Interactive Image Reconstruction from its Incomplete, Irregular and Imprecise Fragments

Bahar Taşkesen,[1] A G Constantinides[2]

[1] Middle East Technical University, Turkey
[2] Imperial College London, U.K.

1 Introduction

The aim of this paper is to present a novel method as an adjunct to a human/expert for assembling unknown images from their fragments. Reassembling an image from its partial and eroded or decayed constituent fragments is of great importance in variety of fields, such as forensics, failure analysis, anthropology, archaeology and art reconstruction. It is a common task, indeed, in archaeological research and artefact preservation. Reassembly from archaeological fragments is a much more involved problem in comparison to other fields due to the following reasons. The number of randomly mixed fragments is normally huge, the fragments are mostly corrupted, irregular, and frequently with uncertain and eroded boundaries. In archaeology the process of assembling may not be required to produce a complete solution. Indeed, it may be required to produce a series of possible options and even partial solutions which the archaeologist would appraise subject to a range of archaeological and often unquantifiable constraints. We propose such a method for archaeological images.

This paper is organized as follows. Prior work related with our proposed method is shared in Section 2, while detailed explanation of the proposed method is presented in Section 3. Finally, this paper ends with the conclusion part in the Section 4.

2 Related Work

The fact that reassembly of an unknown image manually from its irregular and very often eroded fragments is very time-consuming task, motivates researchers to design automated systems [1], [2], [3], [4]. The algorithms produced are usually built upon several subsystems which generate reliable or accurate scores between fragments. Although, these algorithms show great success in specific cases, they have, so far, limited success in reassembling incomplete images with vast numbers of eroded pieces automatically.

For 2D matching the algorithms proposed so far can be divided into two categories: a) colour based methods, and b) geometry based approaches. Colour based methods use the colour information present in the fragments and mainly at the boundaries [4], [5], [6], [7]. These algorithms are mostly efficient, but they fail when only the mean of the colour intensities belonging to a fragment is taken, or only when the boundary of the pair of fragments is considered. Geometry based approaches assemble the fragments by matching the boundary curves of the fragment pairs [8], [7], [9], [10]. The geometry based approaches can accurately match the adjacent pairs if there is no corrupted or eroded geometry at the broken parts of the fragments. Since the fragments of interest in our case have exposed to the natural elements, perhaps for thousands of years, it is inevitable that the edges of the pieces are eroded/corrupted, and their geometry is thereby destroyed. Moreover, the geometry based approaches are relatively slow as compared to other methods and they can fail in their search to lock at a local minimum of the objective function.

The latest research published focussed on initially finding matches between the fragments by checking the patterns at the boundaries of adjacent fragments. This pattern alignment is enhanced by several methods which can be listed briefly as follows, a) by using the colour information [11], b) by describing fractured edges using the new 'ribbon' idea [12], c) by normal maps [13]. These methods have great success when the features of the fragment pairs are significantly different. We construct an integrated approach with the above-mentioned methods, in which mostly the boundary correlations between the potential pairs dominate the matches. Matching scores are derived and the scores between the potential pairs are adjusted with each of the several algorithmic iterations in a graph based approach.

3 Proposed Method

Our proposed method is not meant to replace the experts' (in our case the archaeologist's) skills and knowledge but rather to be an aid and an additional tool. This aspect is incorporated in the structure of the algorithm. In our approach we support the most likely match with periodicity check, line continuity and inner layer correlation all of these terms are explained in this paper below. At the end, the most meaningful match is selected as decided by the expert/archaeologist. The method is explained in the following subsections which cover formulation of the problem, border and boundary extraction, score assignment, decision unit and score rearrangement.

Formulation of the Problem

Given the set of fragments G = G_i of the image (I), we wish to find the most likely match between fragments. In the composition of the image I each fragment is identified by its pixel boundary vector B_i, i = 1...N, where N is the total number of available fragments. The image (I), has more fragments than N, such that some of the fragments are lost, however, only N number of fragments are accessible. Because of the possibly of imprecise, eroded or corrupted boundaries we also check the inner layers. A layer is a sequence of pixel values once or several times removed off the boundary of a given fragment. Hence, each fragment is defined by its boundary matrix for each layer, B_{ij}, where j is the layer number *1…ly*, and ly is the maximum layer number. To locate the maximum correlation exactly in the whole boundary matrix of the fragment we also detect the corners (C) of a fragment and divide the boundary vector into borders (sub-boundaries) F_k, where k = 1...C, F_k □B_{ij}□ G_i □I, as shown in the Figure 1.

