Chapter 1

Digital Reconstruction and Mosaicing of Cultural Artifacts

EFTHYMIA TSAMOURA, NIKOS NIKOLAIDIS, IOANNIS PITAS

Department of Informatics
Aristotle University of Thessaloniki
Thessaloniki, Greece
Email: etsamour@csd.auth.gr,nikolaid@aiia.csd.auth.gr,pitas@aiia.csd.auth.gr

1.1 Introduction

Reconstruction of broken or torn artifacts of archaeological, historical or cultural importance is an indispensable tool in the hands of researchers in these fields. Indeed, ceramic pot fragments are often the most important source of information from which archaeologists make inferences about the civilization that existed at a given archaeological site (e.g., its chronology, the population's socioeconomic standards, etc.). For murals and mosaics, see Figure 1.1, accurate reconstructions provide important information to archaeologists about the iconography, stylistic and drawing tools developments, etc. In the case of stone tablets and other artifacts that bear text inscriptions (e.g., papyrus), reconstructions that allow epigraphists to read part of the entire text provide important information, about the society's organization and the scientific and cultural evolution (e.g., poetry, drama), of ancient civilizations such as Greek, Persian, Egyptian, etc.

During the last years, researchers in the fields of image processing and analysis and computer vision have dealt with computer-based approaches for the reconstruction of such artifacts, providing a valuable helping hand to archaeologists, historians and restoration experts. There are many advantages in using computers for the reconstruction of fragmented artifacts. Some of them are listed below.

- Computer-aided reconstruction (either automatic, or human supervised) may lead to huge time savings for archaeologists who spend many hours assembling broken artifacts by hand.
- In most cases, the reconstruction of broken ancient artifacts is extremely difficult for many reasons; the fragments are highly spoiled (e.g., from environmental and other effects), or are fragmented into many tiny pieces. Furthermore fragments from many different pots, murals or mosaics are found mixed during the excavations. Unless the fragments are visually similar and discovered in a context that is local in both time and space, archaeologist very rarely proceed to reconstruction.
- Manual reconstruction of three-dimensional objects is even harder. In that case, the employment of a software tool for 3D fragments visualization, virtual manipulation or reconstruction is extremely

useful.

- The correct computer-based reconstruction of artifacts provides the ability to conduct precise measurements on reconstructed objects. These measurements can sometimes provide a better insight for historical events and historical evolution. When such measurements are performed by archaeologists using empirical techniques they usually have a large margin of error [1]. As an example, archaeologists used to estimate a pot's type, volume and other characteristics from its neck (or a portion of it). In such cases, variations in estimated curvatures, especially for large pots, may lead to volume estimation errors of up to 30%. Such significant errors, if they are repetitive for an excavation site or a number of sites, can lead to false typological and chronological conclusions.

- Finally, the development of computer-based 3D fragmented objects reconstruction methods may have potential applications in other domains such as anthropology or forensic science (e.g., in the reconstruction of a human skeleton from broken bones).

A number of automatic or semiautomatic reconstruction methods have been proposed in the literature. Some of these techniques were actually used in projects for the reconstruction of fragmented objects. A human guided method for the reassembly of murals at the Akrotiri archaeological site in the Greek island of Thera (Santorini), dated back to 1600 BC, was presented in [2]. Although the fragments are 3D, only their 2D boundary (fracture curve) was utilized in the reconstruction project. The murals of Thera were preserved because they were covered by pumice from the eruption of a volcano. The walls that were decorated with the murals collapsed before the volcanic eruption, due to strong earthquakes. Thus, a single painting is usually scattered into many fragments, sometimes mixed with the fragments from other murals, rendering the restoration of the murals from its constituent fragments a very painstaking and time consuming process. The method has been applied to two fragmented murals consisting of 262 fragments. Another important project that dealt with the reconstruction of 3D objects is the Stanford's Digital Forma Urbis Romae project [3]. The project aimed to reconstruct the Seventh Marble of Rome or Forma Urbis Romae. This enormous map of Rome, measuring 18×13 meters, constructed between 203 and 211 AD, was carved onto 150 marble slabs installed on a wall of an aula of the Templum Pacis. The map was destroyed in the 5th century, resulting in 1186 pieces. Since its rediscovery in 1562, scholars have focused on joining the fragments and reconstructing this great monument, but this was a very difficult puzzle to work with since the map is drastically incomplete (only 10 percent of the map survives). In addition, many fragments are huge and heavy, while others are so small that their carved surfaces hardly provide any identifiable information. The proposed reconstruction method required human supervision. Finally, a project dealing with the reassembly of documents of archaeological importance is "The Sinaitic Glagolitic Sacramentary (Euchologium) Fragments" [4]. The project deals with the recording, investigation and edition of two very important medieval Slavonic manuscripts, discovered in 1975 in St. Catherine's monastery on Mountain Sinai: "Euchologii Sinaitici pars nova" and "Missale (Sacramentarium) Sinaiticum". The folios of these manuscripts are degraded due to water.

Another example of fragmented objects, where the computer assistance would be valuable is ripped-up documents. This problem arises in archival study and investigation science. Documents may be torn by hand or shredded by a machine, called shredder. Similar to archaeological fragmented objects, manually ripped-up documents may also suffer damages at several levels, such as, torn edges, moisture, obliteration, charring, and shredding. Thus, their manual reconstruction may require days or even months, depending on the number of fragments, and their conservation state. Another problem that occurs when reassembling documents by hand lies in their manipulation. The physical reconstruction of a document modifies the original document, since products like glue and adhesive tape are added into it. Thus, the computer aided virtual reconstruction of ripped-up documents may reduce or eliminate the time-consuming manual effort, and also lead to reassembled documents that exhibit small degradation with respect to the original.

A special case of fragmented 2D objects, which are not related to Cultural Heritage and its preservation, are the jigsaw puzzles. A jigsaw puzzle is a tiling puzzle that requires the assembly of numerous small, oddly shaped, interlocking pieces, that, when properly assembled, form a picture without gaps and overlaps. Jigsaw puzzles were originally created by painting a picture on a flat, rectangular piece of wood, and then cutting that picture into small pieces with a jigsaw, hence the name. John Spilsbury, a London mapmaker and engraver, is credited with creating the first jigsaw puzzles around 1760. The characteristics of jigsaw puzzles, (e.g., that each puzzle piece is perfectly cut and well preserved) are different from those of archaeological fragmented artifacts. However, puzzle reassembly and 2D object reconstruction methods have many common characteristics. In general, reassembly of puzzles is considered less difficult than reconstruction of ancient artifacts or paper documents , due to the following reasons [2]:

- There is no a-priori knowledge about the picture content drawn on the surface of the unbroken object.
- The size and the shape of the fragments varies dramatically in contrast to what happens in jigsaw puzzles.
- Gaps may exist between adjacent fragments of a fragmented object, due to degradation. Furthermore, one or more fragments may be missing.
- The fragments are often highly degraded from various factors (e.g., environmental effects), in contrast to jigsaw puzzle pieces.
- There is no unique solution concerning the matching of two fragments. As will be discussed later, there is no criterion to confirm that the matching of two fragments is correct.

In summary, the types of fragmented objects whose reconstruction has been studied in the corresponding literature, can be classified into the following two broad categories:

- 2D objects, such as 2D puzzles, paper documents, paper images, mosaics, murals.
- 3D objects, symmetric or asymmetric, such as tiles, bowls, amphorae or other types of ceramic pots.

Another problem that is closely related to that of 2D fragmented object reconstruction is that of mosaicing or stitching. Mosaicing is the process of reconstructing or re-stitching a single, continuous image from a set of overlapping sub-images. Image mosaicing is essential for the creation of high resolution digital images of architectural monuments and works of art (especially of those with considerable dimensions like frescoes and large-size paintings) for archival, digital analysis and restoration purposes [5],[6]. In such applications, the required very high resolution image acquisition stresses the limits of acquisition devices. To overcome this obstacle, digitization procedures that utilize the acquisition of different, overlapping views, sometimes with the aid of sensor arrays and positioning mechanisms, are used. The acquired sub-images (Figure 1.2) are subsequently processed by a mosaicing algorithm.

Several image mosaicing techniques have been proposed in the literature [5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15] since mosaicing is also important in other areas that include the creation of high-resolution large-scale panoramas for virtual environments, image-based rendering, consumer photography, medical imaging [7], [12], aerial [13], satellite and underwater [14] imaging etc. The mosaicing process comprises of two steps. The first step involves the estimation of the optimal transformation (translation, rotation, scaling etc) that aligns each sub-image with respect to each neighboring one. If one assumes that only translational camera motion takes place during the acquisition then only the relative displacement has to be evaluated. This step is the most computationally intensive part of the entire process. In the general case of a set of $M_1 \times M_2$ sub-images, optimal mosaicing (i.e., mosaicing by concurrently searching for the optimal position of all sub-images), would require a search in a m-dimensional space, where $m = 2(2M_1M_2 - M_1 - M_2)$, the term in parenthesis representing the number of all pairs of neighboring sub-images. Obviously the computational cost becomes prohibitive, as the number of sub-images increases. The second step of the mosaicing process utilizes displacement information found in the previous step in order to combine each pair of neighboring sub-images with invisible seams and thus reconstruct the whole image (Figure 1.15).

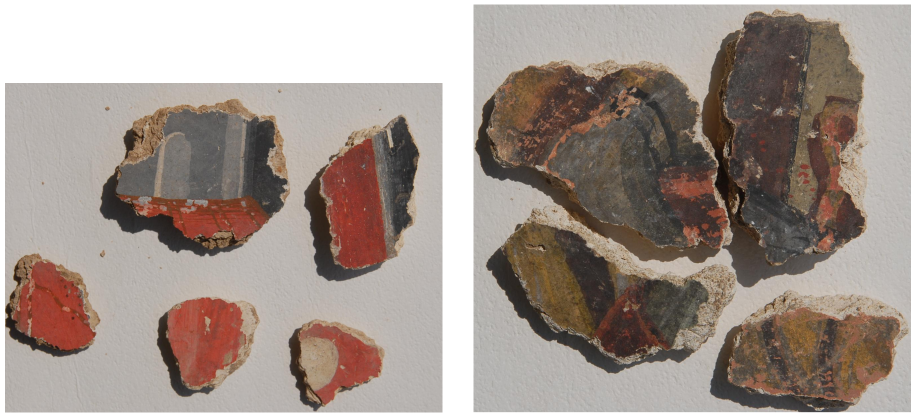

(a) (b)

Figure 1.1: Fragments from Byzantine murals.

In this Chapter we will try to provide a concise review of algorithms proposed so far for the 2D or 3D reassembly of fragmented or torn objects of cultural importance. Section 1.2 deals with the three steps of an object reconstruction procedure whereas reconstruction methods will be reviewed in Section 1.3. Despite the fact that several methods were developed for the reassembly of shredded documents (e.g., [16], [17], [18], [19]), in this Chapter we will review only methods proposed for manually ripped-up (hand torn) documents, since the former have no application in Cultural Heritage preservation. Furthermore, although computer-aided reassembly of puzzles (which attracted significant interest since mid 80's, [20], [21], [22], [23]) is not directly related to the topic of this Chapter, we will briefly review a small number of such techniques, since there is a close relation between puzzle reassembly and 2D object reconstruction methods and the two areas share many principles. Subsequently, in Section 1.4 we present an automatic four-step method for the reassembly of torn 2D paper images or fragmented paintings [24]. Finally, in Section 1.5 we will describe two methods that aim to reduce the number of computations required for the evaluation of the sub-image displacements in mosaicing, without affecting significantly the matching error [25]. Readers interested to get more details for the computational reconstruction of 2D/3D ancient artifacts and mosaicing techniques can delve into the thorough review and tutorial papers of Willis and Cooper [1] and Szeliski [26] respectively.

Throughout this Chapter we will use the term reassembly for 2D objects and reconstruction for 3D objects. Furthermore, the term fracture surface will be used hereafter to denote the boundary surface of a 3D fragment that was created through the fracturing whereas the term fracture curve will be used to denote the boundary of a 2D fragment or the curve delineating the fracture on the outer or inner surface of a 3D fragment (Figure 1.3).

1.2 The Three-step Object Reconstruction Procedure

The purpose of this Section is to present the steps required for the overall reconstruction of an object from its consistent fragments, as well as a comparative study of the methods proposed for each step (e.g., the features and the techniques utilized to identify the matching of two fragments).

