COMPUTER AIDED PUZZLE ASSEMBLY BASED ON SHAPE AND TEXTURE INFORMATION

by

MAHMUT ŞAMİL SAĞIROĞLU

Submitted to the Graduate School of Engineering and Natural Sciences
in partial fulfillment of
the requirements for the degree of
Doctorate of Philosophy

Sabancı University

June 2006

COMPUTER AIDED PUZZLE ASSEMBLY BASED ON SHAPE AND TEXTURE INFORMATION

APPROVED BY

Prof. Dr. Aytül ERÇİL ……..
(Thesis Supervisor)

Prof. Dr. Lale AKARUN ……..

Doç. Dr. Uğur SEZERMAN ……..

Yard. Doç. Dr. Hakan ERDOĞAN ……..

Yard. Doç. Dr. Selim BALCISOY ……..

DATE OF APPROVAL: ……

COMPUTER AIDED PUZZLE ASSEMBLY BASED ON SHAPE AND TEXTURE INFORMATION

Mahmut Şamil SAĞIROĞLU

EECS, PhD Thesis, 2006

Thesis Supervisor: Prof. Dr. Aytül ERÇİL

Keywords: Puzzle assembly, reconstruction of the artifacts in archaeology, expanding images, Fourier based image registration

Abstract

Puzzle assembly's importance lies into application in many areas such as restoration and reconstruction of archeological findings, the repairing of broken objects, solving of the jigsaw type puzzles, molecular docking problem, etc. Puzzle pieces usually include not only geometrical shape information but also visual information of texture, color, continuity of lines, and so on. Moreover, textural information is mainly used to assembly pieces in some cases, such as classic jigsaw puzzles.

This research presents a new approach in that pictorial assembly, in contrast to previous curve matching methods, uses texture information as well as geometric shape. The assembly in this study is performed using textural features and geometrical constraints. First, the texture of a band outside the border of pieces is predicted by inpainting and texture synthesis methods. The feature values are derived by these original and predicted images of pieces. A combination of the feature and confidence values is used to generate an affinity measure of corresponding pieces. Two new algorithms using Fourier based image registration techniques are developed to optimize the affinity. The algorithms for inpainting, affinity and Fourier based assembly are explained with experimental results on real and artificial data.

The main contributions of this research are:

- The development of a performance measure that indicates the level of success of assembly of pieces based on textural features and geometrical shape.
- Solution of the assembly problem by using of the Fourier based methods.

DİZİLİM PROBLEMİNE ŞEKİL VE DOKU TABANLI BİLGİSAYAR DESTEKLİ YAKLAŞIM

Mahmut Şamil SAĞIROĞLU

EECS, Doktora Tezi, 2006

Tez Danışmanı: Prof. Dr. Aytül ERÇİL

Anahtar Kelimeler: Dizilim problemi, kırık arkeolojik parçaların geri çatılması, imge genişletme, Fourier tabanlı imge çakıştırma

Özet

Arkeolojik parçaların birleştirilmesi ve onarılması, kırık nesnelerin tamiri, parçalanmış dokümanların yeniden oluşturulması ve hatta moleküler kenetlenmenin çözümlenmesi genel olarak dizilim problemine dayanmaktadır. Görüntü işlemede dizilim; geometri ve doku olarak birbiriyle ilişkili parçaların birleşerek en iyi bütünü ortaya çıkarması olarak tanımlanmaktadır. Bugüne kadar dizilim problemi üzerinde yapılan çalışmalar sadece geometrik şekil bilgisine dayalı olarak ele alınmış, parçacıklar üzerindeki görsel bilgi kullanılmamıştır.

Bu bildiride daha önceki eğri uyumlama yöntemlerine dayalı geometrik yaklaşımlardan farklı olarak hem resim hem geometri bilgisinin kullanıldığı bir çalışma sunulmaktadır. İlk aşamada parçaların etrafındaki bir bantta doku öngörüsü yapılmaktadır. Öngörülen bu dokudan elde edilen öznitelikler bir uyum ölçüsü bulunmakta ve parçaların birbirlerine birleştirilmeleri Fourier tabanlı imge çakıştırma yöntemleri kullanılarak çözülmektedir. Geliştirilen yöntemler yapay ve gerçek datalar üzerinde sınanarak performansları incelenmiştir. Bu çalışmanın ana katkıları şu şekilde özetlenebilir:

- Doku ve şekil bilgisine dayalı olarak dizilimin başarımını sayısal olarak ortaya koyan bir performans ölçütü geliştirilmesi
- Dizilim probleminin Fourier metodları kullanılarak çözülmesi

Acknowledgements

I wish to express my deepest gratitude to my supervisor Aytül ERÇİL for her valuable advice and guidance of this work. I am grateful to her not only for the completion of this thesis, but also for her unconditional support. I feel myself privileged as her student.

I am greatly indebted to Alparslan BABAOĞLU, the Vice President of the National Research Institute of Electronics and Cryptology, for his encouragement and patience. His support of my work was beyond that of a manager.

Special thanks are due to the staff of my Institute of TUBİTAK and particularly to M. Oğuzhan KULEKCİ, Hüseyin DEMİRCİ, and Ali Said YAVUZ for their friendship and assistance.

My sincere thanks to all my friends and colleges, particularly Hakan BÜYÜKBAYRAK, in VPA Lab who cooperated nicely during the collection of data and laboratory work.

I am grateful to my thesis committee members Hakan ERDOĞAN, Selim BALCISOY, Lale AKARUN, Uğur SEZERMAN for their valuable review and comments on the dissertation.

I would like to thank to my family for their unlimited support and trust made everything possible for me.

Finally, I am particularly grateful to my wife, Buket Zeynep SAĞIROĞLU, for helping and assisting me in all the stages of this work. Without her help this study would never have been possible. Her constant and continuous co-operation proves her love and support during the whole course of this work. Special thanks to my children, Emir and Azra, for their patience.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

FFT	:	Fast Fourier Transform
CSTFT	:	Circular Short Time Fourier Transform
CH	:	Circular Harmonic
3D	:	Three Dimensional
2D	:	Two Dimensional
TV	:	Total Variation
CCD	:	Curvature Drive Diffusion

Chapter 1

INTRODUCTION

The aim of this research is to develop a method for the automated assembly of broken objects that have surface texture from their pieces. The task of reassembling has great importance in the fields of anthropology, failure analysis, forensics, art restoration, bioinformatics [1] and reconstructive surgery. It is also used frequently in archaeology. The fact that performing reconstruction of archaeological objects from fragments manually is very time consuming motivates automatic techniques for the reassembly of fragments. In general, reconstruction of objects can be regarded as a puzzle-solving, which is known as a subtitle of the matching problem. It contains many problems endemic to pattern recognition, computer vision, feature extraction, boundary matching and optimization fields.

Previous works on assembly problem have focused mainly on the geometrical properties of puzzle pieces represented by their boundary curves. As the fractions of boundaries are adjacent and thus similar, a pairwise affinity measure is computed by partial curve matching. Some approaches especially related to standard toy-store jigsaw puzzle solver use feature based matching methods. The fragment assembly problem is similar to that of automatic assembly of jigsaw puzzles, which has been addressed before. However, the problem of jigsaw puzzle solving is a reduced and restricted version of the general assembly problem. Its computerized solution was first introduced by Freeman [2], who successfully solved a 9-piece jigsaw puzzle. Other works [3][4][5][6] also use feature based matching approaches. These methods are relatively fast so that they manage to assemble the pieces, even if they are numerous. The main drawback of this approach is that they cannot provide detailed matching of boundaries and overlapping regions. Moreover, the general assembly problem does not satisfy the assumptions for standard jigsaw puzzle. Researches involving classical jigsaw puzzle ignore texture or color information to the attack assembly problem. There are a few

pictorial approaches, which use only the color values of pixel on boundary contour. However, this is not practical for real applications.

More general partial curve matching algorithms that solve the global 2D and 3D assembly problems based on geometrical properties were presented in [7][8][9]. The problem of 3D curves is addressed by [10]. The accuracy of matching technique depends on the perfect extraction of the trace of a curve and the computation of curvature and torsion. It is potentially a non-robust process and only tested on artificial data. Another study [11] matches 2D and 3D break curves by combining a coarse-scale representation of curves and refine iteratively via a fine-scale elastic matching. Afore mentioned research that achieved global assembly of pieces based on curve matching did not attempt to combine the geometrical methods with textural information.

There is great scientific interest in the archaeological community in reconstructing objects from fragments. In archeological sites, we may encounter a large number of irregular fragments resulting from one or several broken objects. The reconstruction of the original objects is a tedious and laborious task. The artifacts are free-form, multiscale individually and, with respect to one another, they are geometrically and photometrically highly complex and highly variable, and huge in number. An automatic tool that assists archeologists in reconstructing monuments or smaller fragments is being designed. Such a tool is designed in order to avoid unnecessary manual experimentation with fragile and often heavy fragments, and to reduce time required for assembly. Currently, the Digital Michelangelo team is tackling the problem of assembling the Forma Urbis Romae [12]. It is a marble map of ancient Rome that has more than a thousand fragments. Their investigation is based on broken surface border curves, possibly texture patterns, and additional features of the fragments. The University of Athens has developed Virtual Archaeologist system [13], relying on the broken surface morphology to determine correct matches between fragments. This method detects candidate fractured faces, matches fragments one by one and assembles fragments into complete or partially complete entities. The Shape Lab in Brown University [14][15][16] presents an approach to automatic estimation of mathematical models of axially symmetric pots made on a wheel. This technique is based on matching break curves, estimated axis and profile curves, a number of features of groups of break-curves. Finally, the assembly problem is solved by maximum likelihood performance-based search. Fornasier and Toniolo have developed a pattern matching algorithm for comparison of digital images by discrete Circular Harmonic expansions based on

sampling theory [17]. The assumption for this method is that the photographs of the original puzzle exist. At the Technical University of Vienna, a fully automated approach to pottery reconstruction based on the fragments profile is given [18]. In the study in the University of Vienna, a color specification technique [18] is proposed. The method using the colorimetric information assumes that the final object is known a priori and fragments are categorized in the beginning. In this field, an approach for assembly of pieces by using of textural and pictorial information of fragments also does not exist, even if the last research is related to color.

Neglecting the continuity of color and texture for adjacent fragments is a *waste* of valuable information for many cases. Humans use all-important information to decrease the possibilities for matching pieces. Automatic systems also have to use all pictorial features to attack the assembly problem. Actually, the pictorial information on a fragment consists of various components, and different specifications of surface image of pieces are dominant according to implementation field. In classical jigsaw puzzles, the essentials of assembly depend on the alignments of object edges (e.g. picture of a house), the similarity of colors (e.g. cloud drawing) and continuity of textural properties (e.g. grass of a garden) for the adjacent pieces. In the archeological field, the pictorial features may include highly directional marble veining, the pattern of surface incisions, paintings on the outer and inner surfaces, carvings and horizontal circles due to finger smoothing while the pot is spinning on the wheel. The texture-based approaches have to consider all these situations to match images of adjacent pieces. To solve the problem of continuity of texture between neighboring fragments, two main straightforward methods can be used. First one is to calculate a common discriminator from whole image on each pieces and test the affinities between possible pairs whether they share same segment of complete puzzle image or not. This is not so realistic. Because the pieces usually include more than one different texture segments, a common descriptor cannot represent the image of pieces. Moreover, some specifications, especially on the archeological findings such as continuity of marble veining and incisions, become unusable although they are significant pictorial properties. The second method is using of pixel values at regular intervals on the borderline of the pieces. Then, the matching pairs are formulated by this color characteristic along the contour of pieces. The border of the fragments is often eroded and may not be suitable to exact matching so that this approach is also not applicable. Furthermore, using pointwise relations on contour reduces effectiveness of matching pieces based on pictorial information when the puzzle

image has large regions covered by complex textural patterns, such as complex marble structures. Our proposed approach overcomes all mentioned textural problems by the use of completely different algorithm from previous methods. We design a texture prediction algorithm, which generates possible image outside the border of pieces. As we consider the completed puzzle, the predicted region outside of a piece overlaps the interior regions of others. Hence, the features of predicted texture outside a piece are correlated with original pictorial specifications of possible neighbor pairs. Also, a confidence measure depending on texture patterns is defined. Then, we reached an affinity measure of corresponding pieces that utilizes all kinds of image information, such as continuity of edges, textural patterns, and color similarities.

If we attempt to solve puzzles by computers, we need the following conditions:

Continuity of colors: Color continuity is an important factor on the solution of puzzle problems. If asked which combination below is the right solution of the puzzle, the second case seems to be more appropriate for most observers. Usually, the colors are not homogenous, so right color spaces, noise filters; shadow eliminators should be used to work with real images [19].

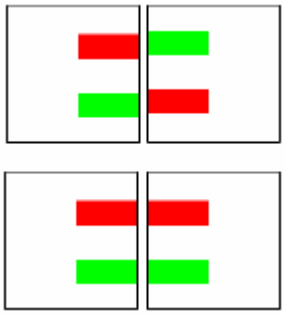


Figure 1.1 : The color continuity

Continuity of edges: The eyes want to see continuous lines, like a derivative operator. The problem here is usually that the broken real object pieces have occluded parts at the boundary zone [19].

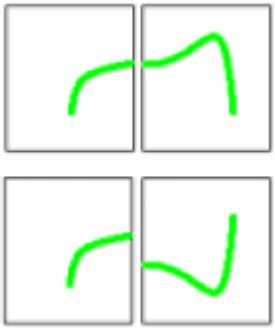


Figure 1.2 : The continuity of edges

Continuity of textural information: The two stripes below have the same color, but one is horizontal textured and the other is vertical. It is clear that the human vision also tracks the continuity of the textural information like the color. Therefore, characterizing a texture with numerical metrics is needed and as in the edge case, the changes at the boundary affect our success to label the pieces as continuous [19].

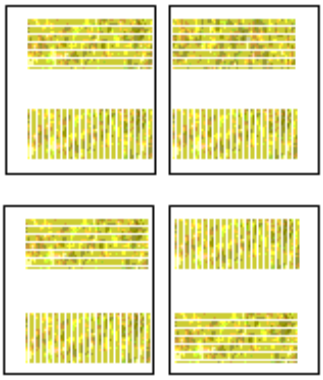


Figure 1.3 : The continuity of textures

Object memory: Just to see is not enough, we conceptually add some meaning to the objects; the biological similarity analysis is not 1 or 0 function but a most probable searcher. Below you see two pieces; it cannot be claimed to contain any cues on how these pieces should be associated. In this case, the only decision taker is our memory, which favors the first combination by remembering past experiences. It is probably the most important and difficult factor, if we would like to computerize biological vision systems [19].

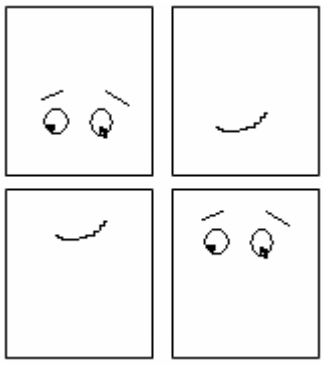


Figure 1.4 : The object memory

Boundary matching and generalization of the problem: The boundary is another dimension for our feature space: the pieces of various sized, occluded and missing parts together increase the complexity exponentially [19].

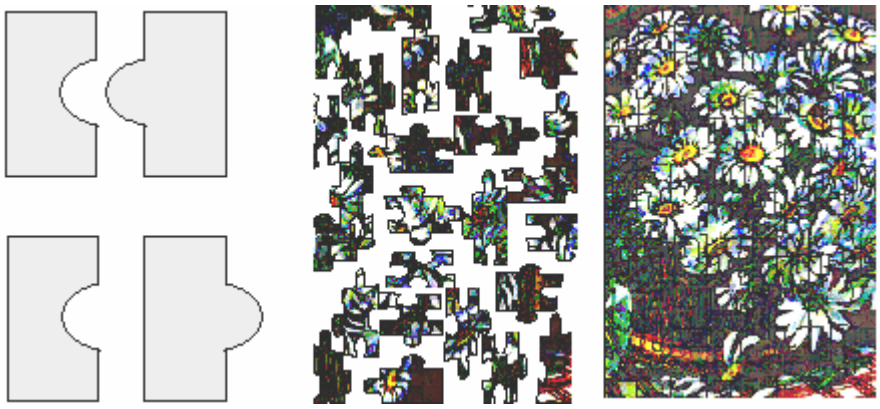


Figure 1.5 : The boundary features

Generally, the reconstruction of arbitrary objects from their fragments can be regarded as a puzzle, taking into account the following considerations:

Parts (fragments) have arbitrary shapes: General assembly problem differs from the standard toy-store jigsaw puzzle problem. The toy puzzles obey certain rules that make the problem more tractable that it would otherwise be. Standard rules include: (1) the puzzle has a rectangular outside border; (2) pieces form overall rectangular grid do that each interior pieces interlock with their primary neighbors by tabs, consists of an "indent" on one piece mating with an "outdent" on its neighbor; (4) each piece has no neighbors except its primary neighbors, that is, the cutting lines between pieces meet only at +, - junctions rather than a mix of +,-,T and Y junctions.

Figure 1.6 : The puzzle pieces may have arbitrary shapes. (An archaeological fragment from Forma Urbis Romea)

The shape and the number of final objects are unknown: In some works including standard jigsaw puzzles, it is assumed that the final shape of the puzzle is known. For especially archaeological problems, this assumption is not realistic. The pieces shall be assembled without the a priori knowledge of final shape. But there are some archeological researches that first estimate the final shape from fragments and than

reconstruct this shape from these fragments. But this assumption is also fails if fragments that belong to more than one broken objects exist in our initial set.

Figure 1.7 : The assembled form of the pieces may also have an arbitrary shape.

Some fragments may be missing and surfaces are probably flawed and eroded: The erosion of fragments is another important fact for which textural approaches are more suitable than geometrical methods. The borders of a fragment may disappear or erode in a real assembly application except artificially cutted toy-store jigsaw puzzles. In archeology, Erosion, impact damages or undesired events cause fragments to vanish or to deteriorate, such as the Forma Urbis Romae. A piece of a broken object (e.g. a broken marble) most probably may not exist. This reality increases the necessity of pictorial information to solve the reconstruction of all types of puzzles, because the geometrical approaches relying on exact matching of break curves are not applicable to assemble pieces if the border of fragments disappears. The texture methods can manage to estimate possible adjacent fragments, even if there is a gap caused by erosion between two neighbor pieces. Unfortunately, when the borders vanish the image from the surface of the fragments may also be removed. This situation is only possible in archeological findings.

Figure 1.8 : The pieces may be missing and probably eroded.

No strict assemblage rules exist: In general puzzle problem, the pieces can make any transformation in 2D or 3D. In some cases, the pieces have a limited number of transformations. For example, the pieces in a jigsaw puzzle can be transferred from one point to another in a grid.

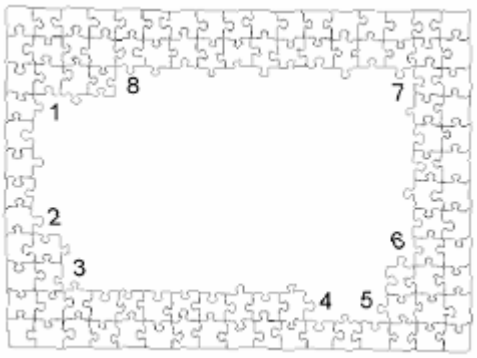


Figure 1.9 : The jigsaw puzzles are assembled with the strict rules.

The pieces may belong to more than one broken or torn objects: In archaeological sites, the fragments usually belong to more than one object. In these cases, the solution method has to consider this situation.

Figure 1.10 : The pieces belong to two different objects.

The main contribution of current research is to use the textural information in the solution of puzzles. Previous works omit this information source and use only the shape in the assembly. The above conditions, continuity of textural structure, the continuity of edges and boundary matching are satisfied and considered in the research. The object memory that is the highest-level application is not used. There is not any assumption that reduces the solutions complexity. The only assumption is that all pieces are acquired in a consistent and meaningful scale. This assumption is reasonable in a puzzle assembly problem because we can ideally acquire the pieces as we have all pieces initially.

1.1 Outline of the Thesis

Our proposed approach is to define a performance measure that represents the appropriateness of the assembly based on textural features and geometrical shape. Then, the best transformations of pieces that maximize harmony of textures of fragments shall be found while the geometrical constraints are being satisfied. Initially, we acquire and preprocess the images of pieces. The color information has to be obtained accurately in addition to other geometrical methods depending only on the border curve. After the collection of visual data, the first step to expose the affinity of fragments is an image prediction algorithm applied to each piece separately. We define a sufficiently large region that includes the original fragment. This region consists of two parts. The central part is the original fragment on which we know all pictorial information. The second part between the original fragment and the border of the predefined region is the

expanding domain. The prediction algorithm automatically fills in this expanding region with information diffusing from the central part.