Another reason for dividing the boundary matrix into borders is the fact that each boundary matrix is huge compared to the likely correct matched pairs. This result in vast and unrelated parts of the boundary matrix. These unrelated parts decrease the correlation between correctly matched pairs and give misleading results subsequently. To avoid these problems, we prefer to check the relatively small but reasonable portions of the boundary vector/matrix borders as explained in the following subsection.

Border and Boundary Extraction

As already mentioned, each fragment is defined by its boundary and related borders for the layer of interest. To extract the boundary of the image we transform the image into binary (for instance the fragment is all white and the background is black), by knowing that the photographic image of each fragment is acquired at a constant colour background. The structuring element disk with the distance from its origin (r) proportional to the intended layer number is used for morphological erosion operation. We simply take the difference of the binary images after two consecutive erosion operations, with r set equal to the desired layer number, *ly* and *ly -1*. Then we obtain the raw boundary vector, without corner indicators.

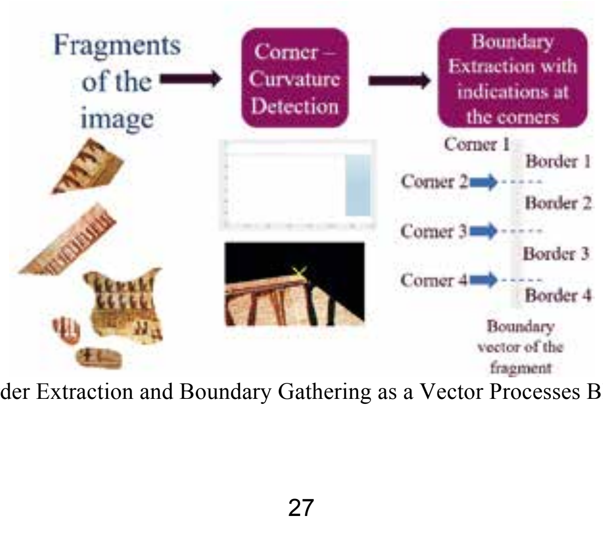

Fragments of the image
Corner – Curvature Detection
Boundary Extraction with indications at the corners
Corner 1
Border 1
Corner 2
Border 2
Corner 3
Border 3
Corner 4
Border 4
Boundary vector of the fragment

Figure 1: Border Extraction and Boundary Gathering as a Vector Processes Block Diagram

Commonly used and well-known functions for corner detection work well if the fragment is close to rectangular in shape with corners at approximately right angles. However, when we have a fragment with other form as in Figure 2a these methods fail and produce false corner alarms, such as in Figure 2b. This is essentially due the fact that we are working in the discrete domain where there are no continuous lines. As a result, when we zoom in the one portion of the fragment as in Figure 2b. Hence for the corner detection purposes we designed our own specific algorithm.

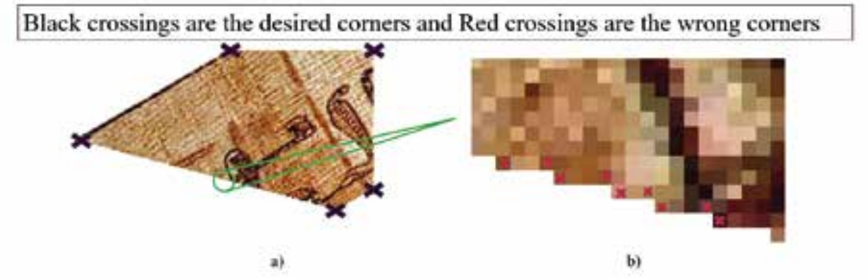

Black crossings are the desired corners and Red crossings are the wrong corners
a)
b)

Figure 2: a. Fragment Sample with the desired corners b. Closed portion of the fragment and the corners detected with the current algorithms

We obtain a slope plot by using the boundary coordinates as in Figure 3. Corners are checked both from periodicity of the slope vector in Figure 3 within the boundary vector, and maximum-minimum slopes in a local area of corner.