Before the execution of a reconstruction method, digitization of fragments must take place. In case of paper documents or color images, the fragments can be either scanned or photographed. The same procedure is followed for 3D objects when only 2D data are utilized for reassembly, as in [2] and [27]. In this case, many conditions must be met when the fragments are photographed: same illumination conditions, no artificial shadows, fixed distance of the fragment plane from the camera focus, object surface parallel to the photographic glass and minimal photo distortion, etc. In the case of actual 3D object reconstruction, the fragments can be either scanned with a 3D laser scanner, or photographed from different viewpoints and reconstructed in 3D using photogrammetry techniques. It must be noted that in all methods that will be

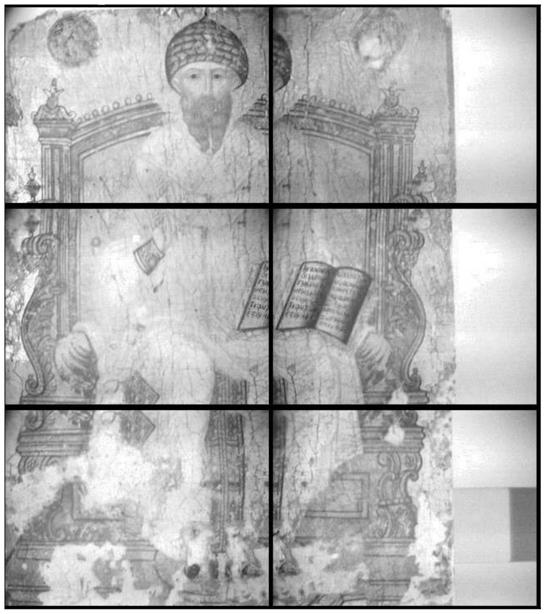


Figure 1.2: A $M_1 = 3$ by $M_2 = 2$ sub-image acquisition of a painting.

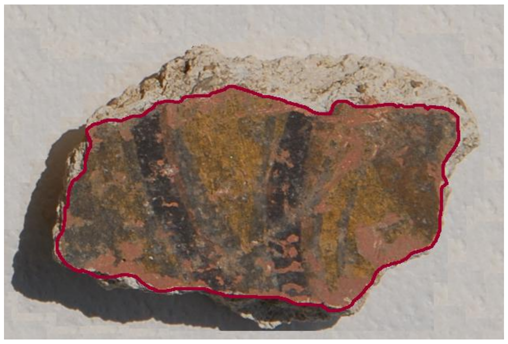


Figure 1.3: The outer surface (the one with depictions on it), the fracture surface (the gray irregular surface on the perimeter of the fragment) and the fracture curve (in red) of a 3D fragment.

reviewed, all input fragments are assumed to belong to the same broken or torn object, unless it is explicitly stated.

Overall object reconstruction methods are either automatic or semi-automatic. In the former category, the object reconstruction process is not guided by users, whose role is limited to the digitization of the input fragments and sometimes the selection of appropriate parameter values. In the latter category (e.g., [2] and [28]), the user selects the correct matches for pairs of fragments among the ones proposed by the system and/or the appropriate affine transformation in order to correctly align them along their matching segments.

All the proposed methods try to identify fragments that were adjacent in the unbroken object. The adjacent fragments should have high similarity with respect to e.g., the shape and/or color of fracture curves in case of 2D objects or the fracture surfaces in case of 3D objects. Similarities may also exist in the depictions on the surface of the fragments. Thus, the existing methods perform pair-wise fragments comparisons based on several criteria such as shape, color, or texture in order to identify candidate adjacent fragments.

In general, the reconstruction of an object is a three-step procedure:

- **Fragments preprocessing**. The purpose of this optional step is to reduce the computational burden of the steps that follow. Two different approaches for fragment preprocessing exist. The first reduces

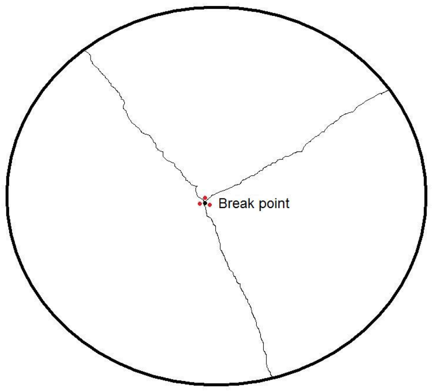


Figure 1.4: A break point on a fragmented object. The break points of the three fragments are marked with red dots.

the number of pair-wise fragment comparisons that will be performed during the second step. Once this step is finished, a set of possibly adjacent fragments is retained. The second approach is to reduce the number of matching possibilities between a certain pair of fragments.

- **Matching and alignment of candidate adjacent fragments**. The second operation that takes place is the matching of pairs of fragments, namely the identification of point correspondences between fracture curves or surfaces on two fragments. If pairs of candidate adjacent fragments have been generated in the first step, then only the matching of these pairs is estimated. The matching process may be guided by information produced during the preprocessing step, see for example [29]. The identification of the matching of two fragments can be done through several approaches. One approach is to utilize information only from a fragment's fracture curve or surface, while another is to utilize information depicted on the surfaces of a fragment. Regarding the former approach, several alternatives for contour/surface comparisons have been proposed. The majority of them perform comparisons utilizing shape and or color criteria, while other utilize pixels/points from the fragments' contours/surfaces. Usually, some candidate adjacent fragments found in the previous step are discarded if the matching quality does not satisfy certain criteria [24]. The output is a (new) set of candidate adjacent pairs that are appropriately matched.

  Many matching methods simultaneously estimate the alignment transformation (i.e., the appropriate affine transformation of one fragment relative to its adjacent one), in order to correctly align and merge them into one. This combined matching and alignment procedure is usually referred to as registration. If the alignment transformation is not found during matching, a separate secondary alignment step is executed. In order to identify this transformation certain criteria are maximized, while information from the matching step is also exploited (see for example [24]). Usually, the affine transformation must satisfy certain constraints (e.g., adjacent fragments must not overlap).

- **Multi-fragments merging**. In this step, the input fragments are appropriately merged in order to reconstruct the overall object. The matching of potentially adjacent fragments and their alignment are utilized to this end. The adjacent fragments are selected among the pairs produced during the previous steps, using heuristic criteria.

1.2.1 Fragments Preprocessing

As mentioned above, the purpose of this step is to speed-up the overall reconstruction process. Most algorithms involve pair-wise matchings in order to discover fragments that were adjacent in the original object. Therefore, for an object consisting of hundreds or thousands of fragments, which is usually the case in archaeological excavations, the cost of these comparisons becomes prohibitively high. Thus, one possible

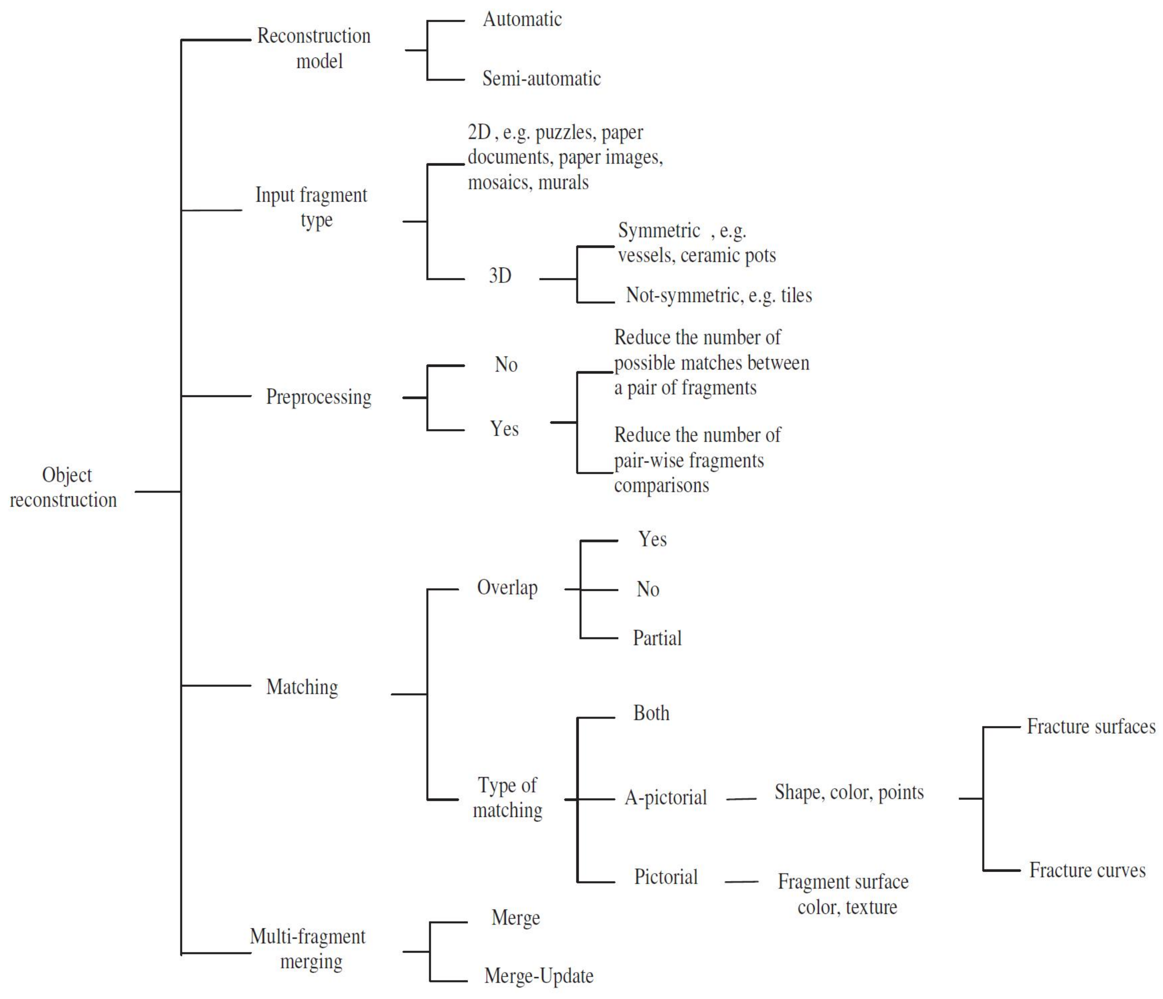

Object reconstruction
Reconstruction model
Automatic
Semi-automatic
Input fragment type
2D , e.g. puzzles, paper documents, paper images, mosaics, murals
3D
Symmetric , e.g. vessels, ceramic pots
Not-symmetric, e.g. tiles
Preprocessing
No
Yes
Reduce the number of possible matches between a pair of fragments
Reduce the number of pair-wise fragments comparisons
Matching
Overlap
Yes
No
Partial
Type of matching
Both
A-pictorial
Shape, color, points
Fracture surfaces
Fracture curves
Pictorial
Fragment surface color, texture
Multi-fragment merging
Merge
Merge-Update

Figure 1.5: A schematic representation of the various steps and characteristics of object reconstruction methods and their variations.

way to speed up the process, is to reduce the number of pair-wise comparisons, either manually, by having the users proposing pairs of potentially adjacent fragments, or through more advanced methods. In [24], for example, a matching approach based on color similarity is employed in order to identify possibly adjacent image fragments. Pre-processing approaches of this type are obviously optional and may lead to reconstruction errors (e.g., two adjacent fragments in the unbroken object may be erroneously classified as not adjacent and not retained for further examination). A second approach is to reduce the matching possibilities between two fragments. This is very useful when the input fragments are three-dimensional. For example, Willis *et. al.* [30], utilize the break points of the 3D fragmented objects, in order to speed up the matching identification process between any pair of sherds. The break points of an object are the surface points where the object has broken. An example of a break point is shown in Figure 1.4. In reality, a break point on the original object is "split" among the various fragments that meet on it. Utilizing the break points of the fragments, the number of the possible contour matchings between a pair of contour curves is significantly reduced. For more details see Section 1.3. In [29], the boundary surfaces of the input fragments are classified into original surfaces, which come from the boundary surface of the unbroken object, and fracture surfaces, which were created when the object broke. After that, only matchings over points on the fracture surfaces are investigated. A similar technique is also employed in 2D puzzles reassembly due to the shape characteristics of puzzle pieces. Indeed, no matching contour segments are searched along a linear contour segment of a puzzle piece, since such a segment obviously belongs to the border of the puzzle [31]. Approaches of this category are beneficial for the correct reconstruction of an object, since an original surface cannot be matched with a break-surface in 3D reconstruction. However, the high degradation of some fragments, especially in archaeology, may lead to erroneous classification of original surfaces to break-surfaces and vice versa.

1.2.2 Matching of Candidate Adjacent Fragments

After the optional preprocessing step, the identification of the matching of two input fragments takes place. If a preprocessing step has been utilized, this procedure is performed only for candidate adjacent fragments. Matching of contour segments can be performed in two ways [32]. In certain cases, only information (e.g., color and/or shape) from the fracture contours or surfaces is considered. We will call this matching non-pictorial (or a-pictorial). On the other hand, when information (e.g., texture, color) depicted on the surfaces of a fragment is accounted to identify the matching of two fragments, the matching is called pictorial. To the best of our knowledge all the methods that were proposed so far for the reconstruction of 2D and 3D objects, except from some methods for 2D puzzles reassembly such as [31],[33] and [34], perform non-pictorial matching. For example, in [31], non-pictorial registration of the pixels in the contours of puzzle pieces is performed, in order to identify the adjacent candidate puzzle pieces. After this step, the set of best adjacent candidates is pruned using pictorial criteria. The features that are employed are related to the RGB colors of the pieces. In [33], only pictorial criteria are utilized to identify the matching of puzzle pieces. Each puzzle piece is expanded in a band around the border of the piece by predicting the pictorial information from the surface outwards, using inpainting techniques [35].

