Received: 18 August 2018 | Revised: 25 May 2019 | Accepted: 12 December 2019

DOI: 10.4218/etrij.2018-0461

ORIGINAL ARTICLE

ETRI Journal WILEY

Reconstruction algorithm for archaeological fragments using slope features

Nada A. Rasheed[1] | Md Jan Nordin[2]

[1]Ministry of Higher Education and Scientific Research, Al-Karkh University of Science, Baghdad, Iraq

[2]Ministry of Higher Education, Universiti Kebangsaan Malaysia, Selangor, Malaysia

Correspondence
Nada A. Rasheed, Al-Karkh University of Science, Baghdad, Iraq.
Email: nadaar@siswa.ukm.edu.my

Funding information
This research was supported by the Universiti Kebangsaan Malaysia of Ministry of Higher Education, Malaysia, (FRGS/1/2014/ICT07/UKM/02/2, DIP-2016-2018).

The reconstruction of archaeological fragments in 3D geometry is an important problem in pattern recognition and computer vision. Therefore, we implement an algorithm with the help of a 3D model to perform reconstruction from the real datasets using the slope features. This approach avoids the problem of gaps created through the loss of parts of the artifacts. Therefore, the aim of this study is to assemble the object without previous knowledge about the form of the original object. We utilize the edges of the fragments as an important feature in reconstructing the objects and apply multiple procedures to extract the 3D edge points. In order to assign the positions of the unknown parts that are supposed to match, the contour must be divided into four parts. Furthermore, to classify the fragments under reconstruction, we apply a backpropagation neural network. We test the algorithm on several models of ceramic fragments. It achieves highly accurate results in reconstructing the objects into their original forms, in spite of absent pieces.

KEYWORDS
3D geometry, archaeological fragments, ceramic, neural network, pottery

1 | INTRODUCTION

Pottery is considered to be the most common material used by humans over the past several thousand years [1]. Archaeology is a discipline that studies the remnants of early human civilization [2], exploring the lives of ancient peoples by examining their remains. Artifacts are often found at archaeological sites in a fragmented state. The process of manually reconstructing large numbers of small pieces of unknown objects—for which thousands of irregular fragments could be involved—is difficult and expensive. Additionally, the process may require years of tedious work [3]—especially in case of absent pieces—or could require laborious effort and experienced archaeologists [2,4]. Significant developments in computer technologies have improved the reconstruction procedures for fragmented archaeological objects [5–9]; a large proportion of this has relied on the fact that pots are axially symmetric. Some researchers have proposed a complete framework with which to automatically assemble pots from 3D fragments [10,11]. These authors have assumed that the objects have been made using a potter's wheel, and as such they rely upon the axial symmetry around the center of an object in order to reconstruct it from its fragments. Therefore, these approaches cannot be applied in the case of axially asymmetrical objects. The first aim of this work is to propose a novel algorithm to form part of a robust prototype procedure for the reconstruction of 3D objects, despite the existence of gaps, by exploiting geometric features (in particular, the slopes of the fragment edges), as well as a method of finding appropriate locations for matching. Finally, this work aims to propose a new method for aligning and matching two different edges of a pair of fragments and finding the corresponding point set, which is important to matching many types of fragments.

2 | SYSTEM OVERVIEW

It has become very popular to use three-dimensional measurements to reconstruct archaeological artifacts [12]. The main focus of this section is to propose a system for reconstructing the original form of ancient 3D objects; to achieve this, we propose a method comprised four major phases:

- 3D model acquisition using a 3D laser scanner device.
- Extraction of features from fragment edges.
- Application of a backpropagation neural network (BNN) algorithm for classification.
- Application of a Euclidean distance algorithm for matching pairs of fragments.

2.1 | Datasets for 3D model acquisition

In order to apply the proposed method, this work uses the 3D point clouds of modern ceramic fragments, obtained via a 3D laser scanner. This study uses the edges of new pottery because they are sharp and easy to analyze; however, for old pottery edges that have been partially erased, an algorithm for optimizing the edges should be applied prior to the proposed method. The 3D laser scanner has an image resolution of 640 × 480 visual grading analysis, and images are taken under invariant illumination to avoid the influence of shadows. In order to obtain a full 3D model, the ceramic object is scanned from all sides [13] (ie, from different angles) as the table rotates slowly in front of the scanner. After the scan has been completed, the fragments are stored in a *Stl* file format using the MATLAB application. We use the real datasets, which consist of two groups; each group represents one vessel and is comprised of three fragments, as shown in Figure 1.

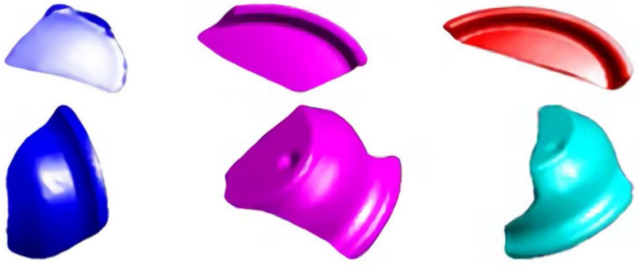


FIGURE 1 Two groups of 3D fragments

2.2 | Feature extraction

Feature extraction is an important component of pattern recognition. Typically, the dimensions of the feature are less than those of the original image size [14]. Therefore, this process facilitates the subsequent classification process. Generally, edges are considered an important feature for image analysis and computer vision [15] because they reflect useful information about the orientation and location of each edge point [16]. One of the first steps archaeological workers undertake when trying to assemble fragments is to match the edges of the fragments, that is, to extract its geometric features [17]. Next, they take the edges of the fragments into consideration and examine their textures. Therefore, we use the edges of the fragments to reconstruct the objects. Detecting the contours of an object from its surface's 3D point cloud is a challenging task, so the following procedure is applied to obtain them.

Step 1: Find all edges in the mesh and note which internal edges are repeated.

Step 2: Determine the uniqueness of the edges.

Step 3: Determine the pixel count for each unique edge.

Step 4: Extract edges that occur only once.

As mentioned above, the goal of this work is to determine the correct location from which to begin matching a pair of fragments, then from there to continue matching the rest of the parts of the fragment pair. If we suppose any object has four directions (North, South, East, and West) such that the contour of the fragment is divided into four parts of as equal a size as possible. This facilitates the identification of the region from which to start the matching process. Each part is divided into sub-contours, and each sub-contour consists of five points. To obtain the features, we use the slope of the sub-contour, which is a basic feature that can be employed to classify fragments. Figure 2 and the following algorithm demonstrate the 3D slope of each part [18].

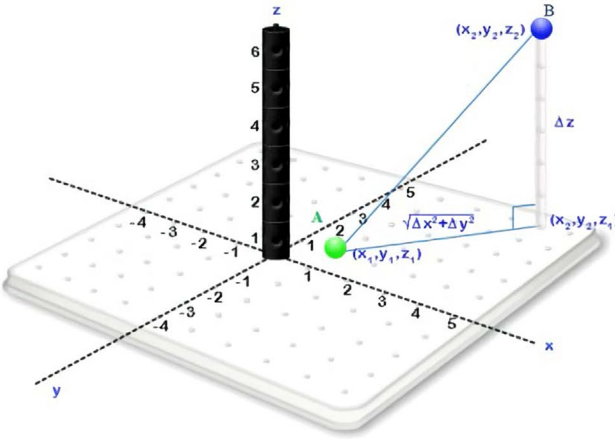


FIGURE 2 Visualization of a 3D slope

Step 1: Suppose that the first point is A, with the 3D coordinates (x_1, y_1, z_1), and that the last point is B, with the 3D coordinates (x_2, y_2, z_2).