In digital world, a line can be defined by its slope period shorter than its size, meaning that each line has a periodic behaviour that determines its starting point to end point slope. The line cannot follow same the same slope at each consecutive point as it can be seen at Figure 2(b), however it follows a period throughout the line. The proposed corner detection algorithm is designed based on this claim. If we assume that a boundary vector is taken, and its slope between two cascaded points are calculated, this brings a slope vector (s) of that boundary as it can be seen at Figure 3. Initially, enough data points (m1) depending on the least border size that is investigating, are taken from the boundary slope vector. After that, period of this data array (s(1)…s(m1)) is calculated, the calculated period of this line is less than its size (m1). Later on, the data is accumulated from "*s*" vector one at a time, and period of the accumulated data array (s(1)…s(m1), s(m1+1), s(m1+2)…s(m2)) is calculated. Moreover, if the calculated period is still less than the size of the array (m2), then this indicates that "s(m2)", belongs to the line. On the other hand, if the period is equal to the size of the array (m2) than this indicates that the line might be following another direction, starting from m2. This doubt about m2, that whether it is a corner point, or not, is overcome by accumulating the rest of the s vector size at a time, and its corresponding period, as it is done until m2 point. Furthermore, if they are equal in every time, then this indicates that slope vector is aperiodic after m2, and it is unique. After m2 is marked as a corner, the algorithm is iteratively computed for the *s'* vector, which corresponds to [s(m1+1), s(m1+2)….s(end)].

Within the local area next maximum-minimum and the previous maximum-minimum slope values are checked and the detected corner is verified. This approach is also supported by the local standard deviation calculations of the differences at boundary's x values and y values. Peaks in the sliding standard deviation vector over the slope vector indicate a possible corner. These two checks, supports the proposed corner detection algorithm for the worst cases that it might fail.

Corner extraction results for an ideal fragment is shown in Figure 4(a), and corners for a fragment with curves can be found at Figure 4(b). We put an indicator (*NaN* in our case) for the detected corners in the boundary vector. Hence, we obtain a single vector to represent each G_i.

Score Assignment

We used a dynamic graph based approach to track the relations between fragment pairs, and their corresponding borders. We basically use a cross correlation metric to determine the weights in the dynamic score graph.

Cross correlation calculated from the equation 1, where c_{y1y2} is the sample covariance function calculated from 2 and s_{y1}, s_{y2} are corresponding sample standard deviations of the time series y_{1t} and y_{2t} at lags k = 0, ±1, ±2, … calculated from equation 3. μ_{y1} and μ_{y2} are the sample means of the time series.

$$r_{y_1y_2}(k) = \frac{c_{y_1y_2}(k)}{s_{y_1}s_{y_2}} \quad k = 0, \pm 1, \pm 2, .. \quad (1)$$

$$c_{y_1y_2} = \begin{cases} \frac{1}{T}\sum_{t=1}^{T-k}(y_{1t} - \mu_{y1})(y_{2t,t+k} - \mu_{y2}) \ if\ k = 0, 1, 2 \ldots \\ \frac{1}{T}\sum_{t=1}^{T-k}(y_{2t} - \mu_{y2})(y_{1t,t+k} - \mu_{y1}) \ if\ k = 0, -1, -2 \ldots \end{cases} \quad (2)$$

