, a term for measuring total difference between two line segments, and $Z^2(r, s)$, a term for penalizing highly irregular pairs. In these terms a and b are defined as vectors of sample points for the two line segments. By (r, s) the authors mean a pairing correspondence between two samples a_{r_k} and b_{s_k}, for each value of $k \in \{0, ..., p\}$.

Algorithm 2 [RRCL13]

STEP 1: Initialization. Initially at time t=0, each fragment F_i is considered a cluster by itself, i.e., $c_i = \{F_i\}$ and $C^{(0)} = \{c_0, c_1, ..., c_{|V|}\}$. Each edge of the document graph is weighted according to its alignment score and each Q_i is initialized as an empty sequence of alignments. Also, $\hat{E}$ is an empty set of edges.

STEP 2: Combining Clusters. Let t be the current iteration. In order to find the best alignment $\hat{a}$ between any pair of clusters, we simply determine the edge with the largest weight that connects two distinct clusters. Let e be this edge and, without loss of generality, let it connect clusters c_i and c_j, $i < j$. We add e to the spanning tree: $\hat{E} = \hat{E} \cup \{e\}$. We combine both clusters into $\hat{c}_i = c_i \cup c_j$ by aligning their fragments according to $\hat{a}$. As only fragments of cluster c_i are aligned in this step, we append $\hat{a}$ to their sequence of alignments, i.e., $\forall k F_k \in c_i$: append $\hat{a}$ to Q_k. We obtain the set of clusters for iteration t by replacing the clusters that were combined, i.e., $C^{(t)} = \frac{C^{t-1}}{c_i \cup c_j} \cup \hat{c}_i$. Since we reduce the number of clusters by one during each iteration, the algorithm terminates after iteration $t = |V| - 1$.

STEP 3: Removing Unpromising Alignments. To speed up the computation during the remaining iterations, we aim to reduce the number of alignments to be considered. For this purpose we apply two heuristics that remove any pair of support points from $\hat{c}_i$ that became obsolete due to alignment $\hat{a}$. First, we disable support points along coincident border of the new cluster. Intuitively, aligning another fragment to one of these points naturally produces high intersections. Second, we remove all pairs of support points which are evidently incorrect due to their clearly insufficient alignment score.

STEP 4: Updating Edges. As mentioned before, combining two clusters provides additional evidence about the document at hand. Therefore the weight of all edges originating from the combined cluster $\hat{c}_i$ need to be updated.

STEP 5: Return. After $|V| - 1$ iterations we accumulate the sequence of affine transformations for each fragment F_i into $\hat{Q}_i$.

The term $Y^2(a,b;r,s)$ itself is acquired from

$$Y^2(a,b;r,s) = \frac{1}{4}\sum_{k=0}^{p-1}(\epsilon_k + \epsilon_{k+1})(\tau_{k+1} - \tau_k). \quad (1)$$

whereas the irregularity penalizing term is given by

$$Z^2(r,s) = \frac{\zeta^2}{2}\sum_{k=0}^{p-1}|(r_{k+1} - r_k) - (s_{k+1} - s_k)|. \quad (2)$$

Combined together they form a quadratic dissimilarity measure for the pair to be considered

$$S^2(a,b;r,s) = Y^2(a,b;r,s) + Z^2(r,s), \quad (3)$$

After having obtained pairwise dissimilarity measures, we still need a way to evaluate their performance and choose the most likely ones to be correct. In other words we are seeking for a pairing (r_*, s_*), which minimizes S^2. Separation to false and possibly true pairings in done in a binary fashion by considering the result of

$$\triangle_*(a,b) = S^2_*(a,b) - \xi^2((m+n)/2 - n_{min}). \quad (4)$$

When $\triangle_*(a,b) < 0$ a candidate pair (a,b) is marked as true and false otherwise. Effectiveness of this approach depends on constant variables ξ, ζ and n_{min}, the values of which should be empirically experimented on to fit the data of the fragments to be matched.