The non-pictorial matching methods employ shape (e.g., curvature, torsion) or color features of the fracture curve or surface, positions of 2D/3D points on the fracture curve or surface, or combine shape and/or color features, as well as points. In particular, methods for non-pictorial matching of 3D fragments can be further classified into those that perform matching by employing characteristics or points from the fracture surfaces and those that employ characteristics or points from the fracture curves. Examples of methods in the latter category include [30], [36] and [37].

As already mentioned, some methods perform registration (concurrent mapping and alignment) of the contour or surface points, while others perform the alignment of the matched contour or surface points in a subsequent step. For example in [30] an affine transformation is found that registers the 3D points from the fracture boundaries of two fragments, in [38], the alignment of the matched contour pixels is estimated using a brute force method, while in [24], alignment is estimated using an Iterative Closest Point (ICP) [39] variant.

In order to correctly reconstruct the fragmented object, the adjacent fragments must not overlap in the reconstructed object. The methods proposed in the literature, both pictorial and non-pictorial, are divided into three categories with respect to this feature: those that do not allow the overlap of adjacent fragments, those that allow the overlap of a small percentage of the contours or outer surface segments, and those that allow the overlap of adjacent fragments. Obviously this categorization does not apply in semi-automatic object reconstruction methods (e.g., [2], [28]), where users select the correct matching among those proposed by the system, since in this case users are supposed to arrange fragments so that no overlap occurs.

1.2.3 Multi-fragment Merging

The last step for the overall object reconstruction is the multi-fragment merging. Here, the correct adjacent fragments among the candidates produced during the previous step are selected. The proposed approaches follow either the merge or the merge-update paradigm. In the merge approach, the matched adjacent candidate fragments are iteratively merged into bigger fragments, until one object is produced. In the second category, two or more fragments are merged forming one or more new and bigger fragments and subsequently the reconstruction procedure is applied on the new fragments. In more detail, pair-wise fragment comparisons are performed in the new set of fragments, namely those that either consist of two or more initial fragments or fragments that are not yet merged with others. The new adjacent candidate fragments are properly aligned and merged and the procedure is repeated. The procedure ends when all input fragments are merged into a single object. The advantage of this method is that some pair-wise matches have a better chance to be discovered after previously found adjacent fragments have been merged into a new bigger fragment. On the other hand, the cost of the overall reconstruction process increases significantly, especially in cases where the input fragments are relatively big and the fragments are digitized in high resolution. Figure 1.5 summarizes the basic characteristics of the overall or partial object reconstruction techniques.

1.3 Approaches for Object Reconstruction

In this Section, some automatic and semi-automatic methods for object reconstruction are briefly reviewed. Table 1.1 reports for every method, the object reconstruction model that is employed (i.e., automatic or semi-automatic), the type of fragments that it operates on (e.g., paper documents, puzzle pieces, etc.), whether fragments preprocessing is performed or not and how fragments are merged during the overall object reconstruction step (i.e., through a merge or a merge-update technique). The last three columns of Table 1.1 provide details regarding the properties (see Figure 1.5) of the utilized matching techniques (e.g., the fragments features that are utilized, whether fragments overlap in the reconstructed object or not, etc.).

1.3.1 Torn by Hand Document Reassembly

An automatic method for the reassembly of paper documents torn by hand is proposed in [40]. This method does not virtually reassemble the entire paper document, but proposes pairs of possibly adjacent fragments as well as their matching. Thus, the paper fragments are not aligned, nor checked for overlapping surface segments. The fragment contours are initially approximated through polygonal lines. The discovery of the matching of two paper fragments is performed using a non-pictorial approach. Shape features, namely the angle and the Euclidian distance of a contour point with respect to its two neighboring contour points are estimated from the fragments' contours and a simple point-wise matching approach is introduced; for every pair of contour points, belonging to two different fragments, the corresponding angle and distance features are compared. Each comparison is assigned a value and the sum of these values characterizes the matching. Finally, the merge-based method proposed in [27] is utilized in order to reassemble the entire paper document. In every iteration, the method merges the document fragments with the highest matching similarity, as described above.

A method that targets the same problem is proposed in [41]. A non-pictorial method identifies the matching contour segments of the paper fragments; shape features, namely turning functions [42], are estimated from every fragment contour and a turning function matching algorithm is introduced to discover matching contour segments [41]. The alignment transformation of the fragments is found during the turning function matching procedure using the method proposed in [42]. In a subsequent step, the estimated matchings are disambiguated (i.e., it is verified whether an estimated matching is correct or not) through an iterative procedure, namely the gradient projection method. To this end, a *global confidence score* is assigned to each matching and reflects the probability that a found matching is correct. The fragments whose matching has confidence score equal to one, are merged and the whole procedure starts again, namely the matching contour segments are identified for all pairs of fragments and so on.

1.3.2 3D Objects Reconstruction

Willis *et. al.* [30], in STITCH project, proposed a method for the automatic reconstruction of 3D rotationally symmetric archaeological fragmented objects (e.g., ceramic pots). The matching between pairs of input 3D fragments is identified through a non-pictorial approach, which employs both 3D shape features (e.g., the

Paper	Reconstruc.	Application	Preproc.	Fragment merging	Type of matching	Overlap	Reg.
Justino [40]	Automatic	Paper documents	No	Merge	Non-pictorial: shape features (angle and distance of a contour pixel, with respect to its two neighboring contour pixels)	Yes	No
Zhu [41]	Automatic	Paper documents	No	Merge-Update	Non-pictorial: shape features (turning functions estimated from the contour pixels)	No	Yes
Willis [30]	Automatic	3D symmetric objects	Yes	Merge-Update	Non-pictorial: shape features (axis/profile curve of a fragment) and 3D points of the fracture curve	Partially	Yes
Papaioannou [43]	*	3D free-form objects	No	*	Non-pictorial: 3D points of the outer surfaces	Yes	Yes
Andrews [36]	Automatic	3D symmetric objects	No	Merge	Non-pictorial: shape features (axis of rotation of a fragment) and 3D points of the inner and outer surface fracture curves	Yes	Yes
Kampel [37]	*	3D symmetric objects	No	*	Non-pictorial: 3D points of the fragments' profile curves	Yes	Yes
Huang [29]	Automatic	3D free-form objects	Yes	Merge-Update	Non-pictorial: shape features (volume descriptor, volume distance descriptor and deviation descriptor) estimated from the fracture surfaces	No	Yes
Lu [28]	Semi-automatic	3D free-form objects	No	Merge-Update	Non-pictorial: shape features (curvature and torsion) estimated from the fracture curves	No	User-def.
Papaodysseus [2]	Semi-automatic	2D objects	No	Merge-Update	Non-pictorial: pixels of the contours	No	Yes
Aminogi [38]	*	2D objects	No	*	Non-pictorial: color and shape features (curvature) estimated from the contour pixels	Yes	++
Leitao [27]	+	2D objects	No	+	Non-pictorial: shape features (curvature) estimated from the contour pixels	Yes	No
Yao [31]	Automatic	2D puzzle pieces	Yes	Merge	Pictorial (color on narrow strips) and non-pictorial (gap area)	No	Yes
Sagiroglu [33]	Automatic	2D puzzle pieces	No	Merge-Update	Pictorial (color and texture)	No	Yes
Nielsen [34]	Automatic	2D puzzle pieces	No	Merge	Pictorial (color)	Yes	No

Table 1.1: Characteristics of the reviewed object reconstruction methods. Methods marked with '*' can reconstruct objects that consist of only two fragments. The method marked with '+' identifies only the matching contour segments of two 2D fragments. The method marked with '++' performs fragments registration in a subsequent step. Abbreviations: reg.=registration, reconstruc.= reconstruction, preproc.=preprocessing.

axis of rotation and the profile of sherds) and points from the fracture curves in the outer surface of the fragments. The matching for every pair of fragments is identified in two-phases; first, an affine transformation is found that aligns the points from the fracture curves of the two fragments. The break points (Figure 1.5) of the 3D fragments are utilized to speed up the matching process. The latter are manually located in every fragment. Thus, given a pair of fragments, two of their corresponding break points, one from each fragment, are matched. Then, an affine transformation is found that aligns the two fracture curve segments located in a small neighborhood around each break point, using a Maximum Likelihood Estimation (MLE) [44] approach. If, after the alignment, the fragments overlap significantly, the matching is rejected. The above procedure is repeated for every pair of break points of the two fragments. In the second phase, the alignments are refined. This step takes into consideration both the points from the fracture curves, the axis of rotation and the profile curve (i.e., the silhouette of the pot when seen from the side). Each refined matching is then assigned a score. The overall object reconstruction follows the merge-update method. After matching and alignment, fragments are heuristically selected and merged into bigger fragments.

In [43] and [37], two automatic methods are proposed for matching and alignment of pairs of 3D, free-form and rotationally symmetric objects, respectively. In [43], the matching is performed through a non-pictorial method whose basic concept is that, given two 3D models, the best fit is likely to occur at their relative position and orientation, which minimizes the point-by-point distance between their mutually visible fracture surfaces. For this reason the authors introduce an error measure of the complementary matching between two fragments at a given relative pose, based on this point-by-point distance. Matching-alignment is performed by minimizing this measure through simulated annealing [45]. It must be noted that in the resulting matching-alignment the fragments may overlap. The same problem is considered in [37] for fragments coming from a rotationally symmetric object. Fragments that are candidates for matching are pre-aligned by aligning their axis of rotation, thus reducing the matching and alignment problem to that of finding a rotation (around the axis of rotation) and a translation value. The best matching-alignment is found by finding the relative pose that minimizes the point-by-point distance of the points lying on the profile curves of the fragments.

Andrews *et. al.* [36], propose an automatic method for the reconstruction of pairs or triplets of 3D symmetric archaeological fragments. No fragments pre-processing is performed. The matching is found through a non-pictorial two-phase method. During the first phase several matchings-alignments are estimated for every pair of fragments. The 3D points of the fracture curves in the outer and inner surface of the fragments as well as the axis of rotation of the fragments are utilized to this end. In the second phase, these matchings are refined using the quasi-Newton method, and evaluated according to several criteria namely the angle formed by the fragments rotation axes, the perpendicular distance between the rotation axes and the distance of the matched fracture curves points. Eventually, one matching is retained for every pair of fragments. The aligned fragments are not checked for overlapping. Finally, in the overall object reconstruction step, a greedy merge strategy selects pairs of fragments to form triplets.

Huang *et. al.* [29], proposed an automatic method for the reconstruction of free-form archaeological fragmented objects. In the pre-processing step, the boundary surface of the input fragments is segmented into a set of surfaces bounded by sharp curves. These are classified into original surfaces, which come from the boundary surface of the unbroken object (and are in general smoother), and fracture surfaces, which were created when the object broke. The segmentation and classification are performed through multi-scale edge extraction and normalized graph cut algorithms. After this step, only information extracted from 3D points of the fracture surfaces is utilized for the matching and alignment of the fragments. In the second step, several shape descriptors [46], namely the volume descriptor, the volume distance descriptor and the deviation descriptor are estimated from the points lying on the fracture surfaces. Subsequently, the points of the fracture surfaces are clustered into feature clusters. Each feature cluster is a cluster of points with similar descriptor values and is represented by a set of (i) single-valued parameters, (ii) vectors (the principal axes evaluated through principal component analysis on the x, y, z data of the cluster points) and points (e.g., the barycenter of the points in the cluster). These feature clusters are used to perform fracture surfaces matching and alignment. The matching dictates the correspondences between feature clusters (point clouds) of two fragments. The fragment matching and alignment is performed in two phases. Coarse feature cluster correspondences are initially found and then refined. A matching of a pair is rejected if the fragments overlap after the alignment. The reconstruction of the overall object is performed through a merge-update approach. The matches and relative alignment of fragments are modelled by a graph, [47]. Using a greedy approach this graph is partitioned and the fragments of each resulting sub-graph are merged into one fragment. After that, the entire reconstruction procedure is repeated using the new set of fragments.

In [28], a human-supervised collaborative reconstruction system is described. The aim of the system is to propose a potential matching between any pair of input fragments. The matching is found by integrating shape features (curvature and torsion) estimated from all 3D points in the fracture curves of the fragments under a cyclic distance algorithm [48]. Then the users select to merge or not the proposed fragments. The fragments alignment is performed interactively (in a VRML environment) by the users. The object reconstruction procedure follows the merge-update paradigm.