$$s_{yi} = \sqrt{c_{y_iy_i}(0)}, c_{y_iy_i}(0) = Var(y_i) \quad (3)$$

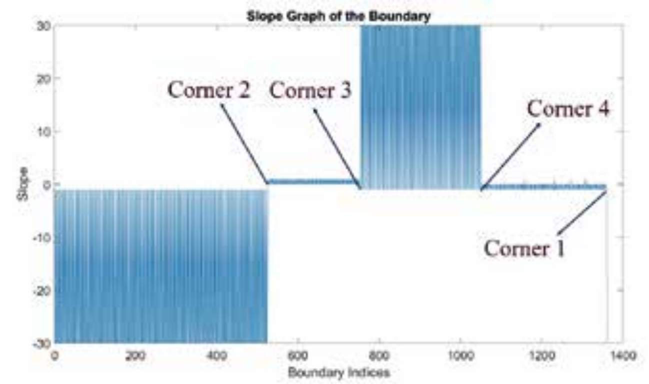
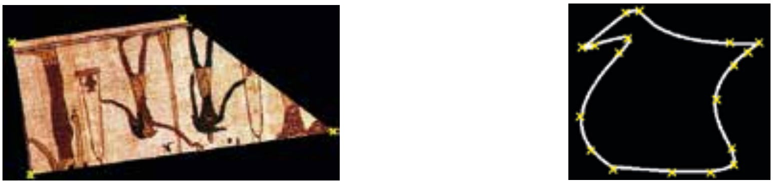


Figure 3: Slope Plot of the Consecutive Points at the Boundary of a Fragment

Initially, the scores are calculated over the first layer. Elimination with a predetermined threshold is performed and most likely solutions are gathered. With this knowledge of the most possible matches, we supervise the inner layer correlations. However, the correlations at the inner layers are relatively small as it can be seen at the Figure 5. As a result, the threshold that eliminates the wrong matches and indicates frame borders of the image, (at the frame borders of the overall image I, we do not expect any match) is decreased. The whole idea is represented in the Figure 6.



(a) (b)

Figure 4: Corner detection for the non-ideally shaped fragment and for the ideally shaped fragment

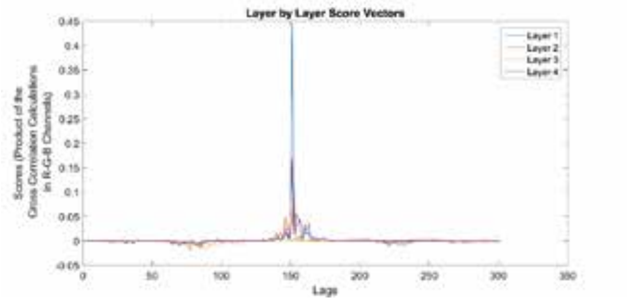


Figure 5: Layer by Layer Score Vector Plot

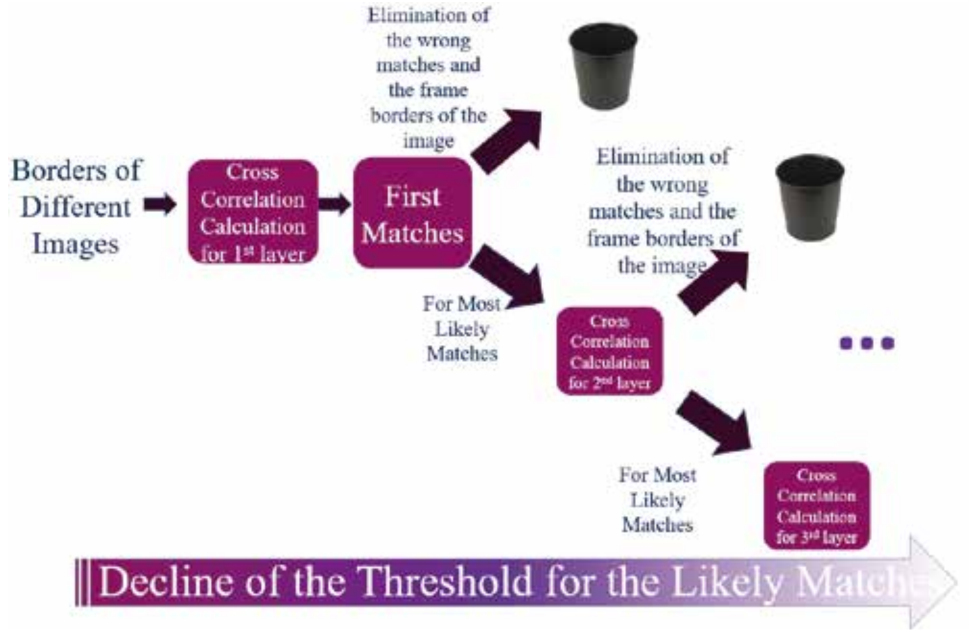

Borders of Different Images
Cross Correlation Calculation for 1st layer
First Matches
Elimination of the wrong matches and the frame borders of the image
For Most Likely Matches
Cross Correlation Calculation for 2nd layer
Elimination of the wrong matches and the frame borders of the image
For Most Likely Matches
Cross Correlation Calculation for 3rd layer
Decline of the Threshold for the Likely Matches