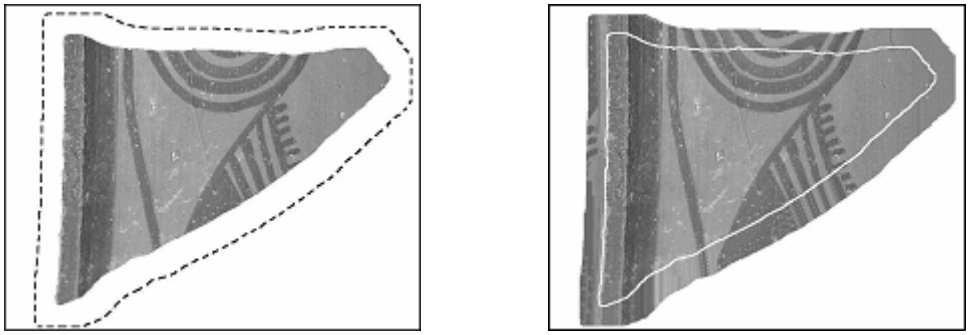


Figure 1.11 : (left) The original piece with dashed line limiting the expansion band, (right) the expanded piece with white line representing the border of the original piece.

The main idea to expand the fragment outwards is that the correlation between the features of the predicted region and the right neighbor is significantly higher than the alternative pairings. We use the mixture of inpainting and the texture synthesis methods for prediction. Image inpainting is the process of filling in missing data in a designated partition of image or video from the surrounding area, and texture synthesis creates a new image with the same seed texture but of different shape to a sample region. While expanding the fragment image, we introduce the confidence of expansion as a new parameter in prediction phase of assembly problem. This parameter represents the reliability of expanded values and will be used by later processes. The confidence depends on the structure of texture such as the continuity of edges, the roughness of texture and the distance to the border of original fragment.

Then, we derive the feature values of both the original fragment and the expanded region. The proposed approach does not bound the number of features nor it restrict the type of image features. Any textural feature believed to improve the success of assembly can be easily inserted to the process. The next step is to determine the similarity or cost function between two textural regions. There is no restriction on the distance function to be implemented by our method. The final goal of the proposed approach is to expose an affinity measure of corresponding pieces by the combination of the feature and confidence values. The harmony of pieces and the achievement of the assembly become better while optimizing this affinity measure. Actually, the assembly of fragments is to find the right transformations of pieces. Initially, each fragment has a random position in the space. To improve the assembly, we have to able to sense

whether any arrangement of pieces becomes better or worse. We use total affinity to be able to decide. The total affinity is defined as the sum of affinity measures of all points in the space. In addition to total affinity, two new functions are defined in this approach. The functions calculate the updated confidence and color values when two or more pieces are merged and a unique fragment is generated.

Although, the puzzle assembly problem can be stated as the optimization of the above cost function, the optimization problem is too computationally costly. We will therefore use the FFT shift theory to find a solution that will maximize the correlation between the predicted parts of a piece and other pieces. Two new assembly algorithms using the FFT based approach are introduced in this research. The first one is the semi-automated algorithm. This is developed for the interactive usage. The second one is a fully-automated assembly method. The last study on the assembly problem is to test the developed methods in 3D.

The rest of this thesis is organized as follows. Previous research is introduced in Chapter 2. Chapter 3 presents image inpainting and texture synthesis methods in literature and our implementations of these algorithms to predict expanding regions of the pieces. Subsequently, we try to find the best transformation of pieces that maximizes the harmony of textures of fragments by using an FFT-based algorithm. Finally, the developed methods for the 2D pieces are applied to 3D problems in the Chapter 5.

Chapter 2

LITERATURE SURVEY

2.1 Introduction

The task of reassembling has great importance in the fields of anthropology, failure analysis, forensics, art restoration and reconstructive surgery [20], and particularly in archaeology. The automated assembly of broken objects or puzzle problem was examined in many areas in the literature. The researchers have worked on the assembly problem to overcome the restoration and reconstruction of archeological findings, repairing of broken objects, solving standard jigsaw puzzles, molecular docking problem and the medical puzzle problem. Each work has only taken into account the considerations relevant to its application. So, the solutions are dependent on the assumptions of a particular application. For example, both the floor plan design of VLSI circuit design [6] and the archaeological fragment assembly [8] can be considered as a puzzle problem. But the main assumptions are completely different. In VLSI design problem, the fragments have regular shape and there is no textural and boundary relation between each pieces. Actually, the problem is equivalent to a placement problem. However, the adjacent pieces have strong textural and geometrical relations in an archaeological problem.

According to this point of view, it may be useful to study the previous works by classifying the research. All research can be categorized into three groups. The first group involves studies which aim to solve the standard puzzles; the second group of research involves studies that propose to develop a curve matching method. That is, by using boundary curves, global matching of pieces indicates the solution of puzzle; the last group consists of works in the field of archaeology. The fact that performing manual reconstruction of archaeological objects from fragments is very time consuming motivates automatic techniques for reassembly of fragments.

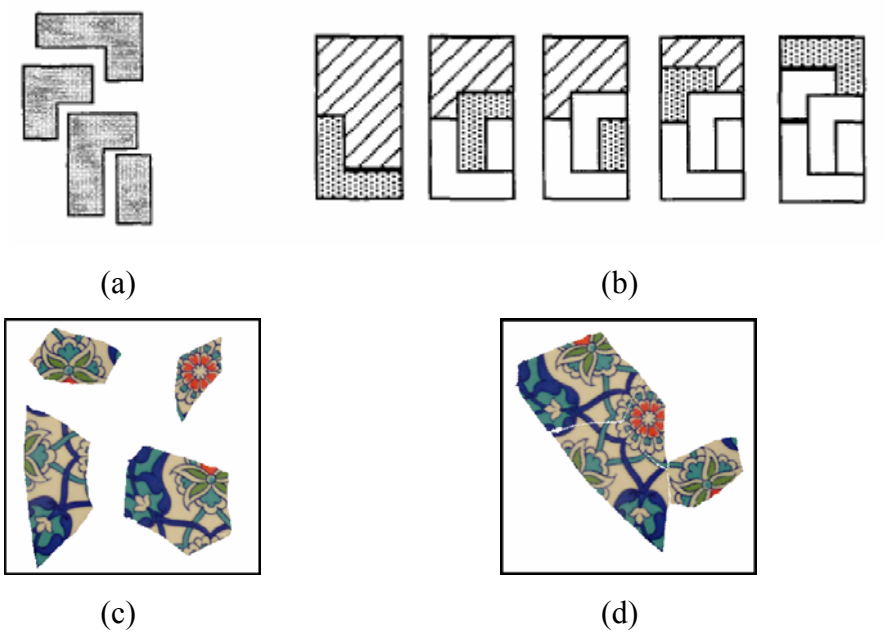


Figure 2.1 : (a) Placement problem (b) Solution of the placement (c) Textural pieces with non-standard shape. (d) Solution of ceramic puzzle with 4 pieces

Another classification criterion involves whether in a study uses textural information. Previous works on the assembly problem have focused mainly on geometrical properties of the pieces. Even there are a few works using pictorial information, the texture-based approaches are not improved.

In general, the reconstruction of objects can be regarded as a puzzle-solving problem, which contains many problems endemic to pattern recognition, computer vision, feature extraction, boundary matching, and optimization fields.

2.2 Literature about Approaches for Solving of non Archaeological Puzzles

The problem of jigsaw puzzle solving is a reduced and restricted version of the general assembly problem. Its computerized solution was first introduced by Freeman [2], who successfully solved a 9-piece jigsaw puzzle. In this work, the piece boundary is broken into sub-boundaries. The task in [21] is to find one-to-one correspondence between each of these sub-boundaries and a sub-boundary from another fragment. This is reasonable if the sharp corners in standard jigsaw puzzles delimit the matching boundaries. However, as Freeman [2] explains, this is not applicable when the problem is generalized.

A set of puzzle works [6][22][23][24] is defining the puzzle problem as a placement of pieces. For these approaches, there is no unsolved problem for the scope of computer vision, computer graphics or geometrics. So, probable solution methods in such puzzles are completely different from the concept of this study.

In [25], the method consists of feature extraction, local matching and global solution. The local matching makes use of new boundary and color matching operation to compute local matching scores between every pair of partial boundaries. Three algorithms are tested to find the global solution. These algorithms are called assignment problem based approach, the traveling salesman problem and assignment problem based approach, and the traveling salesman problem and K-best based approach. While extracting boundary information, heuristics are derived from the shape of standard puzzle pieces.

In [4], a system called automatic puzzle solver (APS), derives a new set of features based on the shape and color characteristics of puzzle pieces. A combination of shape dependent features and color cues is used to match the puzzle pieces. Matching is performed using a modified iterative labeling procedure in order to reconstruct the original picture represented by the jigsaw puzzle. As in [25], corner detection is used in the algorithm to separate the individual sides of each puzzle piece. The assembly method is not briefly examined.

In [26], a method is proposed for solving the rectangular jigsaw puzzle assembly problem. The puzzle is only painted in black and white. It is assumed as a binary image. The assembly of the puzzle is performed only using information of the pixel value on the borderline of the pieces. The proposed method utilizes a genetic algorithm to search for the optimum piece arrangement, because a genetic algorithm has the ability to find the global solution in the large optimization space. This method is tested on only an artificial puzzle set, and the boundary information is not used because the puzzle pieces are assumed rectangle. The pixel color values are directly used to generate fitness function. This usage without any feature extraction is not proper in real implementations.

In the thesis [27], they work on extracting the boundary points around the edge of the piece into an ordered list known as chain code; identifying straight edges in the boundary so they can be ignored when matching occurs; finding the internal angle around each boundary point; comparing the angles and colors of each boundary point to other similarly rendered pieces to try and find a match. The color information near the

boundary is also directly used in this study. The assembly algorithm does not handle the ambiguities and is also tested on only standard and small sets of jigsaw puzzle pieces.

In another and interesting paper [5], it is first stuck to the document reconstruction problem where the pieces are strips. Later, the ideas generalize to jigsaw and the other shapes. To judge the fit between two strips, they run them through a fitness function to compute a score. They coded pieces so that the score indicates the degree of mismatch. Thus, the higher the score, the worse the match is between the pieces. The ideal matching function would evaluate to zero for any two strips that were supposed to be adjacent and to infinity for all other pairs.

The principle at work is coherence. In this context, coherence says that any given column of pixels in an image is going to be a lot like the columns immediately to its left and right. After all, if the images were random noise (that is, just black and white dots with no features), then matching up strips would be hopeless because statistically no pairs of strips would be any better that other pair.

To reconstruct the image, it uses an assembly algorithm. It replaces the idea of a strip's side with the color values running around the perimeter of one side of the piece. This approach is to look for edges that are the same color, within some threshold. If an edge is a single color (or almost a single color) then it does not take part in the matching process. But if not, it influences the scoring, sorting and clustering processes.

In [28], Hopfield neural networks are used to perform the matching of outer contours of the puzzle pieces. The dominant points extracted from the boundary of pieces are defined. Then, all jigsaw puzzle pieces are described using attributed relational graph representation. A number of unary vertex attributes and binary edge attributes such as curvature values at break points, curve-wise distance between break points, angle between two adjacent dominant points, etc. are extracted to be represented in attributed relational graph structure. After applying the developed method, the global solution of puzzle is not examined and defined as another research. The texture information is not used in the research.

One of the latest papers [3], proposed to solve standard jigsaw puzzles works with the larger set of puzzle pieces as many as 200. As in the work of Wolfson et al. [21], algorithm in [3] first assembles the border pieces using a heuristic for the Traveling Salesman Problem. They depart from other works in how we place the interior pieces. Because they do not assume that pieces have well-defined sides, they require a more global matching technique. At all times, they maintain an optimized planar embedding

of the current partial solution. They fit a piece into a pocket not by independent pairwise fitting with top and side neighbors as in [21], but by fitting it into the embedded partial solution, thus allowing for any number of neighbors around the pocket. They reported that

> "Wolfson et al. [21] rejected global embedding because of the possibility of accumulated errors, but we found to the contrary that global embedding gave more accurate results than pairwise matching, enabling a greedy placement algorithm—without any backtracking or branch-and-bound—to solve the jigsaw puzzles. (For more complicated puzzles, we could easily add backtracking or branch-and-bound.)"

They define fiducial points (specifically the centers of ellipses fit to the indents and outdents) to find the best translation and rotation of a piece to match a pocket. The fiducial points approach, however, worked quite well and is significantly faster, because it does not need to test all subcurve or substring starting points. Another advantage of fiducial points is that they are more robust to scanning noise than some of the other techniques.

They fill pockets in highest confidence first order position an eligible pocket if it has at least two primary neighbors that have already been placed. Initially, when only the border pieces have been placed, there are four eligible pockets; later there may be quite a few eligible pockets. At each step they fill the eligible pocket that has the highest ratio of the score of best fitting piece to the second best fitting piece. This order turned out to be more reliable than the best-first order. After fitting a piece, they reoptimize the global embedding of all pieces. They do this by minimizing the squares of the distances between corresponding points on neighboring pieces, for all neighboring pieces at once. Global optimization distributes the matching inconsistencies throughout the partial solution, and in experiments outperformed a smoothing procedure that moved one piece at a time. This work also contributes to the standard jigsaw puzzle solution. The main contribution in the study is that the numbers of the pieces in the puzzle used in experiments are quite high with respect to the other studies. The textural information is also not considered in the study.

In [29], jigsaw puzzle pieces are represented by their medial axis from which certain features are detected. The most significant is the isthmus or neck, which of the isthmus is then used when comparing puzzle pieces and only male/female pairings

whose widths are very close to the same are considered, thus greatly reducing the search space.

The jigsaw puzzle problem is also addressed briefly in [30], a paper which is mainly concerned with creating a canonical representation of shape for object recognition. This representation is based on shape concavities and is invariant to affine and planar transforms and robust against occlusion. However, its precision is limited and most of these features are not currently needed for puzzle solving because a properly scaled, un-occluded image of each piece can always be obtained.

In [31], the boundary curve of each piece is divided into four subcurves corresponding to the four sides of the puzzle piece and these curves are later used in the matching procedure. This division is based on finding four, so called breakpoints on the boundary curve of each piece. Each curve is sampled at equal arc-length and represented by the sequences of coordinates at its sample points. Pieces having an almost straight section between adjacent corners are identified as frame pieces. Then a local curve matching procedure is applied. For each pair of two different puzzle piece boundary subcurves, the subcurves are matched using a matching algorithm like in [21]. Also, as mentioned before, the common boundary of jigsaw puzzle pieces tend to be delimited by sharp corner and feature interlocking curves, which greatly reduce as the number of possible matching configurations. However, none of these specific constraints generally apply to puzzles that consist of fractured natural materials and thus cannot be used for general assembly problem.

In [9], points on a curve are extracted at regular intervals of distance from the central critical point. Using polar coordinates, these points are represented only by their angle term. This low cost representation for curves that allows sequences of similar features to be found quickly is another method called feature-based matching algorithm. A best first technique is used in which the puzzle is assembled using the best local pairwise match at each step. However, this method does not work for a puzzle with larger set, where the introduction of ambiguous matches will almost always result in an incorrect configuration.

2.3 Curve and Surface Matching Techniques

Curve matching methods based on an elastic model has been presented in [32]. These techniques, provide highly accurate matching at a very fine scale, but they cost computationally too much to be used directly. As mention before, [11][33][34] papers that also deal with puzzles composed of natural materials present curve matching using an elastic model somewhat similar to [32].

In the paper [35], they present an outline-based recognition method, which relies on finding the optimal correspondence between 2D outline (or curves) by comparing their intrinsic properties, namely length and curvature. The basic premise of approach is that the goodness of the optimal correspondence can be expressed as the sum of the goodness of matching subsequences. Then, the problem of finding the optimal correspondence is applied to an efficient dynamic-programming algorithm as an energy minimization. They also introduce the notion of an alignment curve to ensure a symmetric treatment of the two curves being matched.

In [36], the algorithm proposes to solve the (partial) surface matching and the (partial) volume-matching problem, either with volume overlap or with volume complementary. First, they associate with each point of the two sets a footprint. This value should be invariant under rotations and translations, and should be "descriptive," in the sense that points of the two sets whose local neighborhoods admit a good match should have similar footprints, whereas points whose local neighborhoods do not fit well together should have significantly differing footprints. Next, they define a scoring function that measures the "goodness" of a specific rotation (of one set relative to the other), and is invariant of the relative translation. In an ideal setting, this function has a global maximum at the correct rotation and does not have any other local maxima. This enables to advance from any rotation toward the correct rotation, by invoking the scoring function iteratively, and by deciding locally in which direction to advance. Finally, they compute the best translation associated with the final rotation. The various applications of algorithm mainly differ in the definition and computation of the footprints. Needless to say, the choice of footprints is a crucial factor that influences the success of this method. The main contribution of this study is the new observation that the density of votes in translation space can be used for computing the correct relative rotation of a model and an image.

The paper [8] approaches the problem of 2D and 3D puzzle solving by matching the geometric features of puzzle pieces three at a time reconstruction. First, they define an affinity measure for a pair of pieces in two stages, one based on a coarse-scale representation of curves and one based on fine-scale elastic curve matching method. Second, triples arising from generic junctions are formed from this rank-ordered list of pairs. The idea is that generic breaks in puzzles only produce T and Y junctions, thus motivating to merge three pieces at a time in this process. The puzzle is solved by a recursive grouping of triples using a best first search strategy, with backtracking in the case of overlapping pieces. Initially, the complete list of contour of pieces enters to search algorithm. If there are more than two pieces in the list, all local triple groups are generated by using local shape analysis algorithm. Then, all local triple groups are ordered such that the most likely local triple will be checked first. Each local group will be tested to see if it can construct a global solution. If it cannot, it backtracks. They also generalize aspects of this approach to matching of 3D pieces. The main difficulty in generalizing the curve matching process to space curves is that it requires the robust computation of curvature and torsion, involving up to second and third order derivatives, respectively.

In [7], two algorithms to find the longest common subcurve of two 2D curves are presented. These algorithms are based on the conversion of the curves into shape signature strings and the application of string matching techniques to find long matching substrings. Then direct curve matching is applied to the corresponding 'candidate' subcurves to find the longest matching subcurve. Here, all possible combinations of sub-boundaries, which have a complexity equivalent to the number of samples along each contour, are considered. This complexity is reduced in one approach by converting each contour into a sequence of feature strings based on a polygon approximation and using geometric hashing to compare sub-sequences. The main drawback of the method is that they cannot provide global information such as region overlap and they are not enough to distinguish between several close ambiguous matchings. Thus, the search space for large puzzle sets becomes larger.

In the work [10], 3D surface piece objects are represented by their boundary curves. These closed curves are parameterized by their *curvature* and torsion *scalars*, which are calculated from the discrete 3D boundary curve data and quadratically added to form a circular string of a single value. For each pair of the representation, a similarity matrix with elements defined as the Euclidian distance between the creature

vectors is constructed. The matrix elements, which are less than a noise threshold, are considered as matching points. By processing of the similarity matrix, all matching fragments are determined. Among sequences of such matching points the longest sequence of matching fragments will be determined in the noise tolerant manner. Considering the start and end information of the fragments the longest non-overlapping sequence of fragments is determined. Then, the algorithm joins the matching portions and removes the parts of the joints. So, a single piece is obtained from two fragments. These operations continue until one piece is left. But, in many of those real world problems a perfect match between two subjects is not possible. Environmental aging effects, imperfections in digitization environment, the accumulation of systematic errors in numerical operations all contribute to this imperfection. Therefore, fault tolerant partial matching is required.

A multi-scale technique is used in a series of studies [11][33][34], where the contours are first re-sampled at a very coarse scale, so that an exhaustive search is manageable. From those, only the best matches are kept and matched again at recursively finer scales. These methods are able to handle our more general type of puzzle. The detailed examination of these studies will be reported below. The semi-automated search using pairwise information are suggested and performed in this technique.

In paper [11], an algorithm for reassembling one or more unknown objects that have been broken or torn into a large number N of irregular fragments is described. The algorithm works by comparing the curvature-encoded fragment outlines, using a modified dynamic programming sequence-matching algorithm. By comparing the outlines at progressively increasing scales of resolution, they manage to reduce the cost of the search form $O(N^2L^2)$ (where L is the mean number of samples per fragment) to about $O(N^2L)$; which, in principle, allows the method to be used for problems of practical size ($N=10^3$ to 10^5 fragments, $L=10^3$ to 10^4 samples).

Their experimental results demonstrate the possibility of automatically identifying adjacent fragments by matching the shapes of their outlines. Those results also validate the basic premise of the multi-scale matching method, namely that the false candidates are quickly eliminated, as they are re-tested with increasing resolution. Their methods depend on the randomness of the fracture lines, and therefore work best for granulated material like unglazed ceramics, stone, stucco, etc.