Typically global puzzle solving processes are presented, but Leitão and Stolfi opt to focus on producing a set of initial pairing candidates. As an input their algorithm, shown in algorithm 3, takes raw fragment outlines $C = \{C_0, ...C_N - 1\}$, minimum sampling step δ, the minimum length L_{min} for reliable matching, constant variables $\xi^{(k)}$, $\zeta^{(k)}$, and $n^{(k)}_{min}$, for each scale k and a corner blurring factor α. At the end a set of possibly true candidates are outputted. The main idea of is to calculate initial samples on a coarse scale, then keep only the good results, and repeat while refining these candidates on finer levels of detail. In order to find the pair that minimizes S^2, the problem is formulated as finding the minimum cost path in a acyclic directed graph G and solved with a variation of the dynamic programming algorithm.

6.3.2 Domain Specific Solver

McBride and Kimia's [MK03] solution builds on top of the techniques introduced in the last section. However, instead of focusing on a generic solution, the authors

take advantage of the dominance of triradial junctions appearing in archaeological puzzles. These triple junctions appear in the form of "T" and "Y" junctions, "Y" junctions being the less common one [MK03].

In their method pairwise affinity is computed by looking at possibly matching sub-boundaries while also incorporating a solution for the partial curve selection problem. Complexity is reduced by using a multi-scale technique similar to that of [dGLS02]. Partial curve matching is started from corners and the end points are dynamically determined from alignment-based elastic curve-matching. The end points are determined by keeping two factors in mind: sub-contour similarity and as long reach for sub-contours as possible. Curve matching energy, sum of stretching and bending energies, is increased by extending the match. This is why shorter matches will be rated better to their longer counterparts. Extending has been balanced by making extensions contribute negatively if in the end results the curves are locally similar, and positively in the case of local dissimilarity in the curves. Matching is done in multiple granularities. The finer levels are restricted to those pairings with a high enough rank to warrant further refinement in re-evaluating the measurement of similarity. Using least-squares a suitable alignment between the points, gained through curve matching, can be attained. Finally a cost factor with three parts is evaluated for the matching: 1) distance measure between the points 2) length measure of the arc for the common boundary and 3) measure of boundary complexity. The longer and more complex the boundary, the higher is our confidence for the correctness of the match. As a final result of matching every pair of fragments, a list of adjacency candidates with measures of affinity is outputted. Final cost for pairings are calculated by combining these measures:

$$C_{total} = \lambda_1 * C_{distance} + \lambda_2 * \sqrt{C_{length}} + \lambda_3 * \sqrt{C_{diagnostic}}. \quad (5)$$

This matching affinity measure tries to overcome two main issues in the pairings: 1) poor results for damaged, very short or flat boundaries, and 2) high-ranked matches that are similar by chance.

Global assembly is approached with a best-first strategy by using both local pairing information and a global confidence measure. Possible errors during assembly are handled by computing multiple solutions simultaneously and memorizing k different arrangements at each stage instead of backtracking a procedure. The approach relies heavily on simplifying the problem with domain specific assumptions for the the kinds of junctions and the way the matching typically occur between the pieces.

The solution is started by ordering the pairings by rank and selecting the best alternative first as the initial step. After this step more constraints are introduced as the proposed solution expands. First of all only fragments that connect to the existing object are considered. Using these added features, a global confidence measure is created, which will tell us how good the final solution is. Fragments forming triple junctions contribute strongly to the global confidence, because they are statistically more likely to be correct. The strategy moves from best-first to best-global-first, which means that the fragment being added that contributes to the best result for global confidence will be added to the solution next.

Instead of backtracking, beam search is used to deal with the possible errors that might occur during the assembling process. This technique picks the k top matches, which combined with the already placed pieces, form states. For each of the states we find Y fragments to match with the best-global-first strategy. Then we order the states and keep only the X best. This process is rinsed and repeated. A confidence threshold is added for additional pruning of the search space and it also serves as a stopping condition.

In this section we saw how the fragments could be described digitally. We also went through two different pairwise affinity measures used in ceramic tile reconstruction. Leitão and Stolfi [dGLS02] provided a way for finding a set of possible matching candidates. This step was enhanced by a description for a global assembly attempt by McBride and Kimia [MK03] given the candidates as an input.

