Summer 2022

# Identifying and Discovering Curve Pattern Designs from Fragments of Pottery

Jun Zhou

# Identifying and Discovering Curve Pattern Designs from Fragments of Pottery

by

Jun Zhou

Bachelor of Engineering
Nankai University, 1996

Master of Science
Loyola University at Chicago, 2001

---

Submitted in Partial Fulfillment of the Requirements

for the Degree of Doctor of Philosophy in

Computer Science and Engineering

College of Engineering and Computing

University of South Carolina

2022

Accepted by:

Song Wang, Major Professor

Michael N. Huhns, Committee Member

Yan Tong, Committee Member

Lannan Luo, Committee Member

Karen Smith, Committee Member

Tracey L. Weldon, Vice Provost and Dean of the Graduate School

# Acknowledgments

I want to thank my advisor, Prof. Song Wang, for the tremendous help and support of my Ph.D. study and research, for his patience, motivation, enthusiasm, and immense knowledge. His guidance helped me in all the time of research and writing of this dissertation.

I want to show my sincere gratitude to Dr. Karen Smith for sharing her pearls of wisdom on the research project, her insightful suggestions, and for encouraging the pursuit of this research.

Thank the committee members for my doctoral dissertation, Dr. Karen Smith, Prof. Michael Huhns, Prof. Yan Tong, and Prof. Lannan Luo, for their help and constructive suggestions. It is my great honor to have them as my committee members.

I want to thank my colleagues: Yuhang Lu, Dhaval Salvi, Kang Zheng, Hongkai Yu, Haozhou Yu, Zhenyao Wu, Xinyi Wu, Canyu Zhang, Lan Fu, Rabab Abdelfattah, and other fellow lab-mates for their assistance.

Finally, I want to show my appreciation to my family for their continuous support and encouragement. I dedicate this dissertation to my husband, Chun-Hui Miao, and my parents for the most cherished love they gave me.

# Abstract

The surface of many cultural heritage objects, such as pottery sherds found in the Southeastern Woodlands, were embellished with curve patterns. The original full designs of these patterns reflect rich historical and cultural information. However, in practice, most objects are fragmentary, making the complete underlying designs unknowable at the scale of the sherd fragment. The challenge to reconstruct and study complete designs is stymied because 1) most pottery sherds contain only a small portion of the underlying full design, 2) curve patterns detected on a sherd are usually incomplete and noisy, and 3) in the case of a stamping application, the same design may be applied multiple times with spatial overlap on pottery, resulting in a *composite* pattern. Our study aims to address these challenges and better identify and discover the full designs from fragmented pottery sherds. In this research, we study two important computer vision problems: design identification that identifies a sherd underlying design and sherd identification that clusters unidentified sherds to discover unknown designs. We focus both problems on curve patterns, where the underlying full design and the partial pattern on the sherd are curve structures, and develop new algorithms to address them.

For design identification, we formulate this problem as matching: a binary curve pattern image segmented from a sherd depth image is matched to each known full design, and the best matching proposes its underlying design. We develop two curve-pattern matching algorithms for this purpose. First, we develop a new curve matching method by extending Chamfer matching, which decomposes a composite pattern into multiple candidate components as long as these components match a partial design.

The optimal combination of these components defines a new matching cost. Second, we develop a new patch-based curve pattern matching method to locate the most similar regions between the sherd and the considered full design. Specifically, we apply uniform sampling for constructing patches and employ a learning-based curve feature descriptor to derive a heatmap for the local similarity between the sherd and the design. With this heatmap, we locate the best matching portions by region growing and define a new matching cost considering the overall similarity of these portions.

For sherd identification, we develop a new clustering algorithm to identify and group sherds with the same design. Given the segmented curve-pattern images of a collection of sherds, we first conduct patch-based pairwise matching between each pair of sherds to construct a similarity matrix. The pairwise matching is based on the best-matched patches between the two sherds to handle possible composite patterns. We build a fully connected graph based on this similarity matrix and partition the graph into subgraphs/clusters by adaptive thresholding. An iterative cluster refining strategy is developed, with curve-pattern stitching in the iteration, for identifying and refining the sherd clustering.

We collect a set of pottery sherds from the heartland of the paddle-stamping tradition with a subset of available paddle-stamped designs from southeastern North America to evaluate the developed algorithms. Moreover, we developed the Snowvision, a computer-aid system that includes sherd digitization, preservation, curve structure segmentation from a digitized sherd depth image, design identification, and sherd identification based on the developed algorithms.

# Table of Contents

# List of Tables

# List of Figures

# Chapter 1

# Introduction

## 1.1 Problem Formation

All around the world, the archaeological record is filled with fragmentary objects of bone, pottery, shell, stone, wood, and cloth variously embellished with realistic and abstract designs. These designs may include figural imagery like that seen on ancient Maya [57] or the carved marine shell gorgets of late prehistory in North America [55]. Humanities and social science scholars have put these designs to many uses, such as building chronologies, tracking trade networks, reconstructing aspects of style and the creative process, and understanding the creation and expression of identity.

Without question, most of these topics are best addressed using complete designs rather than design fragments. Traditionally, complete designs are reconstructed by fitting fragments from the same design; fragments of designs are then identified as belonging to complete designs manually by visual assessment. The tasks of design reconstruction and identification can be highly time-consuming. As a result, millions of broken cultural heritage objects stored in museums worldwide remain unstudied from a design perspective, and large numbers of decorated objects found in the archaeological record contribute little to our understanding of style, production and use, and meaning.

Computer-aided reconstruction and identification of the designs from fragmented cultural objects have attracted great interest among archaeologists and computer scientists in recent years [29, 39]. This research presents the Snowvision project and our efforts to digitally preserve and analyze paddle-stamped pottery produced throughout southeastern North America. Snowvision focuses on the Swift Creek Complicated Stamped pottery type, created during the Middle and Late Woodland periods between roughly AD 100 and 800 [66]. Figure 1.1 (c) shows four such pottery fragments from the period 350-650 AD, when the stylistic tradition of carving wooden paddles (a sample shown in Fig. 1.1 (a) and a small fraction of their curve patterns shown in Fig. 1.1 (b)) was at its most complex (i.e., multiple different design elements

arranged and combined in unique ways on the paddle).


Figure 1.1 (a) A sample paddle. (b) Four paddle designs reconstructed by Frankie Snow. (c) Four sample pottery fragments. (Original design reproduced with permission, courtesy of Frankie Snow, South Georgia State College.)

Carved paddle designs were primarily composed of connected and intertwined curved lines. The same paddle was usually applied to many different locations on the pottery vessel's exterior surface to achieve the desired decorative effect. Also, the same paddle may be applied to many different vessels and sometimes applied to pottery found at different sites, indicating the movement of paddles or people. Current existing designs were composed manually by experienced archaeologists [8, 67]. With access to extensive collections from key excavation sites, the archaeologists assembled sherds belonging to the same paddles, traced or measured sherd curves on individual sherds, and looked for regions with overlap and new design information. Slowly the sherd information is compiled until a paddle design is complete. As an example of design reconstruction illustrated in Fig. 1.2, the archaeologists first visually identify a number of pottery sherds from the same paddle and then match the curve patterns on the sherd surfaces to construct larger design pieces until the final full design is

constructed.

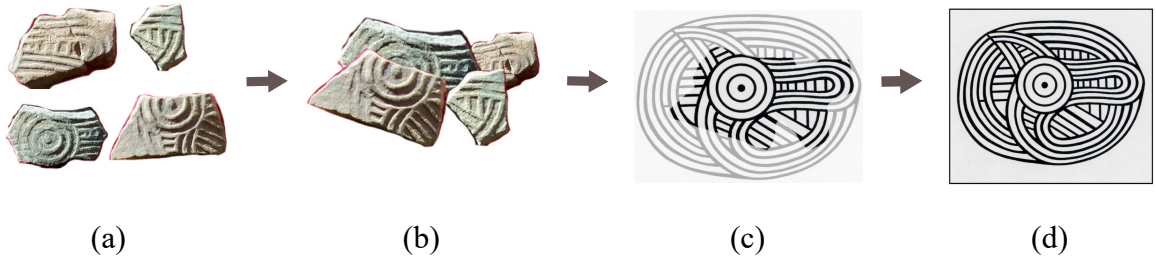



Figure 1.2 An illustration of design reconstruction: (a) four sherds that belong to the same paddle; (b) stitch sherds by their overlapping regions; (c) compile curve pattern on the stitched sherd; and (d) a constructed complete design. (Original designs reproduced with permission, courtesy of Frankie Snow, South Georgia State College.)

More fragmented pottery sherds can then be identified as belonging to their complete designs by matching their curve patterns to a set of composed designs [26], as its procedure illustrated in Fig. 1.3. Specifically, archaeologists overlay a sherd on a design and slide around until the curves on the sherd match those on a design.

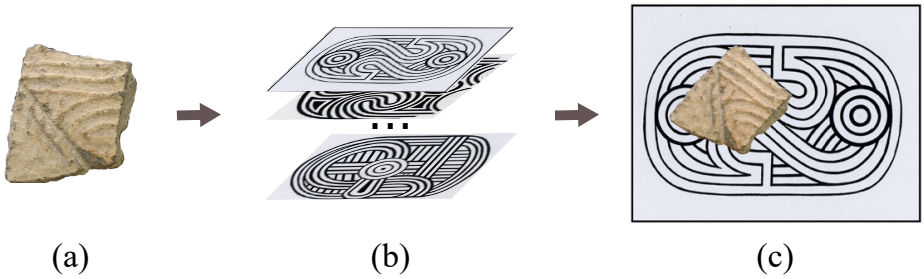



Figure 1.3 An illustration of design identification: (a) a sherd; (b) a set of complete designs; (c) matching result of the sherd to its underlying design. (Original designs reproduced with permission, courtesy of Frankie Snow, South Georgia State College.)

Associating Swift Creek paddle designs and pottery sherds provide unique evidence on the fundamental hunter-gatherer-forager process in the southeast. Broyles [8] recognized the same paddle used on different vessels from archaeological sites up to

100 miles apart. Although paddles are not people, much like vessels are not people, recognizing this individual, unique connection between sites separated in space raises the possibility that scholars can track the movement of individuals (e.g., paddle owners/keepers) for the first time. Snow [67] noted a high number of Swift Creek design matches among sites within the big bend area of the Ocmulgee River (called the Ocmulgee-Big Bend) and between the Ocmulgee-Big Bend and the headwaters of the Satilla River to the south. These matches occurred within an approximately 30-mile (48-kilometer) radius, providing, as Snow said, clues to settlement patterns (i.e., mobility) within the region. He also noted a limited number of design matches between those sites and sites much further to the northwest, southwest, and southeast, well outside the Ocmulgee-Big Bend region. These longer-distance, inter-regional matches suggest another social process may be at play (e.g., long-distance foraging or interregional aggregation). An example of the design connections between a site (Milamo, 9 WL1) and its inter-and intra-regional neighbors is shown in Fig. 1.4. Increasing our ability to discern paddle design matches among sites, as our research does, will further contribute to research that addresses how foragers organized and moved about on the landscape 1,500 years ago.

Manually assessing many such objects can be extremely time-consuming, especially when these objects are highly fragmented. For example, to identify the underlying design of a pottery sherd, archaeologists have to compare this sherd with every design in a design corpus. The smaller the fragment of design preserved or the more diverse the design corpus, the more difficult it is to match a sherd to a complete design. Identifying pottery sherds for new design reconstruction can be even more labor-intensive. This sherd has to be matched to every sherd in a sherd collection, usually consisting of hundreds of thousands of pieces. Such tasks may require months or even years of daily effort to identify the fragments of certain complete designs. Therefore, it is essential to use computing technology to facilitate this process. Our

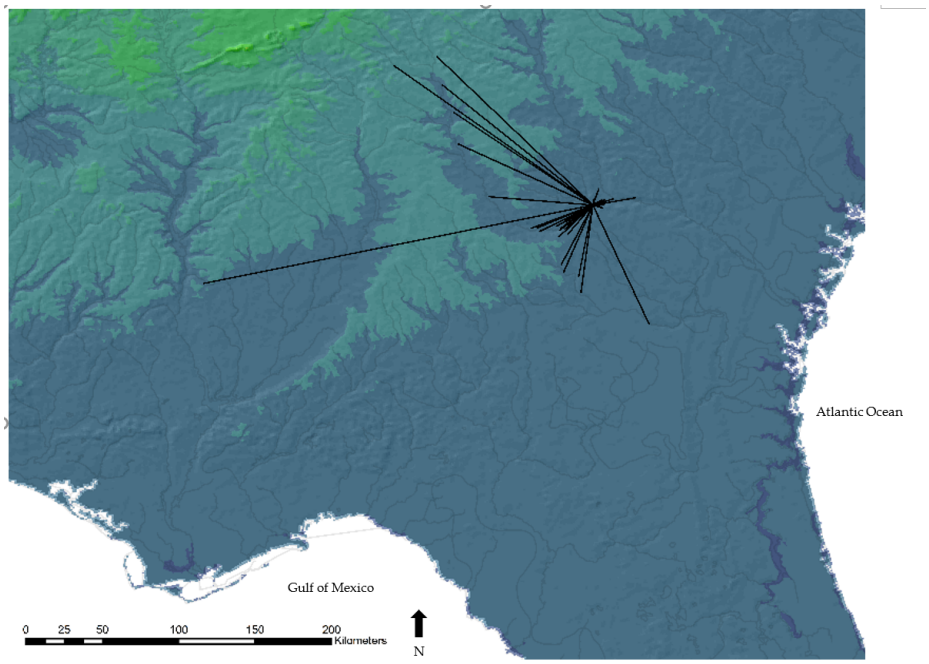



Figure 1.4 Design matches between Milamo (9WL1) and other sites in the region. Black lines represent one or more paddle design matches. Most matches with Milamo occur within the Ocmulgee-Big Bend.

research is such an effort. Our research studies two important computer-vision problems: design identification that matches a sherd to every design in a known design database and sherd identification that identifies and groups sherds with one but unknown design for new design discovery. We investigate these important problems by focusing on *curve* patterns, where the underlying full design and the partial pattern on the sherd are curve structures. New algorithms are developed for each problem, respectively.

## 1.2 Challenges

Although the computer-aided application of the designs from fragmented cultural objects has attracted great interest among archaeologists and computer scientists in recent years [29, 39], identifying the full curvilinear paddle design from fragmentary

sherds remains challenging for several reasons. We will describe these challenges in detail in the following.

### 1.2.1 Highly Fragmented Sherds

Pottery sherds can be highly fragmented, and each sherd contains only a small portion of the underlying full paddle design. While multiple sherds may have the same design, i.e., stamped by the same paddle, they are usually from different vessels and varies in shape, color, texture, and level of degradation. Therefore, they cannot be assembled by fitting their sherd boundaries or grouping them by color and texture for more complete curve patterns as in many other studies. Figure 1.5 shows several such sherds and their complete design.


Figure 1.5 Fragmented pottery sherds (left) and their underlying design (right). (Original designs reproduced with permission, courtesy of Frankie Snow, South Georgia State College.)

### 1.2.2 Multiple Stamping

Designs carved onto these wooden paddles are primarily composed of connected and intertwined curved lines. The same paddle is usually applied to many different locations on the pottery vessel's exterior surface to achieve the desired decorative effect

before the vessel is fired. One carved paddle may be applied multiple times on the pottery surface with spatial overlap, which archaeologists have come to call overstamping. As a result, a sherd may contain a *composite* pattern, i.e., a small fragment of multiple, partially overlapping copies of the same design, as shown in Fig. 1.6. Such a composite pattern is not simply a portion of the full design. Therefore, matching it to the underlying full design is not a simple partial-to-global matching problem.

Figure 1.6 Two sherds with composite patterns (left) and their underlying design (right). Each single pattern in a composite pattern is indicated in blue, green, or red. (Original designs reproduced with permission, courtesy of Frankie Snow, South Georgia State College.)

### 1.2.3 Complexities on Sherd Surfaces

Sherd surfaces can show complexities due to many reasons. First, the carved paddles used for stamping are usually flat, while the pottery surfaces are usually not. As a result, the paddle typically does not well fit the pottery surface, which leads to shallow curves at many locations, as two examples shown in Fig. 1.7 (a). Second, purposeful smoothing of the stamped surface during vessel manufacture or weathering and erosion after vessel discard can lead to deformed or weak curve structure, as two examples shown in Fig. 1.7 (b). Third, on a sherd with a composite pattern, multiple stamping can lead to large overlaps, where curves can be cut into small trunks or

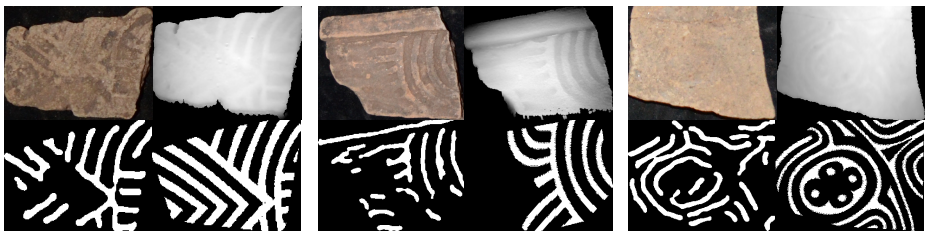
severely deformed, as two examples shown in Fig. 1.7 (c). Commonly, these three complexities may present on one sherd surface.


Figure 1.7 (a) Sherds with shallow stamping. (b) Eroded sherd (up) and sherd with purposeful smoothing (down). (c) Sherds with broken curves (up) or deformed curves (down) in overlapping areas. (Original designs reproduced with permission, courtesy of Frankie Snow, South Georgia State College.)

### 1.2.4 Imperfect Curve Structure Segmentation

For design identification and sherd identification, we will first need to segment curve structures in binary curve pattern images from design and sherd original digitized formats. The digitized designs are gray-scale images scanned from the paper drawings by archaeologists. The curve patterns in the images have an intensity value, and the background has another intensity value. We obtain their binary curve pattern images by a pre-set threshold. The digitized sherds are sampled depth images from 3D scanners (details of scanning process included in Chapter 2). We applied a Fully Convolutional Network (FCN) based method [47] to segment curve structure from a sherd depth map to a binary curve pattern image, where intensities of the curve and non-curve regions correspond to their underlying design curve pattern images. However, as shown in Fig. 1.8, due to the above complexities, even the most advanced

state-of-the-art deep learning method can not produce perfect segmentation.



(a) (b) (c)

Figure 1.8 An illustration of various noise and deformations in the curve structures segmented from sherds: (a) A sherd with missing curve segments due to erosion; (b) A sherd with noisy curves due to weathering and/or shallow paddle stamping on rough pottery surface; (c) A sherd with deformed curve patterns due to the drying process in making the pottery. For each case, we show the sherd's RGB image (top-left), the depth image (top-right), the segmented curve structure (bottom-left) and the corresponding portion in the underlying design (bottom-right).

In the following, We elaborate on the proposed research on design identification and sherd identification by considering the above complexities.

## 1.3 Scope of Research

In this research, we study the problems of design identification and sherd identification. The binary curve pattern images segmented from sherds and designs are the inputs for both identification tasks. For design identification, we formulate this problem as matching; the curve pattern from a sherd matches each known full design. Only a small number of designs with the lowest matching costs can then be considered the design used in the sherd, and they are presented to archaeologists for a final decision. In this study, we develop two curve-pattern matching methods, a Chamfer-matching based method and a patch-based matching method. For sherd identification, we formulate this problem as clustering; curve patterns from sherds are clustered. Each cluster is expected to represent a design that may be a new unseen one. We develop a new curve pattern clustering method consisting of pairwise curve

matching, graph partitioning, and sherd stitching to group sherds with the same design. Lastly, we present the Snowvision project, its digitization pipeline, and each component in its Science Gateway system, including its core backend services: design identification and sherd identification.

### 1.3.1 Design Identification

The first challenge we want to address in design identification is composite curve pattern matching. It is a unique partial-to-global matching, for the sherd may contain multiple incomplete copies of the design and the boundary of each copy is unclear and priorly unknown. Traditional partial-to-global matching methods, such as template matching, may perform poorly as they do not consider composite pattern matching. Therefore, we first develop a new curve pattern matching method by extending Chamfer matching. Specifically, we thin both sherd and design curves into one-pixel wide. Then we decompose a composite pattern into multiple candidate components as long as these components match a partial target design. The optimal combination of these components defines a new matching cost used for design identification. This algorithm can automatically recognize whether or not the pattern on a sherd is a composite one and identify each single design copy in a composite pattern. This low-level matching method can be easily adapted to many shape matching and edge map matching applications.

Second, we develop a patch-based matching method by decomposing curve patterns into patches and establishing dense patch correspondence in the deep feature space. It combines the advantages of template matching and keypoint matching by finding local matching without detecting keypoints. Specifically, we apply uniform sampling for constructing patches and employ a learning-based curve feature descriptor to derive a heatmap for the local similarity between the sherd and the design. We consider both the heatmap value and the overlap area in deciding the matching

between sherd and design. With this heatmap, we locate the best matching portions by region growing and define a new matching cost considering the overall similarity of these portions. The patch-based matching method is chosen for the Snowvision system for it is less computationally intensive and more accurate than the Chamfer-matching based algorithm.

### 1.3.2 Sherd Identification

Given a collection of sherds with unknown designs in sherd identification, we cluster them in terms of their underlying (unknown) designs. To identify and recreate new designs, we expect the sherds in the same cluster to show only partial/local overlap or even no overlap but the local similarity to other common sherds in the cluster. Many existing clustering algorithms rely on global image features and therefore cannot handle our problem well. In this study, we formulate pottery sherd clustering as a graph partitioning problem and develop a new iterative clustering pipeline consisting of pairwise matching, graph partitioning, and sherd stitching. The developed algorithm integrates local feature matching into the image clustering pipeline. More specifically, given the segmented curve-pattern images of a collection of sherds, we first conduct patch-based pairwise matching between each pair of sherds to construct a fully connected graph and then partition it into subgraphs using an adaptive thresholding algorithm. We finally develop an iterative cluster refining strategy, with curve-pattern stitching in the iteration, for identifying and refining the sherd clustering.

In the experiments, we quantitatively evaluate the developed algorithms for the two tasks on a collection of pottery sherds from key excavation sites. We also report comparison experiments with other state-of-the-art methods to justify the effectiveness of the developed algorithms.

### 1.3.3 Snowvision

*Snowvision*, a project named after the archaeologist Frankie Snow [67], is an interdisciplinary effort to leverage state-of-the-art computer vision techniques to explore and study paddle designs from large collections of pottery sherds. It provides standards and guidelines for digitizing sherds and developing rich and accurate metadata, a front-end website from which archaeologists at different institutions can submit and share their sherds. A set of backend advanced computer vision tasks to assist archaeologists in more efficiently exploring the sherds' underlying designs. An acquisition procedure including sherd digitization and preprocessing is developed, and a collection of pottery fragments and reconstructed designs is available to the public for study. The computer vision tasks include curve structure segmentation, design identification, and sherd identification in our research. Please note that curve structure segmentation is not a contribution of this dissertation and will be discussed in the background chapter. The remaining two tasks, design identification and sherd identification, are the focus of this dissertation.

In curve structure segmentation, we apply an FCN-based segmentation method to segment curve structures from the sherd surface depth images. The resulting binary curve pattern images are fed to design identification and sherd identification, and archaeologists can take the identification results produced by these two tasks for design study, including discovering and reconstructing new designs. In practice, these three tasks run in sequence: a sherd is digitized and converted into a depth map, and the depth map is segmented to a curve structure image. We match this sherd image to a known design set in design identification. If we cannot find its underlying design, we cluster this sherd with others to discover unknown designs in sherd identification.

## 1.4 Structure of the Dissertation

The remainder of this dissertation is organized as follows. In Chapter 2, we introduce some essential background knowledge for this research. In Chapter 3, we review previous works related to this research. Chapter 4 presents a new Chamfer matching algorithm to identify design on a sherd with a composite pattern. Chapter 5 describes a patch-based matching method to address the problem of partial matching in terms of noisy binary images for design identification. Chapter 6 discusses a curve pattern clustering method consisting of pairwise curve matching, graph partitioning, and sherd stitching for sherd identification. Chapter 7 introduces the Snowvision system and its components. Finally, we conclude this dissertation in Chapter 8.