1.3.3 2D Puzzles Reassembly

In [31], a method for automatic 2D jigsaw puzzles reassembly is proposed. This three-step method, pre-processes input puzzle pieces in order to limit the matching possibilities between pairs of pieces. Thus, each piece is classified into three categories; corner, boundary, or interior. After that, non-pictorial matching-alignment is performed, in order to identify the correspondences of the pixels in the contours of puzzle pieces. The matching criterion is to minimize the gap area formed after the pieces alignment. The best adjacent candidates (those that are best matched and aligned) are further matched using pictorial criteria. In order to judge the quality of matching of two pieces along one of their sides, the following statistical measure is being evaluated on two narrow strips Φ_1, Φ_2 (one on each piece) that extend from each side towards the interior of the piece:

$$1 - \frac{\sigma_b^2}{\sigma_T^2} \tag{1.1}$$

where

$$\sigma_b^2 = n_1(\bar{P}_1 - \bar{P}_m)^2 + n_2(\bar{P}_2 - \bar{P}_m)^2 \tag{1.2}$$

n_1, n_2 are the number of pixels in regions Φ_1, Φ_2 respectively, $\bar{P}_1$, $\bar{P}_2$, $\bar{P}_m$ are the average R,G or B color values in Φ_1, Φ_2 and $\Phi_1 \bigcup \Phi_2$, and σ_T^2 is the R, G or B color channel variance in $\Phi_1 \bigcup \Phi_2$. This measure (taken from [49], which utilizes $\frac{\sigma_b^2}{\sigma_T^2}$) takes values close to one if Φ_1, Φ_2 are very similar and can be merged and close to zero when they are very dissimilar. Only adjacent candidates that result in large (close to one) values of this statistic are retained. In the final step, the puzzle is reassembled using a greedy-like approach. One piece at a time is added to the puzzle. This piece is the best adjacent candidate of one or more pieces added in the previous iteration.

In [33], an automatic approach that exploits only pictorial information for the matching of pieces is proposed. More precisely, each puzzle piece is expanded in a band around the border of the piece by predicting the pictorial information in this band. Inpainting [35] techniques are used for this task. The method does not perform pair-wise comparisons in order to identify matches. Instead, a global puzzle reassembly procedure takes place. Local mean and variance values are estimated for every pixel of the original and the predicted part of the piece, and a feature image is created by assigning to each pixel a 2-D vector containing these values. Feature images are registered using an FFT-based correlation maximization approach. Each registration is assigned a score equal to the correlation between the predicted parts of one piece and the parts (either predicted or original) of the other piece. In this registration the puzzle pieces must not overlap. Each time the best registered- with respect to pieces already added in the puzzle- piece is added.

In [34], a method that utilizes color features is proposed. The method relies on the observation, that if two puzzle pieces are adjacent in the puzzle, their pictorial information in an area very close to the matching contours of the two puzzle pieces is approximately the same. The overall puzzle reassembly is performed using the merge-like approaches proposed in [20], [23].

1.3.4 2D Objects Reassembly

In [2] a system for the semi-automatic reconstruction of the murals of the Greek island of Thera is described. The mural fragments are photographed and so the problem of mural reconstruction is mapped to the 2D space. Given a pair of fragments having a certain relative orientation, the system estimates a possible matching (i.e., contour pixels correspondences) using only the pixels from the fragments' contours. This matching is evaluated according to several criteria such as the number of overlapping pixels or the number of pixels lying in the gap after aligning the fragments. The latter is related to the translation of the fragments, since their relative orientation is considered fixed. After that, their relative orientation changes by a small amount and the procedure described above is repeated. The best matching is the one which optimizes the criteria previously described. The user selects the matches-alignments that will be retained and merges the corresponding fragments. The new fragments are added in the set and the procedure is repeated from the beginning.

In [27], an automatic, time-efficient non-pictorial matching approach is presented. The fragments' contours are initially sampled in a coarse scale. The curvature of the coarse contours is estimated and a dynamic programming algorithm is employed in order to identify contour pixels correspondences. Each pair of matching contour segments is assigned a two-term quadratic mismatch cost. The first term estimates the total difference between the curvatures, while the second one is used to penalize matches that are too irregular. A pair of contour segments are considered to match if this mismatch cost is less than a user defined threshold. The pairs of candidate matching segments are mapped back to the original contours which are sampled using a finer scale this time, and the dynamic programming technique is employed again.

In [38] the matching of the image fragments is based on both the shape and the color characteristics of the fragments contours. One contour pixel sequence is overlaid on another one and, for each such placement, the curvature and color differences of the corresponding contour pixels are estimated. If their total sum is less than a user defined threshold, the contour segments are considered to match. After that, their alignment is found using a brute force approach that searches all possible alignment angles between the two fragments.

As it is obvious, the literature on 2D object reassembly is rather limited. A novel contribution towards this direction will be presented in the next Section.

1.4 Automatic Color Based Reassembly of Fragmented Images and Paintings

In this Section a novel automatic four-step method for the reassembly of torn 2D paper images is presented. The identification of possibly adjacent image fragments is the first step. It is considered that two fragments were adjacent in the original image if the color similarity of the depictions on their outer surfaces is high. As a result of this step, for each image fragment, the L fragments having the highest color similarity with it, are selected. Subsequently, the matching contour segments for every pair of potentially adjacent input fragments are identified. The matching algorithm is non-pictorial and utilizes color information within a dynamic-programming framework [50]. At the end of the second step, for each fragment, its matching contour segments with K other fragments are retained. The third step aims to align each fragment with respect to its adjacent ones along their matching contour segments, using a variant of the ICP algorithm. Once the matching contour segments of pairs of image fragments are identified and aligned, the reassembly of the overall image takes place. To this end, a novel feature, namely the alignment angles found during the previous step, is introduced and a merge-like paradigm is employed. The main steps of the proposed method are shown in Figure 1.6.

1.4.1 Discovery of Adjacent Image Fragments

Let $\mathcal{F} = \{f_1, f_2, \ldots, f_N\}$ be the set of N image fragments. The aim of this step is to speed up the overall image reassembly process by reducing the number of pair-wise fragment comparisons that take place in the

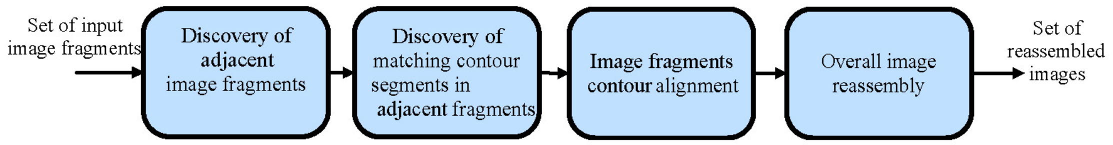

Set of input image fragments
Discovery of adjacent image fragments
Discovery of matching contour segments in adjacent fragments
Image fragments contour alignment
Overall image reassembly
Set of reassembled images

Figure 1.6: The steps of the automatic color based paper image reassembly approach proposed in [24].

second step. It is assumed that the color similarity of the depictions on the outer surfaces of the fragments that were adjacent in the original image is high. Thus, a matching approach based on color similarity is used to identify spatially adjacent image fragments. In particular, color quantization is performed on every image fragment f_k using the Gretag Macbeth Color Checker palette [51]. This palette consists of twenty-four colors chosen so as to represent a variety of naturally occurring colors. After performing color quantization, the normalized color histograms of the image fragments in $\mathcal{F}$ are estimated. These normalized color histograms are compared with the histogram intersection similarity measure, [51]:

$$d_{HI}(h_k, h_l) = \sum_{i=1}^{24} \min(h_k(i), h_l(i))(1 - |h_k(i) - h_l(i)|) \tag{1.3}$$

where h_k and h_l denote the normalized color histograms extracted from image fragments f_k and f_l, respectively. $h_k(i)$ denotes the value of the i-th bin of h_k and is defined as the number of pixels having color i divided by the total number of pixels. Once this step is finished, a list of the L most chromatically similar fragments is retained for every image fragment. It must be emphasized that this step is optional and can be skipped, especially if the number of image fragments is rather small.

1.4.2 Discovery of Matching Contour Segments of Adjacent Image Fragments

In this step, a novel algorithm is utilized for identifying the matching contour segments of pairs of input fragments. Note that this step is performed only for the fragment pairs identified in the previous step or for all pairs of fragments, if the first step has been skipped. In both cases, we denote the set of image fragment pairs by $\mathcal{E}$. The utilized fragment matching algorithm is non-pictorial: it utilizes information exclusively regarding the color of fragments' contours.

A color quantization preprocessing step is initially performed on the contour pixels of the image fragments, in order to avoid comparing directly pixel colors that may contain noise. Kohonen Neural Networks (KNN) [52] are employed for this purpose. KNNs consist of an input and an output node layer. In the input layer, the number of nodes equals the dimension of input vectors, while the number of nodes in the output layer is equal to the number of produced clusters.

In order to perform color quantization of the contour pixels of the image fragments by means of the KNNs the following steps take place; Initially, a number of N_p pixels is randomly sampled from the image fragments in $\mathcal{E}$ and mapped to La^*b^* color space. N_p is empirically set to 25% of the total image fragment pixels [24]. After that, a $[3 \times C]$ KNN is defined, where 3 corresponds to the dimension of the input La^*b^* space and C to the predefined number of color clusters and the KNN iterative learning procedure is applied on the subset of N_p pixels. After training the network, the color of every contour pixel is represented by its color cluster label c_j, $j = \{1, \ldots, C\}$.

Let $U = [u_i]_{i=1}^{n}$ and $V = [v_j]_{j=1}^{m}$ be two sequences that correspond to the contours pixels of two different image fragments. Let also a_i, b_j denote the quantized colors assigned to pixels u_i, v_j respectively. We define a similarity function F between two contour pixels as follows:

$$F_{u_i,v_j}[a_i, b_j] = \begin{cases} e > 0, & a_i = b_j \\ d < 0, & a_i \neq b_j \end{cases} \tag{1.4}$$

Our goal is to find a contour pixel mapping function Φ, given U and V, that satisfies the following conditions:

- $\Phi[u_i] = v_k$ and $\Phi[u_{i+1}] = v_l, \forall i, k \leq l \leq m$.
- $\Phi[u_i] \neq \emptyset$.

The first condition implies that more than one contour pixels in U can be mapped to the same contour pixel in V. However, pixels in contour U that are mapped in the same pixel in $\mathcal{V}$ must be strictly consecutive. This condition guarantees that no "folding" of the U contour pixel sequence will occur during its matching with V. The second condition ensures that every contour pixel in U is mapped to a contour pixel in V. The algorithm that is used to identify the mapping function Φ is a variant of the Smith Waterman dynamic programming algorithm [50] . Each mapping between segments of input sequences is assigned with a score $S > 0$. Large values of S imply high color similarity. A $n \times m$ similarity matrix $\mathbf{H}$ is set up, where the i^{th} row of matrix $\mathbf{H}$ corresponds to u_i, while the j^{th} column corresponds to v_j. $H_{i,j}$ is the best mapping score S of the pair of all sub-sequences ending at pixels u_i and v_j. The algorithm gradually fills matrix $\mathbf{H}$ and forms the mapping function Φ. During filling, each matrix cell is assigned with the highest possible value, as our purpose is to maximize the mapping score S.

Being a dynamic programming algorithm, the solution to an instance of the problem is given in terms of solutions to its smaller sub-instances. Thus, matrix $\mathbf{H}$ is recursively filled-up using the formula

$$H_{i,j} = \max \left\{ H_{i-1,j-1} + F_{u_i,v_j}(a_i, b_j), H_{i-1,j} + g, H_{i,j-1} + g, 0 \right\}, \tag{1.5}$$

where $g < 0$.