7 Overview of the Reconstruction Tool

A tool for assisting researchers, working with the reconstruction of The Dead Sea Scrolls, in the field of theology was created as a by-product of this thesis. The focus in the development of the program was to automate some of the manual steps introduced in section 3.1 as well as to test the viability of applying curve matching techniques in scroll reconstruction process. The program's graphical user interface implementation was done with the help of the Qt library , and the image processing and computer vision side utilized the openCV project.

Algorithm 3 Multi-scale matching [dGLS02]

\\ Multi-scale filtering

$C^{(0)} < -C$; $k < -1$; $\delta^(1) < -\delta$

while $\delta^{(k)} <= L_{min} - 8\alpha\delta^{(k)}$ **do**

Filter and re-sample the curves $C^{(k-1)}$ with step $\delta^{(k)}$, producing $C^{(k)}$.

$k < -k + 1$; $\delta^{(k)} \leftarrow 2\delta^{(k-1)}$.

end while

Set $K \leftarrow k - 1$

\\ Initial matching

Delete any non-fracture segments from the curves $C^{(k)}$

Determine the initial candidate set $R^{(k)}$ among the curves $C^{(K)}$.

\\ Refinement and pruning

for $k = K, K-1, ..., 1$ **do**

Refine the raw candidates $R^{(k)}$, obtaining a new set $S^{(k)}$.

Remove from $S^{(k)}$ all candidates with positive discriminant $\triangle_*$.

For each pair of curves, collect all the corresponding candidates in $S^{(k)}$, retain only the 2^{k-1} candidates with smallest (most negative) $\triangle_*$, and discard the rest.

Merge any overlapping candidates among those remaining in $S^{(k)}$.

Map the candidates $S^{(k)}$ from the curves $C^{(k)}$ to the curves $C^{(k-1)}$, obtaining the raw candidates $R^{(k-1)}$.

end for

Refine $R(0)$, obtaining the final candidate set S.

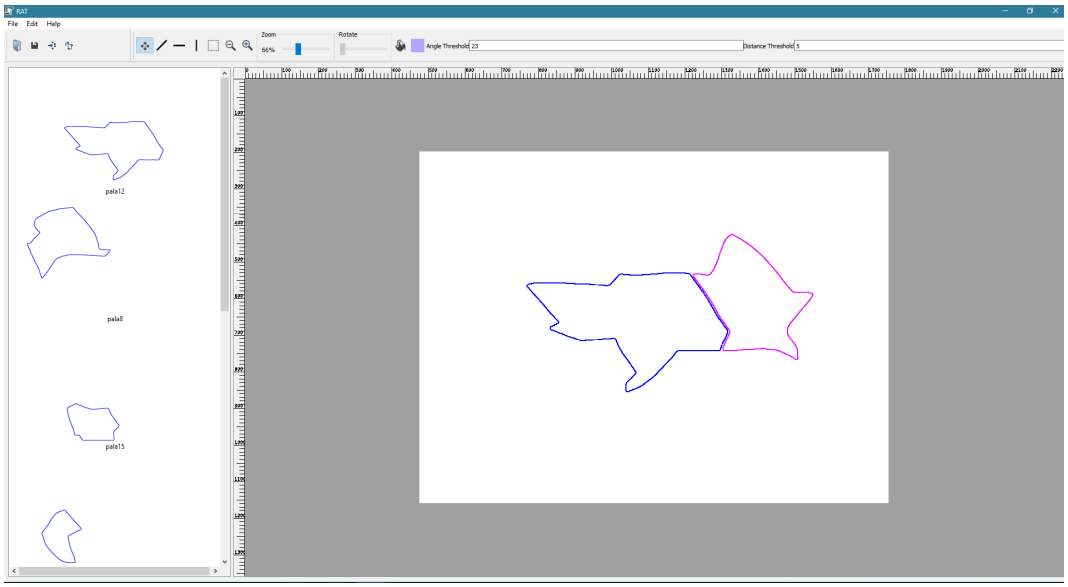

Figure 2: Screenshot showing a manual pairwise comparison being performed with the program.