Figure 6: Block Diagram of the Proposed Method

Interpolation

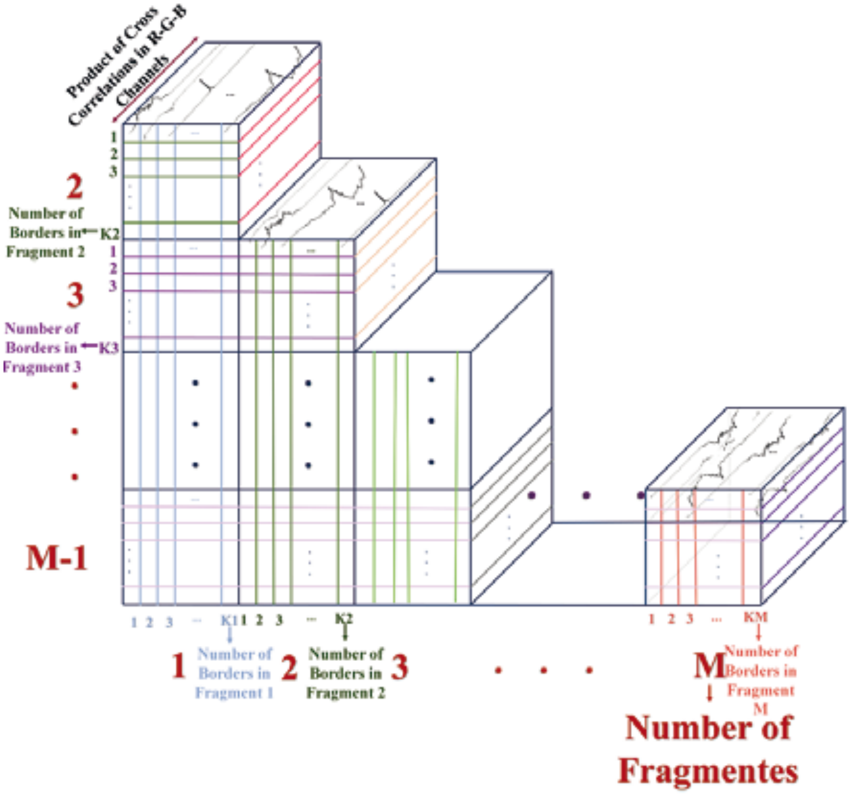
At the inner layers the size of the corresponding border vector is smaller (fewer samples). This reduction in the length of the inner layers (pixel changes in the layer 1 to layer 3 cross correlation calculation) results in changes of the maximum correlation location within the two corresponding borders. To overcome this problem, the smaller border vector is linearly extrapolated |layer1 - layer2| times. Here, layer1 is the layer number of the first border and layer2 is the layer number of the second border.

Calculation of the Cross Correlation in 3D

Cross correlation is calculated between each fragment pairs at the desired layer (j). Especially, in each fragment pair (a1 and a2), all F_ks of the related Ba_1j and Ba_2j. However, since RGB colour space intensities are used throughout the work presented, three dimensional vectors are obtained for the boundary vectors. The cross correlations between the corresponding channels of the border vectors are calculated. (red channel of a1 - red channel of a2) Then, each cross-correlation vector for red, green and blue for the related pair at the predetermined layer are multiplied and to yield a score vector. Since, the orientations of the fragments are unknown, the cross correlations should be controlled in two directions. First, the cross correlation is calculated in the same manner as the borders located in the boundary vector, and then one of them is reversed and then the cross-correlation calculation is performed again.