Step 2: Draw a line passing through the points A and B.

Step 2.1: The components of the vector AB are:

$$\Delta x = \left(x_b - x_a\right), \tag{1}$$

$$\Delta y = \left(y_b - y_a\right), \tag{2}$$

$$\Delta z = \left(z_b - z_a\right). \tag{3}$$

Step 2.2: Compute the vertical difference between the two 3D points, as

$$\text{Rise} = \Delta z. \tag{4}$$

Step 2.3: Compute the horizontal distance between the two 3D points, as

$$\text{Run} = \sqrt{(\Delta x^2 + \Delta y^2)}. \tag{5}$$

Step 3: Calculate the slope using

$$\text{Slop} = \frac{\text{Rise}}{\text{Run}}. \tag{6}$$

The other features adopted in this work are the minimum, maximum, mean, and variance between the points of each part, which are calculated for the coordinates of each point (x, y, and z). This work follows the algorithm below.

Step 1: Calculate the minimum x_i, minimum y_i, and minimum z_i.

Step 2: Calculate the maximum x_i, maximum y_i, and maximum z_i.

Step 3: Calculate the mean x_i, mean y_i, and mean z_i.

$$\bar{x} = \frac{\sum_i^n x_i}{n} \tag{7}$$

where $n = 5$ points.

Step 4: Calculate the variance of x_i, variance of y_i, and variance of z_i.

$$V(x) = \frac{\sum_i^n \left(x_i - \bar{x}\right)}{n}. \tag{8}$$

2.3 | Recognition process

Our approach aims to reconstruct pottery fragments by exploiting the fact that the two edges of the fragments match when they have the same geometry [19]. Therefore, to detect the joints between a pair of sub-contours according to the similarity features, we use a BNN algorithm; this is a powerful mapping network that has been applied successfully to a wide range of problems [20]. The BNN algorithm is a supervised training network, it is composed of two fundamental procedures: feed-forward propagation and feed-backward propagation. Feed-forward propagation involves sending signals through the network from the input neurons and obtaining an output. Backpropagation allows the network to learn from its mistakes. After training, a testing phase is carried out by implementing the feed-forward procedure without changing the weight matrices. The BNN algorithm is as follows [21,22].

Step 1: Initialize weights and offsets.

Set all weights to small random values between (–1 and 1).

Step 2: Select inputs and desired outputs.

Select a continuous-valued input vector (x_0, x_1, ..., x_{n-1}) and specify the desired outputs (d_0, d_1, ..., d_{m-1}). If the net is used as a classifier then all desired outputs are typically set to zero except for the output corresponding to the class the input is from, for which the desired output is 1. The input can differ for each trial or samples from a training set can be presented repeatedly until weights stabilize.

Step 3: Calculate actual outputs.

Use the sigmoid nonlinearity of the following equation to calculate the outputs y_0, y_1, …, y_{m-1}.

$$f(\text{net}_i) = \frac{1}{1 - e^{-\text{net}}}. \tag{9}$$

Step 4: Adapt weights.

Use a recursive algorithm that starts at the output nodes and works backwards to the first hidden layer. Adjust weights using the following equation.

$$w_{ij}(t+1) = w_{ij}(t) + \eta \delta_j x'_i. \tag{10}$$

In this equation, $w_{ij}(t)$ is the weight between the hidden or input node i to node j at time t; w'_j is either the output of node i or is an input; η is a gain term; and δ_j is an error term for node j. If node j is an output node, then

$$\delta_j = y_j(1 - y_j)(t_j - y_j), \tag{11}$$

where d_j is the desired output of node j and y_j is the actual output.

If node j is an internal hidden node, then

$$\delta_j = x'_j(1 - x'_j) \sum_k \delta_j^m w_{jk}. \tag{12}$$

Here k is summed over all nodes in the layers in front node j. Internal node thresholds are adapted in a similar manner by assuming that they are the connection weights on links from auxiliary constant-valued inputs. Convergence is sometimes

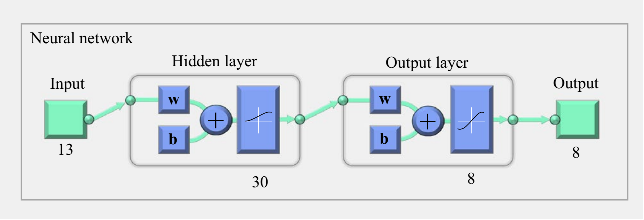


FIGURE 3 Schema of network implementation

faster if a momentum term is added and the weight change is smoothed out by

$$w_{ij}(t+1)=w_{ij}(t)+\eta\delta_j x_i'+\alpha\left(w_{ij}(t)-w_{ij}(t-1)\right) \tag{13}$$

where $0<\alpha<1$.

Step 5: Repeat by returning to Step (2) until $y \approx t$, that is, when y is roughly equal to t.

After extracting feature vectors, we use matrices (13×144), where the second fragment consists of 67 sub-contours and the third fragment consists of 77 sub-contours; thus the sum is equal to 144 sub-contours to represent the two fragments; each one consists of four parts that are available in the system's database. Before training a neural network, we must normalize the inputs and the network. Furthermore, the database contains the values of the desired output layer. The outputs of the appropriate 8-bit code are arbitrarily assigned to each original input pattern. The network structure consists of three layers: the first is the input layer, consisting of 13 nodes in addition to the weights and biases; the second layer represents the hidden layer, consisting of 30 nodes and was found suitable for this problem through experiment; the last layer is the output layer, consisting of eight units that represent the eight classes of patterns. It was noted that a learning rate of 0.05 and a momentum term of 0.9 can be used to give the best recognition results. Test data are in the range of (1, –1) [2]. The input will be the first layer of the network, so it will pass signals to the hidden layer nodes of backpropagation. The implementation of the network is shown in Figure 3.

2.4 | Aligning 3D point cloud of fragments

Immediately after recognizing the sub-contours used to match fragment pairs, an alignment algorithm is applied to match the pairwise 3D point clouds representing the fragments. The two fragments have smooth surfaces of various shapes and sizes; each mesh contains $3 \times N$ and $3 \times M$ matrices, where M and N are different lengths ranging between 3000 to 9000 points. The 3D alignment method is considered a difficult task by many authors [23]. The reason for this is that it needs to identify the angle by which the fragment surfaces rotate around the z-axis. Moreover, a transformation algorithm needs to be applied to one of the clouds in order to bring it as close as possible to the others, to achieve the best-fitting alignment. This is shown in Figure 4.

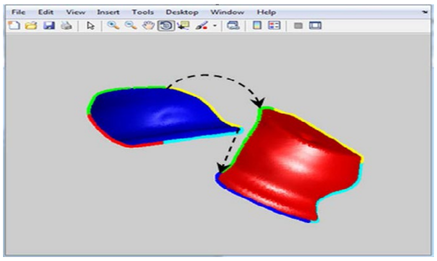


FIGURE 4 Rotation and transformation of one fragment onto another

To address the problem of the alignment of 3D cloud points, several authors have introduced methods of applying various algorithms, such as the iterative closest point and normal distribution transform algorithms [24]. Unfortunately, most of these methods are only able to align two or more scans via their surfaces, whereas this work needs to match the two sides of different fragments. Therefore, this work applies an algorithm to match two edges from different fragments. For the two fragments A and B, one is considered fixed, and the other follows. Recognizing pairs of sub-contours between the fragment edges of A and B will indicate the same point in the space and be considered a match. Often this is performed simply by matching each point with its closest neighbor in the other cloud. In this case, the rotation angle R and the translation T between the two fragments must be computed. To obtain a random alignment between the two fragments, the following algorithm is implemented on a fixed fragment (A).