Let Φ be the estimated mapping of $[u_k]_{k=1}^{i-1}$ and $[v_l]_{l=1}^{j-1}$ subsequences. If $H_{i,j} = H_{i-1,j-1} + F_{u_i,v_j}[a_i, b_j]$, then the relation $\Phi[u_i] = v_j$ is appended to Φ. On the other hand, if $H_{i,j} = H_{i,j-1} + g$ or $H_{i,j} = H_{i-1,j} + g$ is selected, then $\Phi[u_i] = v_{j-1}$ or $\Phi[u_{i-1}] = v_j$ is appended to Φ. A *gap* is formed, when one of these last two cases occur. Generally, a gap is formed when one or more contour pixels of the first fragment are mapped to the same contour pixel of another fragment. The percentage of mapping gaps is a measure of the dissimilarity between the input contour segments; a high gap percentage reveals contours with many dissimilarities. The parameter g is negative, in order to penalize mappings with many gaps. When this stage is completed, we identify an area in $\mathbf{H}$ with high similarity values H_{ij}. Diagonal patterns are preferred, since they correspond to mappings without many gaps. Let H_{e_1,e_2} and H_{s_1,s_2} be the lowest right and highest left borders of this area. In the implemented modification, H_{e_1,e_2} is the maximum value of $\mathbf{H}$, while $H_{s_1,s_2} = 0$. Then we select the mapping starting from (s_1, s_2) and ending at (e_1, e_2), which is formed during filling $\mathbf{H}$. In summary, the steps in order to identify the matching contour segments of two image fragments are the following:

- Let $U = [u_i]_{i=1}^{n}$ and $V = [v_j]_{j=1}^{m}$, be the corresponding contour pixel sequences of the two fragments. Quantize the contour pixel colors of the two image fragments using a KNN (note that this step is performed only once for all image fragments in $\mathcal{E}$). Let $[a_i]_{i=1}^{n}$, $[b_j]_{j=1}^{m}$ be the quantized colors sequences assigned to U and V respectively.
- Initialize $H_{i,j}$ according to $F_{u_i,v_j}[a_i, b_j]$.
- Fill $\mathbf{H}$ according to Equation (1.5).
- Select an area in matrix $\mathbf{H}$, as previously discussed. Let H_{e_1,e_2} and H_{s_1,s_2} be the lowest right and highest left borders of this area. Select the mapping starting from (s_1, s_2) and ending at (e_1, e_2).

Details on how to select e, d and g are presented in [24]. An example of a similarity matrix is presented in Figure 1.7(a). The horizontal axis of the matrix corresponds to the quantized colors of the contour pixels sequence of the image shown in Figure 1.7(b), while the columns correspond to the quantized colors of the

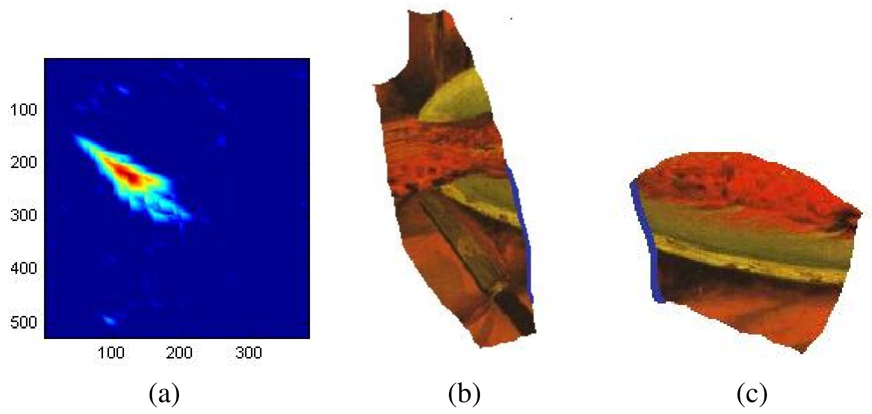


Figure 1.7: Color similarity matrix (a) for the contour sequences of two fragments (b), (c). The correctly matched contour segments are shown in blue in (b), (c).

contour pixels sequence of the image shown in Figure 1.7(c). High values correspond to contour segments with a high degree of similarity (deep red colored areas). The correctly matched contour sequences are shown with blue color in Figures 1.7(b) and 1.7(c). At the end of the second step, for each image fragment f we retain one matching contour segment with K $(0 < K \leq L)$ other image fragments, producing the set of $true$ pairs of adjacent fragments $\{(f, f_i), i = 1, \ldots, K\}$.

1.4.3 Contour Alignment of Fragments

Since the previous step only identifies the matching of the contour segments of image fragments, their alignment transformation must be found. A popular algorithm that is used for the registration of two point sets is the ICP algorithm [39] . The ICP starts with two point sets (in our case pixels from the two contour segments) and an initial guess of their relative rigid body geometrical transformation. It then refines the transformation parameters, by iteratively generating pairs of point correspondences and minimizing the sum of the L_2 distances between points from the two point sets. The original form of the ICP algorithm is not robust to outliers coming from errors during the contour segment matching. These outliers may create serious problems to alignment, if not properly handled.

In order to develop a robust contour alignment procedure, three popular, robust to outliers, variants of ICP, namely [53], [54], [55], have being evaluated. The evaluation of these variants was done by applying them for the alignment of correctly matched contour segments (i.e., contour segments that really match in the original image). These contour matchings may contain misidentified pixels or erroneously non-identified contour pixels. The experiments detailed in [24] have shown that the ICP variant proposed in [55], called ICPIF (ICP using Invariant Features), outperforms the other two variants. ICPIF introduces new features, such as the second order moments and the spherical harmonics, in order to improve the correspondence selection. After the computation of the above features for every point of the two contour segments, the L_2 distance, computed during the second step of ICP, is replaced by the weighted sum of the L_2 distance of the two point sets and distances based on the introduced features. The goal is to estimate point correspondences that are not only based on the Euclidean distance but also incorporate shape invariant features.

1.4.4 Overall Image Reassembly

Once the true adjacent image fragments pairs and their matching contour segments are identified and properly aligned, the final step is the reassembly of the overall image. Approaches that utilize only the mapping scores, such as [22], are inadequate for producing the correct overall image, since the mapping scores of the matching contour segments are not always well correlated with the true ones. Thus, a novel solution has been developed, which is based on the alignment angle, that best aligns the matching contour segments of two image fragments. This approach follows the merge paradigm.

Consider three image fragments f_i, f_j and f_k, where f_i has a matching contour segment with f_j and f_k. We denote by θ_j the rotation angle by which the individual fragment f_j must be rotated, in order to be correctly placed inside the overall reassembled image. The alignment angle, by which we must rotate fragment f_i to align it with the matching contour segment of fragment f_j (before fragment f_j is rotated by θ_j), is denoted by θ_{ij}. In order to align fragments f_i and f_j with respect to each other and place them correctly in the reassembled image, the following steps must be performed:

1. Rotate fragment f_j by θ_j to correctly orient it in the assembled image.
2. Rotate fragment f_i by $\theta_{ij} + \theta_j$ to correctly align its matching contour segment with the corresponding matching contour segment of fragment f_j.

This procedure will simultaneously align fragment f_i with fragment f_j and provide its correct orientation inside the entire image. We can then state that the matching contour segments of pairs (f_i, f_j) and (f_i, f_k) are *compatible*, if and only if:

$$\theta_{ij} + \theta_j = \theta_{ik} + \theta_k. \tag{1.6}$$

Following this rationale, it is considered that an image is fully reconstructed, if all its matching contour segments are compatible, according to Equation (1.6); this image is called *valid*.

Based on Equation (1.6), the so-called relative alignment angle is defined, which will direct the multiple fragments merging procedure. The relative alignment angle ϕ_i^j of a fragment f_i with respect to a fragment f_j is evaluated by the formula:

$$\phi_i^j = \theta_{ij} + \theta_j, \tag{1.7}$$

where θ_{ij} and θ_j are defined as above. It is clear, that for a set of fragments placed in the reassembled image and a fragment f_i having matching contour segments with a subset of the above image fragments (i.e., $I = \{f_1, f_2, \ldots, f_n\}$) the above matching contour segments are compatible if:

$$\phi_i^1 = \phi_i^2 = \phi_i^3 = \ldots = \phi_i^n. \tag{1.8}$$

Consequently, if a new fragment i is to be matched with a partially reassembled image that consists of image fragments, then the number of the new valid partially reassembled images that will be generated is equal to the cardinality r of the relative alignment angle set, $\{\phi_i^l, l = 1, \ldots, r\}$. We have developed a reassembly algorithm that utilizes the aforementioned assumptions to produce a number of possible reassembled images, all being potential solutions to the reassembly problem. The algorithm is initialized by taking M pairs of input fragments that have the highest mapping scores for the corresponding matching contour segments, where $M \leq N * K$ and N is the number of total input image fragments. These pairs are reassembled (correctly aligned) to produce a number of M partially reassembled (two-fragment) images that are further extended by inserting one fragment each time. The selection of the initial M fragment pairs is crucial for the algorithm performance, since the involvement of an erroneous input pair would inevitably lead to a wrong image reconstruction. The reliability of the proposed initialization is both intuitionally expected and experimentally proven.

The set of M partially reassembled images is iteratively updated in order to include further image fragments. At every step and for every partial image (initially consisting of a two image fragments), the fragment having the maximum mapping score with an image fragment that belongs in the reconstructed image, is added. After computing the relative alignment angles of this fragment with respect to the partially reconstructed images, a number of images, equal to the cardinality of these relative alignment angles, are reassembled for each partial image. Those images compose the new set of partially reassembled images (replacing the previous ones), that will be included in the next iteration. Consequently, in the next step a

Table 1.2: Reassembly performance and computational complexity of the proposed method.

	Reassembly performance	Computation time
Without first stage	63.49%	7.58 min
Including first stage	42.15%	3.42 min
Human	-	17.37 min

new fragment will be involved in the update procedure of these images. This iterative algorithm terminates when no more image fragments are left to be inserted into the reassembled images.

The proposed method differs from others found in the literature, as it is based on the alignment angles. Thus, it is sensitive to errors in the estimation of these angles. However, this drawback is ameliorated, by setting a threshold when comparing the relative alignment angles. Thus, two relative alignment angles ϕ_1 and ϕ_2 are regarded to be equal if $|\phi_1 - \phi_2| < \epsilon$. Another "backup" approach is to add manual intervention to the system, and ask the user to select the correct alignment transformation.

1.4.5 Image Reassembly Experiments

The performance of the proposed method was evaluated using 70 paper image prints. Each print had size 25 cm $\times$ 20 cm and was torn by hand into $N = 20$ pieces that were scanned at 300 dpi resolution, in order to create a challenging image fragments dataset. An example is shown in Figure 1.8. Experiments were conducted in a quad-core PC with 4GB RAM and the algorithm has been implemented in C++. For the KNN color quantization in the second step, the number N_p of sampled image pixels was set equal to 25% of the total pixels. The utilized Kohonen nets resulted into $C = 50$ color clusters, while the learning procedure took 750 epochs. Nodes in the output layer of the neural network were organized under a random lattice. The parameters of the Smith Waterman algorithm were set to $e = 1$, $d = -0.5$ and $g = -0.5$. At the end of the second step, for each image fragment, matching contour segments with $K = 10$ fragments were retained.

The performance of the method was estimated as follows. Let I be a manually reassembled image and $\mathcal{I} = \{I_1, I_2, \ldots, I_n\}$ be the set of automatically reassembled images generated by the algorithm for this image. The reassembly is defined to be correct, when there is a $\widetilde{k} \in \{1, \ldots, n\}$ for which $I_{\widetilde{k}}$ is declared similar to I by a human observer. In Table 1.2, the reassembly performance, namely the percentage of the test images that were correctly reassembled according to the above definition, and the computational time of the presented approach, for the 70 test images are shown. The last row shows the mean manual reassembly time (the accuracy of manual reassembly was not measured). Eight people of ages between 23 to 31 participated in this experiment. The average time needed for each person to correctly reassemble each fragmented image was evaluated. As expected, the overall performance is higher when the first stage is omitted. However, in this case the computation cost is higher, since both the second and the third step are executed for every pair of input image fragments. Table 1.2 shows that the choice whether to employ the first step on the reassembly process depends on the computational cost that is associated within the input image fragments. As the amount of image fragments increases, the higher overall performance of the variation that omits the first step gives way to the higher computational cost that is associated with it. Figure 1.9 shows correctly aligned couples of image fragments produced during the second and the third steps of the proposed algorithm. The image resulting from the reassembly of fragments shown in Figure 1.8 is shown in Figure 1.10. It can be seen that its reconstruction is nearly perfect. The white region in the middle of the reassembled image is due to missing pieces of paper that were not scanned.

In the future, we plan to further improve the performance of the proposed method. Improvements can be introduced in each step of the method. For example, the procedure for the discovery of spatial adjacent image fragments may be improved by employing not only color but also textual or semantic features. Another possible extension is to utilize both the color and the shape of the fragments contours in order to perform matching. The evaluation of the proposed method in a real world fragmented image database, such as a

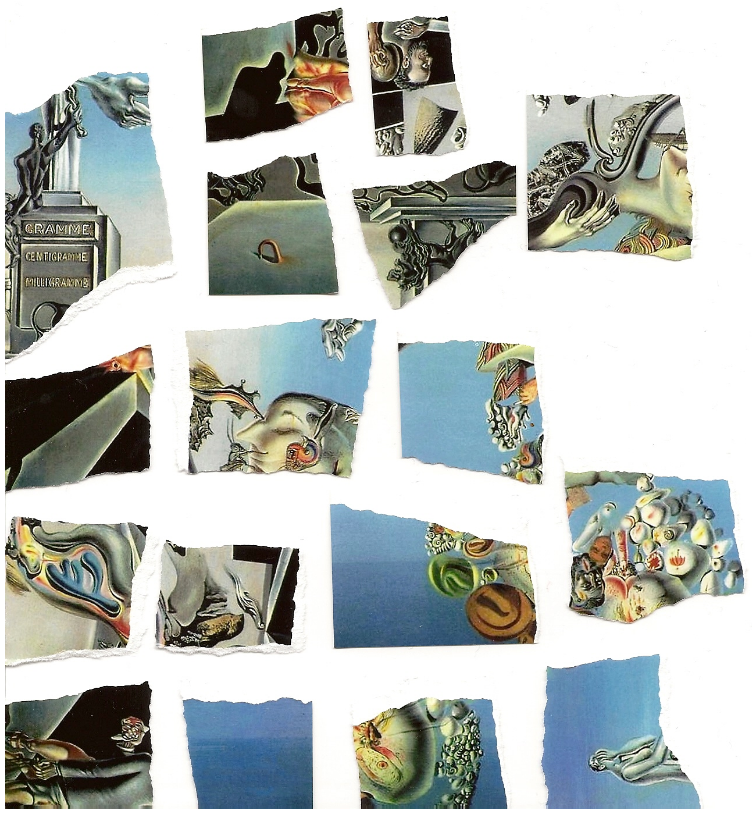


Figure 1.8: A torn by hand image used in the experiments.

database with archaeological data, is worth exploring.