Storage of the Data

Five-dimensional arrays are used to store the scores, and for the graph theory applications, implication tables are constructed. The 3D illustration of the array is shared in Figure 7. In the score array, 2nd dimension is used to construct the implication table. In the 3rd dimension, the cross-correlation multiplication in R-G-B channels vector (score vectors) are stored. The 4th dimension is used to store the score vectors layer by layer. To be more specific, if ly is the last layer that is going to be used, then 4th dimension simply constructs the score vectors for the following combinations: 11, 12, 13… 1ly; 21, 22,...,2ly;ly1,ly2,...lyly. As mentioned earlier, the score vector is calculated first with the same locations of the borders vectors as they are in their boundary vectors and secondly, with one of the border vector is flipped. Hence, another 5th dimension is created for the same implication table, scores vector and layers in which one of the border vector is reversed.



Product of Cross Correlations in R-G-B Channels
1
2
3
2
Number of Borders in Fragment 2
K2
1
2
3
3
Number of Borders in Fragment 3
K3
M-1
1 2 3 ... K1 1 2 3 ... K2
1 Number of Borders in Fragment 1
2 Number of Borders in Fragment 2
3
1 2 3 ... KM
M Number of Borders in Fragment M
Number of Fragmentes

Figure 7: 3D Score Array Illustration

Periodicity of the Lines

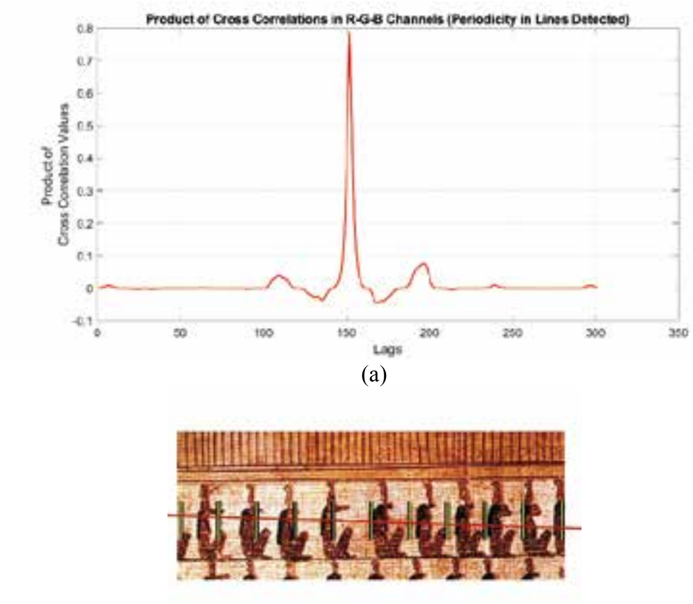
The periodicity of lines detection is also used as a metric which strengthens the score between two fragments. This signals that there is a tight correlation between the boundaries of two fragments. The detection is accomplished by checking the scores vector (product of cross correlations in R-G-B channels). When there are symmetrical peaks around the maximum point, as in Figure 8(a), it can be deduced that precise shifts (amount of the period) of the vectors results in again high correlation, which can be seen at the Figure 8(b). Then, after periodicity detection, the score between the related fragments is increased exponentially.

Decision Unit and Score Rearrangement

The score assignment initially starts automatically. First, the orientation of the vectors of interest are decided by checking the maximum values of the scores vector calculates between the 1st layers of the fragments. The correct orientation (whether the vector is reversed in the boundary vector, or not) although the fragments are not matched at all, yields higher acme among the score vector than the uncorrected orientation of the two border vectors. Subsequently, we also encode the correct orientation of the vectors to the scores array.

The designed algorithm iteratively goes over the layers up to the predetermined ly (maximum number of layers to be used) as it is illustrated in the Figure 8 for the most likely matches. The peak value of the score matrix in each layer is checked by the threshold and the shape of the score matrix is also considered as a decision metric. Mostly for the correct matches, the score vector shows an impulsive behaviour around the principally correlated locations of the border vectors. Moreover, at the inner layers of the score vector, it is expected that the impulsive behaviour will continue with a lower peak, but with the almost same principal location as it can be seen at Figure 5. However, for the wrong matches this impulsive behaviour is not observed.