Step 1: Find the center of fragment A,

$$p=\begin{bmatrix} x \\ y \\ z \end{bmatrix}, \tag{14}$$

where the xyz data of fragment (A) is

$$\text{Centroid}_A=\frac{1}{N}\sum_{i}^{N} p_N^i. \tag{15}$$

Step 2: Accumulate a matrix (H) as

$$H=\sum_{i=1}^{N}(p_A^i-\text{Centroid}_A). \tag{16}$$

Step 3: Use singular value decomposition (SVD) to find the direction of most variance, and rotate the data to make this the x-axis, as follows:

$$[U,S,V]=\text{svd}(H,0), \tag{17}$$

where U has dimensions $m \times n$ and is orthogonal (its columns are eigenvectors of AAT); V has dimensions $n \times n$ and is orthogonal (its columns are eigenvectors of ATA); and D has dimensions $n \times n$ and is diagonal (it contains non-negative real values known as singular values). If H has dimensions $m \times n$ and m is greater than n, then SVD computes only the first n columns of U and S has dimensions $n \times n$.

$$R = V \times U^T. \tag{18}$$

Here, V is the direction of greatest variance.

Step 4: Move the data up the x-axis so all the points are at $x \geq 0$, using

$$T = -R \times \text{Centroid}_A. \tag{19}$$

After bringing the A dataset to the origin by applying the above algorithm, the optimal rotation (matrix R) and the translation T should be determined. As shown in Figure 5, in order to match the edges of fragment B with the corresponding points along the edges of fragment A, the rotation and transformation processes should be applied.

Any 3D object can be rotated around one of the three axes (x-axis, y-axis, and z-axis), according to the rotation angles α_x, α_y, and α_z [23], as shown in Figure 6. To achieve this, the angles between 3D points should be computed.

Once the pairs of 3D sub-contours for different fragments have been recognized, they can be used as the start of the fragment matching procedure. Consider the following three points:

$P_1 = (x_1, y_1, z_1)$ represents the first point on the sub-contour of the moving fragment.

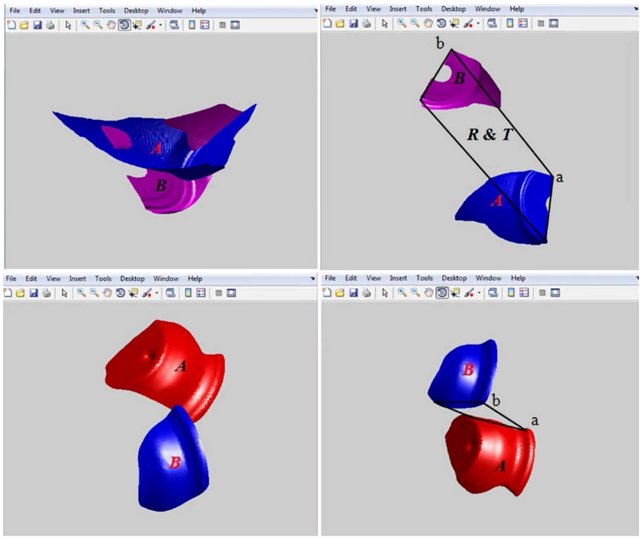


FIGURE 5 (A) Before application of algorithm; (B) After centralization of fixed fragment

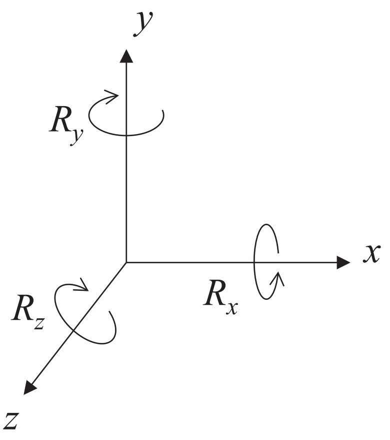


FIGURE 6 Rotation of 3D matrices around 3D Cartesian coordinate axes [23]

$P_2 = (x_2, y_2, z_2)$ represents the first point on the sub-contour of the fixed fragment.

$P_3 = (x_3, y_3, z_3)$ represents the last point on the sub-contour of the fixed fragment.

As shown in Figure 7, the vector from P_1 to P_2 can be obtained by subtracting the coordinates of P_1 from the coordinates of P_2 [25], via

$$\overrightarrow{P_1P_2} = [x_2 - x_1, y_2 - y_1, z_2 - z_1]. \tag{20}$$

The magnitude of a vector between two points can be obtained according to the following formula [26]:

$$\left|\overrightarrow{P_1P_2}\right| = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}. \tag{21}$$

To find the angle between the two vectors using the dot product definition, let $\vec{v_1} = \langle v_{1x}, v_{1y}, v_{1z} \rangle$ and $\vec{v_2} = \langle v_{2x}, v_{2y}, v_{2z} \rangle$ be vectors in 3D space [19], thus

$$\vec{v_1} = \langle v_{1x}, v_{1y}, v_{1z} \rangle, \tag{22}$$

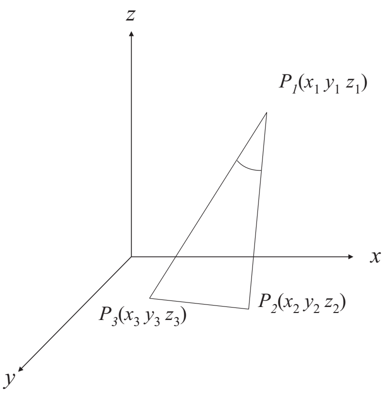


FIGURE 7 Coordinates of three points

$$\vec{v_2} = \left\langle v_{2x}, v_{2y}, v_{2z} \right\rangle, \tag{23}$$

$$|\vec{v_1}| = \sqrt{v_{1x}^2 + v_{1y}^2 + v_{1z}^2}, \tag{24}$$

$$|\vec{v_2}| = \sqrt{v_{2x}^2 + v_{2y}^2 + v_{2z}^2}, \tag{25}$$

$$\vec{v_1}.\vec{v_2} = v_{1x}.v_{2x} + v_{1y}.v_{2y} + v_{1z}.v_{2z}, \tag{26}$$

$$\cos(\alpha) = \frac{\vec{v_1}.\vec{v_2}}{|\vec{v_1}|\,.\,|\vec{v_2}|}, \tag{27}$$

$$\alpha = \cos^{-1}\left(\frac{\vec{v_1}.\vec{v_2}}{|\vec{v_1}|\,.\,|\vec{v_2}|}\right). \tag{28}$$

In this case, the angle α was obtained as the inverse cosine ($\cos^{-1}$, measured in degrees) of the elements of the matrix. Next, the object can be rotated about the *x*-axis, *y*-axis, or *z*-axis, using the matrices below [23].