# Chapter 2

# Background

This chapter briefly reviews the essential background knowledge for this research. Specifically, we will introduce curve structure segmentation, Chamfer matching algorithm, Convolutional Neural Networks (CNNs), and deep learning metric with a triplet network. Curve structure segmentation extracts curve structures from sherd depth images. These curve structures are inputs for design identification in Chapter 4 and Chapter 5, and for sherd identification in Chapter 6. Chamfer matching is a widely used algorithm for image matching, such as shape or curve pattern matching. The developed curve pattern matching method in Chapter 4 is an extension of Chamfer matching. CNNs have been applied to image patch matching and achieved notable performances in various patch matching datasets. The metric learning in deep neural networks is usually called deep metric learning, which guides the network to learn useful features for comparison. CNN and deep learning metric are both applied in the patch-based curve pattern matching method for design identification in Chapter 5 and sherd identification in Chapter 6. We will introduce these important concepts in the following sections.

## 2.1 Curve Structure Segmentation

Curve structure segmentation is an application that segments curves from non-curve regions in color or grayscale images. It can be very difficult for degraded pottery sherds to extract their curve structures from the camera-taken RGB image of these sherds. Given that these curve structures are stamped on the surfaces of pottery vessels by carved paddles, curve structures usually show larger depth than the adjacent non-curve surface. Therefore, 3D scanners are applied to achieve the 3D depth images of the sherd surfaces, as one example illustrated in Fig. 2.1. Specifically, the scanner is placed above the sherd and is perpendicular to the platform where the sherd is seated. After scanning, Point Cloud Library (PCL) [1] is used to convert the 3D point

[1] https://pointclouds.org/

cloud to a depth image, in which curves show larger intensity than non-curve regions. Many segmentation algorithms can be applied to curve structure segmentation, such

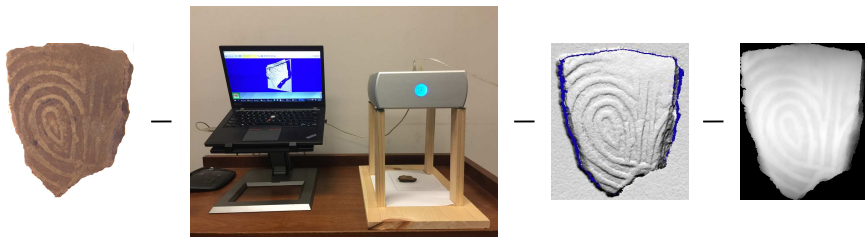



Figure 2.1 An illustration of scanning a sherd for its depth image: (a) RGB image of a sherd, (b) setup of a 3D scanner, (c) 3D point cloud of the sherd surface obtained by the 3D scanner, and (d) depth image of the sherd surface: pixel intensity represents the depth value at a location.

as DoG [11], LevelSet [42], deepLab [17], etc. However, the curve structures on the depth images can be very weak due to many reasons and make the segmentation a very challenging problem. For example, the paddle typically does not fit the object's surface well in stamping applications, leading to shallow curves at many locations. Furthermore, degradation over thousands of years can make the sherd surface highly rough, resulting in a very noisy depth image. In this research, a three-step CNN-based algorithm [47] is applied to more accurately and reliably segment the stamped curve patterns from the depth images of the sherds by learning and incorporating the implied curve geometry, such as curve smoothness and parallelism, in the underlying designs.

In Step I, we detect the skeleton of the curve pattern from a sherd depth image, which we formulate it as a pixel-labeling problem: skeleton pixel has label 1 and non-skeleton pixel has label 0. An FCN network is trained by following the encoder-decoder architecture [45] illustrated in Fig. 2.2. The benefit of this FCN is that it can simultaneously predict the curve skeleton map and estimate the curve width at each skeleton pixel by the encoder scale values.

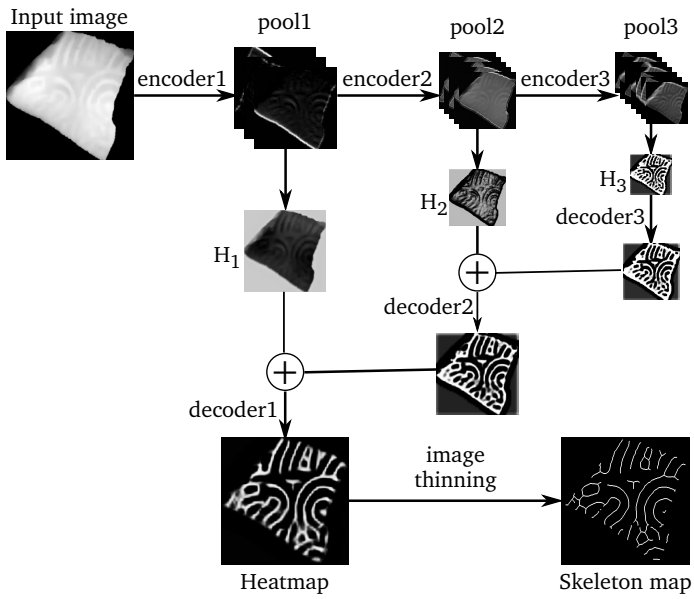



Figure 2.2 Network architecture of the FCN used for skeleton detection.

In Step II, we refine the curve pattern skeletons obtained in Step I, for they contain many false positives. In this step, we further train a supervised classifier to identify and prune such false positives by learning more curve features. Specifically, centered at each skeleton pixel detected by FCN, we take its $45 \times 45$ neighboring region in the original depth image as the input and train a dense prediction ConvNet to determine whether this pixel is a true skeleton pixel or a false positive.

In Step III, we recover the width of the curve structure from the skeleton map derived in Step II with the help of the scale values derived in Step I. Specifically, in a skeleton map, we search in a circular neighborhood of each skeleton pixel with a radius of the predicted width corresponding to its scale value in Step I. The curve pixels are supposed to have greater intensity than non-curve pixels. Then an adaptive thresholding algorithm is applied to decide whether a skeleton map pixel is a curve pixel. This algorithm does not require the detected skeleton to be exactly aligned with the center line of the curves – a slight dislocation between them may not change the final segmentation – it only requires the detected skeletons to be located inside

the underlying curve regions.

Figure 2.3 shows the sample results after each algorithm step.

(a) (b) (c) (d)

Figure 2.3 An illustration of segmenting curve structures from sample sherds: (a) depth images of sherds, where smaller intensity pixels have larger depths. (b) FCN-extracted curve skeletons, (c) refined curve skeletons by using a dense prediction CNN, and (d) final segmented curve structures with recovered curve width, masked by the sherd boundaries (indicated by red contours).

It has been shown in [47] that this CNN-based algorithm can segment the curve structure from a sherd much more accurately than other low-level and high-level image segmentation algorithms.

## 2.2 Chamfer Matching

Chamfer matching is a popular technique to find the best alignment between two image edge maps. It has been widely used in object shape detection and localization in cluttered images. The general concept is illustrated in Fig. 2.4.

Let $U = \{u_i\}$ and $V = \{v_j\}$ be the sets of points in template and query image edge maps respectively. The chamfer distance between $U$ and $V$ is given by the average of distances between each point $u_i \in U$ and its nearest edge in $V$

$$d_{CM}(U, V) = \frac{1}{n} \sum_{u_i \in U} \min_{v_j \in V} |u_i - v_j|, \tag{2.1}$$

where $n = |U|$.

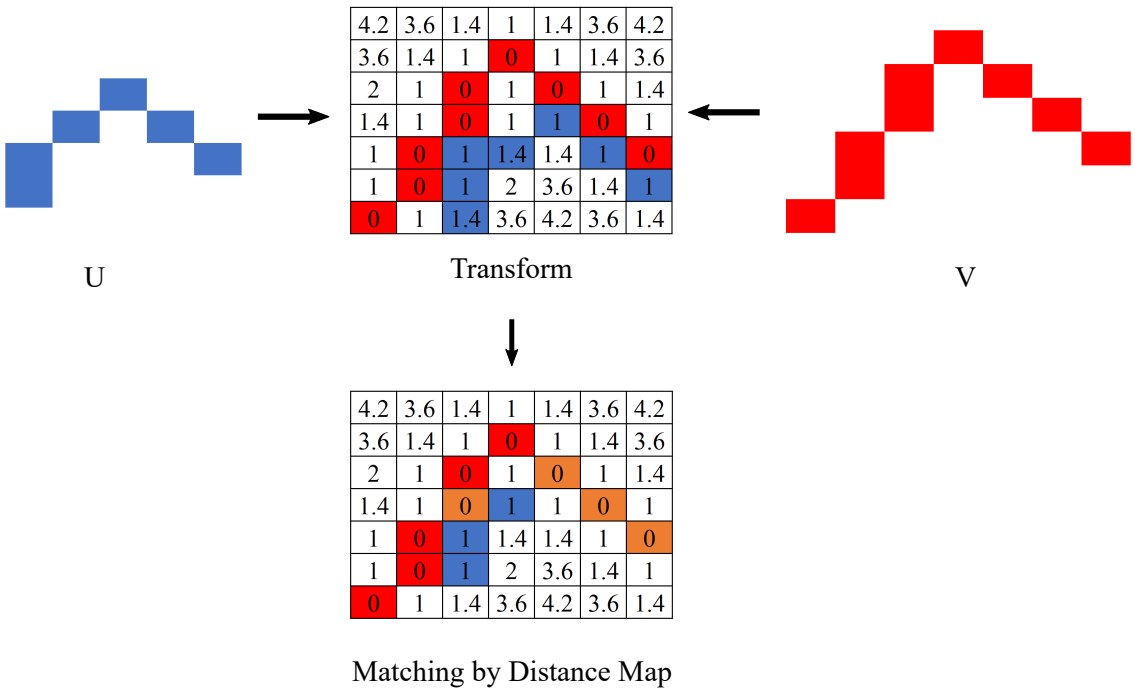



Figure 2.4 An illustration of Chamfer matching. $U$ is a template image, $V$ is a query image. The Chamfer matching cost between $U$ and $V$ is 0.42 by the distance map of $V$ in the middle.

Let $W$ be a warping from $U$ to $V$ defined on the image plane parameterized with $s$. For instance, if it is a 2D Euclidean transformation, $s \in SE(2), s = (\theta, t_x, t_y)$, where $t_x$ and $t_y$ are the translations along $x$ and $y$ axis bounded by the width and height of $V$ respectively, and $\theta$ is the in-plane rotation angle. Its action on template image points $u_i$ is given via the transformation

$$W(u_i, s) = \begin{pmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{pmatrix} u_i + \begin{pmatrix} t_x \\ t_y \end{pmatrix}. \tag{2.2}$$

The best alignment parameter $\hat{s} \in SE(2)$ between the two edge maps is then given by

$$\hat{s} = \arg \min_{s \in SE(2)} d_{CM}(W(U, s), V), \tag{2.3}$$

where $W(U, s) = \{W(u_i, s)\}$.

Chamfer matching provides a fairly smooth measure of fitness and can tolerate small rotations, misalignments, occlusions, and deformations. The matching cost can be computed efficiently via a distance transform map the same size of the query image edge map. Specifically, let us denote $x$ as a pixel in the distance transform map, the value of $x$ is defined as its distance to the nearest edge pixel in $V$, denoted by

$$DT_V(x) = min_{v_j \in V} |x - v_j|. \tag{2.4}$$

The matching cost of $U$ over $V$ can then be computed via

$$d_{CM}(U, V) = \frac{1}{n} \sum_{u_i \in U} DT_V(u_i). \tag{2.5}$$

## 2.3 Convolutional Neural Network

Convolutional Neural Networks (CNNs) are members of the neural network family and are similar to a regular neural network. A regular neural network contains a sequence of layers, each consisting of a group of neurons. Figure 2.5 (a) shows a 3-layer neural network. Each layer of neurons receives some input, performs a dot product, and optionally adds a nonlinearity. The output of one layer is fed to the next layer. Each neuron in one layer is connected to all the neurons in the preceding layer, which is usually called *fully-connected*. The input layer receives input from raw input data, and the output layer (last layer) outputs the predictions. Each layer bears a set of weights/parameters for the dot-product operation. The parameters are to be learned from data by minimizing different loss functions depending on the tasks. Loss functions, such as cross-entropy loss or mean square error (MSE) loss, are commonly used to train the network. We usually reshape the image pixels into a long vector and feed it to the regular neural network to process image data as input. However, regular neural networks do not scale well to larger images because of the full connectivity between neurons of neighboring layers. As image sizes increases, the

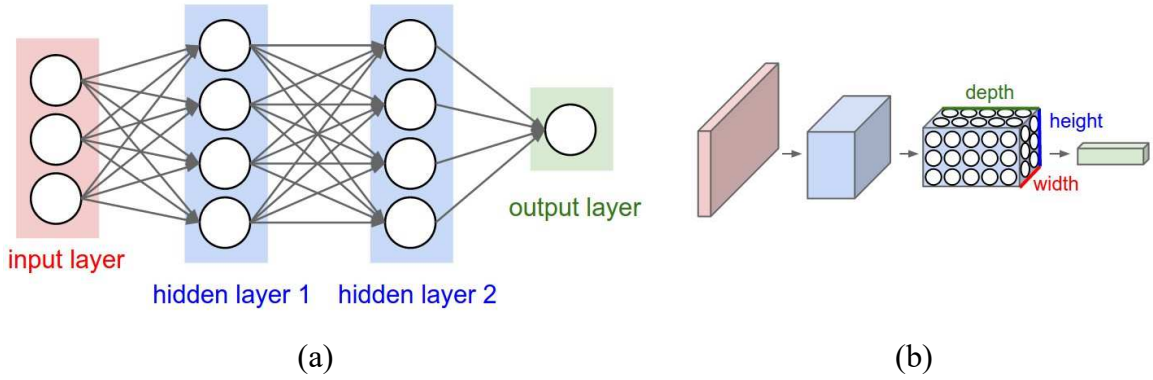



Figure 2.5 (a) An ordinary 3-layer neural network. (b) A convolutional neural network. (Source: http://cs231n.github.io/convolutional-networks/)

number of parameters will increase exponentially as there are many layers in a regular neural network. This will increase the computation and lead to overfitting problem.

Like the regular neural network, a CNN also contains many layers, with many neurons in each layer. However, the neurons in CNN layers are different from the neurons in ordinary NNs. Specifically, the neurons in CNN are arranged in three dimensions: width, height, and depth, as shown in Fig. 2.5 (b). Another difference between CNN and regular neural network is the connectivity between neurons of neighboring layers. In CNN, a neuron in one layer is only connected to a part of neurons in the preceding layer, and the parameters are shared between neurons. Such a design has two advantages: 1) it reduces the computation and the number of learnable parameters significantly; 2) compared to neurons in the regular neural network, it is more natural because neurons should perceive different regions in an image in the same manner. There are three major layers in CNNs: convolutional layer, pooling layer, and fully-connected layer. Figure. 2.6 shows the architecture of AlexNet [41].

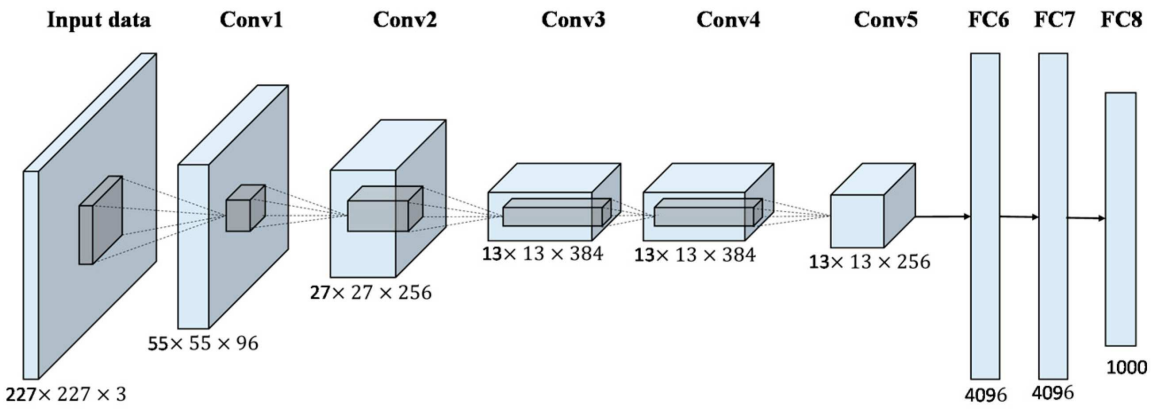



Figure 2.6 An illustration of the AlexNet architecture.

### Convolutional Layer

The convolutional layer is essential to CNN, which changes how a regular neural network operates on input. Taking AlexNet as an example, each layer contains $c_i \times h_i \times w_i$ neurons, where $c_i$, $h_i$ and $w_i$ are the depth, height, and width of $i$-th layer ($l$ layers in total). Usually, the height and width in each layer are the same for CNN architectures. We will use $k_i$ to represent both height and width. In each layer, a neuron is only connected to a local region of neurons in the preceding layer, as shown in Fig. 2.6. Each layer has a set of filters, also known as filters, which determines how each neuron connects with the preceding layer. For example, the first convolutional layer in AlexNet has 96 kernels of size $7 \times 7$. Thus, each neuron connects to $7 \times 7$ neurons in the input layer, which are image pixels. The kernels slide over the whole image and convolve with each region to produce the feature maps. These feature maps are later fed to the second convolutional layer. Padding and striding are often used in the convolution operation. The spatial size of the output feature map is calculated as follows:

$$k_i = \frac{k_{i-1} - m + p}{s} + 1, 1 \leq i \leq l \tag{2.6}$$

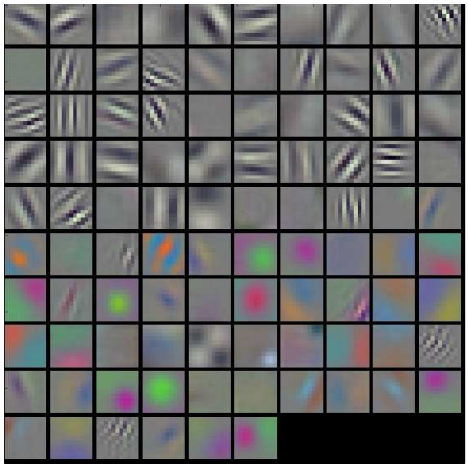


Figure 2.7 Visualization of filters in the first convolutional layer of AlexNet.

where $m$, $p$, and $s$ are the kernel size, padding size and striding size, respectively. For each kernel, there is another bias parameter added to the convolution output. The bias parameter is also learnable.

Figure 2.7 shows the visualizations in the first convolutional layer of AlexNet. Based on the visualization, we can see that the filters can detect edges, colors, and other patterns with different orientations. A later study in ZF-Net [76] shows that low layers in CNN extract low-level features such as edges while high layers extract high-level features such as parts and regions.

After the convolution operation, a nonlinear mapping function is applied to add nonlinearity between input and output of each layer. Such function is usually called *activation function*. The most commonly used nonlinear function is Rectified Linear Unit (ReLU) [41]. It can speed up the convergence of CNN compared to previously used sigmoid function. ReLU function is simply defined as: $f(x) = \max(0, x)$. Later, more advanced activation functions such as ELU [19], Leaky ReLU [50], and PReLU [32] are proposed.

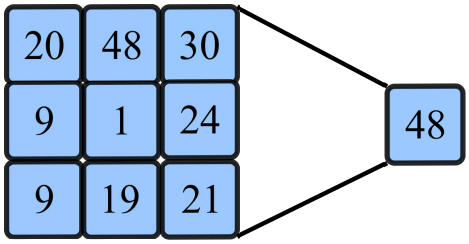



Figure 2.8 An example of pooling operation for a $3 \times 3$ kernel.

### Pooling Layer

Another key component in CNN is the pooling layer. The function of the pooling layer is to reduce the spatial size of feature maps. Specifically, the pooling layer scans the whole feature map region by region and pools a number for each region to produce the output feature map. Like the convolutional layer, two parameters determine how the pooling operates: kernel size and stride. The kernel size determines the size of each region, while the stride determines the space between two neighboring regions. Note that these two parameters are not learnable but predefined. Pooling strategies include max-pooling and average-pooling. Max-pooling produces one number for each region, as shown in Fig. 2.8. In AlexNet [41], kernel size is $3 \times 3$, and stride is 2, which is also the common choice of other CNN architectures.

Despite such a simple operation, the pooling layer has many advantages. First, the computation time is reduced as the spatial sizes are reduced. The computation time reduction is significant considering the number of layers in CNNs can go up to tens, hundreds, and even over a thousand. The pooling layer also increases nonlinearity as the pooling operation is nonlinear. We can perform pooling because neighboring pixels in images are usually similar and not always informative. We can use pooling to reduce some redundancy. The pooling layer also enlarges the receptive field [49], which is essential for CNN to gain more context information for prediction. Finally, the pooling layer can improve the translation-invariance of CNN, which is necessary

for tasks caring more about global contexts, such as classification.

#### Fully-Connected Layer

For a fully-connected (FC) layer, each neuron is connected to all neurons in the preceding layer. FC layers are already used in regular neural networks. Because of the connection pattern, FC layers can obtain information from the global context. As a result, they are usually appended at the end of a network to make predictions. For example, AlexNet has three consecutive FC layers at the end. However, the huge number of parameters in FC layers make them the computation bottleneck of CNN. Given this, recent CNN architectures, such as ResNet [31], use global average pooling to obtain global context information and only keep one FC layer.

### 2.4 Deep Metric Learning with Triplet Network

A metric, or distance function, is a function that defines the distance between each pair of samples. With good metrics, samples of the same class are close, while samples of different classes are far away. Formally, let us define a distance metric as a mapping $D : X \times X \to \mathbb{R}^+$ over a vector space $X$. A metric has to satisfy the following conditions:

$$D(x_i, x_j) \geq 0 \tag{2.7}$$

$$D(x_i, x_j) = D(x_j, x_i) \tag{2.8}$$

$$D(x_i, x_j) \leq D(x_i, x_k) + D(x_k, x_j) \tag{2.9}$$

$$D(x_i, x_j) = 0 \Leftrightarrow x_i = x_j. \tag{2.10}$$

Note that a metric is called a pseudo-metric if the fourth condition is not satisfied.

In the context of deep learning, deep neural networks are used to represent complex nonlinear transformations that effectively represent the structure of data. To learn

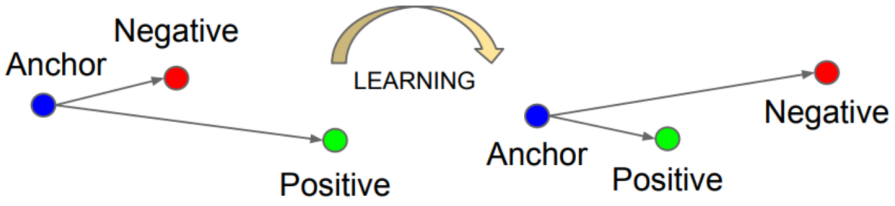



Figure 2.9 The triplet loss [62] optimizes the ranking between anchor-positive pair and anchor-negative pair, such that the anchor-positive distance is smaller than the anchor-negative distance.

the metrics with deep neural networks, various kinds of loss functions were proposed, such as contrastive loss [18] and triplet loss [62].