7.1 Features

The finished program contains several features which seeks to ease the amount manual work the researchers had to previously perform manually. The whole preprocessing pipeline from section 5 was implemented with the added benefit of being able to dynamically adjust different thresholds, constants and algorithms for contour detection. Imported images can be dragged and dropped into an assembly canvas where potential solutions for the puzzle and pairing between pieces can be experimented. Visual aid is supported with colour repainting, line drawing and ruler features as well. Collaboration and sharing of ideas with other researchers can be done, albeit in a rudimentary fashion, by using exporting of images assembled on the canvas or sending program project files. During development, the project files proved to be essential when replicating and fixing bugs. A screenshot of the program in question are shown in figure 2. Lastly, a simple automated puzzle solver was attempted.

7.2 Fragment Pairing and Assembly Model

Due to time constraints, the viability of curve matching techniques applied to Qumran artefacts was tested with the simple feature vector description and global puzzle solving strategy by Justion et. al. [JOF06]. In this section we will focus solely on

the description of the details of the feature vector and global assembly process used. Section 8 will be covering the results of the findings as well as a proposal for a domain specific take on the problem.

The support points in the method by Justino et. al. [JOF06] include information about the vertex index, distances to next and previous support points, x and y coordinates of the point, and the angle between the line segments spanning from the current point to the next and previous ones. Together these form a feature vector for a support point. The pairing evaluation is a simple weight value W_{match}. The weights are calculated by checking whether one or both of any given support points' lengths to a neighbouring support point line up with that of another fragments one. If both of the neighbours distance from point s_i of a fragment F_i are roughly equally as far away as point's s_j neighbours from fragment F_j, then 5 points will be added to the matching similarity W_{match} provided that the angle features of s_i and s_j sum up to roughly 360°. Likewise if only one of the neighbours line up from each point and the angles agree, 1 point will be added to the similarity. Length of the perimeter from these matches are taken into account by adding 2 additional points if a matched contour spans over more than 1/5 of the fragments perimeter or 1 additional points when over 1/10 of the perimeter is covered. The authors explain that these points were determined through empirical experimentation.

Now that the pairwise affinity is defined we continue to explain the global solution process. Similar to that of [RRCL13] we start with each fragment being considered as its own group. All the matchings are calculated between the first fragment F_1 and all the fragments F_i in the range from 2 to n, where n is the total number of fragments. The pair with the overall best matching will be merged into a single fragment. This process will be then repeated by trying to pair each of the fragments with the joined object, until only one fully merged fragment remains or no matching candidates are found. It is apparent from the algorithm description that as a result we might get complete, partial or empty pairing solutions for our problem. Additional details for this algorithm are provided in algorithm 4.

8 Results

The dataset, used for experimenting the reconstruction method introduced during the overview of the reconstruction tool, was created from an rectangular image that has been manually split into 16 arbitrarily shaped pieces. The whole image with

Algorithm 4 Global search [JOF06]

```
D = {F_1, F_2, ..., F_n}
repeat
    best = NULL
    for i = 2ton do
        Compute all possible W_matching for F_1 and F_i
        if 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
```

puzzle piece boundaries is represented in figure 3.

At first our experiments yielded no results. No suitable match for fragment F_1 was found and the algorithm hit a stopping criterion. Once this piece was excluded or the process started with another fragment as the first one, a solution suggestion formed out of most of the puzzle pieces was outputted. Pairing seems to have a slight tendency to favour long matches where the angle α of each individual fragment is close to 180°. Some of these straight matches were false matches that just happened to agree on the features, but the algorithm has no way of making distinctions between these kinds of spurious pairings and real ones.