About the *x*-axis:

$$R_x(\alpha_x) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\alpha_x) & -\sin(\alpha_x) \\ 0 & \sin(\alpha_x) & \cos(\alpha_x) \end{bmatrix}. \tag{29}$$

About the *y*-axis:

$$R_y(\alpha_y) = \begin{bmatrix} \cos(\alpha_y) & 0 & \sin(\alpha_y) \\ 0 & 1 & 0 \\ -\sin(\alpha_y) & 0 & \cos(\alpha_y) \end{bmatrix}. \tag{30}$$

About the *z*-axis:

$$R_z(\alpha_z) = \begin{bmatrix} \cos(\alpha_z) & -\sin(\alpha_z) & 0 \\ \sin(\alpha_z) & \cos(\alpha_z) & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{31}$$

Next, to provide a solution for the object rotation, (31) is applied.

$$C = P \times R \tag{32}$$

where *C* is the new 3D point cloud, *P* is the old 3D point cloud, and *R* represents the rotation matrices. As shown in Figure 8, the optimum rotation is about the *x*-axis, hence this work uses this type. Another problem is how to add the different distances between the two object's coordinates, to transform (*T*) the object and match the two sides of the pairs of different objects.

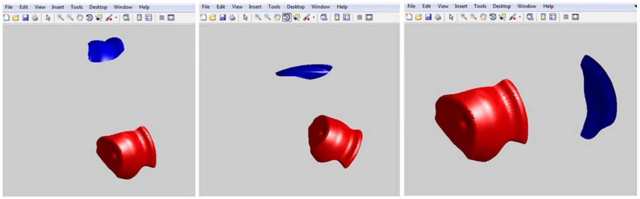

File Edit View Insert Tools Desktop Window Help
File Edit View Insert Tools Desktop Window Help
File Edit View Insert Tools Desktop Window Help

FIGURE 8 Rotation of object around three axes

Therefore, in order to provide a solution for *R* and *T*, as shown in (32), we suppose *B* is the 3D point cloud data of the moving fragment, thus

$$C = P_B * R + T. \tag{33}$$

Here *R* and *T* are the transformations applied to the 3D point cloud P_B, to align it with 3D point cloud *A* as much as possible.

Finally, in order to obtain the optimal match, the Euclidean distance formula is applied between the coordinates of each point on the sub-contour of fragment *B* and the coordinates of each point on the corresponding sub-contour of the fixed fragment, the shortest distance is then chosen. Given two points *a* and *b*, the Euclidean distance is:

$$d(a,b) = \sqrt{(a_x - b_x)^2 + (a_y - b_y)^2 + (a_z - b_z)^2}. \tag{34}$$

Given the point *b*, and a set of points *A*, the Euclidean distance is:

$$d(b,A) = \min_{i \in 1,\ldots,n} d(b, a_i), \tag{35}$$

where *b* and a_i indicate the values of the points that represent the two sub-contours which have been classified as *A* and *B*; *n* is the vector size.

3 | EXPERIMENTAL SETUP

The aim of this experiment is to employ these features, including the slope, to classify and reconstruct objects. The following steps explain the conducted experiment in detail.

3.1 | 3D model

After executing the algorithm, the 3D fragment model files are loaded in the memory. Figure 9 shows the original fragments and the 3D models of the fragments. These three fragments represent one vase.

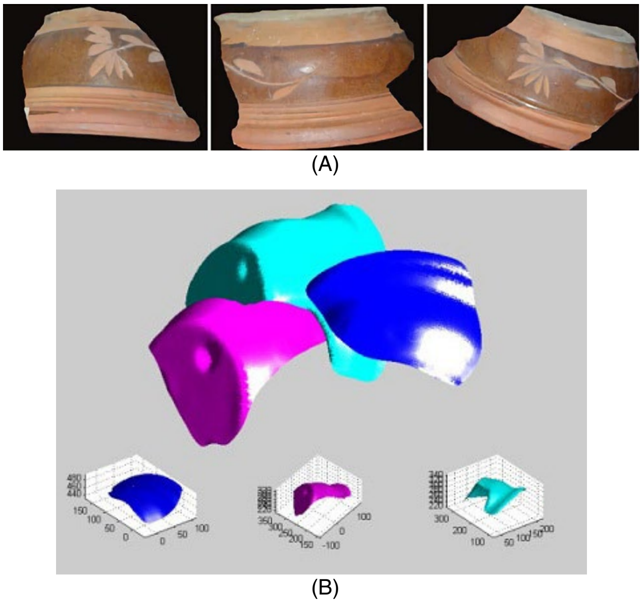


FIGURE 9 (A) Three original fragments; (B) 3D models of fragments uploaded to system

3.2 | Feature extraction

Figure 10 shows the results of applying the procedure used to obtain the contours for each fragment. In order to recognize the region that matches with the corresponding part, each contour is divided into four approximately equal parts, as shown in Figure 11. Each part is independent of the others.

Next, for each fragment, a procedure is applied to divide each part into sub-contours. The five points in the 3D coordinate system (x_i, y_i, z_i) represent the size. The first fragment consists of 51 sub-contours, the second consists of 67 sub-contours, and the last consists of 77 sub-contours, for all four parts. The results of applying the slope algorithm are shown in Tables 1 and 2. Table 1 shows the slope of the first fragment according to the 51 sub-blocks and Table 2 shows the slope of the second fragment according to the 67 sub-blocks, for all four parts.

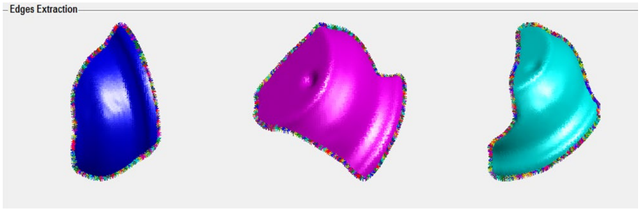


FIGURE 10 Boundaries of three fragments

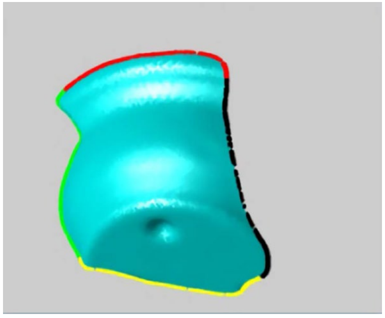


FIGURE 11 Dividing the contour into four equal-sized segments

Figure 12 shows the slope of sub-contours for the pairs of fragments. As can be seen inside the rectangle, the parts of the two curves have approximately similar shapes. This is evidence that the two fragments may be matched in the region between the 17th and 43rd points because here the two different fragments have a similar slope where the two segment curves begin to match. This result is obtained when we apply the recognition algorithm. In addition, for each sub-contour the maximum, minimum, mean, and variance values are extracted; Tables 3–6 demonstrate these quantities for one fragment, respectively.

3.3 | Recognition process

In order to recognize the parts of the fragment that match with the corresponding part of another fragment, we apply the neural network tools in the MATLAB software to the features of the sub-contours.

Figure 13A and 13B display the implementation of the network. During the training of the neural network, updates to the known datasets and desired outputs continuously appear in the window. This window contains the performance, the magnitude of the performance gradient, and the number of validation checks. It is worth noting that the training stops after the mean square error reaches a minimum of 1.00e^{-5}. Furthermore, the window of the neural network training reflects the maximum number of training epochs, which represents the iterations required to complete training. In this experiment, although we assumed 1000 epochs, the network completed training in 391 iterations, in which an epoch is one cycle through the entire set of training vectors; the weights are updated during each epoch until the error between the output and target is at a minimum. From the training window, we can access the regression plot that demonstrates the relationship between the outputs of the network and the targets in four plots (training, validation, test, and all performances). Here, the training *R* is 0.98444, and for all it is 0.9869. This indicates that there is an exact linear relationship between outputs and targets.