Contrastive loss [18] is used to separate samples of different classes by a fixed margin and pull samples of the same class as close as possible. Formally, it is defined as:

$$L_{contrastive} = yd^2 + (1-y)\max(m-d,0)^2, y = 0,1, \tag{2.11}$$

where $d = ||f(x_i) - f(x_j)||_2$ is the $L_2$ distance between features of sample $x_i$ and $x_j$. $y = 1$ indicates $x_i$ and $x_j$ are from the same class, $y = 0$ indicate they are from different classes. $m$ is the fixed margin. Triplet loss [62] relaxes the contrastive loss by comparing a triplet of samples: anchor, positive and negative. The positive sample comes from the same class as the anchor, while the negative sample comes from a different class. As shown in Fig. 2.9, the goal of triplet loss is to learn the ranking between anchor-positive pair and anchor-negative pair. Ideally, the anchor-positive distance should be smaller than the anchor-negative distance. Therefore, the triplet loss is defined as:

$$L_{triplet} = \max(d_{(ap)} - d_{(an)} + \alpha, 0), \tag{2.12}$$

where $d_{(ap)} = ||f(x_a) - f(x_p)||_2$ and $d_{(an)} = ||f(x_a) - f(x_n)||_2$ are the distances between anchor-positive sample pair $(x_a, x_p)$ and anchor-negative sample pair $(x_a, x_n)$

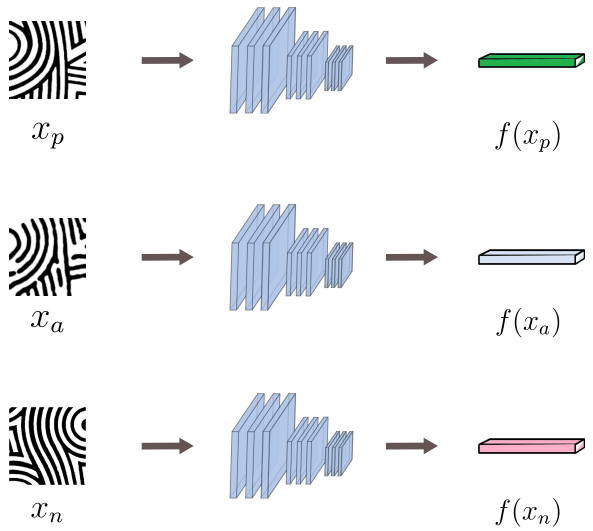



Figure 2.10 The triplet network.

in feature space, respectively. $\alpha$ is a margin that separates anchor-positive pair and anchor-negative pair.

A triplet convolutional neural network is comprised of three branches of the same base network with shared weights, as an example shown in Fig. 2.10. In training, each branch takes a sample in a triplet as input. The triplet network is trained by minimizing the triplet loss defined by Eq. 2.12. In testing, any network branch can be used for deep metric learning.

# Chapter 3

# Literature Review

This chapter briefly reviews prior works related to computer-aided applications on archaeological fragments, design identification by curve-pattern matching, and sherd identification by curve-pattern clustering.

## 3.1 Computer-aided Applications on Archaeological Fragments

Our research aims to **leverage computer-vision technologies to explore pottery sherds** that are important to archaeologists. Such technologies increasingly have been used to foster new insights into humanities and social sciences subjects. For example, the ArchAIDE [28] project developed a web-based system that automatically recognizes archaeological pottery fragments from excavations around the world. The fragment shapes are extracted and classified by their profiles. Then a graph-based fast-marching algorithm is used to match the shape in a comprehensive fragment database. The great Budda project [35] presented preservation and restoration of the original appearance of a Budda statue digitally. A geometric model and a photometric model are developed. The geometric model aligns digitized statue pieces and simultaneously merges them in 3D space. The photometric model renders eigen-texture on the object surfaces.

These projects focused mainly on fragment classification and matching, which aimed to recognize fragments that were originally from the same object, and then used to aid the 3D reconstruction of the whole underlying object, such as a full vessel [21, 27]. For example, Color and texture information has been widely used for fragment classification. Ke-Gang et al. [43] developed texture-based methods for sherd classification by using Gabor wavelet transformation and a non-supervised kernel-based fuzzy clustering algorithm. Smith et al. [66] proposed a ceramic sherd classification method based on color and texture features. In particular, it measures the color similarity based on the joint probability distribution of the color channels, followed by constructing a color histogram in the 3D RGB space and the texture

similarity by using a new texture descriptor similar to the geometric total variation energy (TVG) concept proposed by Burchard [12].

Many geometric features were also used to classify archaeological fragments. Roman-Rangel [59] developed a potsherd categorization system using the Scale Invariant Feature Transform (SIFT) features and the Sprine images in 3D space. The bag-of-words technique was then used to combine all the local features, followed by principal component analysis (PCA) to reduce the feature dimensions. Maiza and Gaildrat [40] proposed an algorithm to use the 3D surface geometry for fragment classification.

Our study can treat design and sherd identification as classification – classifying sherds by designs (known or unknown). However, it has a major difference from these existing works on fragment classification. In our study, the sherds with the same design are rarely from the same vessel. It is impossible to reconstruct an entire vessel, or even larger vessel pieces of a vessel, using available sherds. In other words, sherds with the same design are usually from different vessels with different shapes, sizes, colors, and textures. As a result, we could not use the color, texture, and geometric information in our work as in previous fragment classification research. Instead, we use curve patterns presented on sherd surfaces and on reconstructed designs.

## 3.2 Design Identification by Curve-pattern Matching

At the core of design identification is a **curve-pattern matching** problem, which has been long studied in computer vision and image processing. By thinning all the curve structures to one-pixel wide, many shape matching algorithms have been developed for matching curve patterns. For example, Frenkel et al. [24] proposed to match two curves by deforming them into one identical curve. Then a graph-based fast-marching algorithm was used to find the optimal deformation as the curve matching cost. However, this method requires all the curves to be represented by a set of sampled

points with consistent order. Defining such an order is unfeasible in our problem, given various image-segmentation errors and the nature of partial matching. Belongie et al. [6] proposed a shape context approach, which builds a log-polar histogram around each sampled curve point and then uses this histogram as the feature to match two curve structures. Roman-Rangel [60] extended the shape context algorithm to a Histogram of Orientation Shape Context (HOOSC) algorithm by incorporating the orientation measure into the log-polar histograms. A promising result has been reported using HOOSC to analyze the Ancient Maya Glyph collections. Barrow et al. [4] proposed the widely used Chamfer matching algorithm by pre-computing a distance map for efficiently locating one curve pattern on another curve pattern. By treating curve patterns as binary images, image-based matching algorithms can be directly applied here for matching curve patterns. For example, Brunelli [9] developed a template matching method by treating one as a convolution mask over the other. However, many of these existing matching algorithms do not consider composite pattern matching, performing poorly in our study.

In principle, **local feature matching** [30] can also be used for handling curve-pattern matching. we can extract a set of image features or keypoints, such as the SIFT [7], SURF [5] and ORB [61]; and learning-based methods, such as LIFT [74], LF-Net [53] and RF-Net [63], from the binary curve pattern images from a sherd and a design, and match them by identifying a set of corresponding features. These methods could find partial-to-partial correspondence by matching keypoints. However, most local feature detectors are developed for color or gray-scale images. They are usually very rare in these texture-less binary curve pattern images.

Curve-pattern matching is also related to fingerprint matching [37], in which minutiae features are extracted at ridge bifurcations or ridge endings. The curve patterns on sherds usually carry much less minutia than fingerprints. Therefore, algorithms developed for fingerprint matching usually perform poorly on this task. Curve-pattern

matching can also be treated as a region-based image retrieval (RBIR) problem [14]. The goal of the image retrieval is to find regions in the target image that share similar features with a specified region on the query image. For example, Hinami et al. [33] proposed a region-based network whose multi-task CNN features can jointly handle multi-aspect object specifications. However, these methods assume the query image's region of interest is known. In our study, without knowing the curve-pattern composition in advance on a sherd, directly applying these methods may fail. This research develops a new Chamfer-matching based method and a patch-based matching method to better address the curve-pattern matching problem.

## 3.3 Sherd Identification by Curve-pattern Clustering

Our problem of identifying sherds from the same unknown design can be formulated as a **curve-pattern clustering** problem. Sherds originated from the same design are assigned to the same cluster. Cluster analysis is a long-studied topic, resulting in many classical algorithms. For example, K-means [44] clustering partitions observations into clusters by minimizing within-cluster variances. K-Means++ [2] removes the drawback of K-means, which is dependent on the initialization of the centroid. Affinity Propagation [25] clusters data by passing messages between the data points. Unlike K-means clustering, it does not need to specify the number of clusters in advance. MeanShift [73] basically assigns the data points to the clusters iteratively by shifting points towards the highest density of data points, i.e., cluster centroid. Hierarchical Clustering [38] builds nested clusters as a tree structure and successively merges or splits them to achieve the optimal clustering. Spectral Clustering [64] performs a low-dimension embedding of the affinity matrix between samples, followed by clustering, e.g., by K-means, of the components of the eigenvectors in the low dimensional space.

More recently, deep-learning-based algorithms have been proposed in image clus-

tering. For example, DeepCluster [13] is a self-supervision approach for learning image representations. It iteratively groups the features with a standard clustering algorithm, such as K-means. It uses the subsequent assignments as supervision to update the network's weights. Deep Adaptive Clustering (DAC) [15] recasts the clustering problem into a binary pairwise-classification framework to judge whether pairs of images belong to the same clusters. The similarities are calculated as the cosine distance between label features of images generated by a deep convolutional network (ConvNet). Deep Comprehensive Correlation Mining (DCCM) [72] proposes a framework that explores various kinds of correlations behind the unlabeled data. A pseudo-label supervision method is proposed to investigate category information and learn discriminative features. Graph Convolutional Network (GCN) Clustering [71] exploits the graph clustering structure for SGD-based training. It samples a block of nodes that associate with a dense subgraph identified by a graph clustering algorithm and restricts the neighborhood search within this subgraph. Approximate Rank-Order Clustering (ARO) [54] advances GCN clustering by measuring distances between images based on their shared nearest neighbors.

Unlike conventional image clustering problems, curve-pattern clustering aims at finding sherds with common partial curve patterns. Existing algorithms mostly take the input image as a whole. Thus they are not directly applicable to our problem.

# Chapter 4

# Chamfer-matching based Method for Design Identification

## 4.1 Motivation

The surface of a sherd with a composite pattern contains multiple incomplete copies of its underlying design, with various shapes, deformations, and overlaps. Such sherds are very common in existing archaeological collections. In this chapter, we study the problem of design identification on a sherd with possible a composite pattern. Illustrated in Fig. 4.1, each design copy of a composite pattern has its translation when matched to its underlying design. Traditional matching algorithms, such as template matching or Chamfer matching, require one pattern to be a portion of the other, which is not true as the curve pattern on the sherd is a composite one.

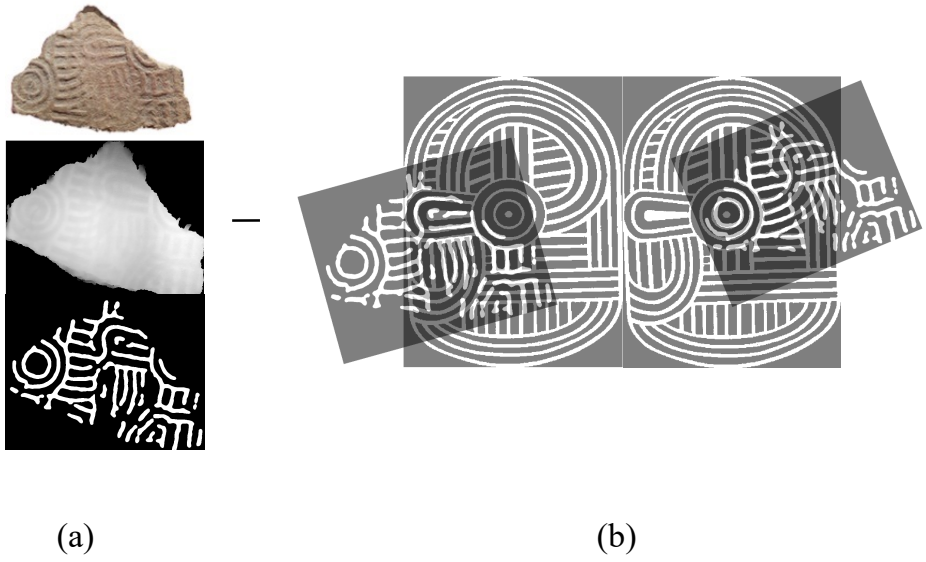



Figure 4.1 An illustration of a sherd with a composite pattern matching its underlying design: (a) a sherd with a composite pattern (from top to bottom: its RGB image, its depth image, and its extracted curve structure), and (b) the matching result of this sherd to its underlying design. (Original design reproduced with permission, courtesy of Frankie Snow, South Georgia State College.)

We develop a new Chamfer-matching based algorithm that can automatically identify underlying design on such a sherd to address this problem. After the curve structure segmented from the sherd, we thin the sherd and design curves to one-pixel

wide. While ideally the curve width can be used as an important cue in matching the sherd and a design, we try not to use the curve-width information in this study because it is very difficult to measure the curve width from a deteriorated sherd surface accurately. Therefore, in this study, we thin curves from both the sherd images and the binary paddle stamp design images, as shown in Fig. 4.2, and the matching distance is then defined based only on the one-pixel wide curves. Note that scale transform is usually not considered in the matching since both sherds and full designs have known sizes in real-world and their matchings are not scale invariant, i.e., two designs are considered different in archaeology even if they are identical after a uniform scale transform.

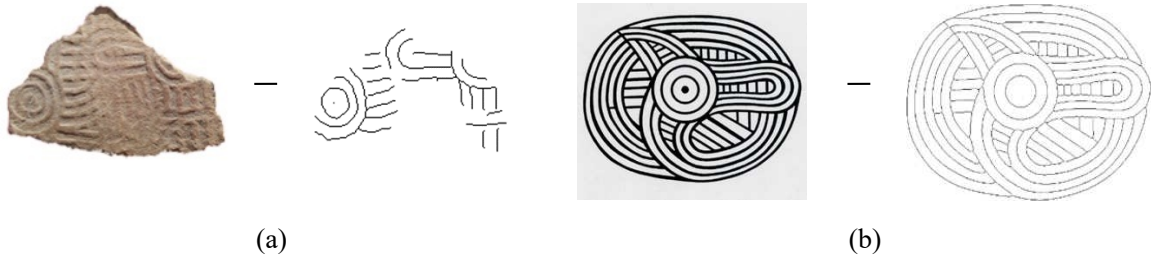



Figure 4.2 An illustration of one-pixel wide curve patterns from a sherd and a design. (a) Curve pattern from a sherd. (b) Curve pattern from a design. (Original design reproduced with permission, courtesy of Frankie Snow, South Georgia State College.)

## 4.2 Method

Chamfer matching [4] is an efficient algorithm that has been successfully used for partial-to-global matching of one-pixel-wide curve patterns. This section briefly introduces the classical Chamfer matching algorithm and applies it to our sherd-to-design curve pattern matching problem. And then, we present the new curve pattern matching algorithm by extending several concepts in Chamfer matching.

### 4.2.1 Chamfer Matching of Curve Patterns

Given a sherd curve pattern $U$ and a full design curve pattern $V$, the goal of Chamfer matching is to decide whether $U$ is a portion of $V$ and if so, find the transform that matches the partial pattern $U$ to the full pattern $V$. Both $U$ and $V$ are one-pixel-wide. For clarity, in the work we use $U$ and $V$ to denote the set of 2D coordinates of the curve pixels in these two patterns, respectively. The transform $\mathbf{T}$ that matches $U$ to $V$ usually consists of a translation $\mathbf{t} = (t_x, t_y)$, a rotation of angle $\theta$, and a scaling with factor $s$. Chamfer matching is based on the Chamfer distance defined between $U$ and $V$. Let $U_{\mathbf{T}}$ be the partial pattern $U$ after the transform $\mathbf{T}$. Aligning $U_{\mathbf{T}}$ and $V$, we can define a Chamfer distance between them as

$$d'_{CM}(U_{\mathbf{T}}, V) = \frac{1}{|U|} \sum_{\mathbf{u} \in U_{\mathbf{T}}} \min_{\mathbf{v} \in V} \|\mathbf{u} - \mathbf{v}\|_2 , \tag{4.1}$$

where $\mathbf{u} \in U_{\mathbf{T}}$ indicates all the curve-pixel coordinates $\mathbf{u}$ in the transformed partial pattern $U_{\mathbf{T}}$ and $\mathbf{v} \in V$ indicates all the curve-pixel coordinates $\mathbf{v}$ in the curve pattern $V$. $|U|$ is the total number of curve pixels in the partial pattern $U$. Eq. (4.1) actually finds the nearest curve-pixel coordinate in $V$ for each curve-pixel coordinate in $U_{\mathbf{T}}$, records its Euclidean distance $\|\mathbf{u} - \mathbf{v}\|_2$ and finally averages over all the curve-pixel coordinates in $U_{\mathbf{T}}$. By trying all possible transforms $\mathbf{T}$'s, the transform $\mathbf{T}^*$ for the best matching can be determined by

$$\mathbf{T}^* = \arg\min_{\mathbf{T}} d'_{CM}(U_{\mathbf{T}}, V). \tag{4.2}$$

The Chamfer matching distance between $U$ and $V$ is then defined as the minimum Chamfer matching distance corresponding to the optimal transform $\mathbf{T}^*$, i.e.,

$$d_{CM}(U, V) = d'_{CM}(U_{\mathbf{T}^*}, V). \tag{4.3}$$

In practice, we can examine the Chamfer matching distance $d_{CM}(U, V)$ – if it is larger than a given threshold, we may consider that $U$ is not a partial pattern

of $V$; otherwise, we can consider that $U$ is a partial pattern of $V$ and $\mathbf{T}^*$ provides the location, orientation and scaling that match $U$ to $V$. In Chamfer matching, we need to search over all possible transform parameters of $\mathbf{T}$. Therefore, the reduction of the degree of freedom in $\mathbf{T}$ can substantially reduce the search space and speed up the algorithm. We match sherd pattern $U$ to the full paddle design $V$ in this work. Thi matching is not scale-invariant, and all the sherd and design images have been preprocessed to have a uniform DPI (dots per inch), as discussed in section 4.1. Therefore, the transform $\mathbf{T}$ in this work only consists of a translation $\mathbf{t}$ and a rotation with angle $\theta$, i.e.,

$$\mathbf{T}(\mathbf{u}) = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \mathbf{u} + \begin{pmatrix} t_x \\ t_y \end{pmatrix}, \tag{4.4}$$

where the search range for $\theta$ is $[0°, 360°)$ and the search range for $t_x$ and $t_y$ can be constrained by the size (width and height) of the bounding boxes that tightly cover the sherd pattern $U$ and design pattern $V$.

Based on Eq. (4.2), we need to calculate the matching distance $d'_{CM}(U_{\mathbf{T}}, V)$ for each possible choice of parameters in transform $\mathbf{T}$. In Chamfer matching, this can be accelerated by pre-computing the distance map for $V$ – the distance map value $M(\mathbf{u})$ at any 2D coordinate $\mathbf{u}$ indicates the Euclidean distance from $\mathbf{u}$ to the nearest coordinate of an edge pixel in $V$. For $\mathbf{v} \in V$, we have $M(\mathbf{v}) = 0$. This way, Eq. (4.1) can be simplified as

$$d'_{CM}(U_{\mathbf{T}}, V) = \frac{1}{|U|} \sum_{\mathbf{u} \in U_{\mathbf{T}}} M(\mathbf{u}). \tag{4.5}$$

An example of the distance map built for a design $V$ and its use for computing Chamfer distance is shown in Fig. 4.3.

### 4.2.2 Composite Pattern Matching

The classical Chamfer matching discussed above requires that the pattern $U$ is a portion of the full design $V$, under a transform $\mathbf{T}$. The developed sherd-to-design matching problem requires that the sherd partially contains a single copy of the full

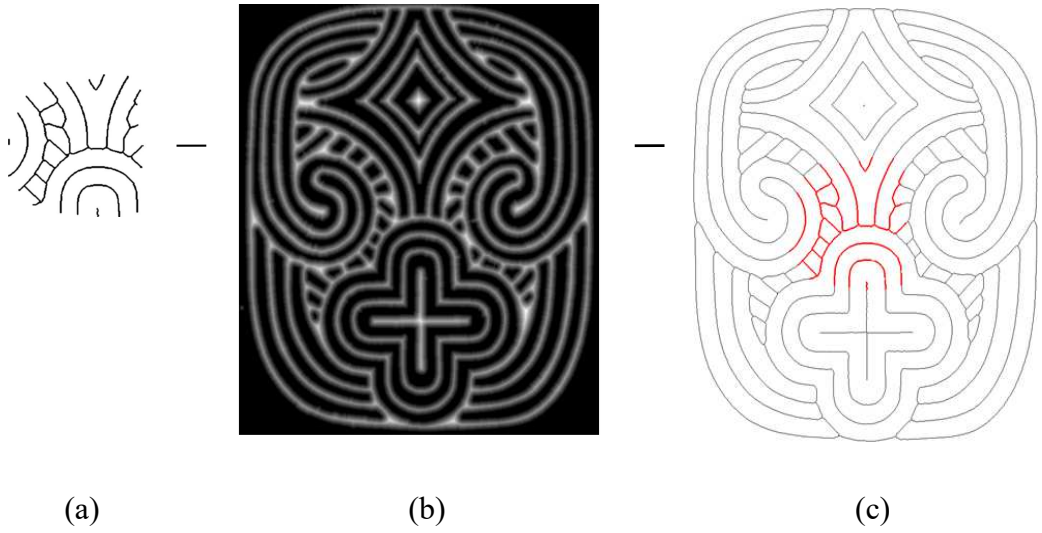



Figure 4.3 An illustration of the distance map and Chamfer matching. (a) Curve pattern on sherd. (b) Distance map of design - brighter pixels indicate higher values in the distance map. (c) Chamfer matching result (in red). (Original design reproduced with permission, courtesy of Frankie Snow, South Georgia State College.)

design. However, in the practice of archaeology, the curve pattern on a sherd may be a composite one – the same carved paddle was applied to the pottery surface multiple times with spatial overlaps, and the sherd may contain multiple partial, spatially overlapped copies of the same design, as an example shown in Fig. 4.4, where the sherd curve pattern consists of two components, corresponding to the two overlapped copies of the same design. Two components (red and green) of the composite pattern are matched to different design parts, with blue curve fragments shared by the two components. In this case, the direct application of the classical Chamfer matching could not find the correct partial-to-global matching between the curve pattern $U$ on the sherd and the design $V$.

To address this problem, we need to allow different parts of the sherd pattern $U$ to be matched to the different parts of the design $V$. Ideally, $U$ can be matched to

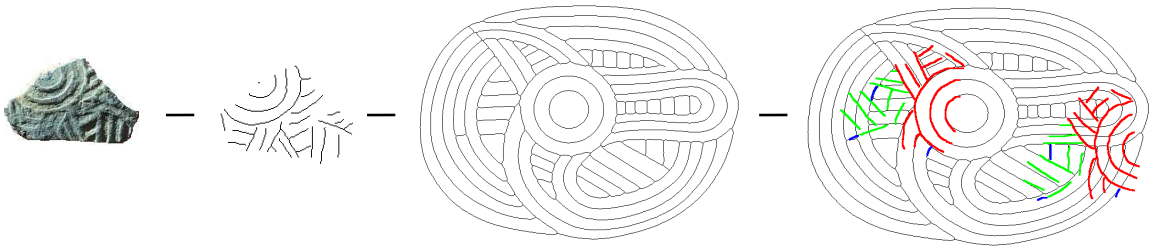



Figure 4.4 An Illustration of a composite pattern, which consists of two components. (a) A sherd with a composite pattern. (b) The extracted composite pattern. (c) The underlying design. (d) Two components (red and green) of the composite pattern matched to different parts of the design, with blue curve fragments shared by two components. (Original design reproduced with permission, courtesy of Frankie Snow, South Georgia State College.)

$V$ by decomposing $U$ into $\{U_1, U_2, \cdots, U_K\}$ such that

$$U = \bigcup_{k=1}^{K} U_k \tag{4.6}$$

$$U_i \bigcap U_j = \emptyset, \forall i \neq j; i, j = 1, 2, \cdots, K, \tag{4.7}$$

and then each component $U_k$ can be matched as a portion of $V$ with its own transform. This way, we can define the matching distance (or score) between $U$ and $V$ by combining the matching distance (or score) between each component $U_k$ and $V$.