To improve the global search algorithm, multiple solutions from different starting points could be run in parallel to alleviate the issues we had early on. Since the algorithm has no mechanism for correcting errors during reconstruction and backtracking can become complex and computationally expensive, this simple approach from McBride and Kimia [MK03] increases fault tolerance and quality of results by allowing to pick the solution with the best measured confidence for being correct. Another thing that could be adopted from McBride and Kimia is a notion of boundary complexity for the point awarding rule set. In essence, without at least these

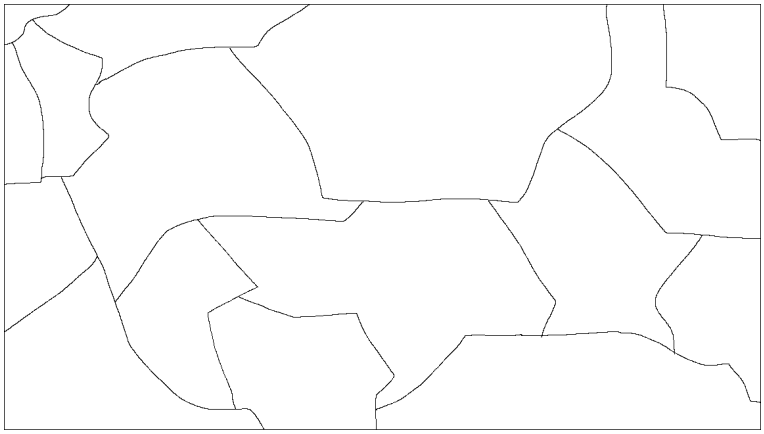

Figure 3: Correct solution for our rectangular shaped test puzzle.

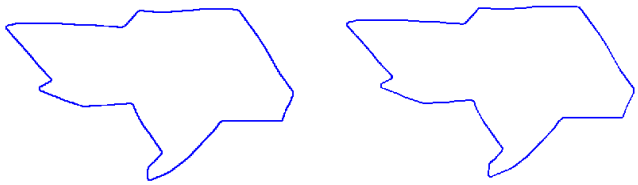

Figure 4: Side by side comparison of image results from original contour extraction and after applying line simplification. As is apparent from the simplified image on the right, no important curvature features have been lost in this process. Contour extraction for all the 16 pieces produced a total of 14076 support points. With the Douglas-Peucker algorithm we were able to reduce this amount to just 314 points.

Figure 5: Example of a spurious result. The darker straight lines represent the suggested matching sub-contours.

changes given the poor performance on our dataset with simplified contours, which still retain all the relevant features, this approach will not be viable in ancient document reconstruction with more evident decay. The algorithm was not tested on the ancient scrolls, because the puzzle pieces turned out to be too sparsily populated. As such, a list of pairing suggestions outputted by the algorithm would have solely contained false correspondences. Even with the changes or the approaches from section 6.3 it is questionable if these general approaches would fare much better. Therefore, like in [MK03] we propose a domain specific process for reconstructing Dead Sea Scrolls. The unique traits exhibited in Qumran scrolls can be exploited to aid in disjoint puzzle solving. The following approach concentrates in global piece arranging while omitting an exact description for puzzle piece features. However, it is assumed that such features contains information about inner, outer and hole borders and pictorial information including colour.

8.1 Domain Specific Solution

Recurring similar shapes of damage, occurring through out the different puzzle pieces from several layers of the scroll, are the key to our approach. The width of these damage patterns increase towards the outer layers and likewise decrease the more

towards the inner layer the fragments reside, because they have been preserved in a rolled up state. This change can be precisely measured in all of the larger Qumran scrolls by $2\pi r$ provided that we know the material, thickness and tightness of the wrap of the scroll needed to determine the radius r [Ste90]. The digital solving process can additionally be aided by providing constraints with evidence from the data for column width, overall scroll length and direction of the rolling.

At the start of the process to reduce the search space, a clustering of pieces that might belong to the same column can be attempted by comparing pictorial information: letter heights, color, etc. Next pairwise evaluation of fragments is done, but instead of looking for similar complementing boundaries we are looking for identical patterns of differing widths from the contours. Exhaustive search is sidestepped by grouping, suggested in [KS09], and multi-scale matching as was done in [dGLS02, MK03]. First of all, pairwise checks don't have to be done between pieces belonging to the same cluster, because they belong to the same layer. The recurring shapes that we are interested in only appear in pieces belonging to other clusters. Unfortunately, we still have to compare pieces that remained unassigned to any cluster against all other pieces. Second, evaluation is done from a coarse to finer level of detail. Expensive computation for clearly poor matches are avoided early on.