In spite of the large estimated speedup provided by the multi-scale method, the algorithm is still somewhat too expensive for practical use. Further speedup will require improving the algorithm itself. In particular, by using geometric hashing techniques, it would be possible to reduce $O(N^2)$ term something closer to O(N log N).

The study [34] describes a method to measure the average amount of information contained in the shape of a fracture line of a given length. This parameter tells us how many false matches we can expect to find for that fracture among a set of fragments. In particular, the numbers that are obtained for ceramic fragments indicate that fragment outline comparison should give useful results even for large instances. Their fragment matching algorithms are specialized for objects with a smooth and locally flat surface, such as tiles, tablets, large vases, frescoes, etc. The algorithms' input consists of the digitized fragment contours or outlines, modeled as a set of plane curves. They assume that two fragments, adjacent in the original object were separated by an ideal fracture line of zero thickness.

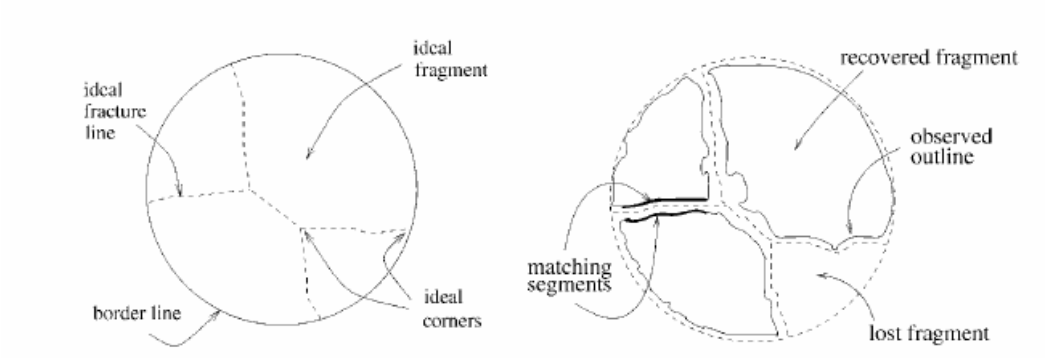
Before they apply the tools of information theory to this problem, they turn each curve into a signal real function of some real parameter t. A well-known rotation-invariant representation of a curve is the graph of its curvature k(t) as a function of its arc-length t measured from an arbitrary reference point.

In [33], an approach based on information extracted from fragment outlines that is used to compute the mismatch between pairs of pieces of contour is represented. They use the technique of dynamic programming. In order to asymptotically reduce the cost of matching, they use multiple scale techniques after filtering and resampling the fragment outlines at several different scales. They look for initial matchings at the coarsest possible scale. They then repeatedly select the most promising pairs, and re-match them at the next finer scale of detail. In the end, they are left with a small set of fragment pairs that are most likely to be adjacent in the original object.

They assume that the fragmented objects have a well-defined smooth surface. This surface is divided into two or more parts, the ideal fragments that are separated by ideal fracture lines, irregular curves with zero width. Two fragments are said to be adjacent if they share a fracture line. The fractures also split the original outline of the surface into one or more borderlines.

The fracture lines can be viewed as a graph G drawn over the object's surface, which they call the fracture network. (Figure 2.2) The point where three or more lines

(fracture or boundary) meet is called ideal corner. The boundary of an ideal fragment is an ideal contour. It is the concatenation of one or more fracture and borderlines.



(a) (b)

Figure 2.2 : (a) The ideal fracture network and (b) observed outlines

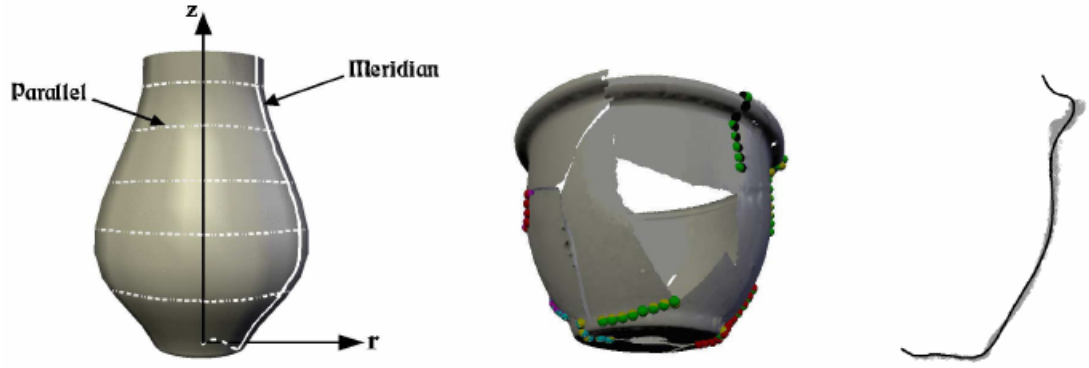
In [37], two different approaches are used for fragment matching. One is curve matching, the other is to compare whole surfaces or volumes depending on the nature of the broken objects. A unified method that combines curve-matching techniques with a surface matching algorithm to estimate the positioning and respective matching error for joining of 3D fragmented objects. It is reported that combining both aspects of fragment matching, essentially eliminates most of the ambiguities present in each one of the matching problem categories and helps provide more accurate results with low computational cost.

First, the fragment meshes are segmented into crude sides and the potentially fractured ones are detected, marked accordingly and stored. At a second stage, potentially fractured sides are processed in pairs, in order to define the geometric transformation that joins the two surfaces in an optimal way. The fractured facet boundary information guides the search for complementary matching between the two fragments but is not sufficient to determine a correct match alone. Therefore, it is used to constrain a local search using the surface similarity criterion. Then, a curve-constrained matching is performed. Finally, a global optimization scheme is employed to arrange the fragment collection in a set of reconstructed objects, based on the pairwise matching errors, and the corresponding geometrical transformations are applied hierarchically to arrange the fragments to the correct pose.

2.4 Literature about Approaches for Solving of Archaeological Puzzle

In the studies [14][15][16][38][39], we find many contributions on axially symmetric pot assembly. In archaeological sites, the findings are mostly the pots made on a wheel. Thus, the reconstruction of the pots from the hundreds of sherds found at an excavation site is one of the most important and unsolved problems. An approach is presented to the automatic estimation of mathematical models of such pots from 3D measurements of sherds. A Bayesian approach is formulated beginning with a description of the complete set of geometric parameters that determine the distribution of the sherd measurement data. The matching of fragments and aligning them geometrically into configurations is based on matching break-curves (curves on a pot surface separating fragments), the estimated axis and profile curve pairs for individual fragments and configurations of fragments, and a number of features of groups of break-curves. Pot assembly is a bottom-up maximum likelihood performance-based search. Experiments are illustrated on pots, which were broken for the purpose, and on sherds from an archaeological dig located in Petra, Jordan. The performance measure can also be an aposteriori probability, and many other types of information can be included, e.g., pot wall thickness, surface color, patterns on the surface, etc. This can also be viewed as the problem of learning a geometric object from an unorganized set of free-form fragments of the object and of clutter, or as a problem of perceptual grouping.

In this paper, a Bayesian approach has been outlined for the estimation of mathematical representations for pots based on sherds found at archaeology sites. The key algorithms for implementing the approach have been developed, and experimental results from these algorithms to real sherd 3D data have been presented and discussed.



(a) (b) (c)

Figure 2.3 : (a)A sample axially symmetric pot (b) A correctly assembled pot and (c) its profile

The framework discussed in this paper is for estimating arbitrary a priori unknown axially symmetric pot models. Hence, it is unsupervised pot geometry learning from sherd data. If, instead, we know a priori that the pot sherds present are not arbitrary but rather that each belongs to one of a group of 10 known pot shapes, the problem is computationally much easier because the sherd alignment problem is then more of a pot shape-recognition problem and less of a shape-estimation problem.

The framework presented can accommodate additional geometric and pattern information, which should result in doing the pot estimation faster, or with fewer sherds, or estimating models for more complex objects.

A complete system which automatically estimates complete mathematical models for 3D ceramic pots given 3D measurements of their fragments is described in the thesis of Andrew Willis [40] as below. This approach is defined as solutions of four problems:

1. An algorithm for accurately estimating the surface geometry of an individual sherd
2. An algorithm for accurately aligning assemblies of sherds, called configurations
3. A Bayesian performance measure for sherd configurations
4. A performance-driven search algorithm

Estimation of the outer surface geometry is implemented as maximum likelihood estimation of the axially symmetric surface parameters given the measured sherd data. Sherd configurations are aligned along break-point segments, which lie on the boundary of the sherd's outer surface. An algorithm is proposed for accurately aligning configurations of N sherds given a hypothesized set of correspondences between the sherd break-point segments. This is also implemented as maximum likelihood estimation where the estimated parameters are the N-1 sherd alignment transformations, the matched break-point segment parameters, and the global configuration surface parameters. A common Bayesian framework provides a performance measure for sherd configurations which is the log of the probability of the measured sherd data given the computed configuration maximum likelihood estimation, referred to as the configuration cost. The search mechanism is of the nature of a uniform cost search. The assembly process starts with a fast clustering scheme, which approximates the maximum likelihood estimation solution for all sherd pairs. More accurate maximum likelihood estimation values based on all parameters are computed when sherd pairs are merged with other sherd configurations. Merging takes place in order of constant probability starting at the most probable configuration.

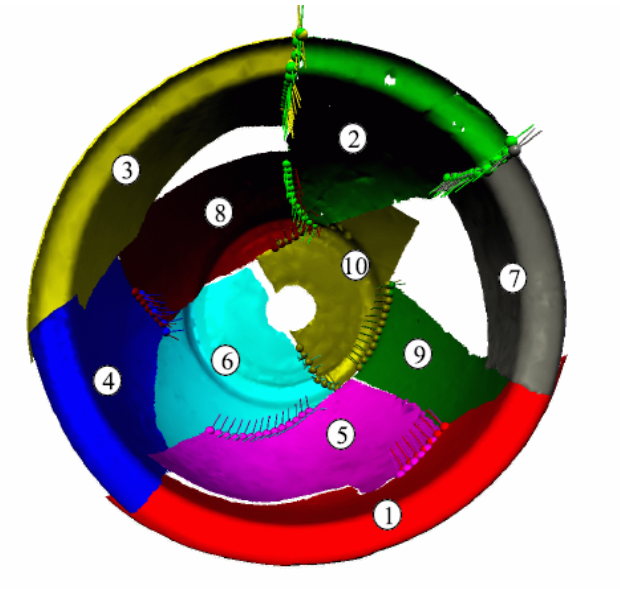


Figure 2.4 : A result from the Willis's thesis represents a correctly assembled pot of 13 sherds where only the 10 matched sherds shown were available.

One of the latest works about the archaeological assembly problem is presented in [17]. An accurate matching algorithm for the comparison of digital images implemented by discrete Circular Harmonic expansions based on sampling theory is proposed. The algorithm and its performance for reassembling fragmented digital images are tested on the art fresco of the Italian Renaissance, destroyed by bombing during the Second World War. The main assumption in their study is that there exists fairly good quality of black and white photographs of the original surface of the archaeological site from 1900 and 1920, because the available pieces from the fresco demonstrate the lack of continuous fragments and makes it extremely improbable that any reconstruction will be successful using methods based on the outline shape of the pieces. The accuracy and the robustness of the proposed procedure are achieved by exploiting the independent (orthogonal) information given by the Circular Short Time Fourier Transform (CSTFT) at different angular and radial frequencies. The CSTFT is here implemented by discrete scalar product of the digital image with respect to location shifted and sampled compactly supported Circular Harmonic (CH) functions, selected among those affected by minimal aliasing. The computational efficiency of the algorithm is given by the combined use of the correlation implemented by fast Fourier transforms for the location/position detection, and of a tricky implicit and fast computation of the mutual angle by exploiting self-steerability properties of CH functions. It is reported that "up to 90% of the fragments are detected in the first 20 best positions the algorithm returns out of more than 7 millions possible. The implementation of combined redundant

computations and registration methods improves further this ratio, realizing in most of the cases an almost completely automatic detection of the fragments"[17].

Figure 2.5 : The picture is taken from the study [17]. It is shown that a fragment is detected and overlapped the corresponding part of the old gray fresco photo dated to 1920.

Another fragment assembly problem attempting to overcome the limitations of many previous techniques has been worked in thesis [41]. Like most previous ones [32], their technique consists of two distinct stages. These are to calculate local pairwise affinity and to search for a globally optimal arrangement of fragments. To compute pairwise affinity, they first address the problem of specifying matching sub-boundaries. They rely on corners to define one end of the sub-boundary instead of using the fragment corners to partition the boundary into matching sub-boundaries. To find the other end, they use a novel normalized energy in elastic curve-matching [32] function in order to determine how far along the sub-boundaries similarity exists. This defines a finite number of adjacency candidates, which is dependent on the number of corners on the respective fragments. These adjacency candidates can then be evaluated by standard curve-matching techniques. Then, they employ a multi scale technique similar to the one described in [11][33][34].

To find a globally optimal arrangement of fragments, they use a best-first strategy, but instead of only using local pairwise information, they also evaluate matches based on their resulting contribution to the global confidence. At each step in process, they first use the fragment triplet representation to generate list of possible choices for the selection and arrangement of the next fragment. Since backtracking is expensive, they consider multiple solutions simultaneously, maintaining a list of possible arrangements of fragments at each stage. This allows errors to occur without causing the failure of the search.

Another chapter in [41] introduces some techniques regarding the use of color information from a fragments image to enhance the pairwise relations. The usage of the textural information is not sufficiently studied in the puzzle problem (especially in the archeological field), as we introduced in the first chapter. Thus, this is an important work in archeological fragment reassembly. The main idea is that a new type of curve matching alignment based on elastic curve matching technique is applied to solve fragment assembly, but instead of trying to minimize the elastic energy between two curves, optimization of the continuation of intensity and texture across a fracture is used.

Another paper [42] related to archeological fragment assembly presents techniques regarding the classification and reconstruction of ceramics based on the profile, which is the cross-section of the fragment in the direction of the rotational axis of symmetry, and can be represented by a closed curve in the plane. This paper compares and combines several methods for the interpolation and approximation of a closed curve by B-splines in the plane. The closed curve, representing the profile, is divided into several parts for which the most accurate method is selected. All the interpolation and approximation methods are compared on the provided data with respect to the achieved precision and complexity of the curve description. The main contribution of this work is to be able to demonstrate which combination of these methods gives the best representation of the reconstructed profile from the data under the smallest possible error and the simplest possible spline representation. Similar methods are used in other studies [18][43][44].

An algorithm for the automatic construction of a 3D model of archeological vessels is presented in [45][46]. The importance of the determination of the exact volume of arbitrary vessels in archeology motivates this study, since the information about manufacturer and the usage of vessel is valuable. The object's silhouette is the

only which is extracted from an input image. Images are acquired by rotating the object on a turntable in front of a stationary camera. The algorithm uses an octree representation of the model, and builds this model incrementally, by performing limited processing of all input images for each level of the octree. This work is not directly related to assembly of pieces. But the development in acquisition systems is important, because the solution of the reconstruction problem of fragments (especially in archaeology) usually deals with the acquisition of 3D objects. Also, the work [12] aims to digitize the shape and the color of large fragile objects under non-laboratory conditions. Their system employs the laser triangular rangefinders, laser time-of-flight rangefinders, digital still cameras, and a suite of software for acquiring, aligning, merging and viewing scanned data. As a demonstration of this system, they digitized 10 statues by Michelangelo, including the well-known figure of David, two building interiors, and all 1,163 extant fragments of the Forma Urbis Romae, a giant marble map of ancient Rome.

The Forma Urbis Romae, also known as the Severan Marble Plan, is a giant marble map of ancient Rome. Measuring 60 feet wide by 45 feet high and dating back to the reign of Septimius Severus (circa 200 A.D.), it is probably the single most important document on ancient Roman topography. Unfortunately, the map lies in fragments - 1,186 of them, and not all the fragments still exist. Piecing this jigsaw puzzle together has also been one of the great- unsolved problems in this situation [12].

The fragments of the Forma Urbis [47] present many clues to the would-be puzzle solver: the pattern of surface incisions, the 2D (and 3D) shapes of the border surfaces, the thickness and physical characteristics of the fragments, the direction of marble veining, matches to excavations in the modern city, and so on. Unfortunately, finding new fits among the fragments is difficult because they are large, heavy, and numerous. It is believed that the best hope for piecing the map together lies in using computer shape matching algorithms to search for matches among the fractured side surfaces of the fragments. In order to test this idea, it is needed 3D geometric models of every fragment of the map. To obtain this data, during June 1999, a team of faculty and students from Stanford University spent a month in Rome digitizing the shape and surface appearance of every known fragment of the map using laser scanners and digital color cameras. Their raw data consists of 8 billion polygons and 6 thousand color images, occupying 40 gigabytes.

The goals of the Digital Forma Urbis Romae Project are threefold: to assemble the raw range and color data into a set of 3D (polygon mesh) models and high-resolution (mosaiced) photographs - one for each of the 1,186 fragments of the map, to develop new shape matching algorithms that are suitable for finding fits between 3D models whose surfaces are defined by polygon meshes, and to use these algorithms to try solving the puzzle of the Forma Urbis Romae. Whether or not they succeed in solving the puzzle, one of the tangible results of this project will be a web-accessible relational database giving descriptions and bibliographic information about each fragment and including links to our 3D models and photographs. Their long-term plan is to make the entire database (1,186 fragments) freely available to the archeological (and computer graphics) research communities, educators, museum curators, and the general public.

Figure 2.6 : A photograph of fragment of the Forma Urbis Romae (Severan Marble Plan). This fragment is roughly 3 feet across, which is 640 feet on the ground, and it weighs about 150 pounds. Each incised line is a wall; thus, parallelograms with gaps in their borders are rooms with doors. The small V's in narrow rooms are staircases, and sequences of round pits are porticos supported by columns [47].

In an article [13], they present a complete method – encapsulated in their Virtual Archaeologist system – for the full reconstruction of archaeological finds from 3D scanned fragments. In Virtual Archaeologist, they regard the reconstruction problem from a general, geometric point of view, relying on the broken surface morphology to determine correct matches between fragments. This approach does not require specific object information, but is versatile enough to exploit any other data available.

This program detects candidate fractured faces, matches fragments one by one, and assembles (clusters) fragments into complete or partially complete entities. The only input data this system requires are the polygonal meshes with 3D scanner or

digitizer. Modeling or curve interpolation may be required in cases where only blueprints of cut sections of the fragments are available as part as the standard archiving procedure. This system does not require human intervention, but users can clarify the reconstruction by interactively fine-tuning the clustering and poses of the fragments.

The general method proposed in another paper [48] from the same study matches and glues fragments or parts belonging to an object, one against one, using only their surface geometry, assuming no information about the fragments' origin, data set sampling distribution or the final model to be reconstructed. The basic concept in their method is that, given two 3D models, the best fit is likely to occur at their relative pose, which minimizes the point-by-point distance between the mutually visible faces of the objects. For this reason, they introduce and calculate an error measure of the complementary matching between two object parts at a given relative pose, based on this point-by-point distance.

This matching error is minimized by employing a standard global optimization scheme to determine the relative positioning of the two fragments that corresponds to their best complementary fit. During the automated assembly, it is assumed that the two object parts can be rigidly attached to one another without having to penetrate each other's surface. For instance, the method cannot be used to connect two links of a chain.

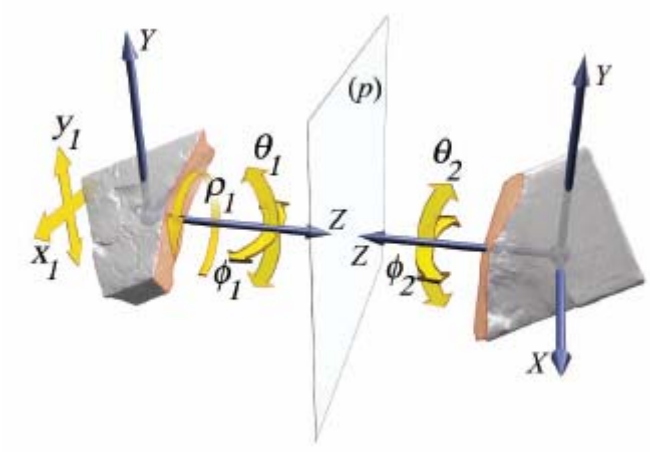


Figure 2.7 : The matching of the fractured faces.

In [49], a strategy for detecting the joint among two potsherds is proposed. Similar to the other studies, the main idea in this method is that the most similar section between two contours must be detected by partial verification. Their implementations consist of five processes: the contour segmentation process, the segment description process, the segment verification process, the candidate extraction process, and the candidate verification process.