The above first condition in Eq. (4.6) reflects the *completeness* of the decomposed pattern components $U_k, k = 1, 2, \cdots, K$. Considering the possible noise in the sherd pattern, we simply seek the decomposition that maximizes the completeness

$$\phi_c(U_1, U_2, \cdots, U_K) = \frac{|\bigcup_{k=1}^{K} U_k|}{|U|}. \tag{4.8}$$

The above second condition in Eq. (4.7) reflects the *disjointness* of the decomposition. Considering the possibility of shared curve fragments across multiple components, as illustrated in Fig. 4.4, in this work we relax this condition to

$$\phi_d(U_i, U_j) = \frac{|U_i \cap U_j|}{|U_i \cup U_j|} < \eta, \forall i \neq j; i, j = 1, 2, \cdots, K, \tag{4.9}$$

where $\phi_d(U_i, U_j)$ is the *disjointness* between two components $U_i$ and $U_j$, and $\eta$ is a preset threshold for the disjointness.

In practice, we also need to limit the number of components $K$. If we over-decompose $U$ to too many very simple curve patterns, e.g., each $U_k$ only contains one edge pixel, we can always perfectly match each $U_k$ to $V$ with zero Chamfer distance. Considering that the sherds are highly fragmented pieces of the pottery and it is not common to see a sherd that partially contains more than two copies of the full design $V$, we only consider the cases of $K \leq 2$ in all our experiments.

Based on these considerations, the main problem we need to address is to find the optimal decomposition $U_k, k = 1, 2, \cdots, K$ for $U$ to match the design $V$ and quantify the matching distance or score. The possible choices of decompositions are very large given a large number of edge pixels in $U$, and we could not try every possible decomposition to search for the global optimum. In this work, we use the Chamfer distance between each component of $U$ and $V$ to reduce the search space of decomposition.

More specifically, for each possible transform $\mathbf{T}$ consisting of a translation $\mathbf{t}$ and a rotation $\theta$, we align $U_{\mathbf{T}}$ and the design $V$. Using the distance map $M(V)$, we can construct a candidate component $U(\mathbf{T}) = U(\mathbf{t}, \theta) \subseteq U$ by collecting all the edge-pixel coordinates $\{\mathbf{u} | \mathbf{u} \in U_{\mathbf{T}}, M(\mathbf{u}) < \alpha\}$, where $\alpha$ is a threshold to determine whether $\mathbf{u}$ has a corresponding matching edge pixel in $V$ under the transform $\mathbf{T}$ and let $d'_{CM}(U_{\mathbf{T}}(\mathbf{T}), V)$ be the Chamfer distance between this candidate component and $V$. For each translation $\mathbf{t}$, we first try all possible values of $\theta$ in $[0°, 360°)$ and keep the one that leads to the minimum Chamfer distance, i.e.,

$$\theta^*_{\mathbf{t}} = \arg\min_{\theta} d'_{CM}(U_{\mathbf{T}}(\mathbf{t}, \theta), V). \tag{4.10}$$

We construct a candidate component $U(\mathbf{t}) = U(\mathbf{t}, \theta^*_{\mathbf{t}})$ for each translation offset $\mathbf{t}$ and

the matching distance between this candidate component and $V$ is defined as

$$\tilde{d}(\mathbf{t}) = d'_{CM}(U_{(\mathbf{t},\theta^*_{\mathbf{t}})}(\mathbf{t}, \theta^*_{\mathbf{t}}), V). \tag{4.11}$$

After constructing a candidate component at each possible translation $\mathbf{t}$, we obtain a new distance map $\tilde{d}$ of the same size as $M(V)$. We also construct a large number of candidate components from $U$, one for each translation offset $\mathbf{t}$. Trying all possible combinations of these candidate components is computationally expensive. In practice, one can expect that the candidate components constructed at two neighboring $\mathbf{t}$'s are very similar. Therefore, we use a minimum-suppression strategy to reduce the number of candidate components further. Specifically, we find the regional local minimum on the new distance map $\tilde{d}$ and only keep the candidate components at $\mathbf{t}$'s corresponding to these local minimums. Assume that the local minimums are found at $\mathbf{t}_1, \mathbf{t}_2 \cdots, \mathbf{t}_p$ and their corresponding candidate components are $U(\mathbf{t}_1), U(\mathbf{t}_2), \cdots, U(\mathbf{t}_p)$ respectively. We can consider the $K$ combination of them for final components. As discussed above, we set the actual number of components in $U$ to be $K \leq 2$. Therefore, we limit the search space for the decomposition of $U$ to the following $p + \frac{p(p-1)}{2}$ cases:

1. The $p$ cases where $U$ only contains one single component, i.e., $U(\mathbf{t}_1), U(\mathbf{t}_2), \cdots, U(\mathbf{t}_p)$.
2. The $\frac{p(p-1)}{2}$ cases where $U$ contains two components, i.e., $U(\mathbf{t}_i)$ and $U(\mathbf{t}_j)$, with $i < j$, $i, j = 1, 2, \cdots, p$.

This can be extended to the cases where $U$ is decomposed into more than two components, but the size of the search space will substantially increase. For each of the $p + \frac{p(p-1)}{2}$ cases in the search space, we evaluate the completeness $\phi_c$ and disjointness $\phi_d$. Finally, we keep the one case with the maximum completeness subject to the constraint that its disjointness is less than $\eta$, as defined in Eqs. (4.8) and (4.9). The completeness $\phi_c$ for this case is then taken as the matching score between $U$ and

$V$, and we denote this score as $\phi(U, V)$. The higher this matching score, the better the partial-to-global matching between $U$ and $V$. Since we consider the cases with $K = 1$ and $K = 2$ components into one unified optimization process, this algorithm can automatically identify whether $U$ partially contains one or multiple copies of the same design. The process of the composite pattern matching is illustrated in Fig. 4.5, and this algorithm is summarized in Algorithm 1.

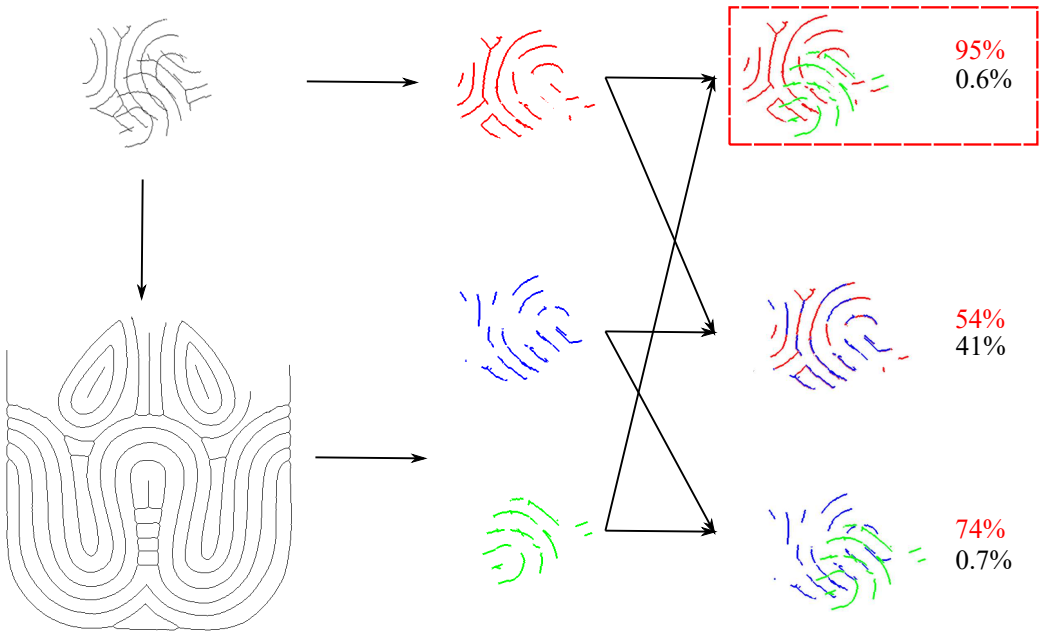



Figure 4.5 The process of combining candidate components for matching to a design ($K = 2$). The optimal result is indicated in the red box. (a) Matching a sherd pattern (top) to a design pattern (bottom). (b) Candidate Components. (c) Combining candidate components (completeness scores $\phi_c$ shown in red and disjointness scores $\phi_d$ shown in black). (Original design reproduced with permission, courtesy of Frankie Snow, South Georgia State College.)

## 4.3 Experiments

To test the developed method, we assembled an image dataset of 100 sherds from archaeological sites in the Snow Collection associated with the Swift Creek paddle stamped tradition of southeastern North America [68, 58]. These 100 sherds have

**Algorithm 1** Algorithm for composite sherd-to-design Matching.

Input: A sherd image and the design image database
**for** all design images in design image database **do**
  Extract the curve patterns $U$ from the sherd image and $V$ from the design image
  **for** all translation $\mathbf{t}$ of $U$ on $V$ **do**
    **for** all $\theta$ in $[0°, 360°)$ **do**
      Calculate component $U_{\mathbf{T}}(\mathbf{T})$ with Chamfer distance $d'_{CM}(U_{\mathbf{T}}(\mathbf{T}), V)$
    **end for**
    Construct a candidate component $U(\mathbf{t})$ by Eq. (4.11)
  **end for**
  Reduce the candidate components by taking the local minimums at the map $\tilde{d}$
  Find the optimal component $U_i$ or combined components $\{U_i, U_j\}$ from the constrained set defined by Eq. (4.9), with the maximum completeness $\phi_c$ defined in Eq. (4.8)
  Store completeness $\phi_c$ as the matching score
**end for**
Sort the matching scores for all designs and find the best matched designs

curve patterns representing 20 unique paddle designs. Samples of these sherds and designs are illustrated in Fig. 4.6. About 80% of the sherds clearly show composite patterns, i.e. sherds shown in Fig. 4.6 (b), (c) and (d), part of the multiple copies of the same design with spatial overlap. We extract curve structures from these sherds manually. Then we perform a thinning operation using MATLAB function `bwmorph` to get one-pixel-wide curves. We further remove small branches (less than 10 pixels) by using the MATLAB function `findendsjunctions` included in LineSegments package[1].

We use the CMC ranking metric to evaluate the matching performance in our experiment. To identify the underlying design of a sherd pattern $U$, we match it against all 20 designs. We then sort these 20 designs in terms of the matching scores and pick the top $L$ designs with the highest scores. If the ground-truth design of this sherd is among the identified top $L$ designs, we treat it as a correct design

[1]http://www.peterkovesi.com/matlabfns/

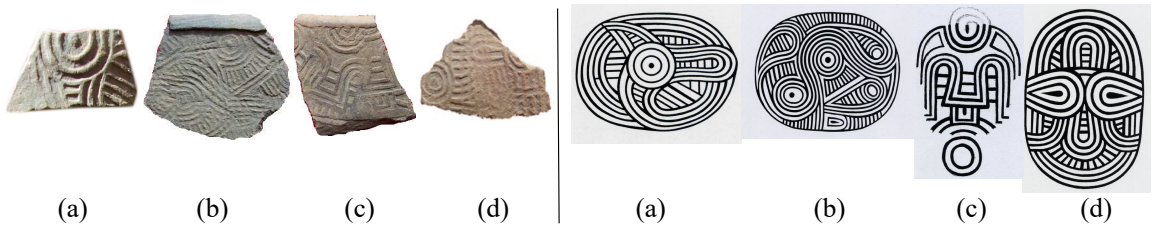



Figure 4.6 Sample sherds (left) and designs (right) in our collected data that are used for performance evaluation. (Original designs reproduced with permission, courtesy of Frankie Snow, South Georgia State College.)

identification under rank $L$. We repeat this identification for all 100 sherds and calculate the accuracy, i.e., the percentage of the correctly identified sherds, under each rank $L$, $L = 1, 2, \cdots, 20$. This way, we can obtain a CMC curve in terms of rank $L$ as shown in Fig. 4.7 to evaluate the performance of a matching algorithm. The higher value in this curve, the better the matching performance. For parameter settings in the developed method, we set $\alpha = 3$ to decide whether an edge pixel in the transformed sherd pattern has been matched to an edge pixel in the design (see Section 4.2.2). We set $\eta = 0.1$ for the disjointness score to allow the possible sharing of some edge pixels between different components in the sherd pattern (see Section 4.2.2).

To justify the effectiveness of the developed method, we pick four classical matching algorithms for performance comparison in the experiments. 1) *(Basedline) Chamfer Matching* without considering composite patterns. It takes the same curve patterns $U$ and $V$ from sherds and designs, respectively, as in our method, calculates the Chamfer distance $d_{CM}(U, V)$ and uses it for computing the CMC ranking. 2) *Image Matching*, in which the sherd image, after converting to gray-scale, is translated and rotated for a best matching to each design in terms of pixel intensity. Denote $I^U$ and $I^V$ as the sherd image (gray-scale) and design image, respectively, and let $\mathbf{T}$ be the

transform of $I^U$ over $I^V$, then the matching score $S$ is defined as

$$S = \max_{\mathbf{T}} \frac{\sum_{x,y} I^U_{\mathbf{T}}(x,y) \cdot I^V(x,y)}{\sqrt{\sum_{x,y}(I^U_{\mathbf{T}}(x,y))^2 \cdot \sum_{x,y}(I^V(x,y))^2}}, \tag{4.12}$$

where the transform $\mathbf{T}$ considers all the possible translation and rotation as in our method. In the experiment, we directly use an OpenCV implementation of this algorithm. 3) *Shape Context*. It uses the same curve patterns $U$ and $V$ as in our method. To deal with partial matching, we use a sliding window technique to match a sherd curve pattern to each window-cropped design curve pattern and then choose the one with the lowest matching distance. The sliding-window size is the same as the sherd image. The Shape Context algorithm implementation directly comes from the OpenCV package. 4) *Histogram of Orientation Shape Context (HOOSC)* [60]. Its setup is the same as Shape Context but incorporates the orientation measure into the log-polar histograms. It is based on Roman-Rangel's [60] paper and implemented by HG Zhao[2] using MATLAB.

Figure 4.7 shows the CMC curves of the developed method and the four comparison methods. All four comparison methods clearly show very poor performance by having CMC curves along the diagonal line. The major reason for their poor performance is that they do not consider and cannot well handle the composite patterns present on the sherds. Our method achieves much better CMC performance by explicitly considering the possible composite patterns.

Figure. 4.8 shows the sample results of the developed method and the four comparison methods when matching two sherds to the designs. In these two examples, we can see that our method can identify the correct designs under CMC rank 1. In contrast, the four comparison methods can only identify the correct designs under much higher CMC ranks.

Figures 4.9, 4.10, 4.11 and 4.12 show the matching results of four sherd samples,

[2]https://github.com/CyberZHG/Sketch-Based/tree/master/HOOSC

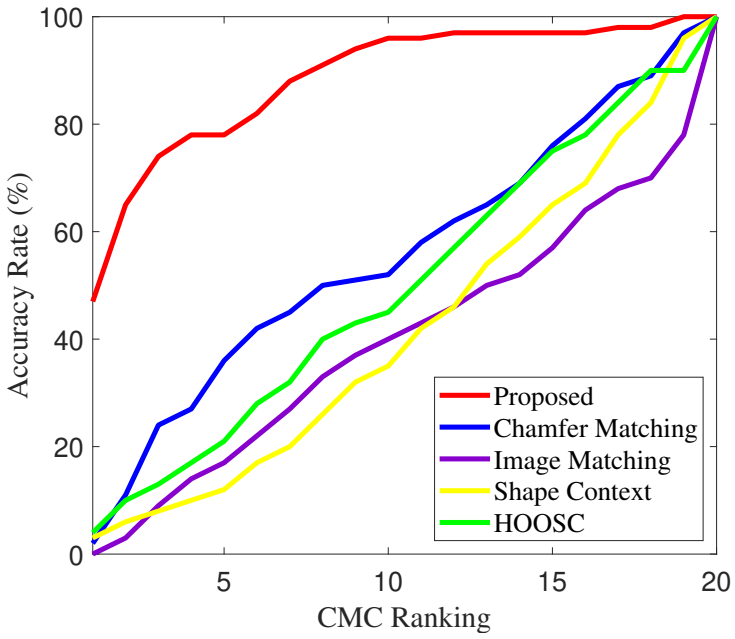



Figure 4.7 CMC curves of the developed method and the four comparison methods.

respectively. On the left column of these figures are the sample sherds and their best-matched designs. We can see that composite patterns are present on all these four sherds. On the right side of these figures, we show the matching results of each sherd over its best-matched design, using our method and the four comparison methods. Specifically, for each sherd, (a) and (b) show the identified two components (in green) of the sherd pattern and their matched locations/orientations on the design, respectively. (c), (d), (e), and (f) show the best-matched locations/orientations of the sherd pattern on the design using Chamfer Matching, Image Matching, Shape Context, and HOOSC, respectively. The values above the results of the comparison methods are their respective matching costs or scores. For Chamfer Matching, this value is the Chamfer distance defined in Eq. (4.3). For Image Matching, it is the matching score defined in Eq. (4.12). For Shape Context and HOOSC, these values are the respective matching distances. Note that, in the four comparison methods,

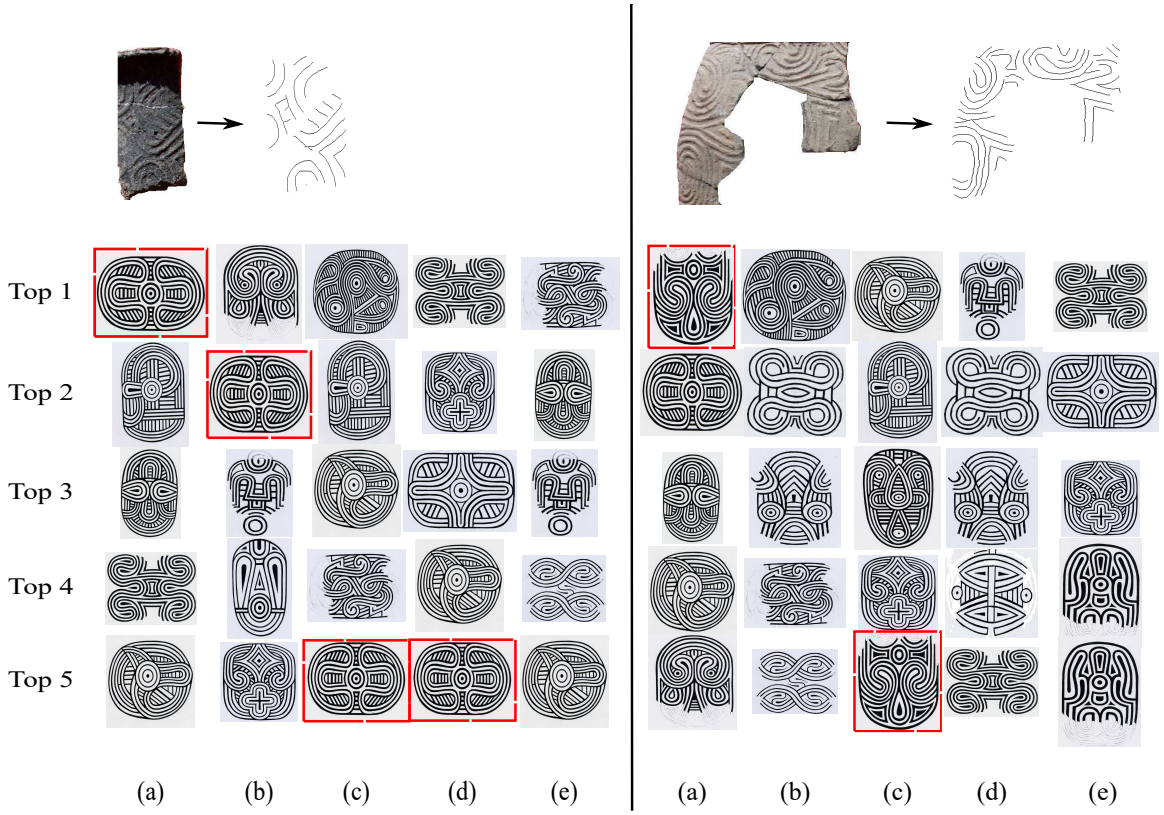



Figure 4.8 The design identification result for two sample sherds. The top matched designs identified by the developed method shown in column (a), while the top matched designs identified by Chamfer Matching, Image Matching, Shape Context and HOOSC are shown in columns (b), (c), (d) and (e) respectively. (Original designs reproduced with permission, courtesy of Frankie Snow, South Georgia State College.)

the sherd pattern is not decomposed into multiple components, and they are matched to the design as a whole. Archaeologists who specialize in studying these designs confirmed that the matching results of these four sherds are correct when using our method. For the comparison methods, the only correct matching is produced by Chamfer Matching on the first sample sherd, as shown in Fig. 4.9 (c).

We also inspected the experiment results to find the failure cases when using our method and the cause of failure cases. Specifically, we examined the sherds whose ground-truth designs are not among the top five matchings, i.e., incorrect matching under CMC rank 5. Examples of these failure cases are shown in Fig. 4.13.

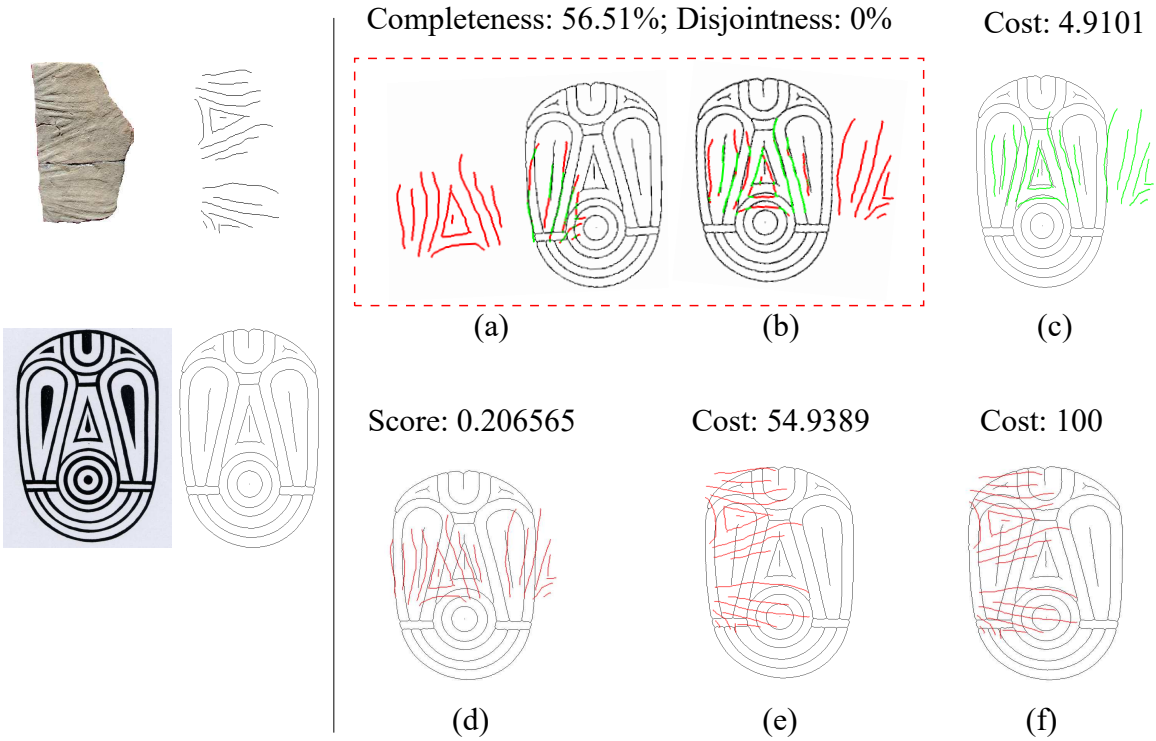



Figure 4.9 Matching results of a sample sherd. (a-b) Two matched components on the design using the developed method. (c) Result from Chamfer Matching. (d) Result from Image Matching. (e) Result from Shape Context. (f) Result from HOOSC. (Original design reproduced with permission, courtesy of Frankie Snow, South Georgia State College.)