The purpose of learning sample data is to classify and predict new data successfully. After completion of all steps, the performance of the trained network is evaluated by testing new data inputs in the network. In order to classify the unknown part of the fragment, we feed the data for the unknown part into the input layer of the network and test it through the feed-forward phase. Next, we compute the actual output; a

TABLE 1 Slope for each part of the first fragment

No	Slope/Part 1	No	Slope/Part 2	No	Slope/Part 3	No	Slope/Par t4
1	0.247112	14	0.048806	27	–0.05442	40	–0.25479
2	0.180028	15	–0.03905	28	–0.1421	41	0.010533
3	0.108561	16	–0.05184	29	–0.19199	42	0.292315
4	–0.06668	17	–0.04528	30	–0.23394	43	0.739107
5	–0.33496	18	0.0225	31	–0.36583	44	0.75811
6	–0.38933	19	0.380067	32	–0.43145	45	0.70935
7	–0.39305	20	0.460829	33	–0.49704	46	0.603671
8	–0.53467	21	0.810121	34	–0.55073	47	0.489838
9	–0.53807	22	0.342211	35	–0.65745	48	0.277209
10	–0.11877	23	0.146291	36	–0.76818	49	0.222547
11	0.05065	24	0.201791	37	–0.79443	50	0.282332
12	0.160863	25	0.093604	38	–0.79318	51	0.27005
13	0.134512	26	0.0173	39	–0.25052		

TABLE 2 Slope for each part of the second fragment

No	Slope/Part 1	No	Slope/Part 2	No	Slope/Part 3	No	Slope/Part 4
1	1.0619	18	–0.0874	35	–0.668	52	0.8772
2	1.1551	19	–0.3539	36	–0.5092	53	0.3118
3	1.1235	20	–0.2252	37	0.1681	54	0.0572
4	0.8297	21	–0.1796	38	0.4203	55	–0.0959
5	0.5592	22	–0.1747	39	0.5484	56	–0.3523
6	0.3583	23	–0.2286	40	0.6561	57	–0.1025
7	0.1263	24	–0.3089	41	0.7012	58	0.1453
8	0.0508	25	–0.3957	42	0.1487	59	0.0568
9	0.0182	26	–0.4395	43	–0.3022	60	0.0485
10	–0.204	27	–0.4758	44	–0.4493	61	0.0707
11	–0.3542	28	–0.5678	45	–0.3509	62	0.1016
12	–0.5275	29	–0.6162	46	–0.2733	63	0.0854
13	–0.6315	30	–0.6296	47	–0.2771	64	0.229
14	–0.675	31	–0.7	48	–0.062	65	0.9327
15	–0.426	32	–0.7533	49	0.2894	66	1.0712
16	0.0555	33	–0.8038	50	0.5936	67	0.6909
17	0.4154	34	–0.9029	51	0.7893		

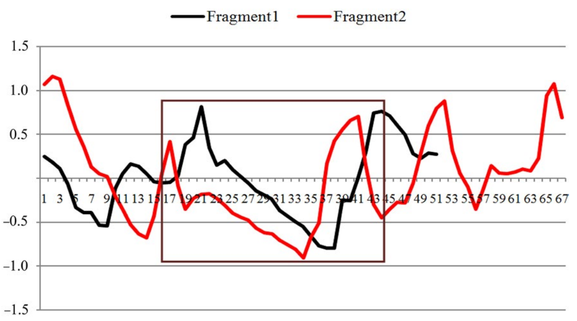


FIGURE 12 Slope for all sub-contours of a pair of fragments

part of the fragment is a successful match if the maximum value between the computed output nodes is close to 1.

Table 7 shows the result of the first part of the first fragment. We can see the matching of a pair of fragments that begins in the first sub-contour and the first node, this represents the first part of the corresponding fragment because it achieves 0.99769, it is a maximum value across eight nodes. Here, we adopt the maximum value across eight nodes because we assign 1 to the present part and zeros otherwise.

In order to match the pairs of sub-contours for different parts, we measure the distance between the sub-contour of

TABLE 3 Maximum values for each part of the first fragment

	Part 1				Part 2				Part 3				Part 4		
No	*X*1	*Y*1	*Z*1	No	*X*2	*Y*2	*Z*2	No	*X*3	*Y*3	*Z*3	No	*X*4	*Y*4	*Z*4
1	–33.23	65.73	483.1	14	28.8	–43.91	465.9	27	108.3	66.75	491	40	40.73	171.7	434.3
2	–32.26	59.26	485.1	15	40.52	–39.77	466	28	105.4	80.03	490	41	34.67	162.9	431.7
3	–30.89	47.81	486.9	16	53.96	–33.36	465.5	29	102.7	91.85	488.4	42	27.52	153.4	434.6
4	–30.75	31.16	487.1	17	63	–28.12	464.8	30	98.74	105.8	485.7	43	20.61	145.7	443.2
5	–31.52	12.19	485.5	18	71.51	–22.95	464.5	31	93.68	118.4	482.3	44	10.54	138.4	448.9
6	–32.63	–1.066	481	19	80.61	–18.23	467.9	32	88.7	126.8	477.4	45	4.594	133.9	456.3
7	–32.23	–10.93	477.3	20	88	–14.49	471.7	33	85.42	137	473.5	46	–3.466	127.4	462.6
8	–31.16	–19.73	473.7	21	97.44	–9.029	479.7	34	80.12	144	467.4	47	–11.47	118.8	468.5
9	–24.86	–29.89	468.1	22	107.4	0.147	484.6	35	75.57	154.6	462.4	48	–18.22	109.3	472.2
10	–14.85	–36.98	462.6	23	112.5	9.942	486.1	36	69.66	160.4	455	49	–23.44	99.78	474.7
11	–3.074	–41.14	462.1	24	113	25.7	489.2	37	65.66	165.1	449.4	50	–28.09	90.39	476.8
12	7.749	–44.57	463.7	25	112.2	40.07	490.7	38	62.42	172	445.1	51	–31.11	82.03	480.1
13	19.37	–45.89	465.3	26	110.7	54.65	491	39	54.44	174.6	436.9				

TABLE 4 Minimum values for each part of the first fragment

	Part 1				Part 2				Part 3				Part 4		
No	*X*1	*Y*1	*Z*1	No	*X*2	*Y*2	*Z*2	No	*X*3	*Y*3	*Z*3	No	*X*4	*Y*4	*Z*4
1	–33.32	60.47	481.8	14	20.8	–45.69	465.5	27	106.4	57.76	490.5	40	35.56	164.3	432
2	–33.16	49.86	483.4	15	30.9	–43.29	465.6	28	103.2	71.15	488.7	41	28.98	155.3	431.4
3	–32.03	34.04	485.4	16	43.57	–38.46	464.9	29	99.68	81.88	486.4	42	21.67	146.7	432
4	–31.3	14.67	486	17	55.38	–32.59	464.4	30	94.52	94.6	482.9	43	12.07	139.5	435.4
5	–32.49	0.887	481.7	18	63.91	–27.56	464.2	31	89.27	107.8	478.1	44	6	135	444.6
6	–32.76	–9.542	477.7	19	73.56	–21.78	464.9	32	85.98	119.9	474.2	45	–2.178	128.6	450.2
7	–32.59	–17.79	474.6	20	83.55	–16.75	469.4	33	81.18	128.1	468.6	46	–9.446	121.2	457.4
8	–32.14	–29.03	468.7	21	91.24	–12.85	473.8	34	77.16	139	464.2	47	–16.56	111.9	464.3
9	–30.87	–35.97	463.5	22	98.91	–7.887	480.6	35	70.45	146.5	456.1	48	–22.57	101.4	469.7
10	–22.6	–40.27	461.6	23	109.1	2.477	484.9	36	66.79	155.9	450.9	49	–27.66	91.34	472.6
11	–12.51	–44.04	461.6	24	112.4	11.83	486.4	37	63.29	162	446.3	50	–30.38	84.44	475
12	–0.829	–46.04	462.3	25	111.1	28.37	489.6	38	56.2	166.1	438.3	51	–32.9	72.57	477.5
13	8.965	–46.3	463.9	26	108.9	43.23	490.8	39	43.67	173	434.1				

the first fragment—which has been classified as *P*—with the sub-contour of a corresponding fragment *Q*; the matching is achieved when the joint (*P*, *Q*) is minimized, as shown in Figure 14. Then, we reduce the distance by applying the algorithm designed earlier for aligning the 3D point clouds of the fragments. Figures 15 and 16 represent the results of the first and second sets, respectively.