The color information is used in the paper [50], unlike the previous methods. Here, the color information does not indicate the color values of the pictorial data. The color specification technique in [50] exploits the fact that the spectral reflectance of materials like archaeological fragments varies slowly in the visible. They present an approach for accurate colorimetric information on fragments, performed on digital images containing archaeological fragments under different illuminants. A characteristic vector analysis of the reference reflectance leads to an algorithm that computes the calorimetrically accurate reflectance out of a video digitizing system.

A EU funded project introduces the 3D Measurement and Virtual Reconstruction of Ancient Lost Worlds of Europe system (3D MURALE) [51]. It consists of a set of tools for recording, reconstructing, encoding, visualizing and database searching/querying that operate on buildings, building parts, statues, statue parts, pottery, stratigraphy, terrain geometry and texture and material texture. The tools are loosely linked together by a common database on which they all have the facility to store and access data. The paper describes the overall architecture of the 3D MURALE system and then briefly describes the functionality of the tools provided by the project. The paper compares the multimedia studio architecture adopted in this project with other multimedia studio architectures.

Recording tools are being developed for measuring terrain, stratigraphy, buildings, building blocks, pottery, pottery sherds and statues on the archaeological site. The results of these measurements are being stored in the 3D-MURALE database system. Reconstruction systems are using a 3D graphics tool to combine the individual measured components and reconstruct building elements and buildings from building blocks, pottery from pottery sherds, statues from statue elements and stratigraphy from all finds within the excavation.

There are also some applications and studies in virtual reality and computer graphics to reconstruct the pieces using human interaction. First, an image of 3D shape and the surface texture is acquired. Then, an interface for connecting broken fragments in virtual space is built as in [52], so that the original model can be visually recovered.

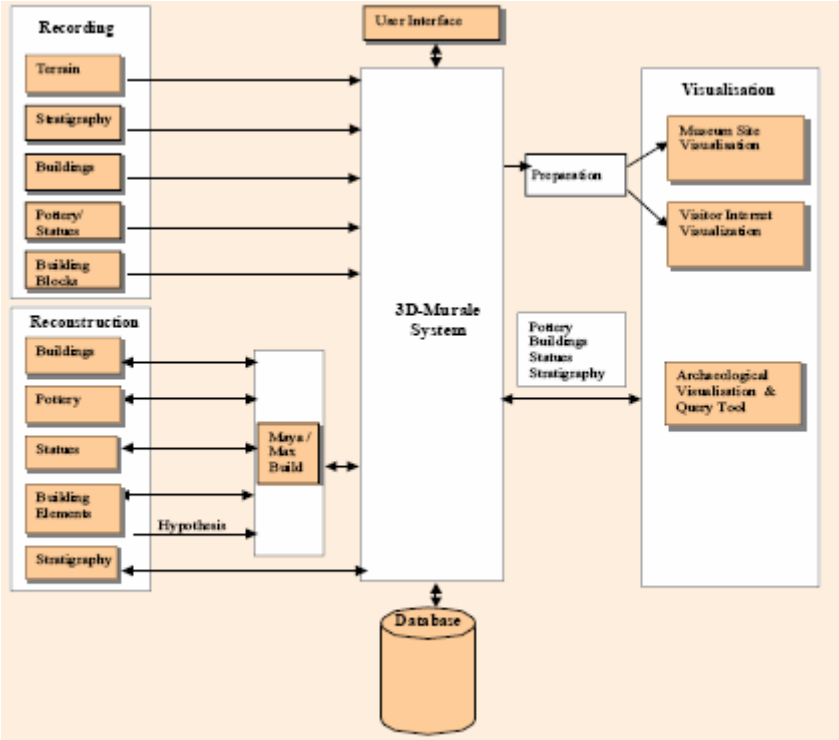


Figure 2.8 : The 3D-MURALE system consists of the Recording, Reconstruction, Database and Visualization components

Chapter 3

EXPANDING PIECES

The essential of matching two or more pieces in an assembly problem is that common features of neighbors are more strongly related than the others. We proposed a pictorial approach based on textural features. The first step is to expand each piece outwards by predicting the pictorial information of outer space. The aim of expansion is to expose relations of the fragments based on textural features. Inpainting and texture synthesis are two main fields in the literature related to this work.

3.1 Introduction

Image inpainting refers to the process of filling-in the missing areas or changing an image in a non-noticeable way by an observer. It is usually applied to the task of restoring photographs, films or paintings, and removing of occlusions, such as subtitles, stamps and text. This ancient practice is used in artwork restoration. However, after the notion digital inpainting is first introduced in [53], it is applied to many fields in image processing, such as image interpolation, zooming [54], and error concealment of wireless image transmission [55][77].

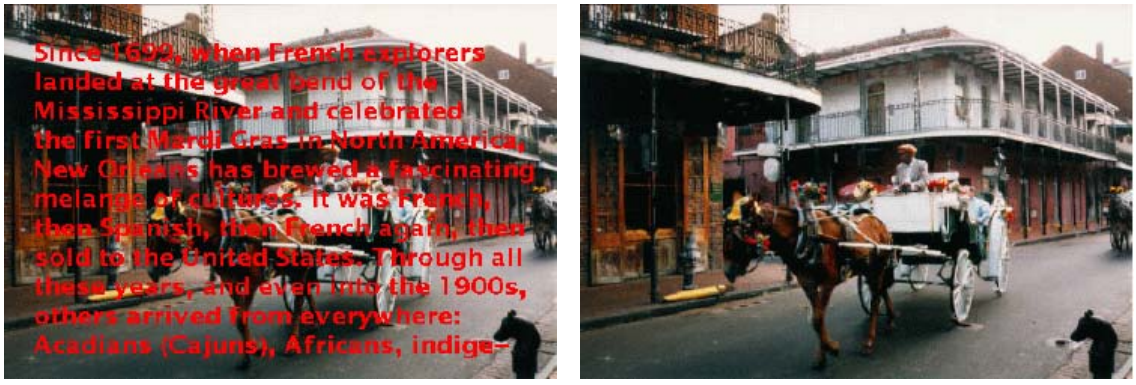


Figure 3.1 : Restoration of a color image by the use of inpainting and removal of superimposed text.

The first work [53] in inpainting used the series of partial differential equations to mathematically model this process. These techniques determine how the linear structures (called isophotes) propagate into the region to be inpainted. Isophote directions are obtained along the inpainting contour by computing the vector perpendicular to the gradient, thus the information is propagated while preserving edges. The intensities of pixels are updated by estimating the variation in color smoothness approximated by a discrete laplacian. This variation is propagated along the mentioned isophote direction. Then, the algorithm smoothes the inpainting region by anisotropic diffusion iterations used to minimize the noise effects. Anisotropic diffusion also preserves boundaries of the inpainted region, and it prevents isophote lines inside the region from crossing. Image inpainting issue is addressed by similar methods in [56][57][58][59][60][61].

The same topic is studied for 3D volumetric data in [62]. This work addresses the problem of hole filling via isotropic diffusion of volumetric data (that is, iterative Gaussian convolution of some distance function to the known data). The approach proposed by the authors addresses holes with complicated topology, a task very difficult with mesh representations. In the paper, a literature review on the subject about the nature of the holes in scanning statues is described. Another work [63], inspired by [62], presents an alternative for the same task. In contrast with [62], they use a system of coupled anisotropic (geometric) partial differential equations designed to smoothly continue the isophotes of the embedding function, and therefore the surface of interest (as the zero level isophotes).

Other inpainting approaches are the Total Variational (TV)[54] and Curvature-Drive Diffusion models (CCD)[64]. TV uses an euler-lagrange equation to minimize total variation and employs anisotropic diffusion. Such a method handles noise well, but does not complete broken edges. CCD is based on the TV algorithm and geometric information of isophotes. The drawback of these methods is the blurring of inpainted image introduced by the diffusion process in the larger filling regions. The abstracts of the papers using the similar methods such as [65][66][67] are collected in the web page [68].

Texture synthesis is an active research topic in computer vision, which has broad applications such as foreground removal, lossy image compression, and texture generation. The problem of texture synthesis is to fill large image regions with a sample texture. This method, which replicates consistent textures, can be used in extension of

images, but it has problems to fill in real image patterns. Linear structures such as a drawing of a line or crossing regions of different textures usually include high frequency components, which prevent the generation of natural images by this approach. The second problem in these techniques is that boundaries between image regions are a complex product of mutual influences between different textures. In contrast to the two dimensional nature of pure textures, these boundaries form what might be considered more one dimensional, or linear, image structures. Another problem of these algorithms is its tendency for some textures to occasionally "slip" into a wrong part of the search space and start growing garbage or get locked onto one place in the sample image and produce verbatim copies of the original. These problems occur when the texture sample contains too many different types of texels (or the same texels but differently illuminated) making it hard to find close matches for the neighborhood context window. Providing a bigger sample image can usually eliminate these problems.

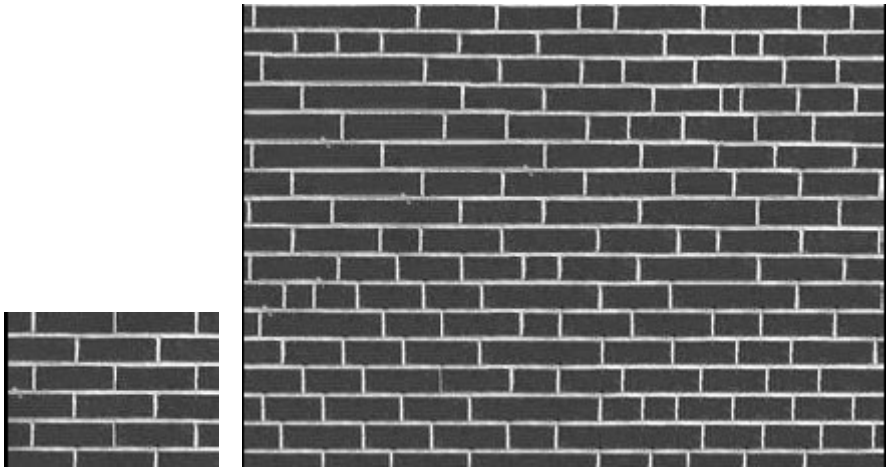


Figure 3.2 : (left) A seed image and (right) synthesized image with texture synthesis methods.

In paper [69], a non-parametric method for texture synthesis is proposed. The texture synthesis process grows a new image outward from an initial seed, one pixel at a time. A Markov random field model is assumed, and the conditional distribution of a pixel given all its neighbors synthesized so far is estimated by querying the sample image and finding all similar neighborhoods. The degree of randomness is controlled by a single perceptually intuitive parameter. The method aims at preserving as much local structure as possible and produces good results for a wide variety of synthetic and real-world textures.

In the study [70], a new method for image completion that interleaves a smooth approximation with detail completion by example fragments is introduced. The

unknown region and fills in the image by adaptive fragments is iteratively approximated. Then, a composition of fragments completes the image under combination of spatial transformations. The completion process can be regarded as a "push-background" process, in contrast to the "pull-foreground" process associated with image matting. This approach requires a relevant training set and a degree of self-similarity within the input image. It is an image-based 2D method that does not incorporate high level information and can therefore produce unnatural looking completions.

An interesting algorithm for the parallel synthesis of composite textures is described in [71]. A special-purpose solution for synthesizing the interface between two "knitted" textures is devised in this work.

To overcome the drawbacks of inpainting and texture synthesis algorithms, the method presented here combines both approaches. There are very recent researches along similar lines. Proposed algorithm in [72] first decomposes the image into the sum of two components with different basic characteristics [73], and then reconstructs each one of these components separately with inpainting and texture synthesis. Another approach by Harrison [74] uses exemplar-based synthesis for object removal process. This approach turns out to perform surprisingly well, making attempts to add refinements using isophote flow irrelevant in many cases. The method assigns pixels information theoretic constraints based upon how much additional information neighboring pixels yield in predicting it (along with some assumed distribution of course). This categorization into neighboring pixels that constrain the central pixel turns out to account for much of the information flow techniques seem to provide already (although not always). Sampling and copying pixels from the source generate the output image in this method. Ordering in which pixels are added to output image is also described.

Another method [75] that our algorithm uses the same overall approach also combines the inpainting and texture synthesis, but the algorithm differs in the implementation. We extend the image outwards, whereas the mentioned method is used to complete inner image portions. It is obvious that the image can be completed more accurately than can be extended, because the inner parts are surrounded by usable information.

3.2 Theory

Our extension implementation is not only a filling-in process as other applications, but also a prediction method. We will predict the texture of a neighbor piece while the current piece is being extended. Therefore, we define the extension of an image outwards as *prediction*, which was not used before in the inpainting literature. We now proceed with the details of the prediction algorithm.

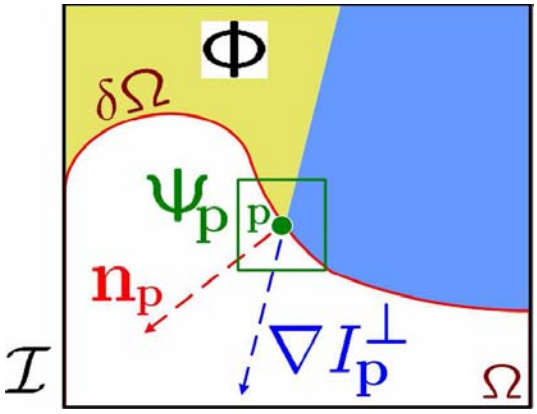


Figure 3.3 : The notations: Original image, with the target region, its contour, and the source region

In this section, we will use some symbols that define the images, contours or variables. For ease of comparison, we try to use and adapt notation to that used in the inpainting literature. The source region, Φ, is an acquired image and remains fixed throughout the algorithm. This region provides the samples used in the filling process. A target band outwards from the source region is indicated by Ω. This region is being filled during the algorithm runs. This target band represents the extension region of the piece. The border between Φ and Ω is indicated by δΩ. This border evolves outward as the inpainting algorithm progresses.

The inpainting algorithm consists of three main steps. These steps are iterated until whole target region or band has been filled. First step is to compute the priority, *P*, which determines the order in which they are filled.

The quality of the output image is highly influenced by the order in which the filling process proceeds. In the work [75], it is described and listed a number of desired properties of the "ideal" filling algorithm.

A comparison between the standard concentric layer filling (onion-peel) and desired filling behavior is illustrated in Figure 3.4. This figure is taken directly from the technical report [75]. Figures 3.4-b,c,d show the progressive filling of a concave target

region via an anti clockwise onion-peel strategy. As it can be observed, this ordering of the filled patches produces the horizontal boundary between the background image regions to be unexpectedly reconstructed as a curve.

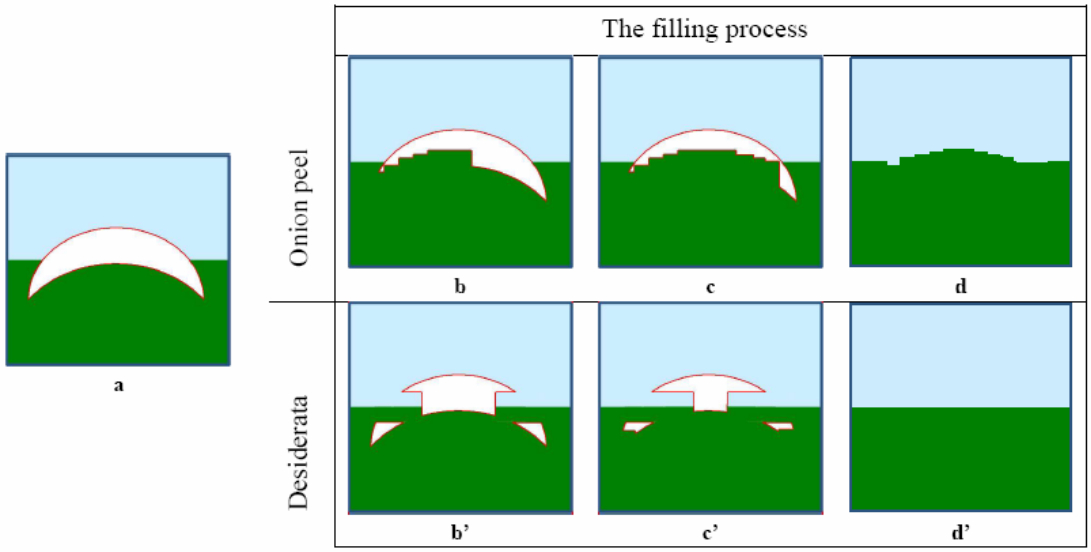


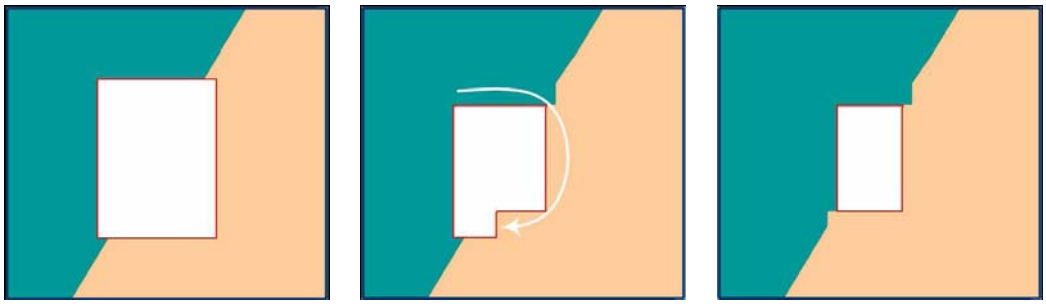
Figure 3.4 : The importance of the filling order when dealing with concave target regions.

A better filling algorithm would be one that gives higher priority of synthesis to those regions of the target area which lie on the continuation of image structures, as shown in Figures 3.4-b',c',d'. Together with the property of correct propagation of linear structures, the latter algorithm would also be more robust towards variations in the shape of the target regions.

A concentric-layer ordering, coupled with a patch-based filling may produce further artifacts such as the ones illustrated in Figure 3.5.

Therefore, filling order is crucial to non-parametric texture synthesis [69][76]. As it is discussed in paper [75], designing a fill order which explicitly encourages propagation of linear structure (together with texture) has never been explored, and thus far, the default favorite has been the "onion peel" strategy.

Another desired property of a good filling algorithm is that of avoiding 'over-shooting' artifacts that occur when image edges are allowed to grow indefinitely. The goal here is finding a good balance between the propagation of structured regions and that of textured regions (Figure 3.4-b',c',d'), without employing ad-hoc strategies. The algorithm achieves such a balance by combining the structure 'push' with a confidence term that tends to reduce sharp in-shooting appendices in the contour of the target region.



(a) (b) (c)

Figure 3.5 : The importance of the filling order in patch-based filling.

As in the paper [75], the filling algorithm proposed overcomes the issues that characterize the traditional concentric-layers filling approach and achieves the desired properties of:

- Correct propagation of the linear structures,
- Robustness to changes in the shape of the target region,
- Balanced simultaneous structure and texture propagation

Initially, each pixel maintains a color value in the source region, and a confidence value, which reflects our confidence in the pixel value, and which is frozen once a pixel has been filled. During the course of the algorithm, patches along the fill front are also given a temporary priority value, which determines the order in which they are filled.

The priority computation is biased toward those patches which: (i) are on the continuation of strong edges and (ii) are surrounded by high-confidence pixels. Priority value is computed for the patches Ψ_p centered at the point p for $p \in \delta\Omega$ (see Figure 3.3).

$$P(p) = C(p).D(p) \tag{3.1}$$

Conceptually, the priority depends on continuation of strong edges, D, and confidence of neighbor pixels, C. We call $C(p)$ the confidence term and $D(p)$ data term and they are defined as follows:

$$C(p) = \frac{\sum_{q \in \Psi_p \cap \Phi} C(q)}{\left|\Psi_p\right|}, \quad D(p) = \left|\nabla I_p^{\perp}.n_p\right| \tag{3.2}$$

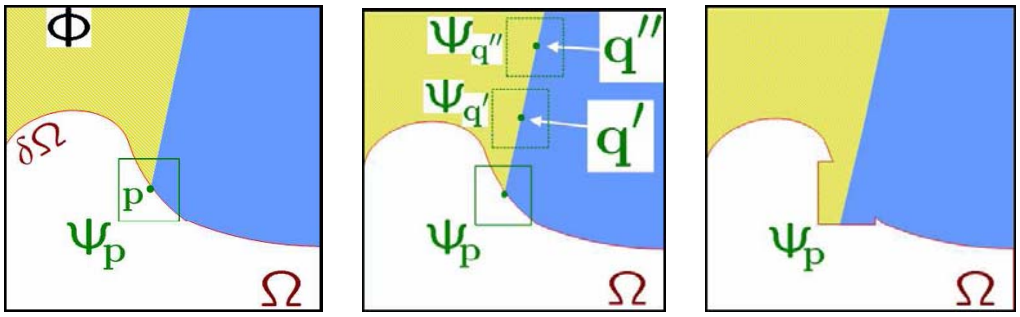
where $|\Psi_p|$ is the area of Ψ_p, n_p is unit vector orthogonal to the front $\delta\Omega$ in the point p and $\perp$ indicates the orthogonal operator. The priority $P(p)$ is computed for every border patch, with distinct patches for each pixel on the boundary of the target region. Initially, we set the 100% reliability to C=1, and assign C=0 if any information does not exist. In a formal representation:

$$C(p)=0 \quad \forall p \in \Omega, \quad \text{and} \quad C(p)=1 \quad \forall p \in \Phi \tag{3.3}$$

The confidence term $C(p)$ ma be thought of as q measure of the amount of the reliable information surrounding the pixel p. The intention is to fill first those patches which have more of their pixels already filled, with additional preference given to pixels that were filled early on (or that were never part of the target region). For example, patches that include corners and thin tendrils of the target region will tend to be filled first, as they are surrounded by more pixels from the original image. These patches provide more reliable information against which to match. Conversely, patches at the tip of "peninsulas" of the filled pixels jutting into the target region will tend to be set aside until more of the surrounding pixels are filled in. At the coarse level $C(p)$ approximately enforces the desirable concentric fill order. As filling proceeds, pixels in the outer layers of the target region will tend to be characterized by greater confidence values, and therefore be filled earlier; pixels far from the source region will have lesser confidence values.