After obtaining similarity measures for all the possible fragment sub-contour pairs, the different clusters can be ordered according to their position in the scroll based on the widths of the damage patterns found in the pieces. On top of that we get an approximated placement for all the pieces from different clusters with matching patterns, because these pieces must lie in the same position relative to each other. Additional pictorial information, like margins, in even one of the fragments in these groups might enable exact placement for the whole group of fragments with similar wear. While it is unlikely that a complete assembly solution will be reached with this kind of manner, the process carries intrinsic value for the researchers in the field, because some of the manual tasks that would have had to been done anyway are now automated. In addition, our process can be combined with the other global assembly algorithms described in this thesis by considering each of the column or page wise clusters as separate puzzles.

9 Conclusion

In this thesis we showed how techniques to digitally solve two-dimensional jigsaw puzzles have been successfully harnessed to reconstruct shredded documents and ancient artefacts. One approach was explained for hand shredded documents and two approaches for objects with tile like characteristics. Despite the successes in these fields we showed the limited usability in employing these techniques to the Dead Sea scrolls due to material deterioration and the disjoint nature the damage tends to cause to the puzzles.

We proposed a new take on digitally solving domain specific puzzles that contain recurring damage patterns in the material. Exhaustive search can be sidestepped through clustering of puzzle pieces prior to global assembly attempts. Pairing process can be skipped for pieces that belong to the same group and placement hints are searched by finding similar recurring patterns from pieces in other groups. The length of those damage patterns tell us the relative position of the fragment in the scroll thanks to the way these scrolls have been damaged while being preserved in a rolled up state. This method can be used in conjunction with other assembly processes such as the curve matching methods described in section 6.

While we did not implement this proposed new strategy, manual labour for the researcher in the field is alleviated with the program that was created together with this thesis. The program is capable of extracting contours for easy pairwise inspection of fragments and providing a workspace for manually experimenting with different possible puzzle solutions. These solutions can then be shared in an interactive format to other researchers in the field.

More emphasis for digitally solving disjoint puzzles is needed in future research. The viability of the methods proposed in this thesis should also be tested in practice. Restoration and reconstruction of ancient objects is an important part for historical reconstructions and progress will remain slow while most of the tasks are done in a manual, laborious and time intensive manner.

References

Can86 Canny, J., A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8,6(1986), pages 679–698.

CFF98 Chung, M. G., Fleck, M. M. and Forsyth, D. A., Jigsaw puzzle solver using shape and color. *Signal Processing Proceedings, 1998. ICSP '98. 1998 Fourth International Conference on*, volume 2, 1998, pages 877–880 vol.2.

CLY10 Cao, S., Liu, H. and Yan, S., Automated assembly of shredded pieces from multiple photos. *2010 IEEE International Conference on Multimedia and Expo*, July 2010, pages 358–363.

CWA+01 Cooper, D. B., Willis, A., Andrews, S., Baker, J., Cao, Y., Han, D., Kang, K., Kong, W., Leymarie, F. F., Orriols, X., Velipasalar, S., Vote, E. L., Joukowsky, M. S., Kimia, B. B., Laidlaw, D. H. and Mumford, D., Assembling virtual pots from 3d measurements of their fragments. *Proceedings of the 2001 Conference on Virtual Reality, Archeology, and Cultural Heritage*, VAST '01, New York, NY, USA, 2001, ACM, pages 241–254.

DD07 Demaine, E. D. and Demaine, M. L., Jigsaw puzzles, edge matching, and polyomino packing: Connections and complexity. *Graphs and Combinatorics*, 23,1(2007), pages 195–208.

dGLS02 da Gama Leitao, H. C. and Stolfi, J., A multiscale method for the reassembly of two-dimensional fragmented objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24,9(2002), pages 1239–1251.

DP73 Douglas, D. H. and Peucker, T. K., Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica: The International Journal for Geographic Information and Geovisualization*, 10,2(1973), pages 112–122.