To evaluate the performance of the proposed method, the same pot fragments used we study [11] were analyzed. Their framework depends on the idea that the pot is axially symmetric, so they use the outer surface of the fragments. The features include a rotation of the surface, the profile of the curve, the outer break curves, and the junctions of the outer break curve, whose types are *Y* and *T*. It is assumed that all measurements are distributed according to a Gaussian distribution and that the fragments are reassembled by applying the maximum likelihood estimation. Moreover, the authors applied a Bayesian approach to formulate and merge the four parameters; they manage to assemble 10 of the 13 fragments belonging to one pot, as shown in Figure 17.

After applying the proposed method, the result is an entire reassembled pot in its original shape without any gaps, as shown in Figure 18.

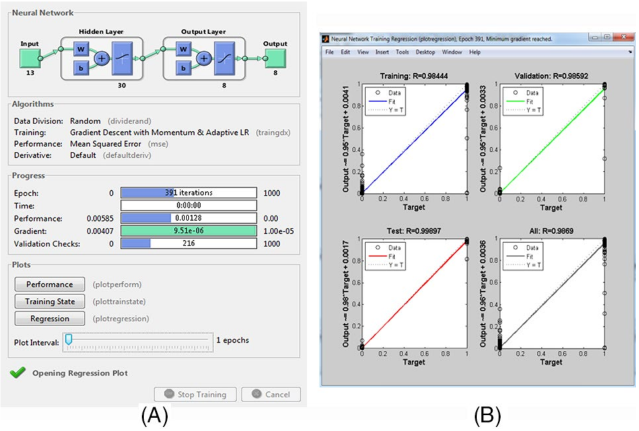
In cases when the pieces were missing due to long storage, because this work uses the advantage of the slope features, it

TABLE 5 Mean for each part of the first fragment

	Part 1				Part 2				Part 3				Part 4		
No	*X1*	*Y1*	*Z1*	No	*X2*	*Y2*	*Z2*	No	*X3*	*Y3*	*Z3*	No	*X4*	*Y4*	*Z4*
1	–33.286	62.826	482.52	14	23.902	–45.052	465.7	27	107.32	62.35	490.78	40	37.65	167.6	433.08
2	–32.798	55.116	484.16	15	35.686	–41.606	465.8	28	104.42	75.186	489.46	41	31.56	158.7	431.52
3	–31.434	41.3	486.18	16	49.636	–35.542	465.16	29	101.30	86.662	487.46	42	24.86	150.24	433.02
4	–30.89	24.44	486.86	17	59.046	–30.458	464.6	30	96.536	100.57	484.24	43	15.92	142.3	439.62
5	–32.03	6.3386	483.6	18	67.882	–25.118	464.32	31	91.044	114.04	479.88	44	8.4356	136.84	446.64
6	–32.698	–4.9458	479.52	19	76.752	–20.164	466.2	32	87.38	123.3	475.84	45	1.1896	131.28	453.3
7	–32.41	–14.338	476	20	85.864	–15.58	470.6	33	83.23	132.8	470.98	46	–6.366	124.48	459.86
8	–31.826	–23.96	471.48	21	94.29	–11.058	476.68	34	78.332	142.08	465.44	47	–14.168	115.24	466.6
9	–27.926	–33.42	465.62	22	103.52	–3.6787	482.92	35	72.488	151.32	458.72	48	–20.552	105.18	471.1
10	–18.22	–38.878	461.92	23	111.02	6.1622	485.46	36	68.302	158.04	453.06	49	–25.466	95.792	473.56
11	–7.7932	–42.654	461.78	24	112.74	19.254	487.96	37	64.58	163.42	447.96	50	–29.358	87.208	475.98
12	3.6864	–45.42	463.04	25	111.7	34.09	490.18	38	59.498	169	441.8	51	–32.244	76.426	479.06
13	14.429	–46.16	464.7	26	109.86	48.854	490.94	39	48.644	173.96	434.86				

TABLE 6 Variance for each part of the first fragment

	Part 1				Part 2				Part 3				Part 4		
No	*X1*	*Y1*	*Z1*	No	*X2*	*Y2*	*Z2*	No	*X3*	*Y3*	*Z3*	No	*X4*	*Y4*	*Z4*
1	0.0016	4.73003	0.277	14	9.9252	0.50582	0.025	27	0.517	11.699	0.037	40	4.2072	8.74	0.887
2	0.1353	14.3953	0.493	15	14.364	1.94143	0.025	28	0.877	13.842	0.303	41	4.6783	8.335	0.017
3	0.2253	32.8453	0.387	16	15.650	3.75457	0.053	29	1.6111	17.625	0.718	42	6.128	7.968	1.262
4	0.0535	39.9185	0.233	17	8.4145	2.89927	0.025	30	3.0935	21.952	1.403	43	12.793	6.7	10.702
5	0.1636	21.7033	2.425	18	8.8159	3.27337	0.012	31	3.1723	18.108	2.817	44	3.2178	1.823	2.873
6	0.0037	10.8305	1.667	19	9.7277	2.46683	1.76	32	1.2623	8.185	1.788	45	7.865	4.807	6.38
7	0.0199	7.04597	1.095	20	2.9075	0.7494	0.79	33	2.8682	12.68	3.877	46	5.8375	6.367	4.378
8	0.1579	13.5323	3.917	21	6.4887	2.44973	6.002	34	1.2986	3.732	1.543	47	4.1606	7.608	2.8
9	6.2066	6.3764	3.712	22	14.104	12.5632	3.092	35	3.8651	9.607	5.747	48	3.0802	10.147	1.015
10	8.9008	1.61282	0.157	23	1.717	8.08593	0.203	36	1.1930	2.948	2.433	49	2.8068	11.151	0.683
11	11.958	1.13398	0.037	24	0.058	31.0951	1.283	37	0.9302	1.597	1.583	50	0.7585	5.0896	0.472
12	11.175	0.339	0.298	25	0.185	21.2297	0.197	38	6.5320	5.885	7.89	51	0.5764	16.044	1.248
13	15.048	0.02755	0.275	26	0.488	19.2861	0.008	39	17.118	0.508	1.403				



Algorithms

Data Division: Random (dividerand)
Training: Gradient Descent with Momentum & Adaptive LR (traingdx)
Performance: Mean Squared Error (mse)
Derivative: Default (defaultderiv)

Progress

Epoch:	0	391 iterations	1000
Time:		0:00:00	
Performance:	0.00585	0.00128	0.00
Gradient:	0.00407	9.51e-06	1.00e-05
Validation Checks:	0	216	1000

Plots

(plotperform)
(plottrainstate)
(plotregression)

Plot Interval: 1 epochs

Opening Regression Plot

Neural Network Training Regression (plotregression), Epoch 391, Minimum gradient reached.