The algorithm presents the most likely matches to the expert/archaeologist after the steps shown in Figure 6, starting from the furthest correlated fragments. Knowledge of the context, such as the archaeological excavation environment, historical cognisance and human intelligence here take place and decide whether the proposed match is correct or not. With this decision the scores are re-adapted and if the given match by the algorithm is rejected the corresponding score becomes *NaN,* and not used again.



(a)

(b)

Figure 8: Plot of the score vector (a) and the periodic lines that we detected by our algorithm(b)

4 Conclusion and Future Work

In this paper we presented a novel interactive image reconstruction algorithm. Our method consists mainly 3 parts. First, we extract the boundary vector and the corners of the fragments. Secondly, we construct a graph based on the correlation between the inner boundaries of the fragments. Afterwards, the expert/archaeologist is introduced to contribute to the process and to control the proposed algorithmic matches. Our results from initial trials show that the correct matches are determined.

Although our study is only a small step, we believe that this model will insert another insight, and will be useful for the construction of future image reassembly algorithms. In the future, the 3D space orientation of the fragments can be added as a metric for the matches of fractured pairs. Moreover, this method can be applied to the complex data-sets with imprecise boundaries and locally corrupted 3D fragments.

References

[1] A. R. Willis and D. B. Cooper, "Computational reconstruction of ancient artifacts," *IEEE Signal Processing Magazine*, vol. 25, no. 4, 2008.

[2] G. Papaioannou, E.-A. Karabassi, and T. Theoharis, "Virtual archaeologist: Assembling the past," *IEEE Computer Graphics and Applications*, vol. 21, no. 2, pp. 53-59, 2001.

[3] D. Koller and M. Levoy, "Computer-aided reconstruction and new matches in the forma urbis romae*," Bullettino Della Commissione Archeologica Comunale di Roma*, vol. 2, 2006.

[4] E. Tsamoura and I. Pitas, "Automatic color based reassembly of fragmented images and paintings*," IEEE Transactions on Image Processing*, vol. 19, no. 3, pp. 680-690, 2010.

[5] M. S. Sagiroglu and A. Ercil, "A texture based matching approach for automated assembly of puzzles," in *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*, vol. 3. IEEE, 2006, pp. 1036-1041.

[6] F. Amigoni, S. Gazzani, and S. Podico, "A method for reassembling fragments in image reconstruction," in *Image Processing, 2003. ICIP 2003. Proceedings. 2003 International Conference on*, vol. 3. IEEE, 2003, pp. III-581.

[7] K. Zhang and X. Li, "A graph-based optimization algorithm for fragmented image reassembly," *Graphical Models*, vol. 76, no. 5, pp. 484-495, 2014.

[8] W. Kong and B. B. Kimia, "On solving 2d and 3d puzzles using curve matching," in *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, vol. 2. IEEE, 2001, pp. II-II.

[9] H. C. da Gama Leiǎto and J. Stol, "A multiscale method for the re-assembly of two-dimensional fragmented objects," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 9, pp. 1239-1251, 2002.

[10] H. Shin, C. Doumas, T. A. Funkhouser, S. Rusinkiewicz, K. Steiglitz, A. Vlachopoulos, and T. Weyrich, "Analyzing fracture patterns in theranwall paintings." in *VAST*, 2010, pp. 71-78.

[11] C. Papaodysseus, M. Panagopoulos, P. Rousopoulos, G. Galanopoulos, and C. Doumas, "Geometric templates used in the akrotiri (thera) wall paintings," *antiquity*, vol. 82, no. 316, pp. 401-408, 2008.

[12] B. J. Brown, C. Toler-Franklin, D. Nehab, M. Burns, D. Dobkin, A. Vla-chopoulos, C. Doumas, S. Rusinkiewicz, and T. Weyrich, "A system for high-volume acquisition and matching of fresco fragments: Reassem-bling theran wall paintings," in *ACM transactions on graphics (TOG)*, vol. 27, no. 3. ACM, 2008, p. 84.

[13] C. Toler-Franklin, B. Brown,T. Weyrich, T. Funkhouser, and S. Rusinkiewicz, "Multi-feature matching of fresco fragments," in *ACM Transactions on Graphics (TOG),* vol. 29, no. 6.ACM, 2010, p. 185