1.5 Reduced Complexity Image Mosaicing Utilizing Spanning Trees

As already mentioned, the mosaicing process comprises of two steps. The first step involves the estimation of the optimal transformation (translation, rotation, scaling etc) that aligns each sub-image with respect to each neighboring one, whereas the second step utilizes displacement information found in the previous step in order to combine each pair of neighboring sub-images with invisible seams and reconstruct the entire image.

In this Section we will describe two methods that aim to reduce the number of computations required for the evaluation of the sub-image displacements, without significantly affecting the matching error. It is important to note that, despite the fact that the methods are illustrated for the particular case where sub-images are only translated with respect to each other, the proposed matching methodology can be easily applied to more complex situations (e.g., cases that involve camera rotation or zooming during the acquisition of the sub-images). Before dealing with the general case of mosaicing an arbitrary number of sub-images, the case of two images will be studied in the next subsection, since it provides significant insight to the problem.

1.5.1 Two-Image Mosaicing

Let us assume that the displacement vector $\mathbf{d}$ is constrained to take values in the following set:

$$\mathbf{d} \in \{[d_1\ d_2]^T : \ d_i \in \{d_{i_{\min}}, \ldots, d_{i_{\max}}\}, \ \ i = 1, 2\}. \tag{1.9}$$

If $I_j(\mathbf{n})$, $j = 1,\ 2$, denotes the intensity of the j-th image at pixel coordinates $\mathbf{n} = [n_1\ n_2]^T \in W(\mathbf{d})$, where $W(\mathbf{d})$ denotes the area where the two images overlap (Figure 1.11), then the matching error $E(\mathbf{d})$

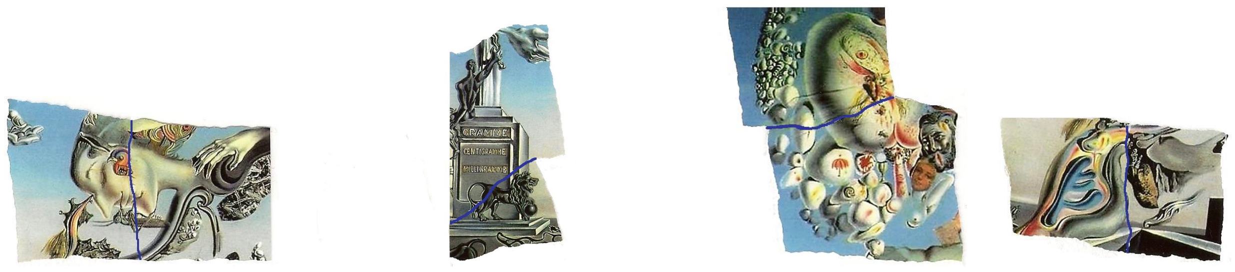

(a) (b) (c) (d)

Figure 1.9: New fragments created by aligning and assembling original fragments along their matching contour segments.

associated with a specific displacement $\mathbf{d}$, can be expressed as follows:

$$E(\mathbf{d}) = \frac{\sum\limits_{\mathbf{n}\in W(\mathbf{d})} |I_1(\mathbf{n}) - I_2(\mathbf{n})|^p}{||W(\mathbf{d})||} \tag{1.10}$$

where $||W(\mathbf{d})||$ denotes the number of pixels in the overlap area $W(\mathbf{d})$. For $p = 1, 2$ Equation (1.10) expresses the Matching Mean Absolute Error MMAE and the Matching Mean Square Error MMSE, respectively. Subsequently, the optimal displacement value $\mathbf{d}_{opt}$ can be estimated through the following minimization:

$$\mathbf{d}_{opt} = arg\min_{\mathbf{d}} E(\mathbf{d}). \tag{1.11}$$

From (1.9) and (1.11) it is obvious that this minimization process requires the evaluation of (1.10) over all possible values of $\mathbf{d}$. Block matching techniques such as 2D logarithmic search, three-point search, conjugate gradient search [56] can be employed in order to reduce the computational cost associated with the exhaustive minimization procedure of Equation (1.11). These procedures may provide estimates $\hat{\mathbf{d}}_{opt}$ of the optimal displacement value $\mathbf{d}_{opt}$. In our simulations, the 2D logarithmic search was employed. With respect to the error metric, experiments showed that similar results were obtained by using either the MMAE or the MMSE criterion. Thus, MMAE was used since it is faster to compute.

1.5.2 Spanning Tree Mosaicing

If $M_1 \times M_2$ sub-images are to be mosaiced, a displacement matrix $\mathbf{D}$ is involved. This matrix plays a role similar to that of the displacement vector $\mathbf{d}$ mentioned in the previous sub-section. The $2M_1M_2 - M_1 - M_2$ columns of $\mathbf{D}$ are two-dimensional vectors, each corresponding to a displacement value between two neighboring sub-images. An expression for the matching quality, similar to the two-image case, can be derived in the multiple image case, by substituting $\mathbf{d}$ with $\mathbf{D}$ in Equation (1.10) and extending the summation over all neighboring images. The optimal value $\mathbf{D}_{opt}$ of the displacement matrix can then be derived as follows:

$$\mathbf{D}_{opt} = arg\min_{\mathbf{D}} E(\mathbf{D}). \tag{1.12}$$

The minimization in the Equation above involves prohibitive computational complexity, since in this case a much larger search space is involved. Indeed, if each of the column vectors in $\mathbf{D}$ is of the form (1.9), $\mathbf{D}$ may assume $\left((d_{1_{\max}} - d_{1_{\min}} + 1)\,(d_{2_{\max}} - d_{2_{\min}} + 1)\right)^{2M_1M_2 - M_1 - M_2}$ different values. Thus, computational complexity increases exponentially with respect to the number of sub-images. Moreover, calculation of $E(\mathbf{D})$ poses additional computational problems, since the overlap area W is now a multi-dimensional set.

In order to avoid exhaustive matching, certain constraints can be imposed on the way images are matched. Indeed, a faster method may be devised by performing simple matches only (i.e., matches between

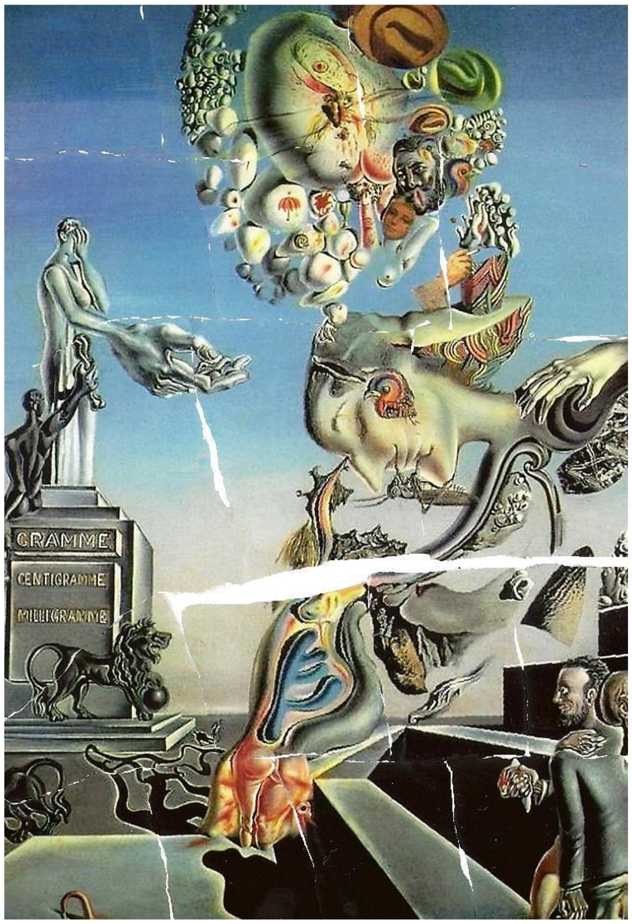


Figure 1.10: The result of reassembling the image depicted in Figure 1.8.

an image and one of its neighbors). The proposed method can be illustrated with the aid of a mosaicing example. In Figure 1.12 (a) a mosaic of $M_1 = 2$ by $M_2 = 2$ sub-images is depicted. If one associates each image with a graph node and each local matching of two sub-images with an edge, the mosaicing of the four images can be represented by the graph of Figure 1.12 (b). Computation of $\mathbf{D}_{opt}$ requires an exhaustive search in 8-dimensional space. To reduce this complexity, the entire mosaicing process is decomposed into simpler steps each involving the mosaicing of two images at a time. Spanning trees can provide a representation of the possible mosaicing procedures, under this constraint. Figures 1.12 (c)-(f) illustrate the four different spanning trees that correspond to the graph of Figure 1.12 (b). For example, in the case depicted in Figure 1.12 (d) three pairwise image matches should be performed: image A to C, D to C, and D to B, while Figures 1.12 (c), 1.12 (e), 1.12 (f) illustrate the other three possible mosaicing procedures. The final image is the one obtained by the procedure which results in the smallest matching error. Obviously, this procedure is sub-optimal but offers a significant decrease in computational complexity. The number of spanning trees that correspond to a certain graph can be calculated by the matrix-tree Theorem [57]:

Let G be a non-trivial graph with adjacency array $\mathbf{A}$ *and degree array* $\mathbf{C}$*. The number of the discrete spanning trees of G is equal with each cofactor of array* $\mathbf{C} - \mathbf{A}$.

Both $\mathbf{A}$ and $\mathbf{C}$ are matrices of size $(M_1M_2) \times (M_1M_2)$. If node v_i is adjacent to node v_j, $\mathbf{A}(i,j) = 1$, otherwise $\mathbf{A}(i,j) = 0$. Additionally, the degree matrix is of the form:

$$\mathbf{C} = \mathrm{d}iag(d(v_1), \ldots, d(v_{M_1M_2})) \tag{1.13}$$

where $d(v_i)$ denotes the number of nodes adjacent to v_i. The number of trees increases rapidly with the grid size. For example the numbers of spanning trees for grids of size 3×3, 4×4 and 5×5 are 192, 100352

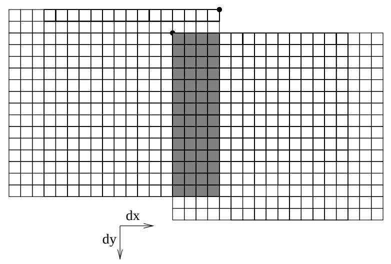


Figure 1.11: Two neighboring sub-images and the associated overlap region $W(\mathbf{d})$ (which is identified by the gray area) for a displacement of $\mathbf{d} = [-4\ 2]^T$.

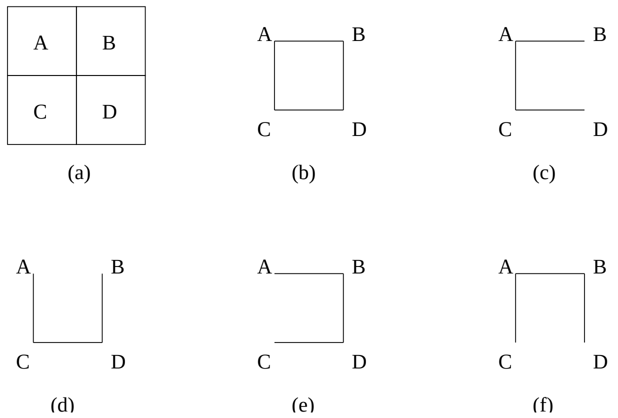


Figure 1.12: (a) A mosaic of $M_1 = 2$ by $M_2 = 2$ sub-images, (b) the corresponding graph, (c)-(f) the four possible spanning trees.

and 5.6×10^8 respectively.