FG64 Freeman, H. and Garder, L., Apictorial jigsaw puzzles: The computer solution of a problem in pattern recognition. *IEEE Transactions on Electronic Computers*, EC-13,2(1964), pages 118–127.

GMB02 Goldberg, D., Malon, C. and Bern, M., A global approach to automatic solution of jigsaw puzzles. *Proceedings of the Eighteenth Annual Symposium on Computational Geometry*, SCG '02, New York, NY, USA, 2002, ACM, pages 82–87.

JOF06 Justino, E., Oliveira, L. S. and Freitas, C., Reconstructing shredded documents through feature matching. *Forensic Science International*, 160,2-3(2006), pages 140–147.

KS09 Kleber, F. and Sablatnig, R., A survey of techniques for document and archaeology artefact reconstruction. *2009 10th International Conference on Document Analysis and Recognition*, July 2009, pages 1061–1065.

KTN+06 Koller, D., Trimble, J., Najbjerg, T., Gelfand, N. and Levoy, M., Fragments of the City: Stanford's Digital Forma Urbis Romae Project. *Proceedings of the Third Williams Symposium on Classical Architecture*, 2006, pages 237–252.

MK03 McBride, J. C. and Kimia, B. B., Archaeological fragment re-assembly using curve-matching. *Proceedings of the IIEEE/CVPR Workshop on Applications of Computer Vision in Archaeology*, Madison, Wisconsin, June 2003, IEEE Computer Society Press, IEEE Computer Society Press.

NDH08 Nielsen, T. R., Drewsen, P. and Hansen, K., Solving jigsaw puzzles using image features. *Pattern Recognition Letters*, 29,14(2008), pages 1924 – 1933.

Paj13 Pajunen, M., *The Land to the Elect and Justice for All: Reading Psalms in the Dead Sea Scrolls in Light of 4Q381.* Journal of Ancient Judaism. Supplements. Vandenhoeck & Ruprecht, 2013.

PPE+02 Papaodysseus, C., Panagopoulos, T., Exarhos, M., Triantafillou, C., Fragoulis, D. and Doumas, C., Contour-shape based reconstruction of fragmented, 1600 bc wall paintings. *IEEE Transactions on Signal Processing*, 50,6(2002), pages 1277–1288.

Pre70 Prewitt, J., Object enhancemet and extraction, picture processing and psychopictorics (b. lipkin and a. rosenfeld, eds, ny. *Academic Pres*, pages 75–149.

RB82 Radack, G. M. and Badler, N. I., Jigsaw puzzle matching using a boundary-centered polar encoding. *Computer Graphics and Image Processing*, 19,1(1982), pages 1 – 17.

RRCL13 Richter, F., Ries, C. X., Cebron, N. and Lienhart, R., Learning to reassemble shredded documents. *IEEE Transactions on Multimedia*, 15,3(2013), pages 582–593.

SE06 Sagiroglu, M. S. and Ercil, A., A texture based matching approach for automated assembly of puzzles. *18th International Conference on Pattern Recognition (ICPR'06)*, volume 3, 2006, pages 1036–1041.

SK85 Suzuki, S. and Keiichi, A., Topological structural analysis of digital binary images by border following. Comput Vis Graph Image Process. *Computer Vision, Graphics, and Image Processing*, 30,1(1985), pages 32–46.

Ste90 Stegemann, H., Methods for the reconstruction of scrolls from scattered fragments. *Archaeology and History in the Dead Sea Scrolls*, Schiffman, L. H., editor, 1990, pages 189–220.

WC08 Willis, A. R. and Cooper, D. B., Computational reconstruction of ancient artifacts. *IEEE Signal Processing Magazine*, 25,4(2008), pages 65–83.

WSKL88 Wolfson, H., Schonberg, E., Kalvin, A. and Lamdan, Y., Solving jigsaw puzzles by computer. *Ann. Oper. Res.*, 12,1-4(1988), pages 51–64.

ZZH08 Zhu, L., Zhou, Z. and Hu, D., Globally consistent reconstruction of ripped-up documents. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30,1(2008), pages 1–13.