We found that most of these failure cases are caused by the local pattern similarity of the designs. Most of the southeastern North American paddle designs are usually combinations of simple curve patterns, such as the concentric circles as shown in Fig. 4.13 (a). When such simple curve patterns dominate the composition of a sherd pattern, it can be easily confused to many designs other than its ground truth design. Furthermore, the curve pattern segmented from a real sherd usually contains noise, missing segments, and inaccuracies because of variable surface smoothing during vessel manufacture, incomplete application of the planar paddle to the curved pottery surface, and surface erosion from post-depositional weathering as shown in Fig. 4.13 (b). Another possible issue is the warping caused by scanning a highly curved sherd,

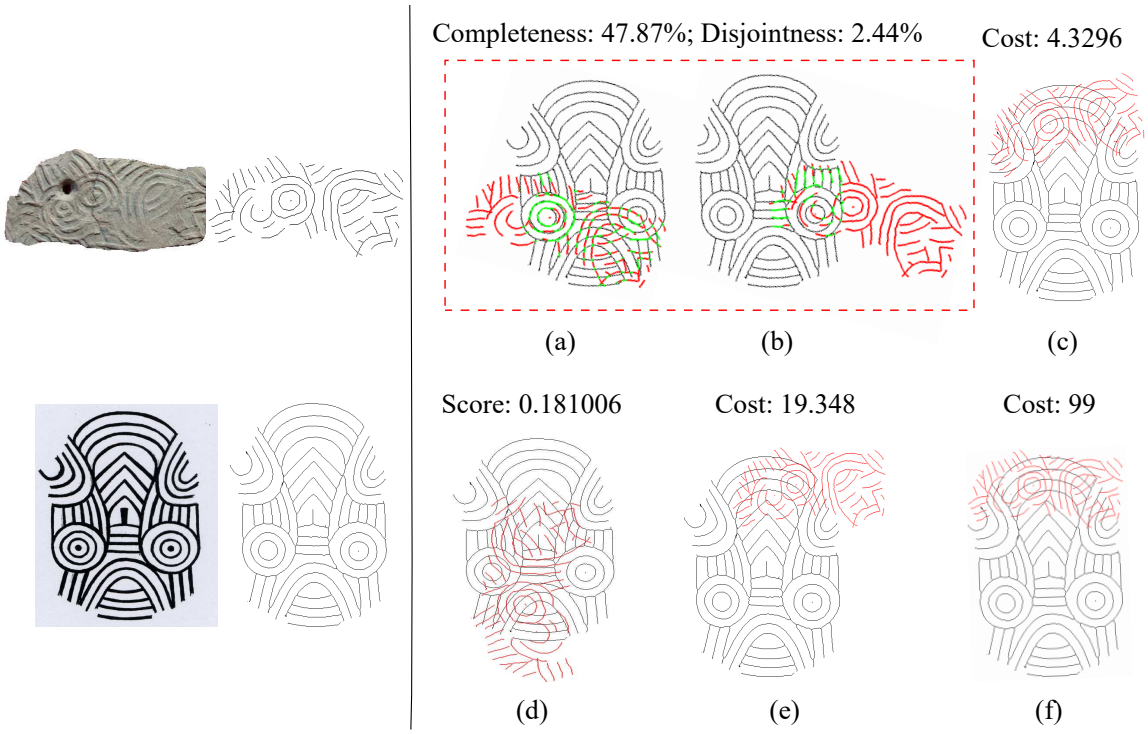



Figure 4.10 Matching results of another (second) sample sherd. (a-b) Two matched components on the design using the developed method. (c) Result from Chamfer Matching. (d) Result from Image Matching. (e) Result from Shape Context. (f) Result from HOOSC. (Original design reproduced with permission, courtesy of Frankie Snow, South Georgia State College.)

as shown in Fig. 4.13 (c). For the issue of noisy curve patterns, we expect that a machine-learning-based algorithm can well address it.

## 4.4 Chapter Summary

In this chapter, we developed a partial-to-global curve-pattern matching algorithm to identify the designs of the carved wooden paddles from unearthed pottery sherds. Unlike previous partial matching problems, the curve pattern on each sherd may be a composite one resulting from multiple, partially overlapped copies of the same design. To address this problem, we extended the classical Chamfer matching to identify candidate components of the sherd pattern and then leveraged two metrics

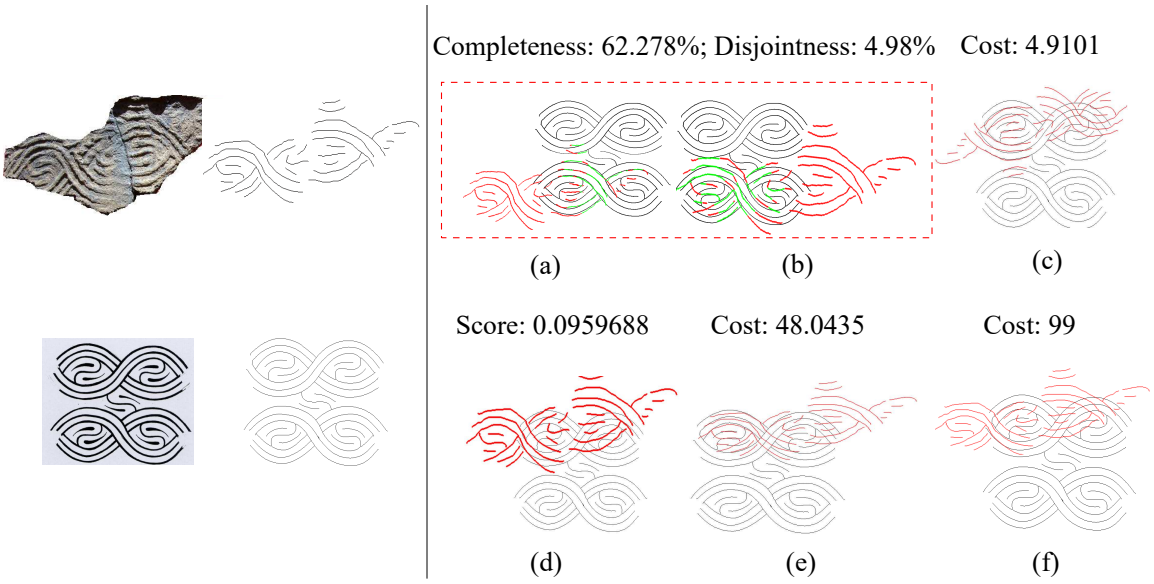



Figure 4.11 Matching results of another (third) sample sherd. (a-b) Two matched components on the design using the developed method. (c) Result from Chamfer Matching. (d) Result from Image Matching. (e) Result from Shape Context. (f) Result from HOOSC. (Original design reproduced with permission, courtesy of Frankie Snow, South Georgia State College.)

of completeness and disjointness to find the optimal sherd-pattern decomposition. In this experiment, we tested a collection of 100 sherds against 20 known southeastern North America paddle designs. The results showed that the CMC performance of our method is substantially better than several traditional images and curve-pattern matching algorithms. The work of this chapter has been published in the Journal of Electronic Imaging [81].

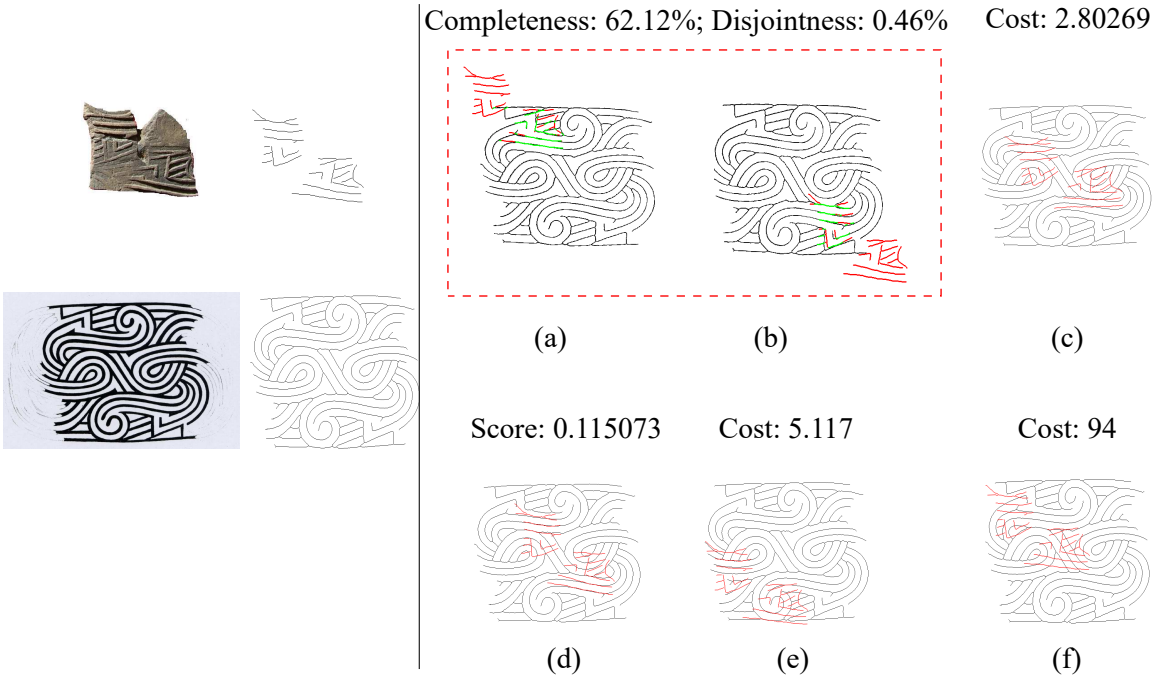



Figure 4.12 Matching results of another (fourth) sample sherd. (a-b) Two matched components on the design using the developed method. (c) Result from Chamfer Matching. (d) Result from Image Matching. (e) Result from Shape Context. (f) Result from HOOSC. (Original design reproduced with permission, courtesy of Frankie Snow, South Georgia State College.)

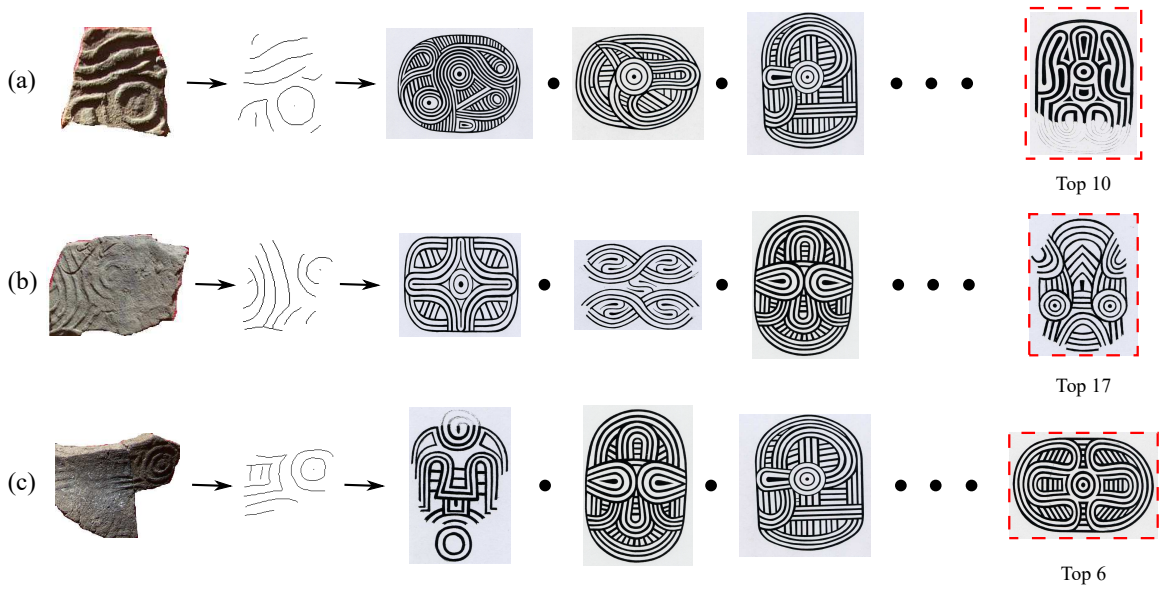



Figure 4.13 Three failure cases where the top five matched designs do not include the ground-truth design. From left to right are the sherd image, curve pattern extracted from the sherd, and the top matched designs returned by the developed method, respectively. (Original designs reproduced with permission, courtesy of Frankie Snow, South Georgia State College.)

# Chapter 5

# Patch-based Matching for Design Identification

## 5.1 Motivation

Pottery sherds found in existing archeological records are usually highly degraded and at the same time consist of composite patterns. Inspired by the recent success of keypoint feature matching in natural images, we wish to find a type of keypoint feature that works for composite-pattern matching, and is robust to noises and errors presented in sherd curve patterns. However, we found that the widely used keypoint-base methods, e.g., SIFT, require keypoints to contain distinctive local features. Such keypoints on the special curve-like structure images can be very scarce, because these keypoint features can be very monotonous, as two samples shown in Fig. 5.1. Unlike matching images by a few points, matching a sherd to designs requires a common continuous region to be matched, and the common region needs to be large enough to contain a distinctive design pattern. We develop a new patch-based curve-pattern

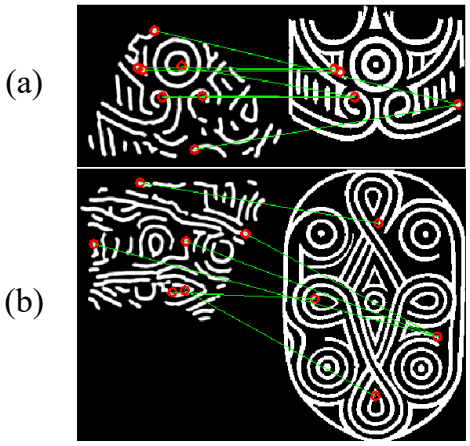



Figure 5.1 (a) Matching a sherd with a single pattern to its underlying design by SIFT. (b) Matching a sherd with a composite pattern to its underlying design by SIFT. (Original design reproduced with permission, courtesy of Frankie Snow, South Georgia State College.)

matching algorithm to identify designs on pottery sherds, discussed in the following.

## 5.2 Method

The new patch-based curve-pattern matching algorithm consists of three steps: 1) dividing and sampling the binary curve-pattern images of the sherd and the design for small image patches, 2) employing a triplet CNN network to evaluate the pairwise matching score between sherd and design, and 3) identifying the matching portions by region growing. The whole sherd-to-design matching process is shown in Fig. 5.2.

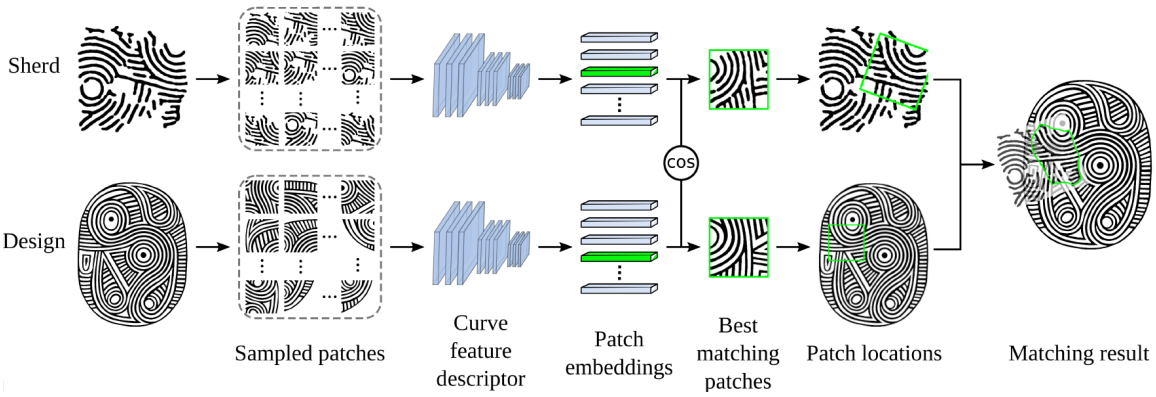



Figure 5.2 An illustration of the matching between a sherd and its underlying design. (Original designs reproduced with permission, courtesy of Frankie Snow, South Georgia State College.)

### Step 1: Patch Sampling from Sherd and Design

For the sherd, the described segmentation algorithm in Chapter 2.1 converts it to a 2D binary image of curve pattern. The design also takes form of a binary image by converting from its digitized gray scale image. We first perform a thinning operation to convert both binary images into one-pixel-wide skeleton images. We then uniformly divide each of the skeleton images to non-overlapping grids of size $g \times g$. If a grid contains one or more skeleton pixels, we take the skeleton pixel closest to the grid center as the patch center, and then take its surrounding $p \times p$ region as a patch, on the curve-pattern image before thinning. This process is briefly illustrated in Fig. 5.3.

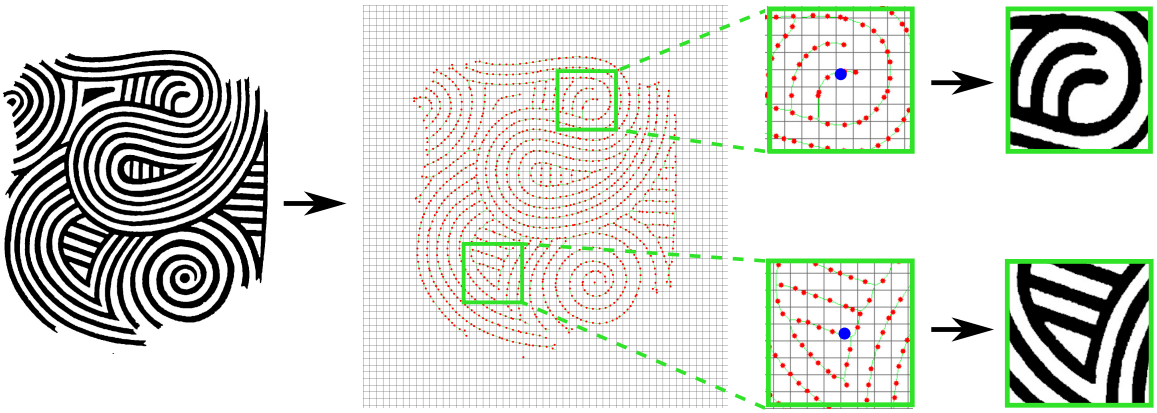


Figure 5.3 Illustration of our patch sampling strategy. We first divide the image into small grids. In each grid, we find the skeleton pixel (red points) that is closest to the grid center. Around each selected skeleton pixel (blue points), we crop a patch of size $p \times p$ on the curve-pattern image. (Original designs reproduced with permission, courtesy of Frankie Snow, South Georgia State College.)

With this patch sampling strategy, we obtain two sets of sherd and design patches for matching. We only need to consider one patch for each grid instead of for each pixel, and it will not miss possible matchings because all curve regions are covered. Finally, considering the rotation transform, we augment each patch constructed for the sherd by rotating it 36 times by $0°, 10°, \cdots, 350°$, respectively.

The patch size $p$ is an important parameter that needs to be pre-set. The smaller the value of $p$ is, the more patches we sample and compare, leading to higher computation time. In addition, overly small patches may lack discriminative curve pattern features for design identification. On the contrary, an overly large patch may still cover composite patterns, which prevents the correct matching between a sherd and its underlying design. In our experiment, we empirically set $p = 300$ pixels such that each patch covers a $3cm \times 3cm$ region, on either the surface of sherd or the design.

### Step 2: Matching Score Estimation by Learning

After Step 1, both the sherd and the design are represented by a set of sampled and augmented patches, each of which is a binary image of curve patterns. We match

Table 5.1 Network architecture of the curve feature descriptor, where n, k, st, pd are the number of output channels, kernel size, stride and padding size, respectively. ReLu and BatchNorm layers are not included.

| **Type** | **Configuration** |
|---|---|
| Convolution | n:128, k:8 × 8, st:1, pd:0 |
| Dropout | ratio:0.5 |
| Convolution | n:128, k:3 × 3, st:1, pd:1 |
| Convolution | n:128, k:3 × 3, st:2, pd:1 |
| Convolution | n:64, k:3 × 3, st:2, pd:1 |
| Convolution | n:64, k:3 × 3, st:2, pd:1 |
| Convolution | n:32, k:3 × 3, st:2, pd:1 |
| Convolution | n:32, k:3 × 3, st:1, pd:1 |
| Input | 300 × 300 curve-pattern image |

every pair of patches sampled from the sherd and the design, respectively, and evaluate their similarity by a matching score. The patch matching score shall be robust to not only the noise and deformation present in sherd images, but also a certain level of displacement, i.e., translation and rotation, because our patch sampling in Step 1 could not cover all possible locations and rotations. For this purpose, we train a CNN-based curve feature descriptor to compute pairwise patch matching scores via deep embedding learning.

The network architecture of the curve feature descriptor is illustrated in Table 5.1. In training, we optimize it in the form of triplet network [34], which consists of three branches with shared weights and the three inputs are an anchor sherd patch $I_S$, a positive design patch $I_D^+$, and a negative design patch $I_D^-$. The positive $I_D^+$ refers to the ground-truth patch on the design $D$ that matches the sherd patch $I_S$ and the negative $I_D^-$ can be any design patch that does not match $I_S$. The three branches of the network then produce three deep feature vectors $F_S$, $F_D^+$, and $F_D^-$. We use the following triplet loss for network training:

$$L_{tri} = \max\left(\left\|F_S - F_D^+\right\|_2 - \left\|F_S - F_D^-\right\|_2 + \alpha, 0\right), \tag{5.1}$$

where $\alpha$ is a constant margin value. This loss function aims at minimizing the distance

between $F_S$ and $F_D^+$, and maximizing the distance between $F_S$ and $F_D^-$.

The selections of $I_D^+$ and $I_D^-$ are crucial to the training of the triplet network. If we only use easy-to-discriminate $I_D^+$ and $I_D^-$ in training, the triplet loss is likely to be 0 and the trained network may not be able to distinguish hard-to-discriminate cases in testing. To address this issue, we adopt two strategies to construct the training samples $I_D^+$ and $I_D^-$. First, to tolerate certain-level of displacement in matching, we spatially transform the window of each positive patch $I_D^+$ and use it to crop the design $I_D$ for more augmented positive design patches. More specifically, we construct 81 positive design patches by taking the combination of rotation of $-10°$, $0°$, or $10°$, scaling of factor 0.9, 1, or 1.1, horizontal translation of -10, 0, or 10 pixels, and vertical translation of -10, 0, or 10 pixels. Second, for each anchor sherd patch $I_S$, we use template matching [10] to search from the patches sampled from all the known designs, except for the ground-truth design $D$, a set of 81 patches with highest template-matching scores. We take them as hard negative design patches $I_D^-$ for enhancing the triplet network training. Using these two strategies, we also achieve a data balance between positive and negative samples. In testing, for each sherd patch $I_S$ and each design patch $I_D$, we can use the trained network to compute their deep feature vectors and calculate their cosine similarity as their matching score:

$$\phi(I_S, I_D) = \frac{F_S \cdot F_D}{\|F_S\|_2 \|F_D\|_2}, \tag{5.2}$$

where $F_S$ and $F_D$ are the extracted curve feature vectors of $I_S$ and $I_D$, respectively.

#### Step 3: Matched-Region Identification

To help the visual examination and confirmation by archaeologists, following the results in Step 2, we further identify the best matched region between the sherd $S$ and the design $D$. Let $I_S^*$ and $I_D^*$ be the best matched patches between $S$ and $D$. By corresponding the coordinate of the four corners of these two patches, we can estimate an isometric transformation $T$ from sherd $S$ to design $D$. We then overlay

the transformed sherd image $T(S)$ (the binary curve-pattern segmentation) to the design $D$ for identifying their maximal matched region by following a region-growing strategy.

More specifically, at each location, we can crop a $p \times p$ patch on both $T(S)$ and $D$ and then apply the algorithm in Step 2 to estimate their matching score. This way, we can obtain a heat map for the area shared by $T(S)$ and $D$, with matching scores as heat values. We then take the location with the largest heat value as seed of the matched region and check its neighboring locations – if their matching scores at a neighboring location is no less than 90% of the matching score at the seed location, they are included into the matched region. We then take each of the newly included locations as seed and repeat the above operation, until no more locations can be added.