The proposed spanning tree mosaicing (STM) procedure is outlined below:

1. For each pair of neighboring images calculate the optimal displacement and the associated matching error.
2. For each spanning tree that is associated with the specific graph, calculate the corresponding total matching error, by summing the local matching errors which are associated with the two-image matches represented by the given tree.
3. Select the tree associated with the smallest total matching error.
4. Perform mosaicing of two images at a time, following a path on the selected spanning tree.

It should be clarified once again that sub-optimal results are obtained by the STM procedure (i.e., only an approximation $\hat{\mathbf{D}}_{opt}$ of the optimal matrix is computed). However, this is compensated by the significant speed gains provided by the algorithm.

1.5.3 Sub-Graph Spanning Tree Mosaicing

As already mentioned, the number of trees grows very fast with respect to grid size and thus for large values of M_1 and M_2, STM becomes computationally demanding. The second approach that is proposed, namely the *Sub-graph STM (SGSTM)* may, partially, address this issue. In SGSTM, a graph is partitioned into sub-graphs, by splitting the original graph vertically and/or horizontally. A sample partitioning of this type is

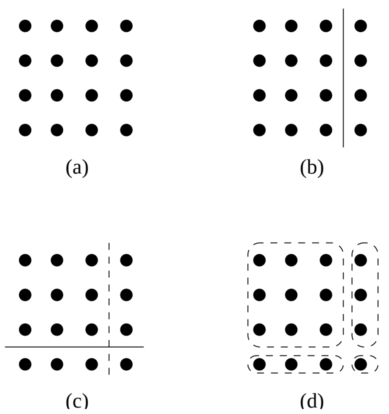


Figure 1.13: (a) A graph corresponding to an $M_1 = 4$ by $M_2 = 4$ image mosaic. (b) Vertical split of (a). (c) Horizontal split of (b). (d) Resulting partition of graph.

depicted in Figure 1.13. By splitting the graph vertically and then horizontally, four sub-graphs result. STM can be applied separately to each one of the four sub-graphs of Figure 1.13 (d). It can be shown that in this case a total of 194 spanning trees should be examined. Since four images will be produced by the STM process (one for each sub-graph), a further STM step will be required, in order to mosaic these four images into the final image. Thus, 4 more trees should be added to the trees examined in the previous step, for a total of 198 trees. In contrast, an STM of the original image set would require matching error calculations for 100352 trees.

If the original graph is of size $M \times M$ ($M = 2^\nu$), the image can be gradually mosaiced by decomposing the original graph into an appropriate number of 2×2 sub-graphs, performing STM on each one, decompose once more the resulting $\frac{M}{2} \times \frac{M}{2}$ graphs and so on, until one image emerges. After mosaicing a partition's sub-graphs, new displacement matrices that correspond to the resulting sub-images should be calculated. The number of 2×2 graphs in this procedure is equal to $\frac{M}{2}\frac{M}{2} + \frac{M}{4}\frac{M}{4} + \ldots + 1 = \frac{M^2-1}{3}$. Since four spanning trees exist for a 2×2 graph, the matching error of $4\frac{M^2-1}{3}$ trees should be evaluated. Theoretical speedup values of SGSTM over STM are depicted in Figure 1.14. It is obvious that SGSTM introduces large computational savings over the STM approach. Obviously, the quality of SGSTM mosaicing may be inferior to the one provided by STM, since SGSTM involves the examination of a significantly smaller number of possible sub-image matches. However, SGSTM can be utilized for fast visualization of mosaicing results (e.g., mosaic previews).

1.5.4 Experimental Evaluation

Simulations were carried out in order to assess the performance of the proposed methods, on several image sets. In the following, comments and results are presented for two of these sets.

The first set consisted of the 6 sub-images of Figure 1.2, which were arranged in $M_1 = 3$ rows of $M_2 = 2$ sub-images, each having resolution of 238×318 pixels. For each one of the $2M_1M_2 - M_1 - M_2 = 7$ pairs of neighboring sub-images, matching errors were calculated using the MMAE criterion. Subsequently, for each one of the 15 spanning trees that correspond to this 2×3 node graph, the total matching error was calculated. The reconstructed images for both STM and SGSTM are depicted in Figure 1.15.

The total time required to find all spanning trees that correspond to the given graph, calculate optimal neighboring image displacements, and output the overall matching error of each tree (for both STM and SGSTM) is tabulated in Table 1.3, in seconds. The corresponding matching errors are also presented in this table. In the case of SGSTM, the original 3×2 graph was partitioned into four sub-graphs. Two of them had sizes 2×1, while the last two consisted of one node each.

In the SGSTM case, matching error calculation is required for 6 trees , instead of the 15 trees of the

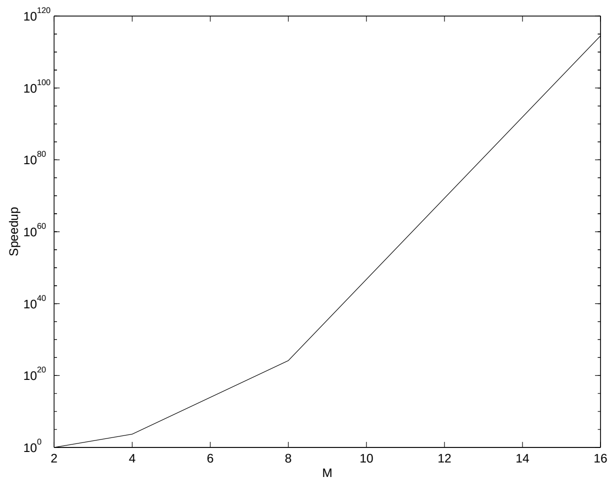


Figure 1.14: Theoretical speedup of SGSTM over STM, for graphs of size $M \times M$ ($M = 2^{\nu}$).

Table 1.3: STM and SGSTM performance results for image set 1.

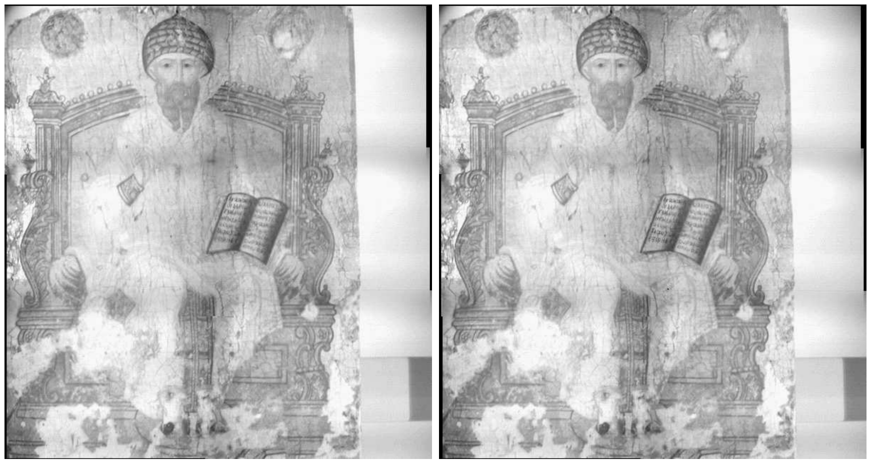
Method	*Error*	*Time*
STM	27.41	2.19
SGSTM	29.60	1.95

STM case. However the approximately 2.5 theoretical speedup factor of SGSTM over STM is not attained, because quoted time figures take into account the amount of time required to re-compute displacement matrices. Due to this overhead a speedup factor of 1.12 was recorded.

The second set consisted of 12 sub-images, which were arranged on a grid of $M_1 = 4$ rows of $M_2 = 3$ sub-images each. Each sub-image had a resolution of 951×951 pixels. For this graph 2415 spanning trees exist. Similar to the previous set, for each one of the $2M_1M_2 - M_1 - M_2 = 17$ pairs of neighboring sub-images, matching errors were calculated and the 2D logarithmic search was utilized in order to obtain the optimal displacement. Subsequently, for each one of the 2415 spanning trees, the total matching error was calculated.

The total time that was required to find all spanning trees, calculate optimal neighboring image displacements, and output the overall matching error of each tree in this case is presented in Table 1.4. In the case of SGSTM, the graph was decomposed into four sub-graphs: two of size 2×2 and two of size 2×1, which required the calculation of the total error for 14 trees, compared to the 2415 of STM. It is evident that, in this image set, SGSTM is more than an order of magnitude faster than STM. More specifically, the speedup factor provided by the SGSTM method over the STM method was 11.4. Of course, this speedup was accompanied by a significant increase in the matching error.

The twenty spanning trees that exhibited the lowest MMAE scores for this image set are depicted in Figure 1.16. By studying these spanning trees as well as the MMAE scores we can conclude that the most characteristic feature of the trees with the lowest error figures is that optimal matching begins from the center and proceeded outwards. In other words, in these trees, the central nodes of the graph were connected. On possible explanation is that the matching quality of the central nodes is more crucial to the overall mosaicing quality, than the matching quality of the other nodes.



(a) (b)

Figure 1.15: (a) Spanning tree mosaicing (STM) reconstructed image. (b) Reconstructed image after sub-graph spanning tree mosaicing (SGSTM).

Table 1.4: STM and SGSTM performance results for image set 2.

Method	*Error*	*Time*
STM	6.4	782.7
SGSTM	9.7	68.9

1.6 Conclusions

Reconstruction of broken or torn artifacts bearing archaeological, historical or cultural importance, such as ceramics, murals, mosaics, papyrus or paper manuscripts, stone tablets etc is a labor intensive procedure when performed manually by experts. Fortunately, computer-based or computer-assisted approaches for the reconstruction of such artifacts have started to emerge during the last few years. This area of research, that lies within the broader field of image processing and analysis and computer vision, is an extremely challenging one, since it has to deal with issues such as high degradation of fragments, missing fragments, lack of a-priori information regarding the correct appearance of the reconstructed object, and so on. As a result, it is expected to attract considerable attention in the years to come, hopefully leading to methods that are highly automated, robust and of general applicability.

Mosaicing or stitching, namely creation of a a single image from a set of overlapping sub-images, is also of high importance in archival, digital analysis and restoration of Cultural Heritage artifacts such as murals or large paintings. Being applicable in other fields, such as consumer photography or aerial, satellite and underwater imaging, mosaicing has attracted the interest of a larger part if the imaging community than the reconstruction of archaeological or cultural artifacts. This led into more solid results, but sufficient room for additional, fruitful research still exists.

In this Chapter, we have attempted to provide a concise review of algorithms for the 2D or 3D reassembly of fragmented or torn objects and have also presented algorithms that we have developed for the automatic reassembly of torn 2D paper images or fragmented paintings and the computationally efficient mosaicing of images. We hope that the content of this Chapter, along with the other Chapters of this book, will be able to stimulate new research in this highly rewarding field, eventually leading into more efficient approaches towards preservation and study of Cultural Heritage.

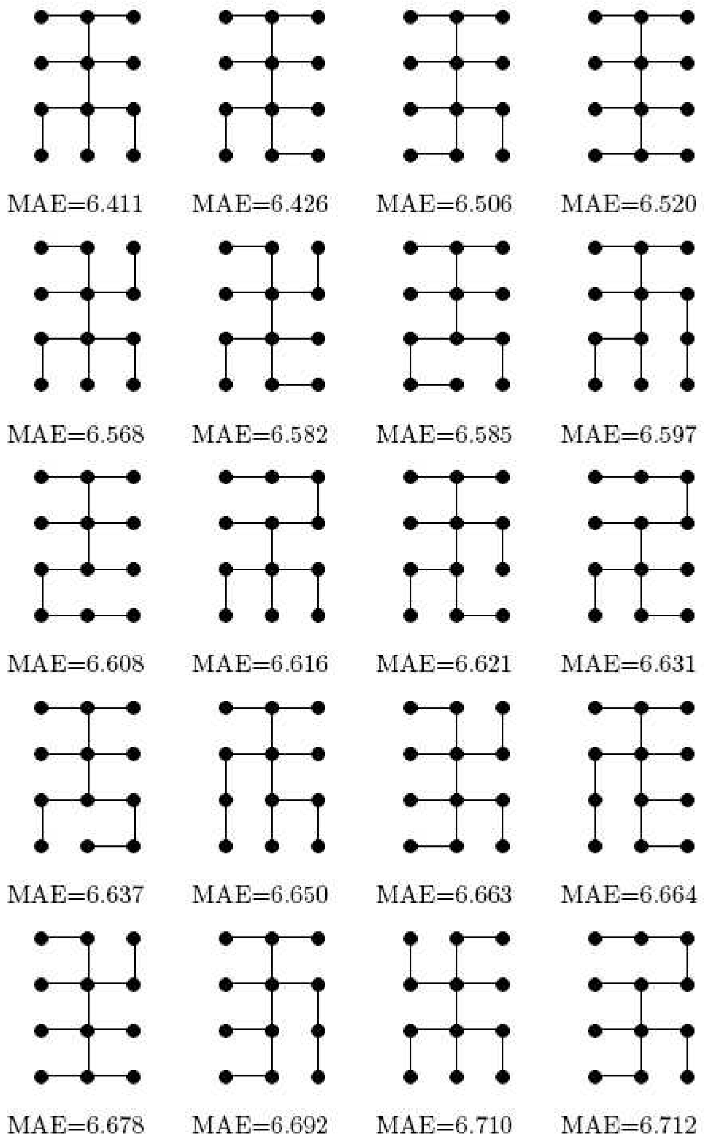


Figure 1.16: The twenty spanning trees that produced the lowest MMAE figures.