This confidence value, first introduced in [75], reflects reliability of a region or a pixel, and it affects filling order during inpainting process. Furthermore, we will also use the confidence values as a critical parameter in assembly step.

The Data term $D(p)$ is a function of the strength of isophotes hitting the front $\delta\Omega$. This term increases the priority if an isophote flows into that patch. This term has an importance in both inpainting and assembly algorithm, because it causes the linear structures to be synthesized or filled first. Therefore, the linear structures orthogonal to border of pieces are completed earlier and these points or patches get higher confidence values. This event is also very important for assembly process that will be described.



(a) (b) (c)

Figure 3.6 : (a) We want to synthesize the area delimited by the patch Ψ_p centered on the point $p \in \delta\Omega$. (b) The most likely candidate matches for Ψ_p lie along the boundary between the two textures in the source region, e.g., $\Psi_{q'}$ and $\Psi_{q''}$. (c) The best matching patch in the candidates set has been copied into the position occupied by Ψ_p, thus achieving partial filling of Ω.

When all priorities have been computed, the highest priority, p', is determined. The second step of inpainting algorithm is the propagating the texture and structure information into the target band. The color information is propagated via diffusion in classical inpainting techniques. In this method as in [75], propagation of the image texture occurs by direct sampling of source region.

It is well-understood that exemplar-based approaches perform well for two-dimensional textures. But, the work [75] note in addition that exemplar-based texture synthesis is sufficient for propagating extended linear image structures as well, i.e. a separate synthesis mechanism is not required for handling isophotes.

Figure 3.6 taken from [75] illustrates this point. We now focus on a single iteration of the algorithm to show how structure and texture are adequately handled by examplar-based synthesis. Suppose that the square template $\Psi_p \in \Omega$ centered at the point p is to be filled. The best-match sample from the source region comes from the patch $\Psi_{q'} \in \Phi$, which is most similar to those parts that are already filled in Ψ_p. In the example in Figure 3.6-a, we see that if Ψ_p lies on the continuation of an image edge, the most likely best matches will lie along the same (or a similar colored) edge (e.g. $\Psi_{q'}$ and $\Psi_{q''}$ in the Figure 3.6-b).

All that is required to propagate the isophote outwards is a simple transfer of the pattern from the best-match source patch (Figure 3.6-c). Notice that isophote orientation is automatically preserved. In the figure, despite the fact that the original edge is not orthogonal to the target contour $\delta\Omega$, the propagated structure has maintained the same orientation as in the source region.

We focus on a patch-based filling approach (as opposed to pixel-based ones) because this improves execution speed. Furthermore, it is noted in [75] that patch-based filling improves the accuracy of the propagated structures.

The mathematical expression to find the most similar patch for sampling is:

$$\Psi_{q'} = \arg\min_{\Psi_q \in \Phi} \quad d(\Psi_{p'}, \Psi_q) \tag{3.4}$$

where $d(\Psi_{p'}, \Psi_q)$ is the distance between already filled pixels of patches in the p' and q points. The Euclidian distance is used in the algorithm. Patch in the q' point is the most similar one and the values of each pixel to be filled in the p' patch $\{ p' \mid p' \in (\Psi_{p'} \cap \Omega) \}$ are copied directly from the patch in the q' point.

This suffices to achieve the propagation of structure and texture information from the source to the target region, one patch at a time. In fact, it is noted in [75] that any

further manipulation of the pixel values (e.g. adding noise, smoothing, and so forth) that does not explicitly depend upon statistics of the source region, is more likely to degrade visual similarity between the filled region and the source region, than to improve it.

The last step for iterations is to update confidence values. After the patch $\Psi_{p'}$ has been filled new values, the confidence values affected by the filling of new patch are updated. This region is the limited by the neighbors of the p' point.

$$C(p) = C(p') \quad \forall p \in \psi_{p'} \cap \Omega \tag{3.5}$$

As filling proceeds and going far from original (or source) region of the piece, the confidence values decay. This indicates that the color values of pixels far from border are less reliable than closer ones. This difference between far and closer pixel has also importance in the assembly algorithm.

3.3 Implementation

As we mentioned before, the main algorithm of the inpainting step in this project is similar with the research [75]. The main differences are in the implementation phase. The algorithm is implemented for each piece separately.

First, a puzzle piece or an archaeological fragment is acquired with a camera or a scanner. The main point in this step is to use an ideal background, because the accuracy of extracting the shape of the pieces depends on the proper capturing.

After acquisition, a standard algorithm is used to find outer boundary of the pieces. The inner part of this boundary is set as source region, Φ. Second step is to find target region, Ω. The target region is found by morphing the source region. The size of window used for morphing the region outwards determine the width of the band that will be predicted.

There are also some parameters used in the algorithm while expanding the source image. The main parameter is the size of the template window Ψ. We provide default window sizes 9x9, 11x11 or 13x13 pixels for different examples. In practice, the user has to set it to be slightly larger than the largest distinguishable texture element, or "texel", in the source region.

A pseudo-code description of the algorithm is shown in Table 3.1. The substring *i* indicates the current piece and superscript t indicates the current iteration.

1. Acquire the image
2. Extract the boundary of the i^{th} piece
3. Morph the source region outwards to find the expanding band
4. Set the confidence of the source region as 1, and the confidence of the expanding band or target region as 0.
5. Repeat until there exist any pixel to be filled in the target region:
    a. Compute new $D(p)$ $\forall p \in \delta\Omega_i^t$
    b. Compute priorities $P(p)$ $\forall p \in \delta\Omega_i^t$
    c. Find the patch $\Psi_{p'}$ with the maximum priority, $p' = \arg\max_{p \in \delta\Omega_i^t} P(p)$
    d. Find the exemplar $\Psi_{q'} \in \Phi$ that minimize $d(\Psi_{p'}, \Psi_{q'})$
    e. Copy image data from $\Psi_{q'}$ to $\Psi_{p'}$ $\forall p \in \Psi_{p'} \cap \Omega_i^t$
    f. Update $C(p)$ $\forall p \in \Psi_{p'} \cap \Omega_i^t$

Table 3.1 : The pseudo code of the expansion algorithm.

The output image of the pieces and the confidence values of the same pieces are saved for the next step of the project. The image will be used for calculating the feature values of the texture. The confidence values are also used in the affinity function that will be defined in the next chapter.

3.4 Results

The effect of the confidence term is that of smoothing the contour of the target region by removing sharp appendices and making the target contour close to circular. It can be noticed that inwards-pointing appendices are discouraged by the confidence term. The presence of the data term in the priority function tends to favor inwards-growing appendices in the places where structures hit the contour, thus achieving the desired structure propagation. But, as mentioned, the pixels of the target region in the proximity of those appendices are surrounded little confidence (most neighboring pixels are un-filled), and therefore, the 'push' due to image edges is mitigated by the confidence term. This achieves a graceful and automatic balance of the effects and an

organic synthesis of the target region via the mechanism of a single priority computation for all patches on the fill front.

Furthermore, since the fill of the target region is dictated solely by the priority function, $P(p)$, it is avoided having to predefined an arbitrary fill order as in existing patch-based approaches. The filling order of the algorithm is function of the image properties, resulting in an organic synthesis process that eliminates the risk of "broken – structure" artifacts and also reduces blocky artifacts without a patch-cutting step or a blur-including blending step.

Figure 3.7 : Image will be divided into four pieces, artificially.

As a result, the edges in the images shown in Figure 3.7 are kept while expanding the pieces outwards. In the images, it is represented that there are some unusual image regions. Those are not meaningful images. Actually, it is not much important because we do not want to serve the pictorial properties of the expanded band, we will use only the features of this expanded regions.

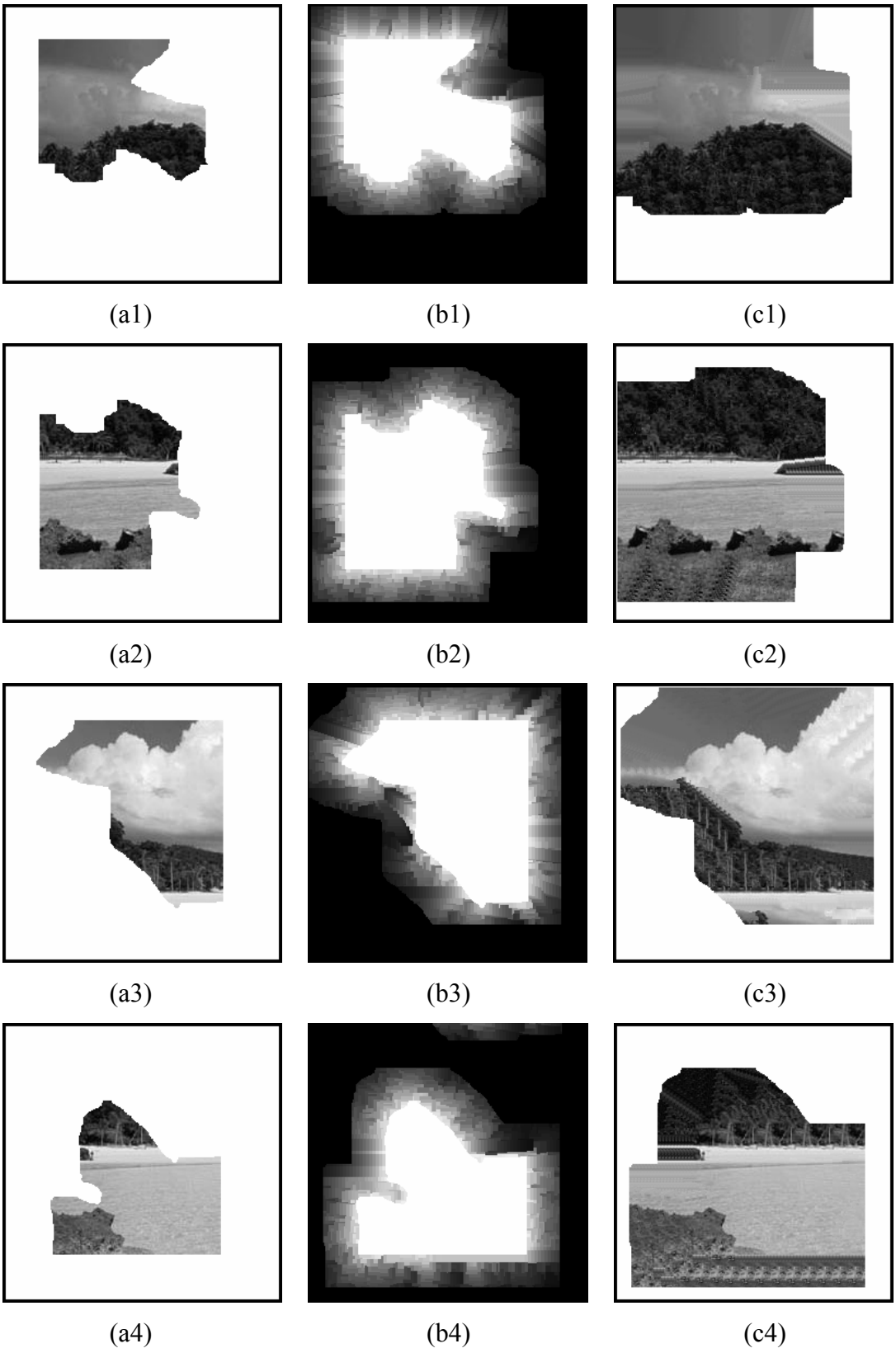


Figure 3.8 : a(1), a(2), a(3) and a(4) show the original images of the pieces. b(1), b(2), b(3), b(4) represent the corresponding confidence images. c(1), c(2), c(3), c(4) show the expanded images of the original pieces.

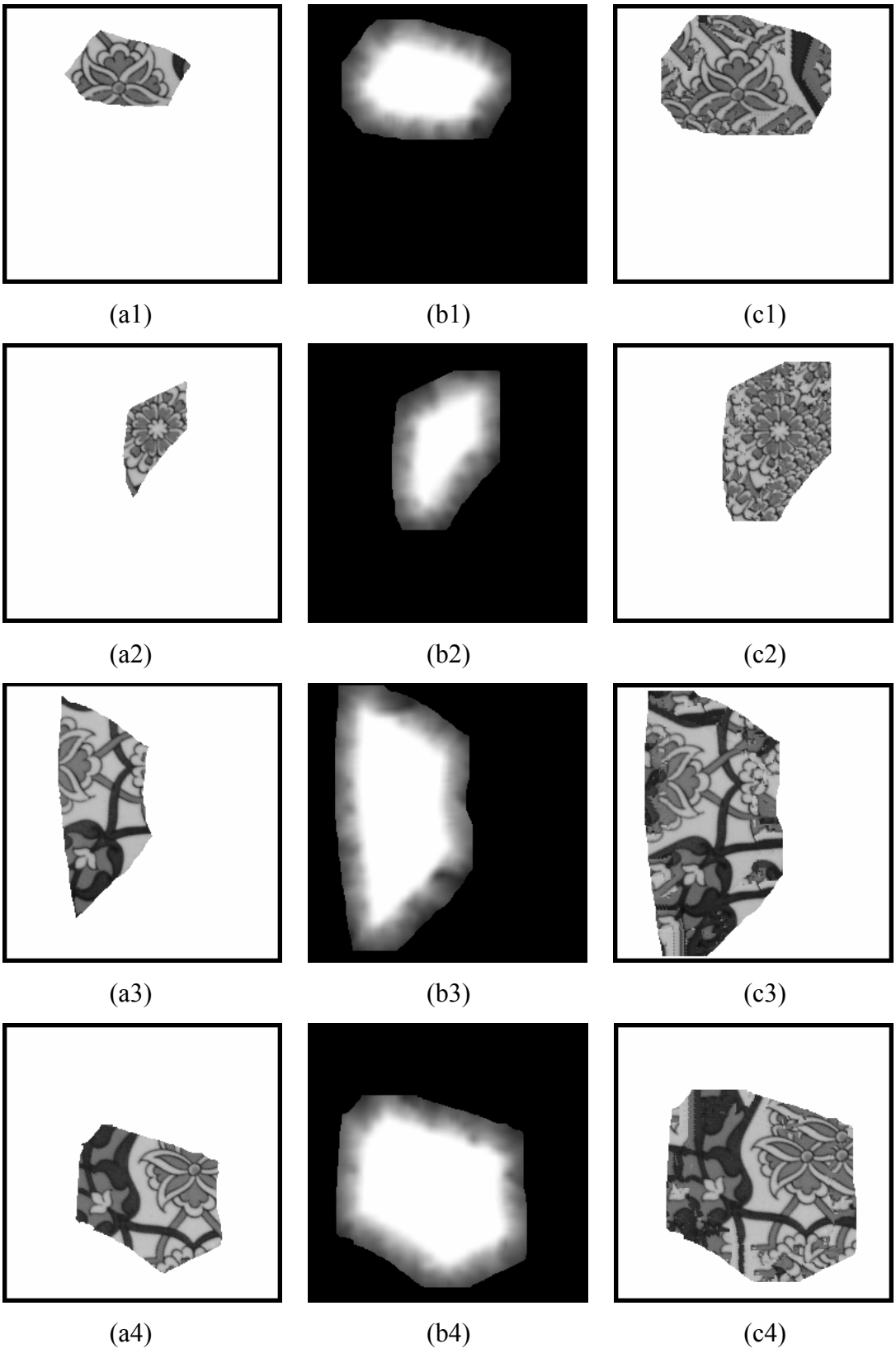


Figure 3.9 : Four pieces of a broken ceramic. a(1), a(2), a(3) and a(4) show the original images of the fragments. b(1), b(2), b(3), b(4) represent the corresponding confidence images. c(1), c(2), c(3), c(4) show the expanded images of the original pieces.

Chapter 4

AN AFFINITY MEASURE FOR COMPATIBILITY OF PIECES

4.1 Introduction

While matching or calculating the similarity of possible two neighboring pieces, pixel-by-pixel comparison of two pieces is not meaningful. Thus, image features are extracted from source and target regions for each piece after predicting the target band. Selection of the features depends on the structure of the image. The images are generally textural in classical jigsaw puzzles. In the archeological field, the images may include highly directional marble veining and paintings on the outer and inner surfaces. Additionally, we can assume the carvings and horizontal circles due to finger smoothing while the pot is spinning on the wheel and the pattern of surface incisions as a textural image.

In the next section, we will represent a short overview on texture in the images. A detailed work [19] examines the textures and makes comparative experiments using different texture algorithms. This study is also valuable that the methods are evaluated according to archaeological applications. The content of the next section is prepared mostly using of this thesis.

4.2 Texture Features

There are many fields of image processing in which texture and color play important roles. The most important areas are probably classification, image segmentation, image encoding and computer graphics. The classification of textures, often real textures, is a common problem in medical image processing and in process control. Some typical applications are the recognition of tissues in microscope images, the quality control of timber, line boards and paper, or the classification of

reconnaissance and remote sensing images. Image segmentation is related to the classification problem; when other features, like shape, tone or color, are not sufficient to discriminate between regions, one criteria of image segmentation might be texture. In image coding, the image is compressed while conserving the information. Recently there have been attempts to improve the compression rate by recognizing the texture in an image and replacing it with a symbolic representation. Computer graphics has another approach to texture. Textures are used to mimic natural scenes. The problem is reversed. How to generate a natural texture? The effort to synthesize textures has also had an effect on texture recognition. Some texture recognition methods based on texture models have been developed. There is also research being undertaken to extract three-dimensional information from two-dimensional images. This extraction is based on determining surface orientations. Surfaces are often textured. The surface orientation is determined by identifying the texture and the texture projection.

Even though texture is an intuitive concept, a formal definition of texture has proven elusive in the literature over the years. Despite this lack of a universally agreed definition, all researchers agree on two points. Firstly, there is significant variation in intensity levels between nearby pixels; that is, at the limit of resolution, there is non-homogeneity. Secondly, texture is a homogeneous property at some spatial scale larger than the resolution of the image. Some researchers describe texture in terms of the human visual system. They claim textures do not have uniform intensity, but are none-the-less perceived as homogeneous regions by a human observer. Texture is an apparently paradoxical notion. On the one hand, it is commonly used in the early processing of visual information, especially for practical classification purposes. On the other hand, no one has succeeded in producing a commonly accepted definition of texture. A complete definition for texture is really challenging, nevertheless many experts successes it over the years.

Due to its wide variability, people usually describe texture as fine, coarse, grained, smooth, etc., implying that more precise features must be defined to make machine recognition possible. Such features can be found in the tone and structure of a texture. Tone is mostly based on pixel intensity properties in the primitive, while structure is the spatial relationship between primitives. A texture primitive, a texel, is a contiguous set of pixels with some tonal and/or regional property, and can be described by its average intensity, maximum or minimum intensity, size, shape, etc. The texels may have various sizes and degrees of uniformity, may be oriented in various directions, may be spaced at

varying distances in different directions and may have various magnitudes and variations of contrast and opacity. The placement of a texel in texture could be periodic, quasi-periodic or random. Natural textures have generally random texels, whereas artificial textures have deterministic or periodic ones. The number, types of primitives and their spatial relationship describe the image texture as a global representation but it does not mean that we have optimal differentiability for all textures.