Denote the matched region between $T(S)$ and $D$ by $R$. We finally take the average heat values in $R$ as the matching score $\phi(S, D)$ between $S$ and $D$:

$$\phi(S, D) = \frac{1}{|R|} \sum_{I_S, I_D \in R} \phi(I_S, I_D). \tag{5.3}$$

Given a set of $M$ sherds and a set of $N$ known designs, we can compute this matching score between each pair of them and construct a $M \times N$ similarity matrix. In our study, for each sherd $S$, we only return a small set of designs $D$ with $\phi(S, D) \geq t$, with $t$ being a pre-set threshold, together with their best matched regions, to archaeologists for confirmation, i.e., examining whether any of the provided designs is the true one stamped on the sherd.

If we have $\phi(S, D) < t$ for all known designs $D$, we mark $S$ as a sherd with unknown design, which means it may lead to new designs that have never been discovered. For such sherds, we cluster them based on their underlying curve patterns to facilitate the reconstruction of new designs, which will be studied in sherd identification.

## 5.3 Experiments

In this section, we conduct experiments to evaluate the performance of the developed method on design identification. Specifically, we first compare the developed method with representative existing methods, and then conduct ablation experiments on the developed method to provide in-depth analysis.

### 5.3.1 Dataset

For this study, we follow the digitization procedure described in Chapter 2.1 to digitize 1,604 pottery sherds from various archaeological sites located in southeastern North America. These 1,604 sherds correspond to 133 unique paddle designs. Each sherd has a curve pattern, either single or composite, from only one design. Their ground-truth designs are manually identified by archaeologists.

We take all $1,604$ sherds for the experiments – 604 for training and 1,000 for testing. Each sherd and design image in the training set are sampled with a patch size set to $300 \times 300$ pixels. In total, 600k triplets were sampled from the training set in the form of image patches. The curve feature descriptor is trained by Adam optimizer with a base learning rate of $10^{-3}$ and a momentum of 0.9. We train it for 50k iterations with a mini-batch size of 64. The margin value $\alpha$ in the triplet loss is set to 1. The sherd and design images in testing are sampled with the same patch size as it in training.

In this task, we use the Cumulative Matching Characteristics (CMC) ranking metric to evaluate the design identification performance, as described in Chapter 4.3. In testing, we calculate CMC values and ranks for all $1,000$ sherds on 133 matching designs, and draw a CMC curve in terms of CMC rank, as shown in Fig. 5.4, to evaluate the performance of the overall matching result.

### Comparison with Existing Methods

To demonstrate the effectiveness of the developed method on design identification, we compare it with nine representative existing image matching methods, including three low-level matching methods template matching [10], Chamfer matching [4], and Shape Context [6]; one fingerprint matching method [69]; three handcrafted-feature-based methods SIFT [46], SURF [5] and ORB [61]; and two learning-based methods LF-Net [53] and RF-Net [63].

The detailed settings of comparison methods are as follows: 1) For template matching, we take sherd images as the source and design images as the target. Besides searching along $x$, $y$ directions, we also search for the angle of rotation from $0^\circ$ to $359^\circ$. The strides along these three dimensions are set as 10. The sum-of-square-difference is employed as the matching cost. 2) For Chamfer matching, we perform image thinning on all sherd and design curve-pattern images to get one-pixel-wide skeleton images, and compute the minimum Chamfer matching cost using the distance transform between each pair of sherd and design. The searching strategy is similar to the one used for template matching. 3) For fingerprint matching, we adopt the minutia-feature-based algorithm implemented in [1] and treat the sherd images as query fingerprints and the design images as the fingerprint database. We perform image thinning to all sherd and design curves to make them 3-pixel-wide to reduce redundant minutia on curve ridges. Euclidean distance is computed as the matching cost by every minutia feature pair. By setting a threshold cost of 15 on each feature pair, we compute the number of matched features normalized over the number of features as the matching score of each sherd-design pair. 4) For Shape Context, we thin the curve structures of the sherd and design to one-pixel wide. To deal with partial matching, we use a sliding window technique to match a sherd curve pattern to each window-cropped design curve pattern and then choose the one with the lowest matching distance. The sliding-window size is the same as the sherd image. The

shape context algorithm implementation directly comes from the OpenCV package. The number of sampled points per sherd is set to be 100. 5) For SIFT, SURF and ORB, we follow the same identification method by taking the best matched design in a design database for a sherd. Specifically, we use their implementations in OpenCV. Keypoints are matched using Brute-Force Matcher for SIFT and SURF, and FLANN-based Matcher for ORB. Every feature in a sherd is compared to all the features in a design, and we compute the L2-norm distance of the feature pair for SIFT and SURF, and normalized Hamming distance for ORB. Then the matching score is computed as the number of best matched keypoint pairs for a sherd-design pair. In searching for the best matched keypoint pairs, the ratio of the best matched keypoint pair and their second best matched keypoint pair is set to 0.75. 6) Both LF-Net and RF-Net are learning-based methods. We use their released pretrained models as initialization and fine-tune them on the training set of the developed method. LF-Net is pretrained on the indoor dataset ScanNet [22], while RF-Net is pretrained on HPatches [3]. Both contain two networks, a detector and a descriptor. Training the detector requires the ground-truth camera pose and the calibration matrix, which are not available in our datasets. Therefore, we fix the pretrained detectors, and fine-tune the descriptors using their default hyperparameters but reduce the learning rate by 10 times. In their source code, they convert RGB images to grayscale images as input, so we do not need to change the number of channels of our images.

CMC curves of the developed method and all comparison methods are shown in Fig. 5.4. The detailed CMC Rank-1/5/10/20 values are reported in Table. 5.2. From Fig. 5.4 and Table. 5.2, we can see that: 1) The developed method achieves the best performance among all, and outperforms the second best method by 13.7% in terms of CMC Rank-1 accuracy. 2) All comparison methods perform poorly on this task, and several of them even have nearly diagonal CMC curves, which is equivalent to random guessing. It demonstrates the challenge of the curve-pattern matching prob-

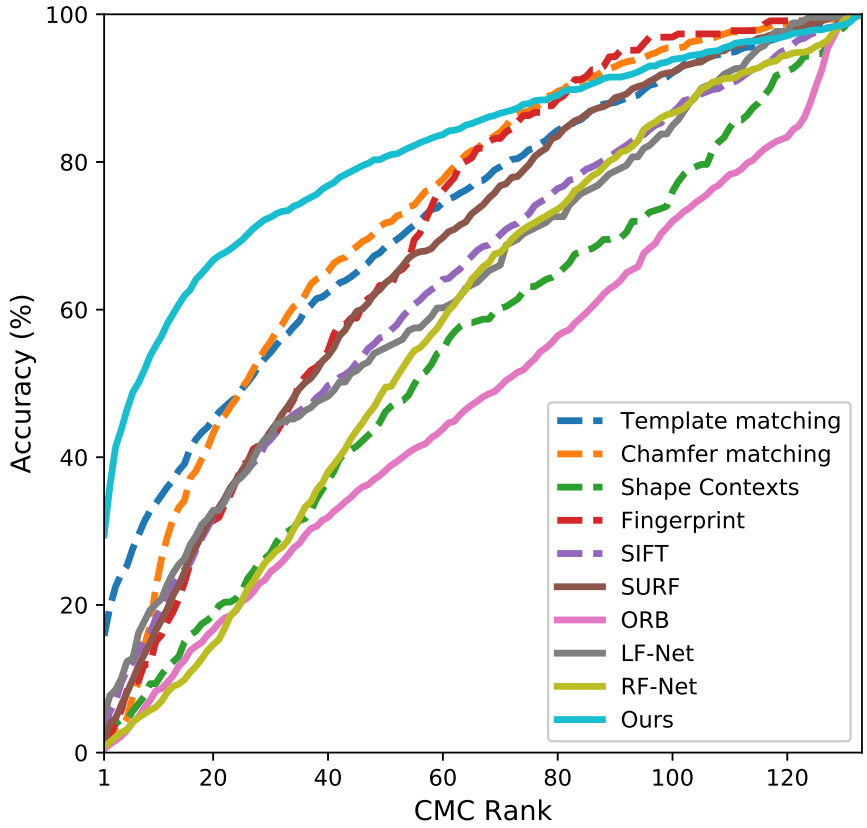



Figure 5.4 CMC curves of all the methods in design identification.

Table 5.2 CMC values (%) of all the methods in design identification.

| **Method** | **Rank-1** | **Rank-5** | **Rank-10** | **Rank-20** |
|---|---|---|---|---|
| Template matching [10] | 15.8 | 25.3 | 33.7 | 45.5 |
| Chamfer matching [4] | 0.5 | 4.7 | 17.1 | 43.2 |
| Shape Context [6] | 1.7 | 4.4 | 9.3 | 18.7 |
| Fingerprint matching [69] | 0.8 | 8.4 | 15.0 | 31.4 |
| SIFT [46] | 2.9 | 11.2 | 18.2 | 31.9 |
| SURF [5] | 1.6 | 6.4 | 16.5 | 32.2 |
| ORB [61] | 0.5 | 3.0 | 8.4 | 16.6 |
| LF-Net [53] | 8.5 | 16.0 | 22.6 | 33.8 |
| RF-Net [63] | 2.5 | 5.1 | 7.6 | 14.7 |
| **Ours** | **29.5** | **46.5** | **55.1** | **66.7** |

lem. 3) Template matching has the second best Rank-1 accuracy among all. After checking the matching results, we find that template matching works well on sherds with single patterns, but performs poorly on composite pattern matching; other low-level matching methods, such as Shape Context and Chamfer matching, are sensitive to noises and perform poorly in both cases. 4) Keypoint-based matching methods including handcrafted-feature-based and learning-based methods do not produce satisfactory results, because they cannot find proper keypoints in our low-texture binary curve-pattern images. By explicitly considering the possible composite patterns, as well as noise, errors, and deformation in curve-patterns images, the developed method achieves much better CMC performance than all other methods.

### Ablation Study

*Without patch sampling:*

Step 1 of the developed method is patch sampling on both sherd and design. As sherds may contain composite patterns, the major purpose of Step 1 is to address this challenge by cropping a patch with only a single pattern that can be matched to a portion of its underlying design. Here we try to directly match whole sherds to whole designs without patch sampling. Specifically, for each pair of sherd and design, we directly use the trained curve feature descriptor to obtain their global image features, and their matching score is computed based on the global features. The CMC values of the developed method with and without patch sampling are compared in Table 5.3. The performance degrades significantly after removing patch sampling. It indicates that curve-pattern matching is a partial-to-partial matching problem and cannot be solved using global features.

*Changing the patch size:*

In Step 1, we use a fixed patch size $p = 300$ to sample patches from sherd and design images. In this experiment, we try different patch sizes to see its effect to

Table 5.3 CMC values (%) of the developed curve-pattern matching method with and without patch sampling.

| | **Rank-1** | **Rank-5** | **Rank-10** | **Rank-20** |
|---|---|---|---|---|
| w/o patch | 15.8 | 25.3 | 33.7 | 45.5 |
| w/ patch | **29.5** | **46.5** | **55.1** | **66.7** |

Table 5.4 CMC values (%) of the developed curve-pattern matching method with different patch sizes.

| **Patch size** | **Rank-1** | **Rank-5** | **Rank-10** | **Rank-20** |
|---|---|---|---|---|
| 200 | 11.7 | 29.9 | 40.3 | 55.1 |
| 220 | 18.5 | 37.4 | 48.6 | 61.6 |
| 240 | 25.2 | 42.9 | 55.3 | 66.6 |
| 260 | 27.9 | 44.1 | 54.0 | 67.3 |
| 280 | 29.1 | 45.8 | 54.8 | **70.0** |
| 300 | **29.5** | **46.5** | 55.1 | 66.7 |
| 320 | 28.6 | 46.4 | **57.1** | 66.2 |
| 340 | 28.4 | 45.1 | 53.5 | 65.3 |
| 360 | 27.1 | 43.3 | 51.9 | 63.2 |
| 380 | 25.4 | 41.1 | 49.2 | 59.3 |
| 400 | 24.2 | 39.1 | 47.5 | 56.5 |

the matching performance. Specifically, we vary the patch size from 200 to 400 with a stride of 20, and the detailed CMC values are reported in Table 5.4. We can see that developed method achieves the best performance at $p = 300$, and the matching performance is quite stable for $p \in [260, 340]$ – CMC Rank-1 values vary from 27.9% to 29.5%. In practice, overly small patches could not carry sufficiently discriminative information for matching, while overly large patches may contain composite patterns or non-curve regions. Both cases may prevent the correct matching between a sherd and its underlying design.

*Using different feature descriptors:*

Although the sherd and design patches are binary images with sparse texture, measuring their curve pattern similarity is non-trivial. To validate the discriminability of the curve feature descriptor trained in Step 2, we try to replace it with other

commonly used similarity metrics/feature descriptors to evaluate curve pattern similarity. In specific, we compare with representative methods, including two pixel-level similarity metrics Sum-of-Square-Difference (SSD) and Normalized Cross Correlation (NCC); three handcrafted feature descriptors SIFT [46], HOG [23], and LBP [52]; and two learning-based feature descriptors HardNet [16] and SKAR [51]. For SIFT, we adopt the implementation in OpenCV, but only compute the feature at the patch center with a neighborhood size of 300 to get the patch features. For HOG, we divide each patch into $8 \times 8$ cells and use 16 bins. For LBP, we adopt the implementation of scikit-image [70]. The radius of circle is 150, and the number of points is set to be 8 times of the radius. For HardNet and SKAR, we use their released models trained on HPatches [3]. The CMC curves of using all the methods are shown in Fig. 5.5, and the detailed CMC Rank-1/5/10/20 values are reported in Table 5.5.

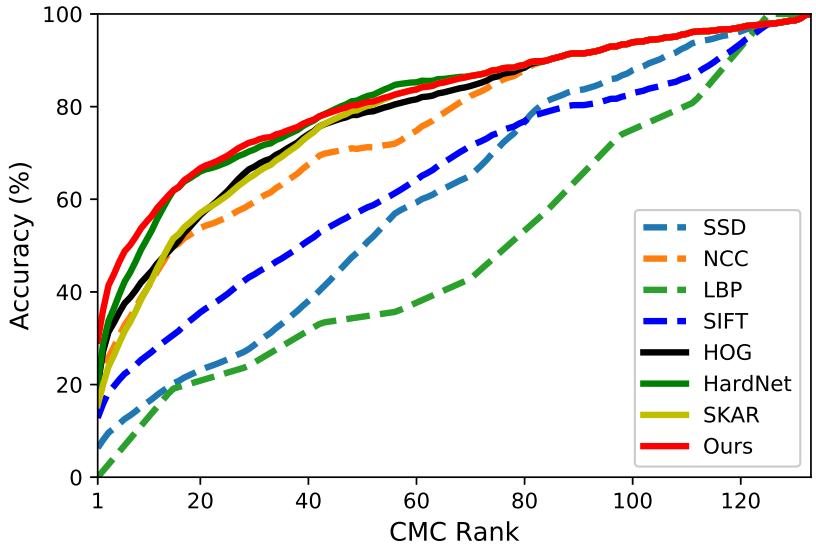



Figure 5.5 CMC curves of using different feature descriptors in the developed curve-pattern matching method for design identification.

From Fig. 5.5 and Table 5.5, we can see that our curve feature descriptor significantly outperforms other alternative methods, especially on the CMC Rank-1 accuracy. Pixel-level similarity metrics and handcrafted features are not robust to

Table 5.5 CMC values (%) of the developed curve-pattern matching method with different feature descriptors.

| **Descriptor** | **Rank-1** | **Rank-5** | **Rank-10** | **Rank-20** |
|---|---|---|---|---|
| SSD | 6.3 | 11.5 | 16.0 | 23.1 |
| NCC | 17.4 | 30.6 | 40.5 | 53.7 |
| LBP [52] | 0.0 | 5.5 | 12.4 | 20.8 |
| SIFT [46] | 12.8 | 21.0 | 26.1 | 35.6 |
| HOG [23] | 19.8 | 35.5 | 43.2 | 56.5 |
| HardNet [16] | 20.0 | 39.4 | 51.1 | 66.0 |
| SKAR [51] | 15.6 | 28.9 | 33.7 | 57.0 |
| **Ours** | **29.5** | **46.5** | **55.1** | **66.7** |

noises and the curve deformations between sherd and design images, so they have inferior performances on this task. Learning-based descriptors have better performance, but they are not specifically trained for curve-pattern matching. With the carefully designed training strategy, our curve feature descriptor is more sensitive to curve structure difference and more robust to minor translation and rotation perturbation.

*Changing the similarity metric in Step 2:*

In design identification and sherd identification, we use two different metrics to measure the similarity of curve patterns – cosine similarity for the former and the metric network $\mathcal{M}$ for the latter. In this experiment, we replace the cosine similarity with the metric network $\mathcal{M}$. After changing the metric, the performance of the developed method drops slightly. The CMC Rank-1/5/10/20 values decrease from 29.5%, 46.5%, 55.1% and 66.7% to 28.9%, 46.0%, 54.7% and 66.6%, respectively. It indicates that, although simple, the cosine similarity is more suitable for design identification.

We use different metrics in design identification and sherd identification as they have different goals. In design identification, we match each sherd to different designs, and the similarity scores incurred by different sherds can be non-comparable. In sherd identification, the similarity scores incurred by different sherds must be comparable – they are embedded into a single fully-connected graph for clustering. Therefore, we

simply use the cosine similarity for ranking in design identification, but train a more comprehensive metric network for thresholding in sherd identification.

## 5.4 Chapter Summary

In this chapter, we studied partial curve pattern matching problem in terms of noisy binary images. We developed a patch-based matching method and applied to design identification problem. Specifically, aiming to locate the most similar regions between the sherd and the considered full design, we applied uniform sampling for constructing patches and employed a learning-based curve feature descriptor to derive a heatmap for the local similarity between the sherd and the design. We considered both the heatmap value and the overlap area in deciding the matching between sherd and designs. With this heatmap, we located the best matching portions by region growing and defined a new matching cost considering overall similarity of these portions. In the experiment, we validated the developed method by using 1604 real sherds together with their corresponding 133 designs from the Woodland Period in southeastern North America. Comparison to nine existing matching methods verified that the developed method can achieve a new state-of-the-art performance. The work of this chapter is included in submission to the International Journal of Computer Vision [48].

# Chapter 6

# Sherd Identification

## 6.1 Motivation

Archaeologists have manually constructed hundreds of complete designs from fragmented pottery sherds in the past decades. For a thousand-year pottery stamping tradition, there are still a huge number of designs undiscovered or unconstructed. In this task, we perform sherd identification. We take a group of unidentified sherds and perform clustering, expecting each cluster to contain the same underlying design. The clustering result is then presented to archaeologists for new design discovery and reconstruction.

Sherd clustering differs from conventional image clustering in two aspects. On one hand, two sherds should be assigned to the same cluster as long as part of them share the same curve pattern, and their remaining parts can be different. On the other hand, we do not require every pair of sherds in each cluster to have the same curve pattern but allow them to be linked through other sherds. For example, if sherd A and sherd B do not show any partial overlap. However, both of them can be matched to sherd C with a certain overlap, then A and B are still considered to have the same underlying design. To address this problem, we formulate sherd clustering as a graph partitioning process presented in the following.

## 6.2 Method

In general, our developed method can be divided into three steps: 1) constructing a sherd graph weighted by sherd-to-sherd matching scores, 2) partitioning the graph into subgraphs/clusters by adaptive thresholding, and 3) sherd stitching for iterative clustering. An illustration using the developed method to cluster four input sherds is shown in Figure 6.1.

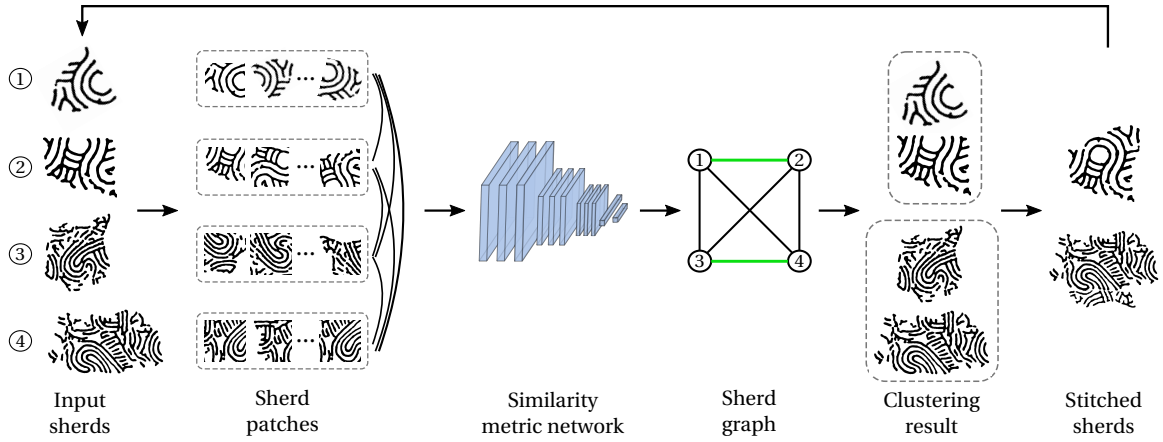



Figure 6.1 The pipeline of the developed sherd clustering method with four sample input sherds.

### Step 1: Constructing Sherd Graph

Given a set of $N$ input sherds, in the form of their segmented binary curve pattern images, $\mathbf{S} = \{S_1, S_2, \cdots, S_N\}$, we first construct a fully-connected undirected graph $G = \langle \mathbf{S}, W \rangle$, where $W$ is the edge weight matrix that measures the linkage of each pair of sherds. For each pair of sherds, their edge weight is determined by the similarity of the best matched parts between them. To find out their best matched parts, we follow the patch sampling strategy developed in Section 5.2 to sample a set of patches of size $p \times p$ from each sherd.

The cosine similarity of the extracted features was employed in design identification as the similarity metric to rank design patches. However, this metric is not sufficiently discriminative for sherd identification because the cosine similarities of positive and negative patch pairs can be very close, which is not desired in threshold-based graph partitioning. Therefore, we train a curve similarity metric network $\mathcal{M}$ to directly predict the similarity of two sherd patches. Specifically, on top of the curve feature descriptor described in Table 5.1, we append a global average pooling (GAP) layer, two fully-connected (FC) layers, and a softmax layer. The output channels of two FC layers are 128 and 2, respectively.

**Algorithm 2** Sherd graph partitioning by adaptive thresholding

**Input:** Sherd set $\mathbf{S}$; Edge weight matrix $W$; Maximum cluster size $t_c$
**Output:** Sherd clusters $\mathbf{C}$

Initialize the weight threshold $t_w = \min(W)$, the queue of undivided clusters $\mathbf{C}' = \{\mathbf{S}\}$.
**while** $\mathbf{C}'$ is not empty **do**
  Let $C$ be the first element of $\mathbf{C}'$ and pop it out
  Update the weight threshold by $t_w \leftarrow t_w + 0.1 * (1 - t_w)$
  Partition $C$ to a set of clusters $\hat{\mathbf{C}}$ using the current weight threshold $t_w$
  **for** each cluster $C_i \in \hat{\mathbf{C}}$ **do**
    **if** $|C_i| \leq t_c$ **then**
      Append $C_i$ to $\mathbf{C}$
    **else**
      Append $C_i$ to $\mathbf{C}'$
    **end if**
  **end for**
**end while**

The training of $\mathcal{M}$ takes a pair of sherd patches and a 0/1 label that indicates whether they are matched or not as input. We minimize the cross-entropy loss to optimize the parameters of two FC layers, while the curve feature descriptor is fixed. In testing, the edge weight $W_{i,j}$ of two sherds $S_i$ and $S_j$ is the highest similarity among all patch pairs sampled in these two sherds:

$$W_{i,j} = \max_{I_{S_i} \in S_i} \max_{I_{S_j} \in S_j} \mathcal{M}(I_{S_i}, I_{S_j}). \tag{6.1}$$

Step 2: Sherd Graph Partitioning

After obtaining the weight matrix $W$, we partition the fully-connected graph $G$ into sherd clusters. Intuitively, we can cut all edges below a pre-set similarity threshold and take the remaining subgraphs as clustering results. However, such clustering result is very sensitive to the pre-set threshold. To solve this problem, we adopt an adaptive thresholding strategy [77] to partition the graph, as shown in Algorithm 2.