FIGURE 13 (A) Neural network training; (B) Neural network training regression

was found more suitable to determine the best positions when matching pairs of parts.

Therefore, to show the visual performance of the proposed method that reconstruct 3D of 13 fragments into original object, we had conducted the experiment and the results are shown in the Figures 18 and 19.

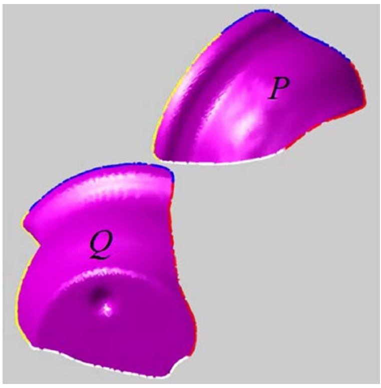


FIGURE 14 Pair fragments of one vessel

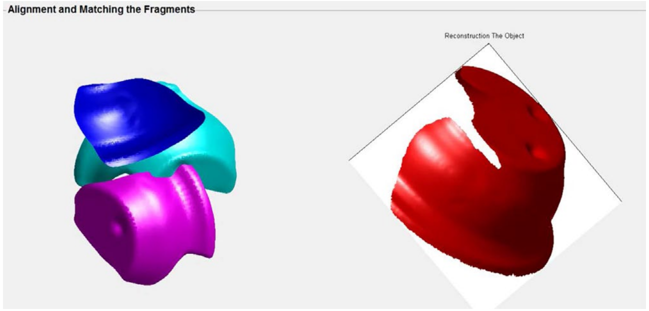

Alignment and Matching the Fragments
Reconstruction The Object

FIGURE 15 Reconstructed object pair fragments of one vessel

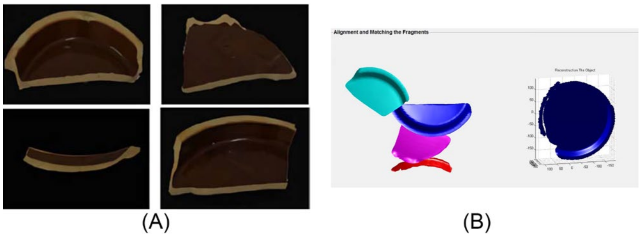

Alignment and Matching the Fragments
Reconstruction The Object
(A)
(B)

FIGURE 16 Matching of fragments in second set: (A) Four original fragments; (B) Reconstruction of object

4 | CONCLUSION

A method for solving the practical problems of reconstructing fragmented objects is proposed in this study. The main principle of the proposed method is to extract the edges of the fragments, using the fact that the edges of the fragments are lines, corners, and curves. The contours of the fragment edges are extracted and divided into four parts. Each part is divided into sub-contours in which each one is composed of five points. Algorithms are applied to calculate the slope, maximum value, minimum value, mean, and variance as the classification features of the BNN. The proposed method achieves impressive results through its employment of the features of each fragment, in particular the slopes of the sub-contours. The effectiveness of the matching partition recognition procedure is remarkable. In addition, the algorithm is tested on a ceramic database and achieves the reconstruction of two sets of broken objects. The first set consists of three fragments and the second set comprises four fragments.

TABLE 7 Results of test process of the first part of the first fragment

No	Sub 1	Sub 2	Sub 3	Sub 4	Sub 5	Sub 6	Sub 7	Sub 8	…	Sub 13
1	0.99769	0.9963	0.9943	0.9943	0.9895	0.9405	0.9432	0.20611	…	1.86E-06
2	0.00565	0.0049	0.0057	0.0057	0.0052	0.0040	0.0047	0.0002	…	1.60E-06
3	5.77E-07	5.17E-07	4.50E-07	4.50E-07	3.74E-07	2.90E-07	1.00E-07	8.89E-07	…	0.42853
4	4.29E-06	3.89E-06	3.08E-06	3.08E-06	2.54E-06	1.36E-06	1.43E-06	1.57E-06	…	4.52E-05
5	2.12E-08	1.71E-08	2.46E-08	2.46E-08	2.61E-08	2.50E-08	1.45E-07	1.12E-07	…	1.24E-06
6	0.00274	0.0029	0.0018	0.0018	0.0015	0.0009	0.0006	0.0013	…	3.75E-06
7	0.00064	0.0016	0.0009	0.0009	0.0013	0.0029	0.0004	0.0330	…	0.91379
8	1.61E-05	1.85E-05	1.32E-05	1.32E-05	1.19E-05	8.51E-06	4.33E-06	5.67E-06	…	0.1788

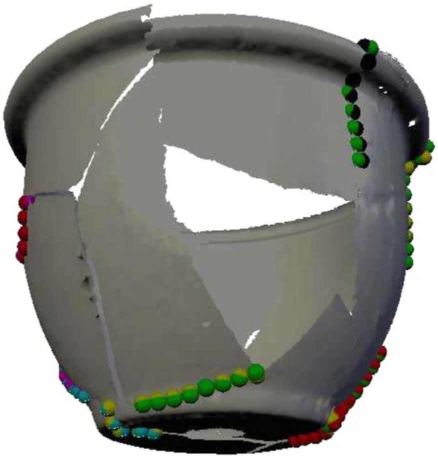


FIGURE 17 Reconstructed pot, with 10 of the 13 fragments matched correctly

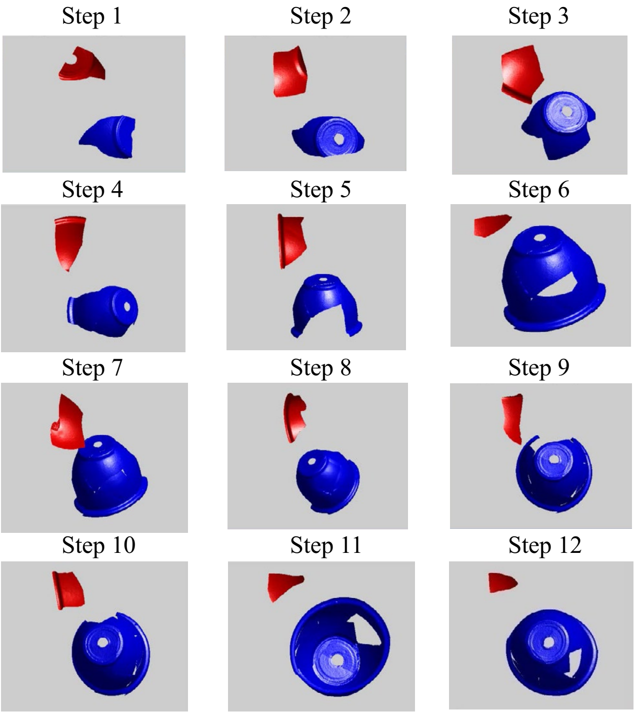


FIGURE 18 Reconstruct the pot that was reconstructed using the proposed method

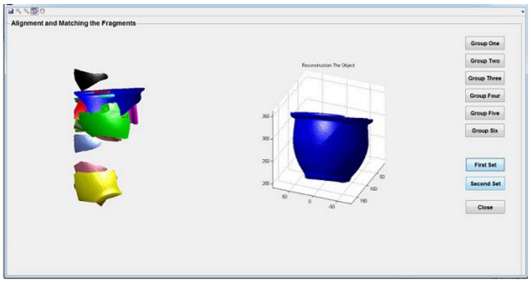


FIGURE 19 Reconstruction of pot achieved using proposed method

We showed that the proposed method outperforms previous methods. In conclusion, we would like to highlight that we achieved promising results using the proposed algorithms.