Texture tone and structure are not independent; textures always display both tone and structure even though one or the other usually dominates, and we usually speak about one or the other only. Tone can be understood as tonal properties of primitives, taking primitive spatial relationships into consideration. Structure refers to spatial relationships of primitives considering their tonal properties as well. If the texture primitives in the image are small and if the tonal differences between neighboring primitives are large, a fine texture results. If the texture primitives are larger and consist of several pixels, a coarse texture results. Note that the fine/coarse texture characteristic depends on scale. Further, textures can be classified according to their strength, which then influences also the choice of texture description method. Weak textures have small spatial interactions between primitives, and can be adequately described by frequencies of primitive types appearing in some neighborhood. Because of this, many statistical texture properties are evaluated in the description of weak textures. In strong textures, the spatial interactions between primitives are somewhat regular. To describe strong textures, the frequency of occurrence of primitive pairs in some spatial relationship may be sufficient. Strong texture recognition is usually accompanied by an exact definition of texture primitives and their spatial relationships. Most texture research can be characterized by the underlying assumptions made about the texture formation process described above. Two main texture description approaches exist: statistical (stochastic) and structural (syntactic). The choice of the assumption depends primarily on the type of textures.

Statistical methods yield characterizations of textures as smooth, coarse, grainy, etc. and are suitable if texture primitive sizes are comparable with the pixel sizes. Textures that are random in nature like sand, water and grass are well suited for statistical characterization. The structural placement paradigm for textures may also include a random aspect; from the stochastic point of view, however, we take a more extreme position and consider that the texture is a sample from a probability distribution

on the image space like noise on a television screen. The image space is usually an NxN grid and the value at each grid point is a random variable in the range {0, 1, …, G-1}.

Syntactic (structural) methods are more suitable for textures where primitives can be assigned a label, meaning that primitive type can be described using a larger variety of properties like shape, size than just tonal properties. Purely syntactic texture description models are based on the idea that textures consist of primitives located in almost regular relationships. As an example, we could think of a strictly ordered array of identical sub-patterns like a checkerboard. The sub-patterns may be of deterministic shape, such as circles, hexagons, or even dot patterns. Macro textures have large primitives, whereas micro textures are composed of small primitives. These terms are again relative to the image resolution.

The broad taxonomy map below (taken from work [19]) is derived by combining various charts from the literature, still does not claim to be an absolute truth due to the variability of the texture definitions.

TEXTURE TAXONOMY

Structural (Syntactic) Approaches	Statistical (Stochastic) Approaches	Hybrid Approches
Grammar Rules	**Spatial Domain Methods**	Textons
Periodic Texture Grammars	Gray Level Cooccurrence	
Random Texture Grammars	Gray Level Variance Matrix	
	Gray Level Gap Length Matrix	
	Gray Level Run Length Matrix	
	Neighbouring Gray Level Dependence	
	Laws Texture Energy	
	Signal Processing Methods	
	2D Gabor Filters	
	Wavelet Transformations	
	Autocorrelation Features	
	Fourier Features	
	Model Based Methods	
	Markov Random Field	
	Fractals	

Figure 4.1 : Texture taxonomy [19]

Feature extraction is the second step after predicting the expanded parts of the pieces. As it is described above, the achievement of an application based on texture directly depends on the selection of the ideal texture method. In this thesis, we proposed the assembly of the textural pieces. While we are developing the affinity measure used for assembly, we try to build a general framework on which a feature can be located.

The selection of a feature extraction method from texture exactly depends on the samples of the pieces. For example, the statistical method (histogram statistics) is better in a jigsaw puzzle. On the other hand, the structural methods (laplacian in a window) are more reasonable in a painting including directional lines on an archaeological fragment. Currently, only first and second moments (mean and variance) are used in experiments. We choose these two most common features that can be used in all cases, because the experiments with different puzzles can be comparable. In the case of using suitable texture features, the serious improvements can be observed. The features are calculated in a window. The window size directly depends on the resolution of the images on the pieces.

After selection of the features, another problem is to find the distance function that gives the appropriateness between two features. In the experiments, we used the Euclidian distance function, because of generality of the experiments as in feature selection.

4.3 Affinity Measure

The first step is to generate feature values of pieces after expanding the pieces, because using directly the pixel values are not meaningful. For example, if we have a pictorial image that has high frequency texture, the distance of the pixel values does not give the proper relation. Currently, only first and second moments (mean and variance) are used in experiments. We set a window size, which is slightly larger than the largest distinguishable texture element or "texel", to calculate mean and variance values. In the case of using suitable texture features, the important improvements can be observed. The features are calculated in a window. The window size directly depends on the resolution of the images on the pieces.

The next step is the computation of confidence values for the features from the confidence of pixels calculated in the inpainting step. When a feature value is extracted by using the pixels in a window, the confidence of this feature for a point depends on the confidences of all pixels in this window. In the algorithm, mean of all confidence of pixels in the window is assigned to confidence of feature, $C_i^{'}$. The last predefinition is to determine a sufficiently large solution board or space. It is not a theoretical, only

practical necessity. Initially, the pieces are randomly inserted onto this board. Then, the calculation of cost function and searching of transformations will be done on it.

Let $D_k(f_{ki},f_{kj})$ is the distance function between the k^{th} feature values of i and j feature values. $T_i=(\Delta x_i,\Delta y_i,\Delta\theta_i)$ denotes the transform of the i^{th} piece and $T_i(f_{ki})$ denotes transform of the k^{th} feature extracted from the i^{th} piece. For the simplicity of expressions, the $(\Delta x_i,\Delta y_i,\Delta\theta_i)$ parameter for each variable will not be shown.

In the current experiments, the Euclidian distance is used for all features. If distances specific to texture and features of pieces are selected, the performance of assembly might improve. Let define a threshold value Th_k for k^{th} feature distance.

$$S_k = D_k(f_{ki}, f_{kj}) - Th_k \tag{4.1}$$

We set a threshold, Th^k, so that the more similar the feature values are, the larger negative value the similarity measure, S^k, will take or visa versa.

$$\sum_k^{n_k} S_k = \sum_k^{n_k} \left[D_k(f_{ki}, f_{kj}) - Th_k \right] \tag{4.2}$$

where n_k is the number of features. (4.2) gives total similarity between i and j pieces. We can transform (4.2) into (4.3) by dividing all S_k into Th_k and normalizing the total constants to 1, so that both of them give related responses for the same inputs.

$$\sum_k^{n_k} (S_k / Th_k) = \sum_k^{n_k} w_k \left[D_k(f_{ki}, f_{kj}) - 1 \right] \tag{4.3}$$

w_k are weight values for the k^{th} feature and inversely proportional to Th_k. Let define the above formula for all j pieces.

$$\sum_{\substack{j \\ i \neq j}}^{n_p} \left[\sum_k^{n_k} w_k D_k(f_{ij}, f_{kj}) - 1 \right] C'_j \tag{4.4}$$

where n_p is the number of pieces in the puzzle. Expression (4.4) denotes that total similarity between i and j pieces are weighted according to j^{th} confidence values, because it should be affected when confidence of a point is small (close to zero), even if two pieces are similar. It is also valid that the cost or affinity function should be more sensitive to texture distance if confidence is high. After weighting the similarities, summation for all j pieces where i different than j shows how much the i^{th} piece fits the other pieces. Let define the above expression for all i pieces.

$$m_1(x,y) = \frac{\sum_{i}^{n_p}\sum_{j=i+1}^{n_p}\left[\sum_{k}^{n_k} w_k D_k(f_{ki}, f_{kj}) - 1\right] C'_j C'_i}{\sum_{i}^{n_p} C'_i} \tag{4.5}$$

This is the first part of Cost or Affinity function and is derived from the weighted mean of (4.4). It is the summation of similarities for possible pairs. This value goes towards negative if there exists a good harmony between images of pieces. Let insert the transforms to the above expression.

$$m_1(x,y) = \frac{\sum_{i}^{n_p}\sum_{j=i+1}^{n_p}\left[\sum_{k}^{n_k} w_k D_k(T_i(f_{ki}), T_j(f_{kj})) - 1\right] T_j(C'_j) T_i(C'_i)}{\sum_{i}^{n_p} T_i(C'_i)} \tag{4.6}$$

The variables of this expression are the transforms of the pieces. When the transforms are changed, the affinity measure in a (x,y) point also changes.

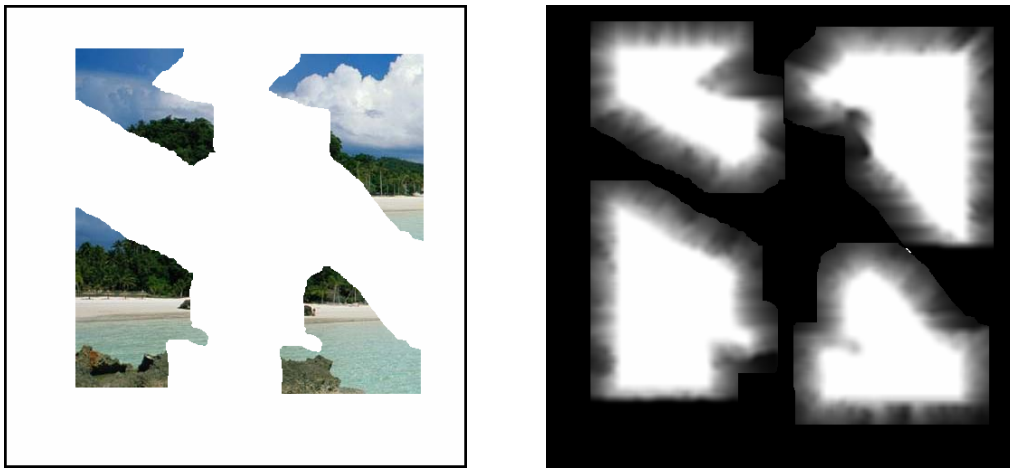
$$m_2(x,y) = \sum_{i}^{n_p}\sum_{j=i+1}^{n_p} L(T_i(C_i)) L(T_j(C_j)) \tag{4.7}$$

$$\text{where} \quad L(x) = \begin{cases} 0 & \text{if } x \neq 0 \\ 1 & \text{if } x = 0 \end{cases} \tag{4.8}$$

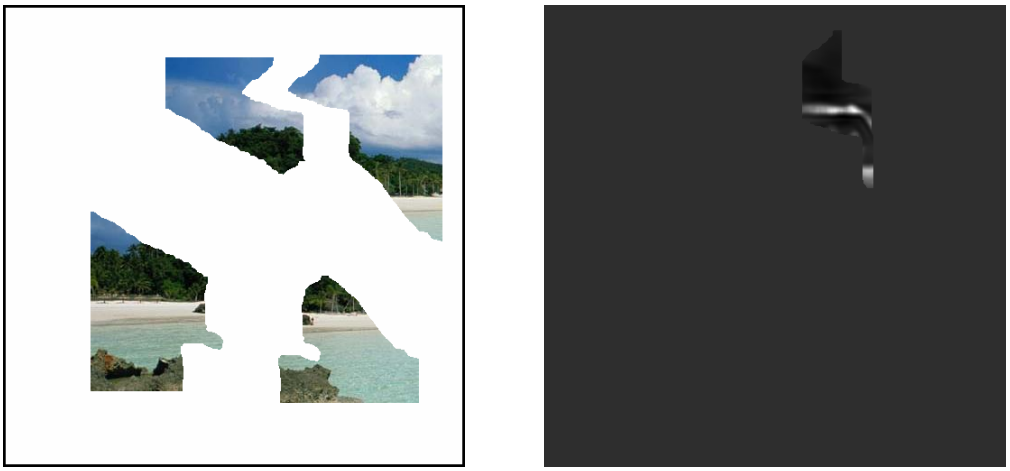
The second part of general F_{cost} function is for embedding the geometrical constraints to Cost or Affinity. In reality, two pieces cannot overlap on any point. In order to prevent this situation, a sufficiently large, w_c, weight or constant is added to Cost function in the overlapping points. The confidence values are used to formulize overlapping operation. The L function produces 1 when only the original part of image is input; otherwise it produces 0 for the predicted regions. Thus, the Cost increases when the original parts of *i* and *j* images overlap.

$$F_{cost} = \sum (m_1 + m_2) \tag{4.9}$$

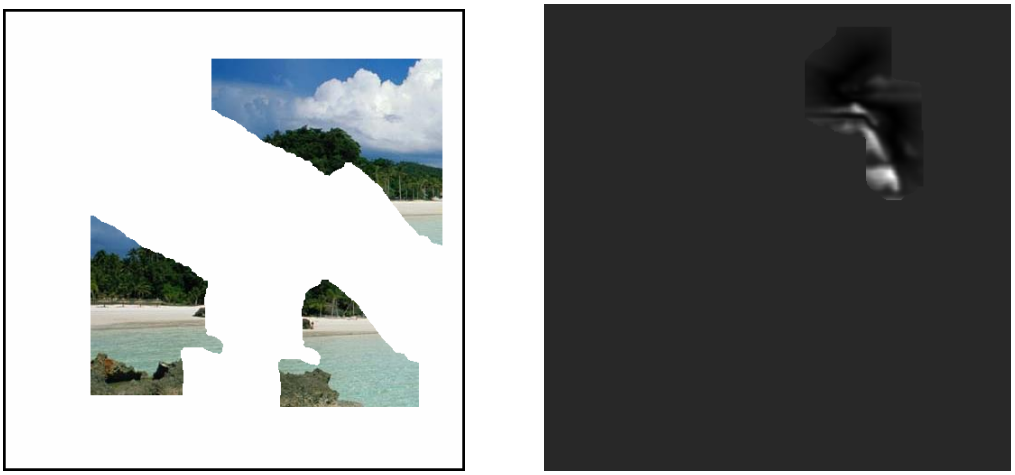
Total cost is summation of similarity and geometrical constraints terms for all points in predefined board or space. The only parameter of this performance measure that represents the goodness of assembly of pieces based on textural features and geometrical shape is the transformation of pieces, T_i.



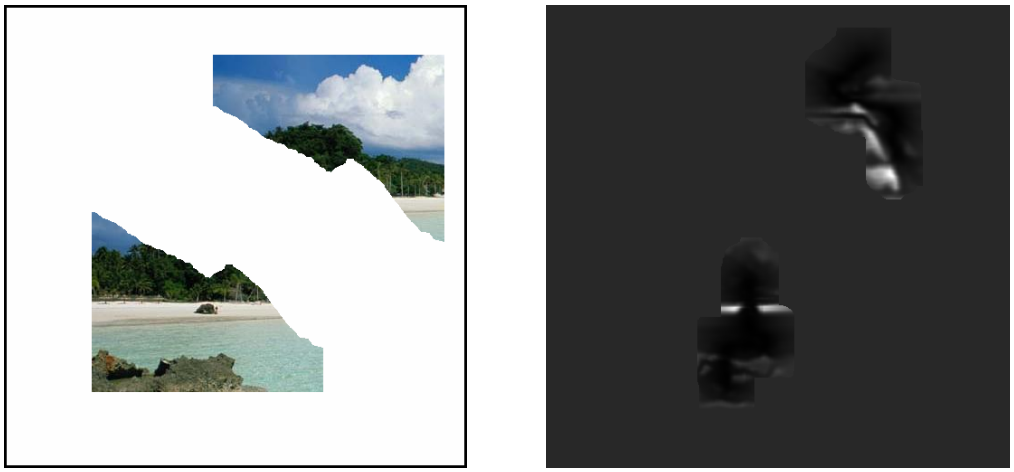
(Left) Four pieces do not intersect in any point in the space. (Right) image represents the corresponding confidence. m_1=0, m_2=0 and F_{cost}=0.



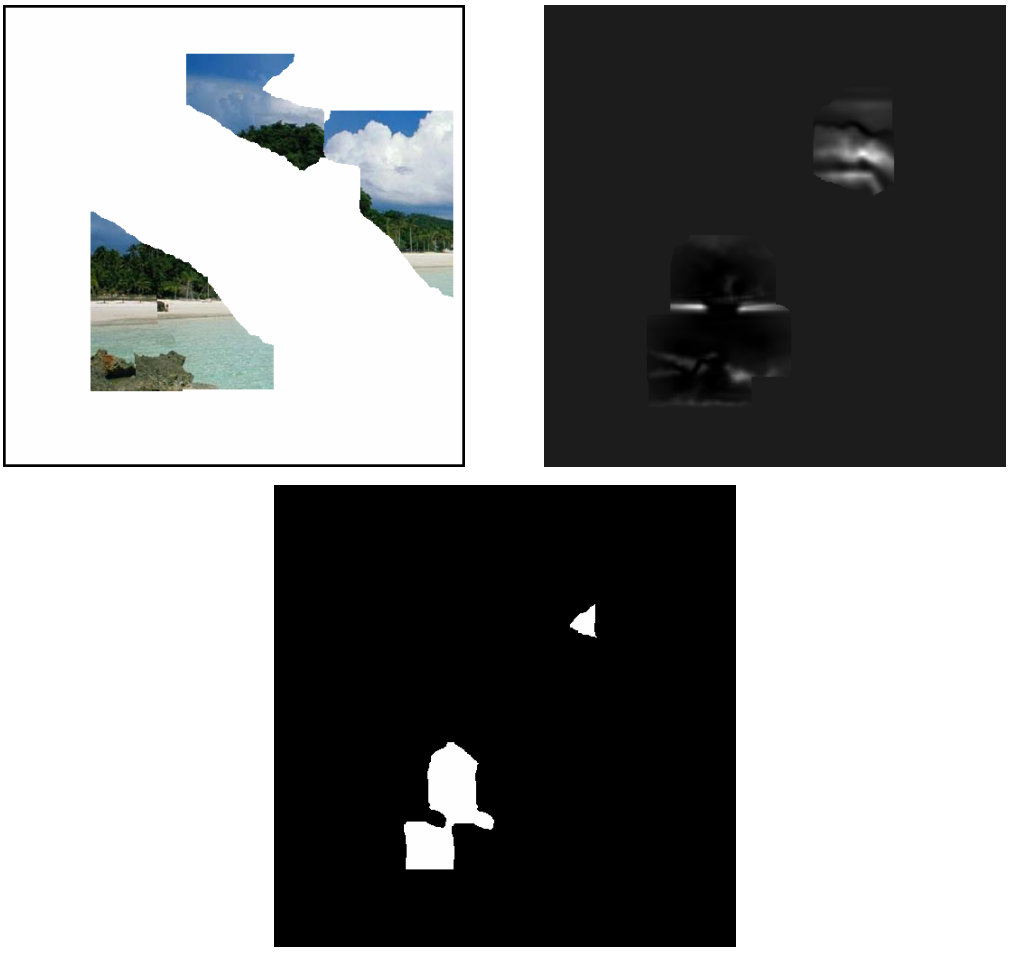
(Left) The expanded regions of first and second pieces are overlapped (Right) Image represents m_1 values for all (x,y). m_2=0 and F_{cost}=-176.



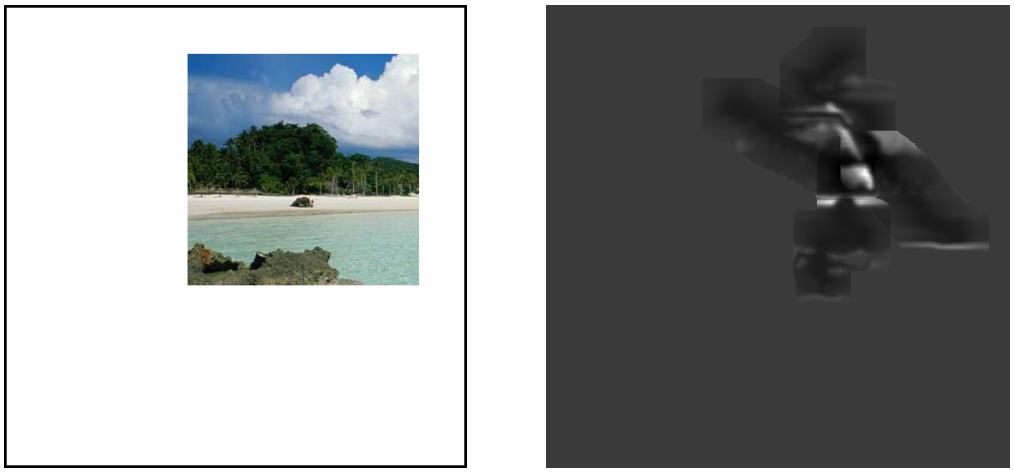
(Left) The first and second pieces are assembled. (Right) Image represents m_1 values for all (x,y). m_2=0 and F_{cost}=-988.



(Left) Two groups of the assembled pieces. (Right) Image represents m_1 values for all (x,y). m_2=0 and F_{cost}=-3552.



(Left) The original region of the pieces are overlapped. (Right) Image represents m_1 values for all (x,y). (Bottom) Image represents m_2 values for all (x,y) and F_{cost}=+77564.



(Left) The final assembly. (Right) Image represents m_1 values for all (x,y). m_2=0 and F_{cost}=-8293.

Figure 4.2 : The response of the m_1 and m_2 functions (Equations (4.6) and (4.7)) for the different assemblies.

The fitness between the pieces is increasing while the Cost function is being optimized. Two types of optimization methods might be used in experiments. The first one depends on the best replacement strategy. Initially, the transformations of pieces are randomly assigned. The algorithm progresses by finding best movement in each step. When the function is stuck into a local minimum, two randomly selected pieces are exchanged. All local minima are buffered to find best assembly. The algorithm is stopped if the function reaches the best value in the local minima buffer more than n times.