We first set the weight threshold to the lowest value of $W$, and maintain a cluster queue initialized by $\mathbf{S}$. Next, we iteratively process each cluster in the queue in the

order of Breadth First Search (BFS). The main idea is to adjust the weight threshold to limit the size of output clusters to a size threshold of $t_c$. In each iteration, we gradually increase the weight threshold, and partition the current cluster into sub-clusters. The cluster partitioning is implemented by cutting low-weight edges and grouping remaining connected sherds by Union-Find. This process is repeated until the cluster queue is empty.

#### Step 3: Sherd Stitching for Iterative Clustering

We finally propose a strategy to iteratively refine the clustering result by introducing the stitched curve patterns. As a side benefit of the patch-based matching in Step 1, we can find the transformation between the two sherds by aligning their best-matched patches. This way, we can transform the curve patterns of all sherds in the same cluster into a common coordinate system and stitch them together to obtain a larger curve-pattern image, as shown in Fig. 6.1, by following a similar process described in Section 5.2. Specifically, in each cluster produced by the previous step, we stitch sherd pairs in the order of their edge weights – from high to low. When stitching two sherds, we overlay the larger one on the top of the smaller one. The stitched curve patterns on individual sherds bear more geometrical information of the underlying design than curve patterns on individual sherds. We put them back as new graph nodes for clustering, and they can help recall sherds that were not clustered or not correctly clustered in the initial clustering. This iterative clustering is repeated until there is no more change to the clustering result.

### 6.3 Experiments

In this section, we construct a dataset of 1,091 sherd curve pattern images from 129 clusters (designs) in the experiment of sherd identification. The training set includes 510 sherds from 60 clusters. To train the similarity metric network, we sample 25k

Table 6.1 Purity, ARI and NMI of all the methods in sherd identification.

| **Method** | **Purity** | **ARI** | **NMI** |
|---|---|---|---|
| K-Means [44] | 0.313 | 0.016 | 0.174 |
| K-Means++ [2] | 0.344 | 0.017 | 0.196 |
| Spectral Clustering [64] | 0.23 | 0.013 | 0.143 |
| MeanShift [20] | 0.165 | 0.001 | 0.041 |
| AHC [38] | 0.27 | 0.028 | 0.18 |
| Affinity Propagation [25] | 0.3 | 0.011 | 0.158 |
| ARO [54] | 0.26 | 0.021 | 0.108 |
| GCN Clustering [71] | 0.353 | 0.017 | 0.198 |
| **Ours** | **1** | **0.148** | **0.644** |

positive patch pairs of size $300 \times 300$ from sherds with overlapped regions in the same cluster. We also sample 25k negative patch pairs by performing template matching on sherds randomly selected from different clusters to mine hard negative pairs. The similarity metric network is trained by Adam optimizer for 20k iterations with a base learning rate of $10^{-3}$, a momentum of 0.9, and a batch size of 32. The threshold on cluster size $t_c$ is set to 20.

In testing, 591 sherds from 69 clusters are used to evaluate the performance of the developed sherd clustering method and existing image clustering methods. Three commonly used metrics are employed to evaluate the clustering results: 1) Purity: The fraction of samples that belong to the same cluster; 2) Adjusted Rand Index (ARI): The fraction of sample pairs that are correctly assigned in the same or different clusters; and 3) Normalized Mutual Information (NMI): The mutual information between the ground-truth clusters and the predicted clusters normalized by the sum of their entropy. For all three metrics, a higher value indicates better performance.

### Comparison with Existing Methods

We compare the developed method with eight representative existing clustering methods, including six traditional methods K-Means [44], K-Means++ [2], Spectral Clustering [64], MeanShift [20], Agglomerative Hierarchical Clustering (AHC) [38], Affin-

ity Propagation [25]; and two deep-learning-based methods Approximate Rank-Order Clustering (ARO) [54] and GCN Clustering [71]. For the comparison methods in sherd identification, we use the curve-pattern descriptor trained in design identification to extract a 128-dimension vector for each sherd and then perform clustering of these feature vectors. We tune their parameters and settings on the training set to maximize the Normalized Mutual Information (NMI). Except for ARO and GCN clustering, all other methods adopt the implementations in scikit-learn.

The detailed experimental settings of comparison methods are as follows: 1) For K-Means and K-Means++, we provide the ground-truth number of clusters in the test images as a favor for them, and set the maximum number of iterations to be 300. 2) For spectral clustering, we adopt the discretized label assignment strategy and construct the affinity matrix by computing a graph of nearest neighbors. The ground-truth number of clusters is also provided as a favor. 3) For MeanShift, the bandwidth used in the RBF kernel is estimated by a built-in function with a quantile of 0.3 and the maximum number of iterations is set to be 300. 4) For AHC, we construct a connectivity matrix for initialization using k-nearest neighbors (k-NN) and set the linkage criterion to minimize the variance of the clusters being merged. The linkage distance threshold is set to be 0.7. 5) For affinity propagation, the damping factor is set to be 0.5, and the number of iterations is set to be 200. 6) For ARO, we adopt the implementation in [77]. It directly takes the feature vectors as input and does not require fine-tuning. We set the number of considered nearest neighbors to be 30 and the clustering distance threshold to be 2. 7) For GCN clustering, we use our own feature vectors and train the clustering model from scratch using the default hyperparameters, except that the numbers of 1-hop and 2-hop nodes are changed to 20 and 5, respectively.

From Table 6.1, we can see that our clustering method achieves the best performance on all three metrics. All the comparison methods produce a poor performance

on this task, even if the ground-truth cluster number is given. This demonstrates both the challenge of this task and the effectiveness of the developed method. We believe that their failure is mainly due to that they only leverage the global image features of the sherds (curve pattern images) in clustering, but cannot handle the special partial-to-partial matching in this particular problem. We also notice that the developed method achieves 100% purity while producing many singleton clusters with only one sherd. We choose high precision in the trade-off between precision and recall. Archaeologists prefer to work on small and pure clusters rather than large clusters with sherds from many different designs. In practice, we return only clusters with more than 2 sherds for design reconstruction.

Using the Hungarian algorithm, we further examine our clustering results by mapping each ground-truth cluster to a predicted cluster. After removing predicted clusters with only one sherd, we find 61 predicted clusters correspond to 41 unique designs. By comparing each ground-truth cluster with the corresponding predicted cluster, we can count the number of found sherds (true positive), missed sherds (false negative), and wrong sherds (false positive) for each design. Averaged over all 41 designs, the developed method can find 41.3% sherds, miss 58.7% sherds, and introduces 0 wrong sherds. The developed method works better on sherds with large and clear curve patterns. Although many sherds are missed, we return the most informative sherds to archaeologists to reconstruct the full design. In Fig. 6.2, we display the sample clustering results of three designs.

### Ablation Study

*Using different similarity metrics:*

In the first step of the developed method, we train a curve similarity metric network $\mathcal{M}$ to obtain edge weights of the sherd graph. We replace it by directly computing the cosine similarity of feature vectors extracted by the curve feature

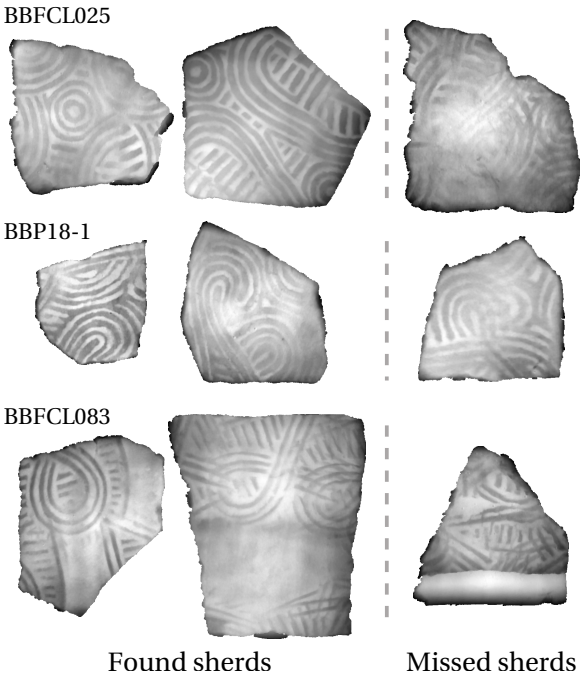



Figure 6.2 Examples of the clustering result of the developed method. For each design (BBFCL025, BBP18-1, or BBFCL083), we show two found sherds and one missed sherd.

descriptor to justify its effectiveness. With this modification, we find that the edge weights of both positive and negative sherd pairs mostly range from 0.8 to 1, too close for the adaptive thresholding in Step 2. Accordingly, the developed method's purity, ARI, and NMI decreased to 0.535, 0.125, and 0.499, respectively.

We also use the angle difference to measure curve-pattern similarities by applying the arccos function to the cosine similarity – arccos similarity varies in a larger range than cosine similarity. With this metric, the resulting purity, ARI, and NMI turn to 0.543, 0.128, and 0.502, respectively. We can see that, although it performs slightly better than the cosine similarity, it is still not comparable with the metric network, of which the resulting purity, ARI and NMI are 1, 0.148 and 0,644, respectively. Therefore, simply enlarging the varying range of similarity cannot solve this problem. It is beneficial to use the similarity metric network to improve the discriminability.

*Changing the graph partitioning strategy:*

In Step 2, we partition the fully-connected sherd graph using an adaptive thresholding strategy. A simple alternative approach uses a pre-set threshold to cut low-weight edges to partition the graph. In this experiment, we compare the performance of using two different thresholding strategies in the developed method. Specifically, we evaluate the performance using a pre-set threshold, which varies from 0.1 to 0.9 with a step of 0.1. The resulting curves of purity, ARI, and NMI are shown in Fig. 6.3. We can see that when the threshold is 0.5, the developed method leads to the best performance, resulting in the purity, ARI, and NMI of 1, 0.156, and 0.643, respectively, which are close to the performance of adaptive thresholding. But the finding of this optimal threshold requires careful tuning on the test set. In contrast, this is not required by adaptive thresholding. Therefore we choose to use the adaptive thresholding strategy in our method.

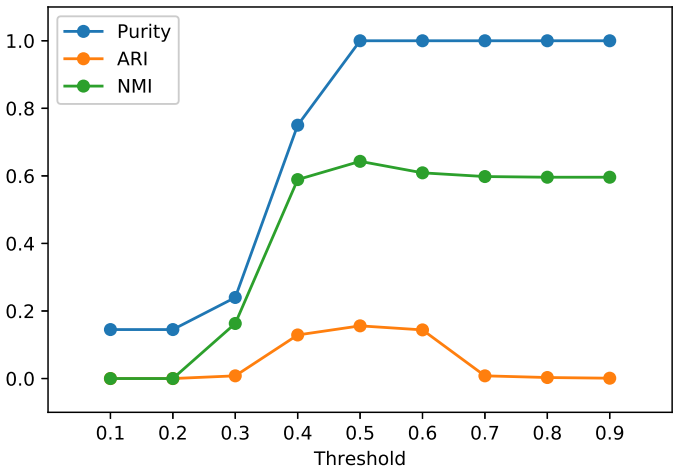



Figure 6.3 Purity, ARI, and NMI by using different thresholds in the pre-set thresholding strategy.

*Removing Step 3:*

After obtaining the initial clustering result, an iterative strategy was adopted to refine the developed method by stitching curve patterns in the same cluster. To justify

its usefulness, we skip it and report the performance of the initial clustering result, of which the purity, ARI, and NMI are 1.0, 0.12, and 0.631, respectively. Comparing the results before and after the iterative refinement, we find that 11 isolated sherds in the initial result are assigned to their correct clusters in the second iteration without introducing additional false positives.

## 6.4 Chapter Summary

In this chapter, we studied sherd identification and formulated it as a sherd curve pattern clustering problem. We developed a new clustering algorithm to identify and group sherds with the same design. This problem is special for curve patterns in a cluster that only partially overlap or do not overlap. We conducted patch-based pairwise matching between each given pair of sherds to construct a similarity matrix. A fully-connected graph was constructed on the matrix and a graph partition with adaptive thresholding was applied to find clusters. We then developed an iterative cluster refining strategy, with curve-pattern stitching in the iteration, for identifying and refining the sherd clustering. We conducted experiments on 1,091 sherds from 129 designs. A comparison of eight existing matching methods verified the effectiveness of the developed algorithm. The work of this chapter is included in the submission to the International Journal of Computer Vision [48].

# Chapter 7

# Snowvision

Some 1,820 archaeological sites with Swift Creek pottery have been recorded across the states of Alabama, Florida, Georgia, and South Carolina [65]. Hundreds of thousands of pottery sherds from these sites reside in curation facilities and archaeological laboratories across the Southeast. Making sense of these large and scattered collections is a daunting task for researchers. The Snowvision project takes on this challenge with state-of-the-art research in computer vision and archaeology. At its core, Snowvision is a backend study of these heritage objects using the identification and segmentation of curve patterns on the pottery fragments. This involves the digital acquisition of pottery fragments and reconstructed paddle designs. These cultural heritage objects are stored for the free public World Engraved digital archive to centralize and share collective knowledge about the diversity and geographical reach of the designs over time.

This chapter introduces the digitization pipeline of Snowvision and the architecture of World Engraved, featuring a science gateway with the Extreme Science and Engineering Discovery Environment (XSEDE) where scientists can interactively share computing resources, data, and expertise in an academic cloud system. The project has digitized more than 8,000 sherds using this Snowvision digitization pipeline. It is expected to increase this number to at least 25,000 as the project grows. More than 800 paddle design reconstructions have been digitized, and 150 have been scaled by calibrating them to the associated pottery sherd depth scans. Detailed metadata about the sherds and design reconstructions are collected and published with the objects.

## 7.1 Digitization Pipeline

Common methods of digitizing cultural heritage objects include RGB photography, depth imaging, point cloud, 3D mesh, etc. All known paddle design reconstructions were manually drawn by archaeologists, primarily Bettye J. Broyles and Frankie Snow.

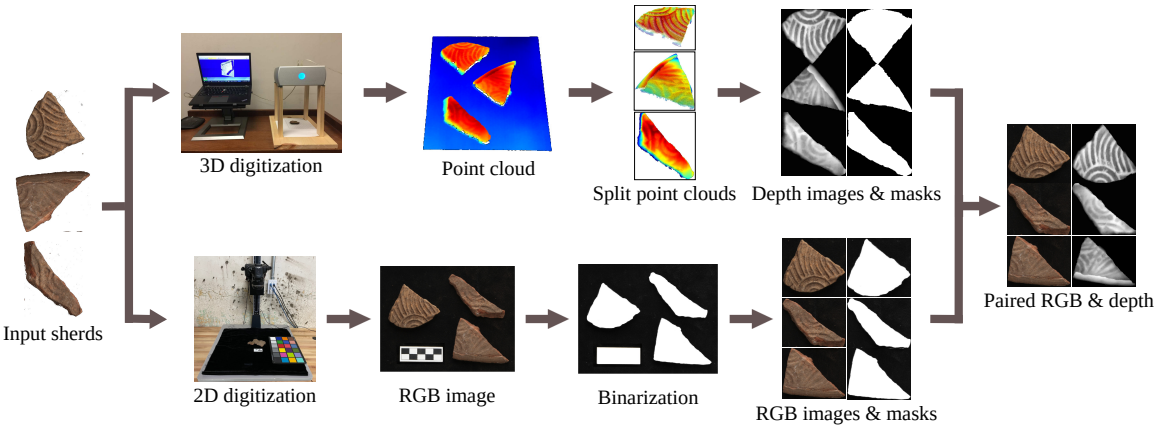



Figure 7.1 The Snowvision digitization pipeline.

The Snowvision digitization pipeline uses a digital camera and a high-precision 3D scanner to collect high-quality RGB images and depth scans of the surface of pottery sherds. The RGB images are easier for the human eye to read than the depth scans. Still, color differences in the RGB images due to lighting, pottery manufacture, and depositional conditions make them unsuitable for curve extraction. Therefore, RGB photographs of the pottery sherds are collected for display in World Engraved but not used in the backend computer-vision tasks. Since the carved paddles impress ridges and valleys on the surface of a vessel and depth imaging records 3-dimensional data from the pottery sherds, a depth scan captures all the information required for analyzing curve patterns.

The depth difference between the ridge and valley on the surface of sherds is quite small, typically in the range of $1mm$ to $3mm$. The NextEngine Ultra HD, a linear array laser scanner with of accuracy of $\pm 0.127mm$, is used to capture this low-level depth difference. A dense point cloud of the sherd surface is captured and then converted into a depth scan. The sherd digitization process is depicted in Fig. 7.1 (up). The scanner is placed on a four-leg stand face-down to capture sherds placed below on a flat surface, with the impressed exterior sherd surface facing up towards the scanner. Multiple sherds can be scanned at one time for efficiency, as long as they

are not too close to appear whole in the scanning area. An example of three sherds scanned together is shown in Fig. 7.1.

The 3D scanning process captures a point cloud with a density of $6,200$ points per $cm^2$, and each scan takes approximately 5 minutes. The points corresponding to the surface below the sherds are removed using the 3D coordinates. The KD-tree is employed to obtain the point cloud of each sherd when multiple sherds are scanned at one time. In KD-tree, points are grouped for each sherd using a threshold of $1mm$ for the spatial distance. From the point cloud of each sherd, a depth image is generated by grid sampling with a resolution of $10,000$ pixels per $cm^2$, and the values of the depth image are normalized to $[0, 255]$. A mask image is also constructed to identify the irregular shape of the sherd. Finally, as a fragment of a rounded vessel, the sherd surface is curved instead of planar. An adaptive histogram equalization method CLAHE [56] is applied to enhance the contrast between sherd regions so that the vessel curvature can be removed from the depth scan.

RGB photographs taken with controlled lighting and scale cards are standard in archaeological data collection and dissemination. Information about the color and size of an artifact can be communicated through digital imagery. A digital SLR camera with a 60mm lens captures RGB images. Like the 3D scanning process, multiple sherds are placed on a flat surface with the camera above and face-down and captured in one RGB image to expedite the digitization process. A sherd splitting algorithm was developed to divide an RGB image with multiple sherds into multiple RGB images, each with a single sherd, and match such single-sherd RGB images to their corresponding depth scans. Specifically, as illustrated in Fig. 7.1 (bottom), given a multi-sherd RGB image, we binarize it and retrieve the boundaries of sherds by intensity thresholding. We then conduct shape matching between the sherd boundaries extracted from RGB images and the sherd masks extracted from depth scans using Hu-Moment values [82]. This pre-processing procedure can pair the RGB and depth

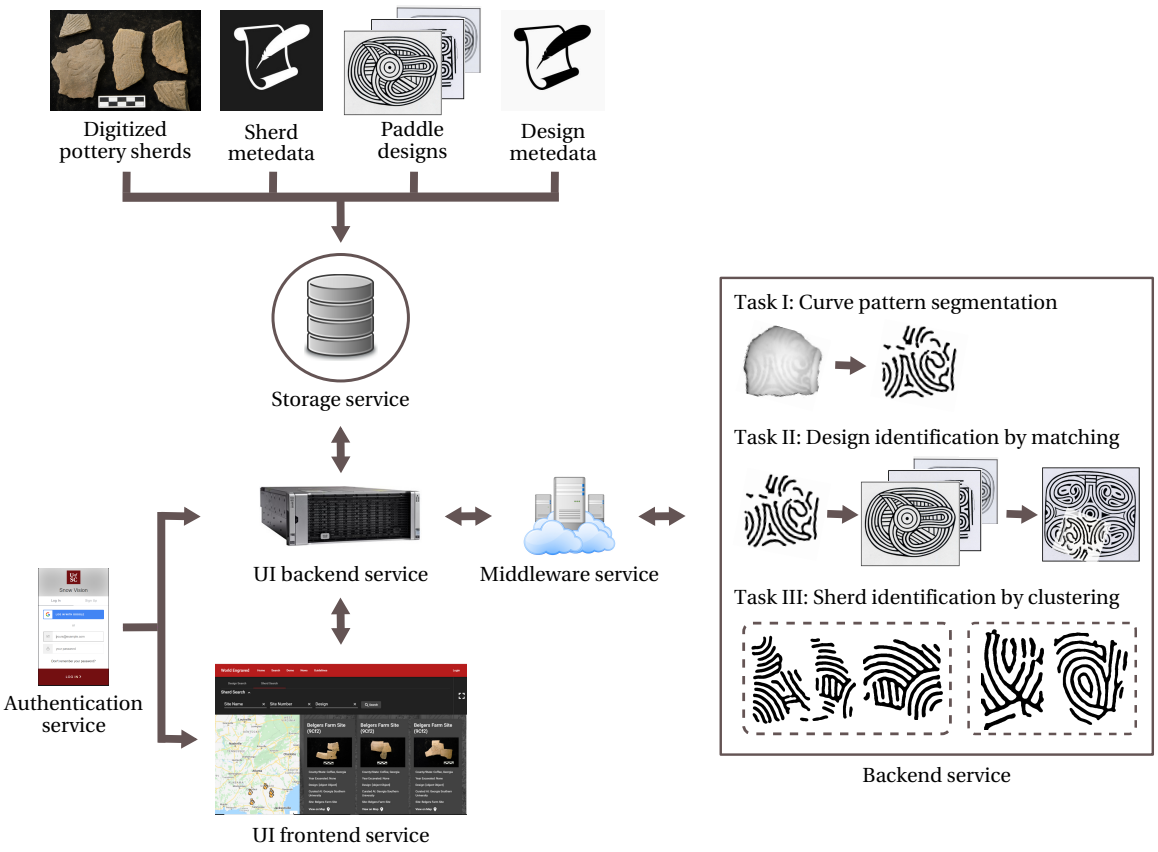



Figure 7.2 The architecture of the Snowvision system. (Original designs reproduced with permission, courtesy of Frankie Snow, South Georgia State College.)

images of the same sherds in the Snowvision database.

## 7.2 Science Gateway Framework

Built as a science gateway with XSEDE, our Snowvision website consists of six modules: a UI frontend service, a UI backend service, a storage service, an authentication service, a middleware service and a backend service. Its architecture is illustrated in Fig. 7.2.

World Engraved allows researchers to submit sherds for Snowvision processing and provides an interface for users to view and search for designs and sherds and the segmentation and identification results. Anyone with an internet connection can view

sherd and design data published on World Engraved. Still, only users with registered accounts can submit sherd data to the database. Snowvision and World Engraved are provided as a free service to encourage open data sharing. The World Engraved UI frontend service was implemented using Angular[1], a single-page app made for dynamic web applications. It offers a high level of abstraction, eliminating the need for redundant code, hence making the creation of dynamic web applications much simpler versus plain HTML and CSS. Moreover, it can keep the interface separate from the UI backend service, allowing for a better workflow during development. Each module can be worked on and changed separately without affecting the other.

The Snowvision UI backend service was developed using the Java Play Framework[2], an architectural pattern that helps create lightweight and highly scalable web applications. By handling every request asynchronously and using short, non-blocking operations, Java Play can quickly scale with a small, fixed thread pool. The feature allows multiple users to make the non-blocking request simultaneously without being forced to wait for each other's operations to complete. This proves to be extremely scalable. As the user base for a site grows, the potential slowdown from repeated requests is lessened.

The Snowvision storage service sits behind the UI backend service. All digital objects and metadata are stored in this service and communicated through the UI backend service. Auth0[3], an easy-to-implement user authentication service, is employed to allow users to create accounts and keep track of user data. Auth0 uses cookies and tokens to authenticate a user. Cookies are stored locally, which removes the risk of overloading the web server. Tokens are assigned by an external remote authentication server. It provides an authentication API that allows users to manage

[1]https://angular.io/

[2]https://www.playframework.com/

[3]https://auth0.com/

all aspects of user identity, such as login, sign up, log out, access APIs and more. It also supports various identity protocols, like OpenID Connect, Auth 2.0, and SAML. Using Auth0 ensures that security and authentication are handled properly without the hassle of implementing and maintaining a reliable, secure system.