In this study, we used pottery with sharp edges, so in the future more feature information will be evaluated, in particular the characteristics of the fragment material to reconstruct objects whose edges have been degraded. All the algorithms mentioned in this study were applied to the edges of pottery that had not been degraded.

ACKNOWLEDGMENTS

The researchers thank the Ministry of Higher Education, Malaysia and Universiti Kebangsaan Malaysia (UKM) for their support of this work via the research grants FRGS/1/2014/ICT07/UKM/02/2 and DIP-2016-018.

ORCID

Nada A. Rasheed https://orcid.org/0000-0002-1904-8958
Md Jan Nordin https://orcid.org/0000-0002-7524-5665

REFERENCES

1. D. Tsiafaki et al., *Virtual reassembly and completion of a fragmentary drinking vessel*, Virtual Archaeology Rev. **7** (2016), no. 15, 67–76.
2. K. Son, E. B. Almeida, and D. B. Cooper, *Axially symmetric 3d pots configuration system using axis of symmetry and break curve*, in Proc. IEEE Conf. Comput. Vision Pattern Recogn. (Portland, OR, USA), June 2013, pp. 257–264.
3. J. Liu et al., *Broken pottery relic reassembling based on mixed feature vector*, Boletín Técnico **55** (2017), no. 5, 1–9.
4. L. D. Angelo and P. D. Stefano, *Automatic dimensional characterisation of pottery*, J. Cultural Heritage **26** (2017), 118–128.
5. C. Piccoli et al., *Towards the automatic classification of pottery sherds: two complementary approaches*, in Proc. Comput. Applicat. Quantitative Methods Archaeology Conf. (Perth, Australia), Mar. 2013, pp. 1–16.
6. G. Üçoluk and I. H. Toroslu, *Automatic reconstruction of broken 3-d surfaces objects*, Comput. Graphics **23** (1999), 573–582.
7. G. Papaioannou, E. A. Karabassi, and T. Theoharis, *Reconstruction of three-dimensional objects through matching of their parts*, IEEE Trans. Pattern Anal. **24** (2002), 114–124.
8. M. Kampel and R. Sablatnig, *Profile-Based pottery reconstruction*, in Proc. Conf. Comput. Vision Pattern Recogn. Workshop (Wisconsin, USA), June 2003, pp. 1–6.
9. Y. Lu et al., *Interactive reconstruction of archaeological fragments in a collaborative environment*, in Proc. IEEE Biennial Conf. Australian Pattern Recogn. Soc. Digital Image Comput. Techniques Applicat. (Glenelg, Australia), Dec. 2007, pp. 23–29.
10. C. S. Belenguer and E. V. Vidal, *Archaeological fragment characterization and 3D reconstruction based on projective GPU depth maps*, in Porc. IEEE Int. Conf. Virtual Syst. Mutimedia (Milan, Italy), Sept. 2012, pp. 275–282.
11. A. R. Willis and D. B. Cooper, *Bayesian assembly of 3d axially symmetric shapes from fragments*, in Proc. IEEE Comput. Soc. Conf. Comput. Vision Pattern Recogn. (Washington, DC, USA), 2004, pp. 82–89.
12. M. Pietikäinen et al., *Local binary patterns for still images, chapter 2 of computer vision using local binary patterns*, Computational imaging and vision 40, Springer, Verlag London, 2011, pp. 13–47.

13. R. M. Haralick, K. Shanmugan, and I. Dinstein, *Texture features for image classification*, IEEE Trans. Systems, Man Cybern **3** (1973), 610–621.
14. S. Ding et al., *A survey on feature extraction for pattern recognition*, Artif. Intell. Rev. **37** (2012), 169–180.
15. C. Bähnisch, P. Stelldinger, and U. Köthe, *Fast and accurate 3D edge detection for surface reconstruction*, in Proc. DAGM Symp. Pattern Recogn. (Jena, Germany), 2009, pp. 111–120.
16. B. D. Nhan and S. P. Oh, *High redshift 21cm line simulation data visualization with canny edge detection*, PhD. Dissertation, University of California, Santa Barbara, CA, USA, 2010.
17. N. A. Rasheed and M. J. Nordin, *A polynomial function in the automatic reconstruction of fragmented objects*, J. Comput. Sci. **10** (2014), 2339–2348.
18. D. R. Maidment and D. Tarboton, *Computation of Slope, GIS in Water Resources Class*, University of Texas, Austin, USA, 2011, pp. 1–9.
19. G. Oxholm and K. Nishino, *A flexible approach to reassembling thin artifacts of unknown geometry*, J. Cultural Heritage **14** (2013), 51–61.
20. R. O. Duda, P. E. Hart, and D. G. Stork, *Multilayer neural networks*, In pattern classification, 2nd ed.; Chapter 6, 2000, pp. 10–74.
21. I. Jung and G. N. Wang, *Pattern classification of back-propagation algorithm using exclusive connecting network*, World Academy Sci., Eng. Technol., Int. J. Electrical, Comput., Energetic, Electron. Commun. Eng. **1** (2007), no. 12, 1777–1781.
22. A. Sharma and A. Chaturvedi, *Gradient descent feed forward neural networks for forecasting the trajectories*, Int. J. Adv. Sci. Technol. **34** (2011), 83–88.
23. H. M. Kjer and J. Wilm, *Evaluation of surface registration algorithms for pet motion correction*, Technical University of Denmark, DTD, Dk-2800 KGS, Lyngby, Denmark, 2010, pp. 28–35.
24. P. Biber and W. Straßer, *The Normal distributions transform: a new approach to laser scan matching*, in Proc. 2003 IEEE/RJS Int. Conf. Intell. Robots Syst. (Las Vegas, NV, USA), 2003, pp. 1–24.
25. D. Lanman and G. Taubin, *Build your own 3d scanner: 3d photography for beginners*, ACM, New York, USA, 2009, p. 14.
26. A. S. Skembris, C. Papaodysseus, and E. Koukoutsis, *2D fragmented object reconstruction with the use of the chromatic and thematic content*, Pattern Anal. **15** (2012), 133–146.

AUTHOR BIOGRAPHIES

Nada A. Rasheed received her BS degree in Statistics from the Department of Statistics, College of Management and Economics, Al-Mustansiriyah University, Baghdad, Iraq, in 1985, and an MS degree in Computer Science from the Department of Computer Engineering, Al-Rasheed College, University of Technology, Baghdad, Iraq, in 2002. She received her PhD degree in Computer Science from the Faculty of Information Science and Technology, Universiti Kebangsaan, Malaysia, in 2016. From 2003 to 2007 she worked for Al-Nahrain University, Baghdad, Iraq. From 2007 to 2018 she worked for the University of Babylon, Babylon, Iraq. Since 2018 she has worked in the Department of Sciences, Al-Karkh University of Science, Baghdad, Iraq, where she is now a Prof. Assistant. Her main research interests are artificial intelligence and digital image processing.

Md Jan Nordin received both BS and MS degrees in Computer Science from Ohio University, USA, in 1982 and 1985, respectively. He received his PhD degree in Engineering Information Technology from Sheffield Hallam University, United Kingdom, in 1995. He is currently an Associate Professor at the Centre for Artificial Intelligence Technology (CAIT), National University of Malaysia (UKM). His current research interests include pattern recognition, computer vision, intelligent systems, and image reconstruction.