The second method depends on pairing of pieces. Initially, the algorithm searches for best pair that gives minimum cost. Then, these paired pieces are merged to produce unique piece. The algorithm is stopped when the all pieces in the puzzle are combined and become one piece. In this method, the algorithm backtracks when the pairing cannot improve the cost. To implement this method, the confidence and feature values of new piece should be defined after merging process.

$$C'_{new} = 1 - \prod_{i \in M} (1 - C'_i) \tag{4.10}$$

$$f_{knew} = \frac{\sum_{i \in M} \left[\prod_{j \in M, i \neq j} (1 - C'_j).C'_i.f_{ki} \right]}{\sum_{i \in M} \left[\prod_{i \in M, i \neq j} (1 - C'_j).C'_i \right]} \tag{4.11}$$

M is the set of pieces that will be merged. (4.10) gives the new confidence value for overlapping points of pieces. The new confidence value is equal to 1 if one of piece has a confidence of 1, otherwise it is geometrical mean of possible confidence values in that point. (4.11) gives the new k^{th} feature values by calculating the weighted mean of pieces in the set *M*.

4.4 Implementation

After the expanded pieces are generated using inpainting methods as described last chapter, the feature values are calculated and stored in a file. This file contains all information that will be used for later operations. This file includes data about the number of the pieces, the original image of each piece, the expanded image of each piece, the number of feature that will be used in assembly operations, the feature images for each feature and for each piece, the confidence image of each piece, and weights of features for affinity measure. We write two programs of which one returns the new cost value when the pieces are replaced manually. This program takes two files initially. First one is data file mentioned before. The second file is the transformation file. This file contains the temporary transformations of the pieces.

By using of this user interface, we test the consistency of numerical value of cost function with harmony of pieces. The behavior of this affinity measure is observed under the different cases. The first of them is whether the edges continue on the neighboring piece or not. In the inpainting phase, the edges obtain the higher confidence values as it was explained. The higher confidence values force the cost function to locate the pieces properly. The second important criterion is similarity of corresponding textures on the neighboring pieces. The distance measure in the cost function attracts the similar textures if the expanded regions of pieces are accurately inpainted. Providing the geometrical constraints is another property of the cost function.

The program first developed in MATLAB 7.0. Then, it is also implemented in VC++ environment. The test computer used in experiments is a PC that has 1.3 GHz and 1GB memory. To visualize the images, the OpenCV utility tool is used. Under these conditions, the affinity measure defined above can be implemented in real time. In another words, the affinity measure is changing while a piece is being moved.

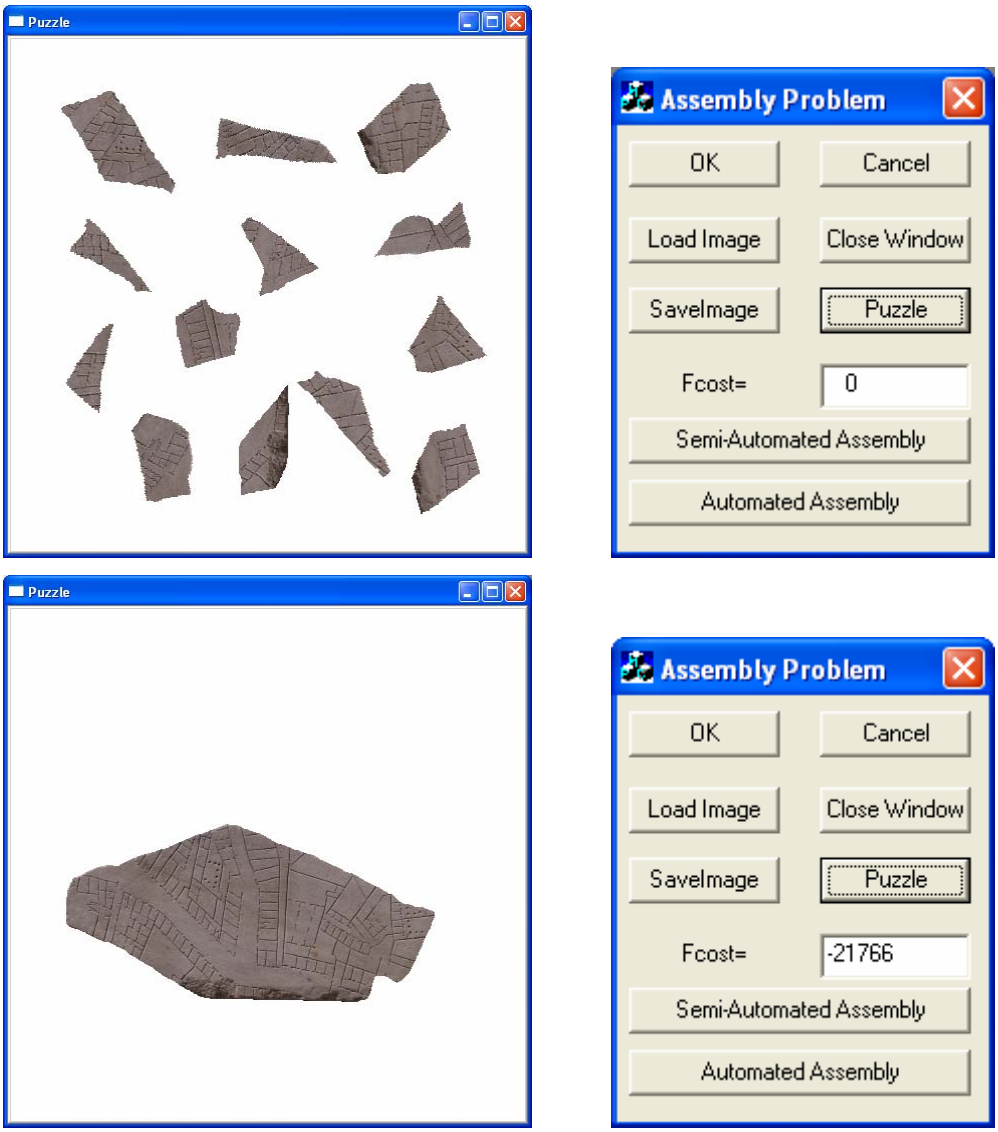


Figure 4.3 : The scenes from the developed program.

We perform a second experiment to test the consistency of the cost function. We artificially prepare puzzles including a few pieces (2,3 or 4). After that, we make an exhaustive search by running the cost function for all possible transformations of pieces. The reason of this experiment is to ascertain whether there exists any placement giving less cost value than right assembly or not. As a result of experiment, all other placements of pieces cost more than the true placement. And also, the values closer to right assembly of puzzle belongs to closer placements of pieces.

4.5 Results

The first test for affinity measure is done subjectively. More than ten different people are used to test the consistency of affinity measure by using the user interface for puzzle assembly. People move the pieces and decide to whether it is more proper arrangement. Then, they compare their decisions and the affinity measure value computed by the program. After approximately 100 trials, we want them to subjectively give a success rate for the consistency of affinity and their decisions. The mean of the success rate is 92%. Actually, this value is not sufficient to numerically prove anything. But this experiment gives an intuition that the affinity measure does not give responses independent from the visual harmony of the pieces.

F_{cost}=-8471

F_{cost}=-22536

F_{cost}=-1891

F_{cost}=-15150

F_{cost}=-17575 F_{cost}=-18967

Figure 4.4 : Images that are shown to humans.

We use two different puzzles, here. Images of four pieces from a broken ceramic are used. The algorithms are tested with the more than 30 pieces, but we prefer to represent this experiment with 4 pieces so that the details of the images can be distinguished. The other experiment (13 pieces) presented in website of Stanford university is a real image of archeological fragment. In Figure 4.4, 4.6 and 4.7, the cost values are also shown for the different placement of pieces.

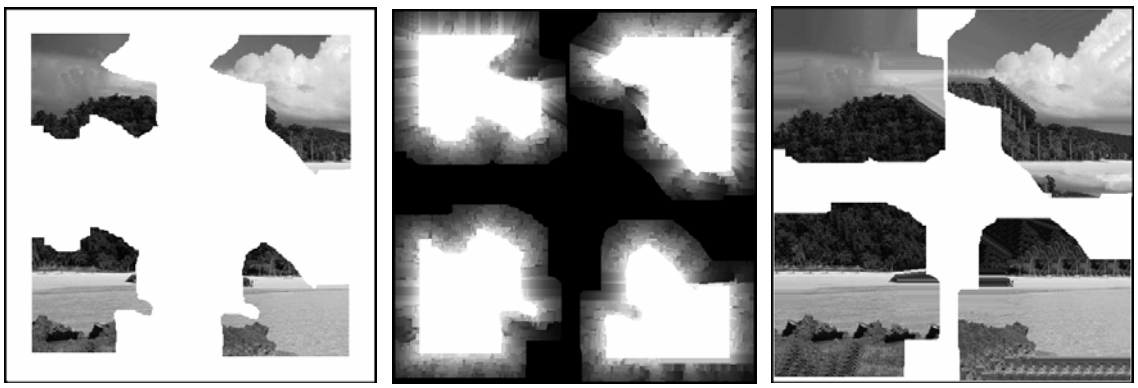


Figure 4.5 : (Left) A puzzle consisting of 4 pieces, (Middle) confidence values of the predicted regions (Right) expanded versions of the pieces. ($F_{cost} = 0$)

In Figure 4.5, the original images, confidence images and expanded images of 4 pieces are placed, respectively. The cost of puzzle in the solution space is equal to zero for this placement, because the expanded or original regions of 4 pieces are not overlapped in anywhere. In Figure 4.6, only original images of pieces are shown with 4 different assembly stages. Two right neighboring pieces are placed closer with a shift in Figure 4.6-a, and their corrected placement is represented in Figure 4.6-b. The main difference between the cost values of two assemblies is because the edges do not

continue in the first experiment although the neighboring textures are mostly similar. In Figure 4.6-c, the third piece is placed to their right positions, but original (real) regions of fourth piece and third piece are overlapped, in other words, the fourth piece violates the geometrical constraints. In this situation, the second part of cost function (m_2) becomes dominant and the cost increases seriously. In the Figure 4.6-d, the puzzle is completed by placing the pieces to their right positions.

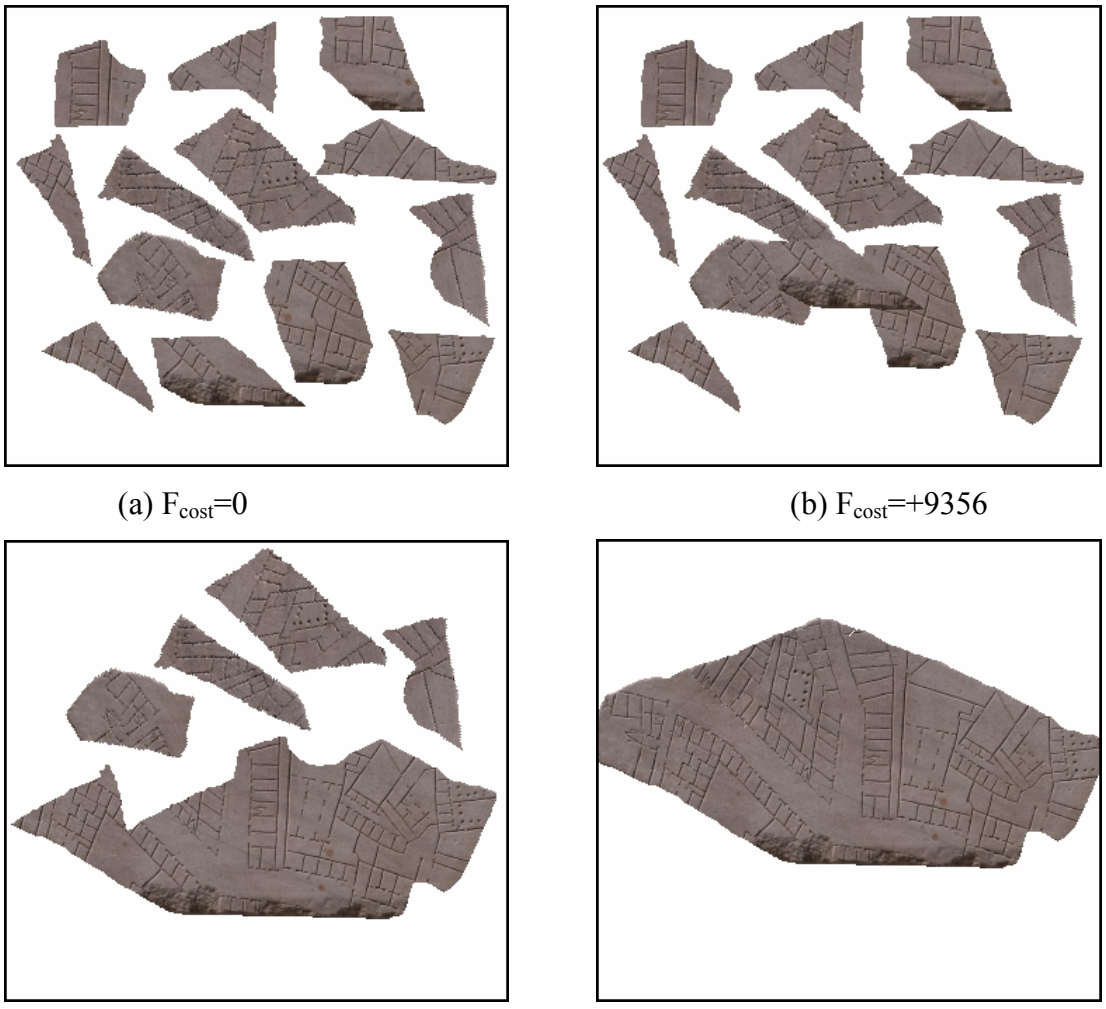
(a) F_{cost}=+269

(b) F_{cost}=-40

(a) F_{cost}=+2987

(b) F_{cost}=-1966

Figure 4.6 : (a), (b), (c) Total cost of ceramic tiles for different layouts (d)Total cost for the completed puzzle



(a) F_{cost}=0 (b) F_{cost}=+9356

(c) F_{cost}=-1554 (d) F_{cost}=-19318

Figure 4.7 : (a), (b), (c) Total cost of artificial pieces for different layouts (d) Total cost for the completed puzzle

The second test is to search all possible translation and rotation space for a piece. The aim of this experiment is to see if the affinity measure gives the best result for a meaningful replacement. The procedure for this experiment consists of two steps. The first is to locate the pieces in a related order and select a piece. Then, second step is to exhaustively search all possible translations and rotations in the space to find minimum affinity measure. A hot key is added to main program to perform this operation. After the translation and rotation that has minimum affinity measure is found, the program automatically replaces the selected piece. Then, it is manually checked the results. The all test for all puzzles give meaningful suggestions.

The second program involves optimizing the cost function to find the best assembly of the puzzle. The program uses one of the optimization method mentioned before. A hot key is also added to main program for this purpose. The program is tested

for the pieces prepared artificially. Although the edges of pieces erode or one of puzzle piece disappears, the program can find the right assembly for puzzles under test. Even the optimization works, we cannot obtain the assembly results in a reasonable time, if there are more pieces in the puzzle (n_p>20), because the backtracking occurs many times while optimizing the cost. In the case of omitting the rotation parameter from the artificial puzzles to continue the experiments for more pieces, the program finds the assembly for (n_p<40) [76][78][79].

Figure 4.8 : Completed puzzle of a ceramic tile that consists of twenty five pieces.

The complexity of affinity calculation depends on the size of the image, number of features, n_k, and the number of pieces, n_p. The equation (4.6) contains the transformations, the distance calculations, the confidence multiplications and the summations. The operations for a transformation of any point are four real multiplications and two summations. The operations of the distance calculation are one summation and one multiplication. The weighting and the subtraction of the threshold are performed by one multiplication and one summation, respectively. Thus, the total operations for the calculation of the distance function of two transformed features are ten multiplications and six summations (Total 16 operation). For all features in an (x_0,y_0) point, calculating the total distance has a complexity $O(10n_k)$. So, the complexity of the m_1 for a point becomes $O(n_p^2 \text{x}(10n_k+1)+10n_p) \approx O(C_1 n_p^2 n_k)$. Similarly, the equation (4.7) has a complexity $O(C_2 n_p^2)$. The summation of these two complexities gives the complexity of the $F_{cost}(x_0,y_0)$ function at the point (x_0,y_0) as $O(C_1 n_p^2 n_k+C_2 n_p^2)$.

Suppose that the images have a size of NxN. Then, the complexity of the total F_{cost} is $O(N^2(C_1 n_p^2 n_k + C_2 n_p^2))$.

The complexity of the optimization of the F_{cost} function by searching exhaustively all possible translations and the rotations is important to decide the following operations. All possible translation set includes NxN translation and suppose that there exists N possible discrete rotation angle. The complexity of the search operation becomes $O(N^5(C_1 n_p^2 n_k + C_2 n_p^2))$. If the number of the pieces is not very big, the dominant variable in this complexity is the size of the board. The fifth degree of the size of a board, which can contain all pieces freely, is very high value to compute. Thus, this is not feasible for practical implementation.

In the next chapter, we will introduce the Fourier methods in order to reduce the time consumption while searching for the best place for a piece in the optimization stage. Image registration methods using 2D Fourier transform can find the image correlations very fast. If it is considered that the original and expanded regions of pieces are correlated, assembly algorithm became faster by implementing mentioned method.

To improve the optimization, some other methods might also be implemented. The self-organizing map and genetic algorithm are reasonable methods for optimizing the above cost function.

Chapter 5

TEXTURE BASED PARTIAL MATCHING USING FFT TECHNIQUES

Although the puzzle assembly problem can be stated as the optimization of the above cost function, the optimization problem is too computationally costly. For all practical purposes, the minimizing of the above distance function, *D*, is equivalent to maximizing the correlation between the pieces. We will therefore use the FFT shift theory to find a solution that will maximize the correlation between the predicted parts of a piece and other pieces.

A survey of image registration techniques is detailed in [80]. Below, we briefly introduce these techniques using this survey and examine the relations between our cases and the applications described in the literature.

5.1 Image Registration Survey

Registration is the fundamental task in image processing used to match two or more pictures taken, for example, at different times, from different sensors or from different viewpoints. Techniques developed over the years have been independently studied for several different applications resulting in a large body of research. As an interesting example, the study [81] implemented the image registration to the field of archaeology.

The need to register images has arisen in many practical problems in diverse fields. Registration is often necessary for:

- Integrating information taken from different sensors;
- Finding changes in images taken at different times of under different conditions;
- Inferring three dimensional information from images in which either the camera or the scene have moved;

- Model-based object recognition.

To register two images, a transformation must be found so that each point in one image can be mapped to a point in the second. This mapping must optimally align the two images where optimality depends on what needs to be matched. The work [80] contain many examples of specific problems in registration for the four main classes of problems taken from computer vision, pattern recognition, medical image analysis and remotely sensed data processing. These classes are multimodal registration that finds the transformations of the images of the same scene acquired from different sensors, template registration that finds a match for a reference pattern in an image, viewpoint registration that finds the replacement of the images taken from different viewpoints and temporal registration that registers the images of the same scene taken from at different times or under different conditions. Our situation has more similarities with template registration problems than the others. According to our assumption, the texture of the expanded region of a piece is nearly identical to the texture on the adjacent piece. So, the best matching of these two pieces is obtained if the templates on the pieces are able to be registered.

Problem definition of image registration:

Image registration is defined as a mapping between two images both spatially and with respect to intensity. Let define these two images as 2D arrays of a given size denoted I_1 and I_2 where $I_1(x,y)$ and $I_2(x,y)$ each map to their respective intensity values, then the mapping between images can be expressed as:

$$I_2(x,y) = g(I_1(T_1(x,y)) \tag{5.1}$$

where T_1 is a 2D spatial coordinate transformation, i.e.,

$$(x',y') = T_1(x,y) \tag{5.2}$$

and g is an intensity transformation. The registration problem is the task involved in finding the optimal spatial and intensity transformations so that the images are matched with regard to the misregistration source. The intensity transformation is frequently not necessary as in our fragment assembly problem. It is also generally expressed parametrically as single –valued functions, T_{1x}, T_{2x}:

$$I_2(x,y) = I_1(T_{1x}(x,y), T_{1y}(x,y)) \tag{5.3}$$

which may be more naturally implemented. In the thesis, we usually use shorter expressions like:

$$I_2(x,y) = I_1(T_1(x,y)) \quad \text{or} \quad I_2 = T_1(I_1) \tag{5.4}$$

Transformations:

The most fundamental characteristic of any image registration technique is the type of spatial transformation or mapping needed to properly overlay two images. Although many types of distortion may be present in each image, the registration technique must select the class of transformation which will remove only the spatial distortions between images due to differences in acquisition and not due to differences in scene characteristics that are to be detected. The primary general transformations are affine, projective, perspective, and polynomial. These are well-defined mappings of one image onto another.