The Snowvision backend service is the core of our project. Through this service, users submit unidentified sherds using a sherd submission template directly through World Engraved. This template allows for the robust collection of standardized metadata. It includes a schema to assist users in completing the template. RGB images are submitted as JPEG files, and 3D scans are submitted as XYZ point cloud files. The point clouds will be first converted to depth maps and separated into multiple individual sherds. RGB images are separated as discussed above. Users can interact with individual sherds to edit, publish, and process sherds for curve structure segmentation, design identification, and sherd identification, result reviewing, or entry deletion. The images and metadata are added to the World Engraved public archive when sherds are published.

In curve structure segmentation, we apply an FCN-based segmentation method to segment the curve patterns from the depth maps into binary images. The resulting binary curve pattern images are fed to design identification and sherd identification. Archaeologists can take the identification results produced by these two tasks for design study, including design identification and sherd identification.

We develop a Chamfer-matching based method and a patch-based matching method in design identification. With the Chamfer-matching based method, matching a sherd to a design takes 29.9 minutes, averaging over 100 sherds with 20 designs. In contrast, the patch-based matching method takes only 0.823 seconds, averaging over 1,000 sherds with 133 designs. This computation time was recorded in experiments conducted on a DELL PowerEdge C4140 server with dual Intel Xeon Platinum 8260 CPUs, 192GB memory, and an nVidia Tesla V100 GPU card. Furthermore, we com-

pare the identification accuracy of the two methods using precision/recall by considering the design identification as a multi-class classification, each class represents a design. We take the top one matched design as the prediction of a sherd. The average precision and recall we obtained for the Chamfer-matching based method are 18.84% and 31.99%, respectively. The average precision and recall for the patch-based matching method are 29.22% and 29.98%, respectively. We do not use CMC curves because their values are determined by the number of designs, which are different in the testing datasets for these two methods. We can see that the patch-based matching method outperforms the Chamfer-matching based method in both computation time and identification accuracy. It has much higher precision than the Chamfer-matching based method. Therefore, with Snowvision, we apply the patch-based matching method in this task.

The backend service is run remotely on a high-performance GPU cluster to offload intensive computation from the web server. The analysis requests from the users are submitted to a middleware service implemented by Apache Airavata[4]. This service can compose, manage, execute, and monitor the jobs submitted remotely on the GPU cluster and data transfer in-between.

## 7.3 Chapter Summary

This chapter described the Snowvision project, an interdisciplinary effort to leverage state-of-the-art computer vision techniques to explore and study paddle designs from large collections of pottery sherds. Standards and guidelines are provided for digitizing sherds and developing rich and accurate metadata. Building into a science gateway, a front-end website from which archaeologists at different institutions can submit and share their sherds, and a set of backend advanced computer vision tasks, including curve structure segmentation, design identification and sherd identi-

[4]https://airavata.apache.org/

fication, can assist archaeologists to more efficiently explore the underlying designs of the sherds. The work of this chapter has been published in Practice and Experience in Advanced Research Computing [79, 80].

# Chapter 8

# Conclusion

In this research, we studied the problems of design identification and sherd identification on heritage pottery fragments. We developed new computer vision algorithms to address them respectively. With the algorithms, we developed a computer-aided system, Snowvision, to facilitate archaeologists to study and explore pottery heritage fragments in southeastern North America.

Design identification aims to find the full design that is partially decorated on the surface of a heritage object, such as pottery sherds found in southeastern North America. Due to degradation, and multiple stamping tradition, curve pattern designs extracted on an object can be very noisy or contain a composite pattern. We develop two curve pattern matching methods to address these challenges and apply them to design identification.

First, we develop a new partial-to-global curve-pattern matching algorithm to identify the designs from pottery sherds with possible composite patterns. Unlike previous partial matching problems, we extend the classical Chamfer matching to identify candidate components of the sherd pattern and then leverage two metrics of completeness and disjointness to find the optimal sherd-pattern decomposition. Experiment results show that the CMC performance of the developed method is substantially better than several classical curve-pattern image matching algorithms.

Second, we develop a new patch-based curve pattern matching method to address the problem of partial matching in terms of noisy binary images. Specifically, to locate the most similar regions between the sherd and the full design, we first apply uniform sampling to construct patches and then employ a learning-based curve feature descriptor to derive a heatmap for the local similarity between the sherd and the design. We consider both the heatmap value and the overlap area in deciding the matching between sherd and designs. With this heatmap, we locate the best matching portions by region growing and define a new matching cost considering the overall similarity of these portions. Comparison to existing matching methods verified that

the developed method can achieve a new state-of-the-art performance.

Sherd identification aims to find sherds for unknown design reconstruction. Formulating this problem as clustering, we develop a new clustering algorithm to identify and group sherds with the same but unknown design. Given the segmented curve-pattern images of a collection of sherds, we first conduct patch-based pairwise matching between each pair of sherds to construct a similarity matrix. The pairwise matching is based on the best-matched patches between the two sherds to handle possible composite patterns. A fully-connected graph is constructed on the matrix and a graph partition with adaptive thresholding is applied to find clusters. We then develop an iterative cluster refining strategy, with curve-pattern stitching in the iteration, for identifying and refining the sherd clustering.

Furthermore, we present Snowvision, a multidisciplinary project to digitize, preserve, segment, identify and discover stamped curve patterns on unearthed pottery sherds. A research pipeline was developed for different users to study and share pottery sherds. Computer vision tasks, including design identification and sherd identification, frame the core of Snowvision.

## 8.1 Future Work

For future work, we would extend sherd identification to unknown design reconstruction. Taking a group of sherds from a same design, we stitch these sherd curve patterns to construct a larger piece that covers more design area. Constructing a full design may not be possible if missing parts are not presented in the existing group of sherds. Inspired by recent works on natural image completion [75, 78, 36], we formulate this design reconstruction problem as design completion, learning features from existing constructed designs and generating missing regions for archaeologists for further assessment. Generative adversarial networks (GANs) have been used in these methods.

However, for design completion, many problems persist. First, our constructed design database is small, about eight hundred. Natural image completion usually needs thousands of images in training. Simply applying these methods would result in network overfitting. Fortunately, designs usually geometrically bear repeating patterns, such as parallel curves and symmetric shapes. Moreover, in a design, these repeating patterns show a higher level of similarity; for example, they may show the same size of shapes symmetrically or the same width of parallel curves. Leveraging these patterns to fit in the missing regions provides an approach to address this problem. Second, natural image completion focuses on shape and appearance at the semantic level. However, our design completion lies at the structure level, where lines, edges, and shapes are our focus. The network architecture needs to be modified to learn features at the structure level. Third, rotation and translation will play an important role in design completion because the sherd images can be in any rotation and translation when overlaid on their underlying design images. Fourth, imperfect curve structure segmentation would affect completion performance. For example, two input sherd curve pattern images from the same design can show subtle structure variation, hence having different completion results. In our study, new algorithms are expected to address these problems. We will start from a synthesized dataset, i.e., generating synthesized sherds by masking existing constructed designs and then moving to real sherds. Output from the sherd identification task can serve as the real sherd dataset. Both datasets will be used to evaluate the to-be-proposed algorithm.

# Chapter 9

# Acknowledgements

This research is supported by the National Endowment for the Humanities (NEH) Digital Humanities Advancement Grant Program (HAA-266472-19), the National Science Foundation Archaeology and Archaeometry Grant Program (1658987), the National Center for Preservation Technology and Training Grant Program (P16AP00373), the Extreme Science and Engineering Discovery Environment (XSEDE) Science Gateway Program (DBS180011), University of South Carolina ASPIRE II Program, and University of South Carolina Social Science Provost Grant Program. We would like to show our gratitude to Frankie Snow at South Georgia State College for sharing his pearls of wisdom and design images with us during this research. We are very grateful to Dr. Matthew Compton, the Curator of the R. M. Bogan Repository at Georgia Southern University, and Dr. Amanda Roberts Thompson, Operations Director of the Laboratory of Archaeology at the University of Georgia, for generously providing access to collections, and Scot Keith at New South Associates for his enthusiasm and encouragement of this research. We would like to thank Research Computing at the University of South Carolina for high-performance computing support, and the South Carolina Department of Natural Resources for institutional support.

# Bibliography


[1] Vahid K. Alilou, *Fingerprint matching: A simple approach.*

[2] David Arthur and Sergei Vassilvitskii, *k-means++: The advantages of careful seeding*, Tech. report, Stanford, 2006.

[3] Vassileios Balntas, Karel Lenc, Andrea Vedaldi, and Krystian Mikolajczyk, *Hpatches: A benchmark and evaluation of handcrafted and learned local descriptors*, IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 5173–5182.

[4] Harry Barrow, Jay Tenenbaum, Robert Bolles, et al., *Parametric correspondence and chamfer matching: Two new techniques for image matching*, IJCAI, vol. 2, Morgan Kaufmann Inc., 1977, pp. 659–663.

[5] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool, *Surf: Speeded up robust features*, European Conference on Computer Vision, Springer, 2006, pp. 404–417.

[6] Serge Belongie, Jitendra Malik, and Jan Puzicha, *Shape Context: A new descriptor for shape matching and object recognition*, NIPS, 2001, pp. 831–837.

[7] Matthew Brown and David Lowe, *Automatic panoramic image stitching using invariant features*, International Journal of Computer Vision **74** (2007), no. 1, 59–73.

[8] Bettye Broyles, *Reconstructed designs from swift creek complicated stamped sherds*, Southeastern Archaeological Conference Bulletin **8** (1968), 49–55.

[9] Roberto Brunelli, *Template matching techniques in computer vision: Theory and practice*, Wiley, New York, 2009.

[10] Roberto Brunelli, *Template matching techniques in computer vision: Theory and practice*, John Wiley & Sons, 2009.

[11] Alan Bundy and Lincoln Wallen, *Difference of gaussians*, Catalogue of Artificial Intelligence Tools, Springer, 1984, pp. 30–30.

[12] Paul Burchard, *Total variation geometry I: Concepts and motivation*, UCLA Computational Applied Mathematics Report **2** (2002), no. 1.

[13] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze, *Deep clustering for unsupervised learning of visual features*, European Conference on Computer Vision, 2018, pp. 132–149.

[14] Chad Carson, Megan Thomas, Serge Belongie, Joseph M Hellerstein, and Jitendra Malik, *Blobworld: A system for region-based image indexing and retrieval*, International Conference on Advances in Visual Information Systems, Springer, 1999, pp. 509–517.

[15] Jianlong Chang, Lingfeng Wang, Gaofeng Meng, Shiming Xiang, and Chunhong Pan, *Deep adaptive image clustering*, IEEE International Conference on Computer Vision, 2017, pp. 5879–5887.

[16] Ping Chao, Chao-Yang Kao, Yu-Shan Ruan, Chien-Hsiang Huang, and Youn-Long Lin, *Hardnet: A low memory traffic network*, IEEE International Conference on Computer Vision, 2019, pp. 3552–3561.

[17] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille, *DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs*, arXiv preprint arXiv:1606.00915 (2016).

[18] Sumit Chopra, Raia Hadsell, and Yann LeCun, *Learning a similarity metric discriminatively, with application to face verification*, IEEE Conference on Computer Vision and Pattern Recognition, vol. 1, IEEE, 2005, pp. 539–546.

[19] Djork-Arné Clevert, Unterthiner Thomas, and Sepp Hochreiter, *Fast and accurate deep network learning by exponential linear units (elus)*, arXiv preprint arXiv:1511.07289 (2015).

[20] Dorin Comaniciu and Peter Meer, *Mean shift: A robust approach toward feature space analysis*, IEEE Transactions on Pattern Analysis and Machine Intelligence **24** (2002), no. 5, 603–619.

[21] David Cooper, Andrew Willis, Stuart Andrews, et al., *Assembling virtual pots from 3D measurements of their fragments*, Conference on Virtual Reality, Archeology, and Cultural Heritage, ACM, 2001, pp. 241–254.

[22] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner, *Scannet: Richly-annotated 3d reconstructions of indoor scenes*, IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 5828–5839.

[23] Navneet Dalal and Bill Triggs, *Histograms of oriented gradients for human detection*, IEEE Conference on Computer Vision and Pattern Recognition, vol. 1, 2005, pp. 886–893.

[24] Max Frenkel and Ronen Basri, *Curve matching using the fast marching method*, International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition, Springer, 2003, pp. 35–51.

[25] Brendan J Frey and Delbert Dueck, *Clustering by passing messages between data points*, science **315** (2007), no. 5814, 972–976.

[26] Clive Gamble, *Archaeology: The basics*, London: Rutledge, 2004.

[27] Ayelet Gilboa, Avshalom Karasik, Ilan Sharon, et al., *Towards computerized typology and classification of ceramics*, vol. 31, Elsevier, 2004, pp. 681–694.

[28] Maria Letizia Gualandi, Gabriele Gattiglia, and Francesca Anichini, *An open system for collection and automatic recognition of pottery through neural network algorithms*, Heritage **4** (2021), no. 1, 140–159.

[29] Radim Halir, *An automatic estimation of the axis of rotation of fragments of archaeological pottery: A multi-step model-based approach*, International Conference in Central Europe on Computer Graphics, Visualization and Interactive Digital Media, 1999.

[30] Mahmoud Hassaballah, Aly Abdelmgeid, and Hammam Alshazly, *Image features detection, description and matching*, Image Feature Detectors and Descriptors, 2016, pp. 11–45.

[31] Kaiming He, Xiangyu Zhang, Shaoqing Ren, et al., *Deep residual learning for image recognition*, IEEE Conference on Computer Vision and Pattern Recognition, 2016.

[32] Kaiming He, Xianyu Zhang, Shaoqing Ren, et al., *Delving deep into rectifiers: Surpassing human-level performance on imagenet classification*, IEEE International Conference on Computer Vision, 2015.

[33] Ryota Hinami, Yusuke Matsui, and Shin'ichi Satoh, *Region-based image retrieval revisited*, ACM International Conference on Multimedia, 2017, pp. 528–536.

[34] Elad Hoffer and Nir Ailon, *Deep metric learning using triplet network*, International workshop on similarity-based pattern recognition, Springer, 2015, pp. 84–92.

[35] Katsushi Ikeuchi, Takeshi Oishi, Jun Takamatsu, Ryusuke Sagawa, Atsushi Nakazawa, Ryo Kurazume, Ko Nishino, Mawo Kamakura, and Yasuhide Okamoto, *The great buddha project: Digitally archiving, restoring, and analyzing cultural heritage objects*, International Journal of Computer Vision **75** (2007), no. 1, 189–208.

[36] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros, *Image-to-image translation with conditional adversarial networks*, IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 1125–1134.

[37] Anil Jain, Arun Ross, and Salil Prabhakar, *Fingerprint matching using minutiae and texture features*, International Conference on Image Processing, vol. 3, IEEE, 2001, pp. 282–285.

[38] Anil K Jain and Richard C Dubes, *Algorithms for clustering data*, Prentice-Hall, Inc., 1988.

[39] Martin Kampel and Robert Sablatnig, *Rule based system for archaeological pottery classification*, Pattern Recognition Letters **28** (2007), no. 6, 740–747.

[40] Avshalom Karasik and Uzy Smilansky, *3D scanning technology as a standard archaeological tool for pottery analysis: practice and theory*, Journal of Archaeological Science **35** (2008), no. 5, 1148–1168.

[41] Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton, *Imagenet classification with deep convolutional neural networks*, Advances in Neural Information Processing Systems, vol. 1, 2012, pp. 1097–1105.

[42] Chunming Li, Chenyang Xu, Changfeng Gui, and Martin D Fox, *Distance regularized level set evolution and its application to image segmentation*, IEEE Transactions on Image Processing **19** (2010), no. 12, 3243–3254.

[43] Qi Li-Ying and Wang Ke-Gang, *Kernel fuzzy clustering based classification of ancient-ceramic fragments*, IEEE International Conference on Information Management and Engineering, IEEE, 2010, pp. 348–350.

[44] Stuart Lloyd, *Least squares quantization in pcm*, IEEE Transactions on Information Theory **28** (1982), no. 2, 129–137.

[45] Jonathan Long, Evan Shelhamer, and Trevor Darrell, *Fully convolutional networks for semantic segmentation*, IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3431–3440.

[46] David G Lowe, *Object recognition from local scale-invariant features*, IEEE International Conference on Computer Vision, vol. 2, IEEE, 1999, pp. 1150–1157.

[47] Yuhang Lu, Jun Zhou, Jun Chen, Jing Wang, Karen Smith, Wilder Colin, and Song Wang, *Curve-structure segmentation from depth maps: A CNN-based approach and its application to exploring cultural heritage objects*, Association for the Advancement of Artificial Intelligence (2018).

[48] Yuhang Lu, Jun Zhou, Sam T. McDorman, Canyu Zhang, Deja Scott, Jake Bukuts, Colin Wilder, Karen Y. Smith, and Song Wang, *Snowvision: Segmenting, identifying, and discovering stamped curve patterns from fragments of pottery*, International Journal of Computer Vision. Manuscript submitted for publication. (2022).

[49] Wenjie Luo, Yujia Li, Raquel Urtasun, et al., *Understanding the effective receptive field in deep convolutional neural networks*, Neural Information Processing Systems, 2016.

[50] Andrew Maas, Awni Hannun, and Andrew Ng, *Rectifier nonlinearities improve neural network acoustic models*, International Conference on Machine Learning, 2013.

[51] Nenad Markuš, Igor Pandžić, and Jörgen Ahlberg, *Learning local descriptors by optimizing the keypoint-correspondence criterion: applications to face matching, learning from unlabeled videos and 3d-shape retrieval*, IEEE Transactions on Image Processing **28** (2018), no. 1, 279–290.

[52] Timo Ojala, Matti Pietikainen, and Topi Maenpaa, *Multiresolution gray-scale and rotation invariant texture classification with local binary patterns*, IEEE Transactions on Pattern Analysis and Machine Intelligence **24** (2002), no. 7, 971–987.

[53] Yuki Ono, Eduard Trulls, Pascal Fua, and Kwang Moo Yi, *Lf-net: learning local features from images*, Advances in Neural Information Processing Systems, 2018, pp. 6234–6244.

[54] Charles Otto, Dayong Wang, and Anil K Jain, *Clustering millions of faces by identity*, IEEE Transactions on Pattern Analysis and Machine Intelligence **40** (2017), no. 2, 289–303.

[55] Philip Phillips and James Brown, *Pre-columbian shell engravings from the craig mound at spiro, oklahoma, part 1*, Peabody Museum Press, 1978.

[56] Stephen M Pizer, R Eugene Johnston, James P Ericksen, Bonnie C Yankaskas, and Keith E Muller, *Contrast-limited adaptive histogram equalization: speed and effectiveness*, Conference on Visualization in Biomedical Computing, IEEE Computer Society, 1990, pp. 337–338.

[57] Dorie Reents-Budet, *Painting the maya universe: Royal ceramics of the classic period (duke university museum of art)*, Duke University Press, Durham, 1994.

[58] Brett Riggs and Christopher Rodning, *Cherokee ceramic traditions of southwestern North Carolina, ca. A.D. 1400-2002: A preface to "the last of the Iroquois potters"*, North Carolina Archaeology **51** (2002).

[59] Edgar Roman-Rangel, Diego Jimenez-Badillo, and Estibaliz Aguayo-Ortiz, *Categorization of aztec potsherds using 3d local descriptors*, Asian Conference on Computer Vision - Workshops, 2014, pp. 567–582.

[60] Edgar Roman-Rangel, Carlos Pallan, Jean-Marc Odobez, et al., *Analyzing ancient Maya glyph collections with contextual shape descriptors*, International Journal of Computer Vision **94** (2011), no. 1, 101–117.

[61] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski, *Orb: An efficient alternative to sift or surf*, International Conference on Computer Vision, IEEE, 2011, pp. 2564–2571.

[62] Florian Schroff, Dmitry Kalenichenko, and James Philbin, *Facenet: A unified embedding for face recognition and clustering*, IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 815–823.

[63] Xuelun Shen, Cheng Wang, Xin Li, Zenglei Yu, Jonathan Li, Chenglu Wen, Ming Cheng, and Zijian He, *Rf-net: An end-to-end image matching network based on*

*receptive field*, IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 8132–8140.

[64] Jianbo Shi and Jitendra Malik, *Normalized cuts and image segmentation*, IEEE Transactions on Pattern Analysis and Machine Intelligence **22** (2000), no. 8, 888–905.

[65] Karen Y Smith and Keith Stephenson, *The spatial dimension of the woodland period*, Southeastern Archaeology **37** (2018), no. 2, 112–128.

[66] Patrick Smith, Dmitriy Bespalov, Ali Shokoufandeh, and Patrice Jeppson, *Classification of archaeological ceramic fragments using texture and color descriptors*, IEEE Conference on Computer Vision and Pattern Recognition - Workshops, 2010, pp. 49–54.

[67] Francis Snow, *Swift creek designs and distributions: A south Georgia study*, Early Georgia **3** (1975), no. 2, 38–59.

[68] Keith Stephenson, Judith A. Bense, , and Frankie Snow, *Aspects of deptford and swift creek of the south atlantic and gulf coastal plains. in the woodland southeast, edited by david g. anderson and robert c. mainfort, jr.*, University of Alabama Press, 2002.

[69] Marius Tico and Pauli Kuosmanen, *Fingerprint matching using an orientation-based minutia descriptor*, IEEE Transactions on Pattern Analysis and Machine Intelligence **25** (2003), no. 8, 1009–1014.

[70] Stefan Van der Walt, Johannes L Schönberger, Juan Nunez-Iglesias, François Boulogne, Joshua D Warner, Neil Yager, Emmanuelle Gouillart, and Tony Yu, *scikit-image: image processing in python*, PeerJ **2** (2014), e453.

[71] Zhongdao Wang, Liang Zheng, Yali Li, and Shengjin Wang, *Linkage based face clustering via graph convolution network*, IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 1117–1125.

[72] Jianlong Wu, Keyu Long, Fei Wang, Chen Qian, Cheng Li, Zhouchen Lin, and Hongbin Zha, *Deep comprehensive correlation mining for image clustering*, International Conference on Computer Vision, 2019, pp. 8150–8159.

[73] Kuo-Lung Wu and Miin-Shen Yang, *Mean shift-based clustering*, Pattern Recognition **40** (2007), no. 11, 3035–3052.

[74] Kwang Moo Yi, Eduard Trulls, Vincent Lepetit, and Pascal Fua, *Lift: Learned invariant feature transform*, European Conference on Computer Vision, Springer, 2016, pp. 467–483.

[75] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang, *Free-form image inpainting with gated convolution*, International Conference on Computer Vision, 2019, pp. 4471–4480.

[76] Matthew Zeiler and Rob Fergus, *Visualizing and understanding convolutional networks*, European Conference on Computer Vision, 2014.

[77] Xiaohang Zhan, Ziwei Liu, Junjie Yan, Dahua Lin, and Chen Change Loy, *Consensus-driven propagation in massive unlabeled data for face recognition*, European Conference on Computer Vision, 2018, pp. 568–583.

[78] Chuanxia Zheng, Tat-Jen Cham, and Jianfei Cai, *Pluralistic image completion*, IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 1438–1447.

[79] Jun Zhou, Yuhang Lu, Karen Smith, Colin Wilder, Song Wang, Paul Sagona, and Ben Torkian, *A framework for design identification on heritage objects*, Practice and Experience in Advanced Research Computing on Rise of the Machines, 2019, pp. 1–8.

[80] Jun Zhou, Karen Smith, Greg Wilsbacher, Paul Sagona, David Reddy, and Ben Torkian, *Building science gateways for humanities*, Practice and Experience in Advanced Research Computing, 2020, pp. 327–332.

[81] Jun Zhou, Haozhou Yu, Karen Smith, Colin Wilder, Hongkai Yu, and Song Wang, *Identifying designs from incomplete, fragmented cultural heritage objects by curve-pattern matching*, Journal of Electronic Imaging **26** (2017), no. 1, 011022–011022.

[82] Joviša Žunić, Kaoru Hirota, and Paul L Rosin, *A hu moment invariant as a shape circularity measure*, Pattern Recognition **43** (2010), no. 1, 47–57.