Bibliography

[1] A. R. Willis and D. Cooper, "Computational reconstruction of ancient artifacts," *IEEE Signal Processing Magazine*, vol. 25, pp. 165–83, July 2008.

[2] C. Papaodysseus, T. Panagopoulos, M. Exarhos, C. Triantafillou, D. Fragoulis, and C. Doumas, "Contour-shape based reconstruction of fragmented, 1600 BC wall paintings," *IEEE Transactions on Signal Processing*, vol. 50, no. 6, pp. 1277–1288, 2002.

[3] D. Koller, J. Trimble, T. Najbjerg, N. Gelfand, and M. Levoy, "Fragments of the city: Stanfords digital forma urbis romae project," *Journal of Roman Archaeology*, vol. 61, pp. 237–252, March 2006.

[4] F. Kleber, M. Lettner, M. Diem, M. Vill, R. Sablatnig, H. Miklas, and M. Gau, "Multispectral acquisition and analysis of ancient documents," in *Proceedings of the 14th International Conference on Virtual Systems and MultiMedia (VSMM 08)*, pp. 184–191, 2008.

[5] F. B. M. Corsini and V. Cappellini, "Mosaicing for high resolution acquisition of paintings," in *Proceedings of the 7th International Conference on Virtual Systems and Multimedia*, pp. 39 – 48, 2001.

[6] W. Puech, A. G. Bors, I. Pitas, and J.-M. Chassery, "Projection distortion analysis for flattened image mosaicing from straight uniform generalized cylinders," *Pattern Recognition*, vol. 34, pp. 1657–1670, August 2001.

[7] V. Swarnakar, M. Jeong, R. Wasserman, E. Andres, and D. Wobschall, "Integrated distortion correction and reconstruction technique for digital mosaic mammography," in *Proceedings of the SPIE Medical Imaging: Image Display*, vol. 3031, pp. 673–681, 1997.

[8] H.-Y. Shum and R. Szelisky, "Construction of panoramic image mosaics with global and local alignment," *International Journal of Computer Vision*, vol. 36, pp. 101–130, February 2000.

[9] H. S. Sawhney and R. Kumar, "True multi-image alignment and its application to mosaicing and lens distortion," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 97)*, pp. 450–456, 1997.

[10] S. Peleg and J. Herman, "Panoramic mosaics with videobrush," in *Proceedings of the DARPA Image Understanding Workshop 97*, pp. 261–264, 1997.

[11] K. S. F. Toyama and J. Miyamichi, "Image mosaicing from a set of images without configuration information," in *Proceedings of the International Conference on Pattern Recognition (ICPR 04)*, vol. 2, pp. 899 – 902, 2004.

[12] Y. Gehua and C. Stewart, "Covariance-driven mosaic formation from sparsely-overlapping image sets with application to retinal image mosaicing," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 04)*, vol. 1, pp. 804–810, 2004.

[13] Z. Z. Yong Li, Daiyin Zhu and L. Wang, "Automatic mosaicing for airborne SAR imaging based on subaperture processing," in *Proceedings of the IEEE International Geoscience and Remote Sensing Symposium (IGARSS 05)*, pp. 4644 – 4647, 2005.

[14] O. Pizarro and H. Singh, "Toward large-area mosaicing for underwater scientific applications," *IEEE Journal of Oceanic Engineering*, vol. 28, pp. 651–672, October 2003.

[15] A. Levin, A. Zomet, S. Peleg, and Y. Weiss, "Seamless image stitching in the gradient domain," in *Proceedings of the Eighth European Conference on Computer Vision (ECCV 2004)*, pp. 377–389, 2004.

[16] K. Shanmugasundaram and N. Memon, "Automatic reassembly of document fragments via context based statistical models," in *Proceedings of the 19th Annual Computer Security Applications Conference (ACSAC '03)*, pp. 152–159, 2003.

[17] A. Ukovich, G. Ramponi, H. Doulaverakis, Y. Kompatsiaris, and M. Strintzis, "Shredded document reconstruction using mpeg-7 standard descriptors," in *Proceedings of the 4th IEEE International Symposium on Signal Processing and Information Technology*, pp. 334– 337, 2004.

[18] A. Ukovich and G. Ramponi, "Feature extraction and clustering for the computer-aided reconstruction of strip-cut shredded documents," *Journal of Electronic Imaging*, vol. 17, no. 1, pp. 1–13, 2008.

[19] M. Prandtstetter and G. R. Raidl, "Combining forces to reconstruct strip shredded text documents," in *Proceedings of the 5th International Workshop on Hybrid Metaheuristics (HM '08)*, pp. 175–189, 2008.

[20] H. Wolfson, E. Schonberg, A. Kalvin, and Y. Lamdan, "Solving jigsaw puzzles by computer," *Annals of Operations Research*, vol. 12, no. 1-4, pp. 51–64, 1988.

[21] A. Kalvin, E. Schonberg, J. T. Schwartz, and M. Sharir, "Two-dimensional, model-based, boundary matching using footprints," *International Journal of Robotics Research*, vol. 5, no. 4, pp. 38–55, 1987.

[22] W. Kong and B. B. Kimia, "On solving 2D and 3D puzzles using curve matching," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 583–590, 2001.

[23] D. Goldberg, C. Malon, and M. Bern, "A global approach to automatic solution of jigsaw puzzles," *Computational Geometry: Theory and Applications*, vol. 28, no. 2-3, pp. 165–174, 2004.

[24] E. Tsamoura and I. Pitas, "Automatic color based reassembly of fragmented images and paintings," *IEEE Transactions on Image Processing*, vol. 19, pp. 680–690, March 2010.

[25] N. Nikolaidis and I. Pitas, "Using spanning trees for reduced complexity image mosaicing," in *Proceedings of the European Signal Processing Conference (EUSIPCO 2006)*, 2006.

[26] R. Szeliski, "Image alignment and stitching: A tutorial," *Foundations and Trends in Computer Graphics and Vision*, vol. 2, no. 1, pp. 1–104, 2006.

[27] H. C. da Gama Leitao and J. Stolfi, "A multiscale method for the reassembly of two-dimensional fragmented objects," *IEEE Transactions Pattern Analysis Machine Intelligence*, vol. 24, no. 9, pp. 1239–1251, 2002.

[28] Y. Lu, H. Gardner, H. Jin, N. Liu, R. Hawkins, and I. Farrington, "Interactive reconstruction of archaeological fragments in a collaborative environment," in *Proceedings of the Digital Image Computing Techniques and Applications (DICTA '07)*, pp. 23–29, 2007.

[29] Q.-X. Huang, S. Flöry, N. Gelfand, M. Hofer, and H. Pottmann, "Reassembling fractured objects by geometric matching," *ACM Transacions on Graphics*, vol. 25, no. 3, pp. 569–578, 2006.

[30] A. Willis, S. Andrews, J. Baker, Y. Cao, D. Han, K. Kang, W. Kong, F. F. Leymarie, X. Orriols, S. Velipasalar, E. L. Vote, D. B. Cooper, M. S. Joukowsky, B. B. Kimia, D. H. Laidlaw, and D. Mumford, "Assembling virtual pots from 3D measurements of their fragments," in *Proceedings of the Virtual Reality Archeology and Cultural Heritage conference(VAST '01)*, pp. 241–254, 2001.

[31] F.-H. Yao and G.-F. Shao, "A shape and image merging technique to solve jigsaw puzzles," *Pattern Recognition Letters*, vol. 24, pp. 1819–1835, August 2003.

[32] F. Kleber and R. Sablatnig, "A survey of techniques for document and archaeology artefact reconstruction," in *Proceedings of the International Conference on Document Analysis and Recognition*, pp. 1061–1065, 2009.

[33] M. S. Sagiroglu and A. Ercil, "A texture based matching approach for automated assembly of puzzles," in *Proceedings of the 18th International Conference on Pattern Recognition (ICPR '06)*, pp. 1036–1041, 2006.

[34] T. R. Nielsen, P. Drewsen, and K. Hansen, "Solving jigsaw puzzles using image features," *Pattern Recognition Letters*, vol. 29, no. 14, pp. 1924–1933, 2008.

[35] A. Criminisi, P. Perez, and K. Toyama, "Object removal by exemplar-based inpainting," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, pp. 721–728, 2003.

[36] S. Andrews and D. H. Laidlaw, "Toward a framework for assembling broken pottery vessels," in *Proceedings of the Eighteenth American Conference on Artificial Intelligence*, pp. 945–946, 2002.

[37] M. Kampel and R. Sablatnig, "On 3D mosaicing of rotationally symmetric ceramic fragments," in *Proceedings of the International Conference on Pattern Recognition*, vol. 2, pp. 265–268, 2004.

[38] F. Amigoni, S. Gazzani, and S. Podico, “A method for reassembling fragments in image reconstruction,” in *Proceedings of the International Conference On Image Processing (ICIP)*, pp. 581–584, September 2003.

[39] Y. Chen and G. Medioni, “Object modelling by registration of multiple range images,” *Image and Vision Computing*, vol. 10, no. 3, pp. 145–155, 1992.

[40] E. Justino, L. S. Oliveira, and C. Freitas, “Reconstructing shredded documents through feature matching,” *Forensic Science International*, vol. 160, no. 2, pp. 140–147, 2006.

[41] L. Zhu, Z. Zhou, and D. Hu, “Globally consistent reconstruction of ripped-up documents,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 1, pp. 1–13, 2008.

[42] H. Wolfson, “On curve matching,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 2, pp. 483–489, 1990.

[43] G. Papaioannou, E. A. Karabassi, and T. Theoharis, “Reconstruction of three-dimensional objects through matching of their parts,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, pp. 114–124, January 2002.

[44] S. R. Eliason, *Maximum Likelihood Estimation: Logic and Practice*. Sage Publications, Inc, 1993.

[45] P. Siarry, G. Berthiau, F. Durbin, and J. Haussy, “Enhanced simulated annealing for globally minimizing functions of many- continuous variables,” *ACM Transactions on Mathematical Software*, vol. 23, no. 2, pp. 209–228, 1997.

[46] H. Pottmann, J. Wallner, Q.-X. Huang, and Y.-L. Yang, “Integral invariants for robust geometry processing,” *Computer Aided Geometric Design*, vol. 26, no. 1, pp. 37–60, 2009.

[47] D. Huber, “Automatic three-dimensional modeling from reality,” *PhD thesis, Carnegie Mellon University*, 2002.

[48] G. Peris and A. Marzal, “Fast cyclic edit distance computation with weighted edit costs in classification,” in *Proceedings of the 16th International Conference on Pattern Recognition (ICPR’02)*, pp. 184–187, 2002.

[49] N. Otsu, “An automatic threshold selection method based on discriminant and least squares criteria,” *IEICE Transactions*, vol. J63-D, no. 4, pp. 349–356, 1980.

[50] T. F. Smith and M. Waterman, “Identification of common molecular subsequences,” *Journal of Molecular Biology*, vol. 147, no. 1, pp. 195–197, 1981.

[51] M. Gavrielides, E. Sikudova, and I. Pitas, “Color based descriptors for image fingerprinting,” *IEEE Transactions on Multimedia*, vol. 8, pp. 740–748, August 2006.

[52] G. A. Carpenter and S. Grossberg, *Pattern Recognition by Self-Organizing Neural Networks*. Cambridge, MA, USA: The MIT press, 1991.

[53] E. Trucco, A. Fusiello, and V. Roberto, “Robust motion and correspondences of noisy 3D point sets with missing data,” *Pattern Recognition Letters*, vol. 20, no. 9, pp. 889–898, 1999.

[54] D. Chetverikov, D. Svirko, D. Stepanov, and P. Krsek, “The trimmed iterative closest point algorithm,” *Proceedings of the 16th International Conference on Pattern Recognition*, vol. 3, pp. 545– 548, 2002.

[55] G. C. Sharp, S. W. Lee, and D. K. Wehe, “ICP registration using invariant features,” *IEEE Transactions on Pattern Analysis and Machine Inteligence*, vol. 24, pp. 90–102, January 2002.

[56] A. N. Netravali and B. G. Haskell, *Digital Pictures: Representation and Compression*. Plenum Press, 1988.

[57] N. L. Biggs, E. K. Lloyd, and R. J. Wilson, *Graph Theory 1736-1936*. Clarendon Press, 1986.

Index