Here, we will briefly define the different transformations that may be deal with our projects. A transformation is affine if *T(x)-T(0)* is linear. Affine transformations are linear, however, in the sense that they map straight lines into straight lines. The most commonly used registration transformation is the affine transformation which is sufficient to match two images of a scene taken from the same viewing angle but from a different position. This affine transformation is composed of the Cartesian operations of scaling, a transformation, and a rotation. It is a global transformation that is rigid since the overall geometric relations between points do not change. It is typically has four parameters, t_x, t_y, s, θ, which map a point (x_1,y_1) of the first image to a point (x_2y_2) of the second image as follows:

$$\begin{pmatrix} x_2 \\ y_2 \end{pmatrix} = \begin{pmatrix} t_x \\ t_y \end{pmatrix} + s * \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} x_1 \\ y_1 \end{pmatrix} \tag{5.5}$$

The general 2D affine transformation has 6 parameters that can account for other spatial distortions, as well such as skew and aspect ratio. But this general term is not necessary in our case. The acquisition of the fragments is a controllable process for the puzzle problem. The pieces can be placed and captured in an ideal environment. Because the light conditions and the other factors that cause noise and distortion can be minimized, the 4 parameter rigid transform (5.5) is suitable in our situation. Actually, the constant s is also invariable. While acquiring the pieces, the real sizes can be obtained by the devices. So we will not use the scaling factor as a variable of our problem. In 2D, three parameters (t_x, t_y, θ) have to be estimated.

$$\begin{aligned} x_2 &= t_x + x_1\cos\theta - y_1\sin\theta \\ y_2 &= t_y + x_1\sin\theta + y_1\cos\theta \end{aligned} \tag{5.6}$$

The perspective transformation, accounting for the distortion that occurs when a 3D scene is projected, and other transformations are not directly related to our work. The descriptions are detailed in the surveys [80][82].

Registration methods can be categorized with respect to various criteria. The ones usually used are the application area, dimensionality of data, type and complexity of assumed image deformations, computational cost and the essential ideas of registration algorithm. Here, we used Fourier methods in our applications. This technique is a subtitle of area-based methods in registration literature. We do not contemplate detailing other methods or describing results of comparisons of the algorithms; rather we will focus our assembly problem and the implementation of the Fourier method in this work. In the studies [80][82][83], all registrations techniques are explained and the comparative experiments are described. And papers [84][85][86] also introduce 2D and 3D registration methods based on Fourier shift theory and applications in the several fields.

5.1.1 Fourier Method for Image Registration

The Fourier transform has several properties that can be exploited for image registration. There are equivalent operations of all transforms like translation, rotation, reflection and scale in the Fourier domain. Furthermore, Fourier domain operations have excellent robustness against correlated and frequency dependent noise. Another main advantage of the Fourier operations is that the transform can be efficiently implemented in hardware or alternatively can be used the Fast Fourier Transform (FFT). We will describe the main methods used to register images using Fourier analysis in this section.

Phase correlation between two images exposes the shifting relative to one another. So it is used to align the images. Phase correlation relies on the translation property of the Fourier transform, sometimes referred to as the Shift Theory. For example, we have two images I_1 and I_2 which differ only by a distance placement (d_x,d_y);

$$I_2(x,y) = I_1(x-d_x, y-d_y) \tag{5.7}$$

Their Fourier transforms F_1 and F_2 are related by:

$$F_2(\omega_x,\omega_y) = e^{-j(\omega_x d_x+\omega_y d_y)} F_1(\omega_x,\omega_y) \tag{5.8}$$

These two images have the same Fourier magnitude but a phase difference directly related to the displacement. The inverse Fourier transform of the phase difference is a delta function centered at the displacement, which in this case, is the point of registration. In implementation, the continuous transform must be replaced by the discrete Fourier transform and the delta function becomes an impulse function.

If I_1 and I_2 images are not identical images or they are noisy, the Fourier transforms differ. So, we have to estimate the best phase difference in those situations. The above function defines the optimized phase difference between two images.

$$\frac{F_1(\omega_x,\omega_y).F_2^*(\omega_x,\omega_y)}{\left|F_1(\omega_x,\omega_y).F_2^*(\omega_x,\omega_y)\right|} = e^{(\omega_x d_x+\omega_y d_y)} \tag{5.9}$$

Here, (5.9) is the computed values of the cross power spectrum of the images. The inverse of this expression gives the peaks in the high correlation regions, and location that has the peak value in the inverse Fourier transform of the (5.9) function indicates the translation parameters. The derivation of the above expression (5.9) is presented in Appendix A. In our implementations for assembly problem, we used this method. The usage of the method will be described in the next section. We will introduce the properties and the advantages of the Fourier based registration method in the remaining part of this section.

Since the phase difference for every frequency contributes equally, this technique is particularly well suited to images with narrow bandwidth noise. Consequently, it is an effective technique for images obtained under differing conditions of illumination since illumination functions are usually slow varying and therefore concentrated at low spatial frequencies. Actually, the illumination is not a critical parameter in the archaeological fragments, because it is possible to acquire the images of the pieces in ideal conditions. However, the corruption in the archaeological fragments causes a change in the textural information of the pieces. This corruption generally has lower frequencies. For example, the one region of the fragment may be a few darker than another region and there are soft transitions between these regions. In these cases, the Fourier registration gives well responses.

Similarly, the technique is relatively scene independent and useful for images acquired from different sensors since it is insensitive to changes in the spectral energy. This property of using only the phase information for the correlation is sometimes referred to as a whitening of each image. Among other things, whitening is invariant to

linear changes in brightness and makes the correlation measure relatively scene independent. With respect to above comments regarding illumination, the different pieces may have different brightness because of the corruption. If we prefer a method that assemblies the fragments according to the exact similarities of the adjacent pieces, it may give inadequate results. Thus, the Fourier registration method has also advantages in such cases. On the hand, if the white noise immunity is desired, a weighting function can be selected before taking the inverse Fourier transform.

Certain assumptions underlie the use of the Fourier transform, which should not be overlooked. Since the images are bounded and discrete, frequency information is also bounded and discrete. By sampling theorem, in using the Fourier transform, it has been assumed that the images are bandlimited and periodic with the image size. In general, images are preprocessed in order to make these assumptions more valid. For example, a smoothing function can be applied to limit the bandwidth. In our application, we do not use a preprocessing. Because we apply the feature values to the Fourier transform, the values applied to the transform are already windowed in the feature generation phase. In another words, the fast transitions in the feature image values do not exist. So, the limited band precondition is partially obtained in feature generation phase.

In extension of the phase correlation technique, [87] has proposed a technique to register image, which are both translated and rotated with respect to each other. Rotational movement, by itself without translation, can be deduced in a similar manner as translation using phase correlation by representing the rotation as a translation displacement with polar coordinates. But translation and rotation together represent a more complicated transformation. In [87], it is suggested that the first angle of rotation is determined and then the translation shift is detected.

Rotation is invariant for the Fourier transform. Rotating the image with an angle in the spatial domain is equal to rotating in the Fourier domain with the same angle. Two images $I_1(x,y)$ and $I_2(x,y)$ which differ by a translation (x_d,y_d) and a rotation ϕ_0 will have Fourier transforms related by

$$F_2(\omega_x,\omega_y) = e^{-j(\omega_x d_x+\omega_y d_y)} F_1(\omega_x \cos\phi_0 + \omega_y \sin\phi_0, -\omega_x \sin\phi_0 + \omega_y \cos\phi_0) \qquad (5.10)$$

By taking the phase of the cross-power spectrum as a function of the rotation angle estimate ϕ and using polar coordinates to simplify the equation we have

$$G(r,\theta;\phi) = \frac{F_1(r,\theta)F_2^*(r,\theta-\phi)}{\left|F_1(r,\theta)F_2^*(r,\theta-\phi)\right|} \qquad (5.11)$$

Therefore, by first determining the angle ϕ which makes the phase of the cross-power spectrum the closest approximation to a unity pulse, we can then determine the translation as the location of this pulse.

Also scaling can be a parameter of the registration. In this situation, log polar transforms can be used. In paper [88], a method using the log polar transformation is developed. In our assembly problem, we use similar registration method to fast match the pieces. The derivation and the usage details of log polar transforms are represented in Appendix B.

While implementing the above method, it should be noted that some form of interpolation must be used to find the values of the transform after rotation since they do not naturally fall in the discrete grid. This might be accomplished by computing the transform after first rotating in the spatial domain. Although this solution is too costly for most applications, it can be used in systems using parallel computing. Another solution for the same problem is to apply the transform to a zero padded image thus increasing the resolution an improving the approximation of the transform after rotation [87]. Other interpolation techniques, for instance, the nearest neighbor and bilinear interpolation, proved to be unsatisfactory. Their method is also costly because of the difficulty in testing for each angle. In implementations for our problem, we used both methods that rotate the image before transformation and after transformation with zero padding. For the first method, it is slower but the results are better. In the second method, it is faster but it may fail in some cases.

The Fourier methods, as a class, offer advantages in noise sensitivity and computational complexity. Many techniques similar to previous methods are developed. For example, an algorithm is used to register images using the power cepstrum (the power spectrum of the logarithm of the power spectrum). First images are made parallel by determining the angle, which minimizes the differences in their power spectra. Then, the power spectrum is used to determine the translation correspondence in a similar manner to phase correlation. This has the advantages over the previous methods of the computational savings gained by adding images instead of multiplying them due to the use of logarithms.

All Fourier methods described above achieve better accuracy and robustness than the classical correlation algorithms. However, they are only applicable for certain well-defined transformations such as rotation and translation, because the Fourier methods rely on their invariant properties. In many applications, the registration methods are

designed to overcome more complex transformations. Thus, the Fourier methods may not be preferred. But we need only rigid transformations (only rotation and translation) in our assembly problem. So, the Fourier algorithms are more applicable and more efficient (according to computational complexity) than the others.

As we mentioned before, we use only rigid transformations. At the same time, the scaling is not a variable parameter of our solution. The exact imaging can be captured in acquisition by using proper resolution. So, it is sufficient to use polar coordinates without logarithmic computations. In Appendix C, the implementation of polar coordinates without logarithmic scale is derived and the rotation problem is changed from 2D registration to1D registration problem.

5.2 FFT Based Solution

For the clarity of the explanation, we will describe the FFT based solution step by step. In the first step, we will define the easiest puzzle problem and its solution. Then, we will add one more property in each step. At the last step, we will expand the solution for the real puzzle problems.

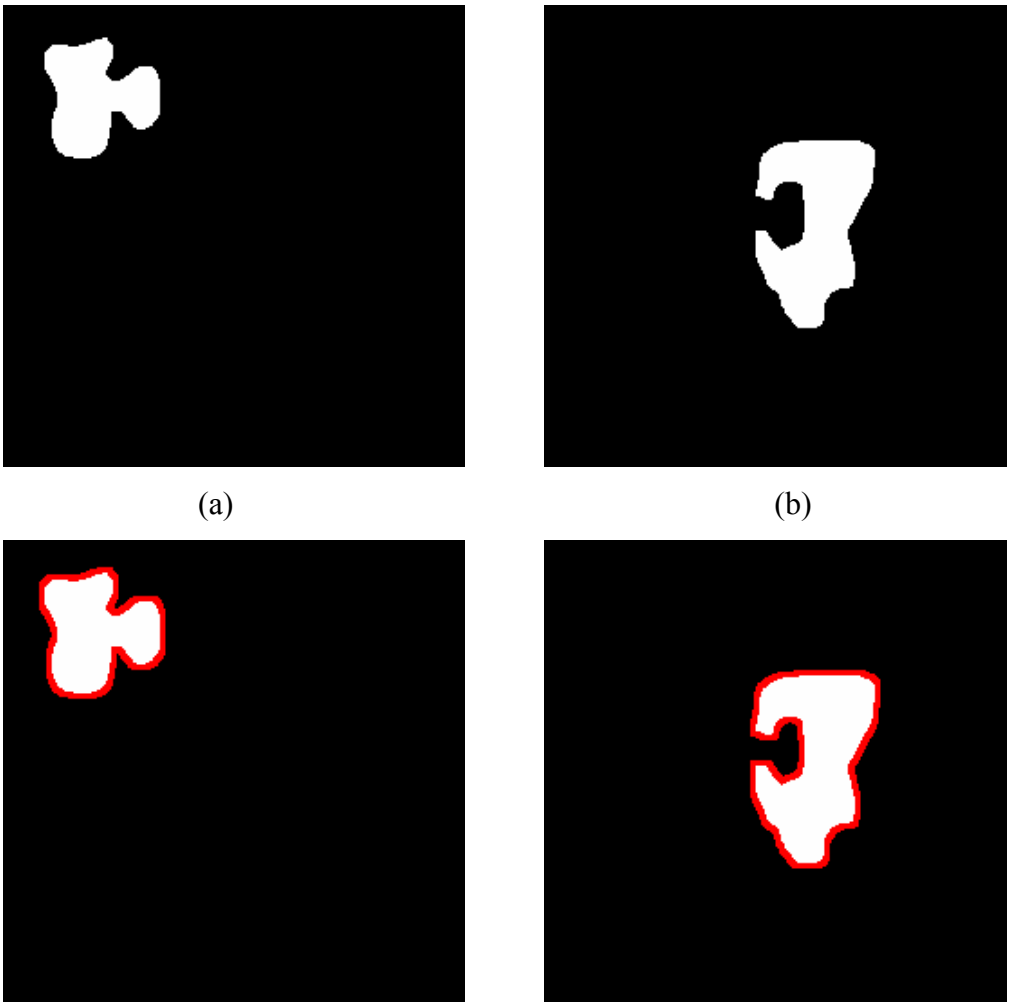
Case 1 : Assumptions of the easiest puzzle problem:

- There are only two pieces in the puzzle
- The puzzle pieces are two-dimensional
- The puzzle pieces do not contain texture, we will use only the shape information
- The pieces are not rotated; all pieces are in their correct orientation
- Only boundaries give the relation between pieces

Definition of the easiest puzzle problem: Let us first consider the solution to a 2 pieces puzzle. The solution set consists of the piece I_0 and the transformed version of the piece I_1. These images are black and white images.

Solution: First step is to expand the images. The expanding operation is equivalent to morph images outwards, because the images are black and white. The window size that also determines the expanding band width is used to filter the images. This operation gives the expanded images, I_0^+ and I_1^+.

The transformation consists of translation (T=(Δx,Δy)). The transformation that gives maximum correlation between I_1^+ and I_0 (and also I_0^+ and I_1) is the best match between the two pieces and, hence, is the solution defined below:



(a) (b)

(c) (d)

Figure 5.1 : (a) and (b) are two pieces from a puzzle and (c), (d) are their expanded forms, respectively. Red band shows the morphed region.

$$S^2_{\text{general}} = \underset{T}{\text{argmax}} \left(C(I_0, T(I_1^+)) + C(I_0^+, T(I_1)) \right) \tag{5.12}$$

C denotes the correlation operator. The S expresses the solution set. The superscript is used to symbolize number of pieces in the puzzle. The subscript general term is used, because the maximum correlation solution does not guarantee the real solution to the puzzle, since it does not incorporate the physical constraint that two pieces cannot overlap. This constraint can be expressed in terms of correlations:

$$(I_0) \cap (T(I_1)) = \varnothing \tag{5.13}$$

This constraint is also written by using the correlation operation like:

$$C(I_0, T(I_1)) = 0 \tag{5.14}$$

Hence the real solution set is given by:

$$S^2_{real} = \left\{ \underset{T}{\text{argmax}} \left(C(I_0, T(I_1^+)) + C(I_0^+, T(I_1)) \right) \middle|\ C(I_0, T(I_1)) = 0 \right\} \tag{5.15}$$

The solution set lies where the original pieces have correlation 0, and the I_1^{th} piece has maximum correlation with the I_0^{th} piece. These correlations can be carried out very fast using FFT operations as in image registration methods [17]. (See also Appendix A for the derivation of the below formula)

$$S^2_{\text{general}} \equiv \text{imax} \left(\overline{F}\left(\frac{F\ (I_0).F^*(I_1^+)}{\left|F\ (I_0)\right|.\left|F^*(I_1^+)\right|} \right) + \overline{F}\left(\frac{F\ (I_0^+).F^*(I_1)}{\left|F\ (I_0^+)\right|.\left|F^*(I_1)\right|} \right) \right) \tag{5.16}$$

$$\text{and also} \qquad C(I_0, I_1) = \overline{F}(F(I_0).F^*(I_1)) \tag{5.17}$$

where imax returns the indices of the max value. F, F^* and $\overline{F}$ denote the Fourier operator, its complex conjugate and the inverse Fourier operator, respectively. If we substitute (5.16) and (5.17) in (5.15):

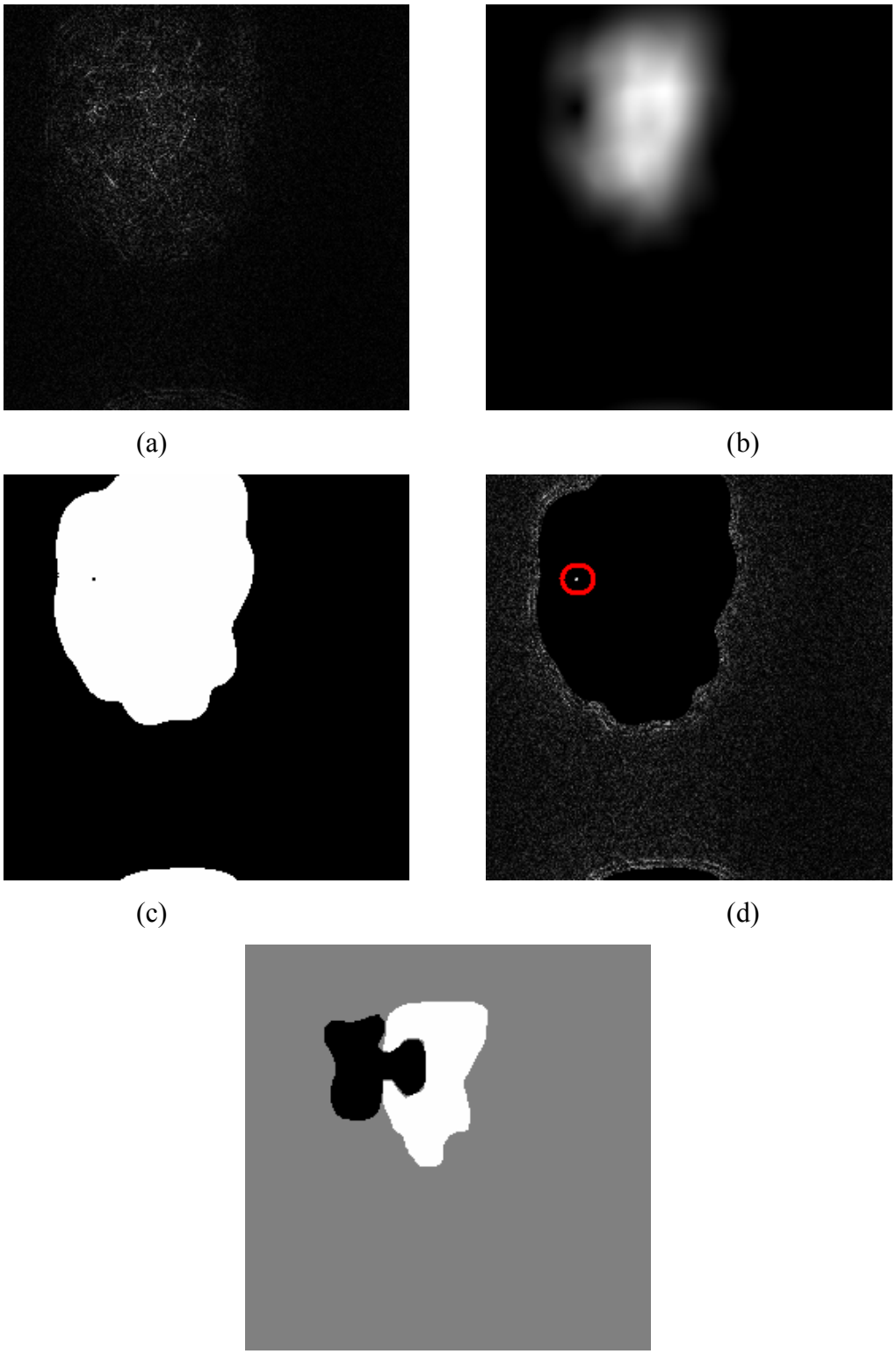
$$S^2_{\text{real}} \equiv \text{imax} \left(\left(\overline{F}\left(\frac{F\ (I_0).F^*(I_1^+)}{\left|F\ (I_0)\right|.\left|F^*(I_1^+)\right|} \right) + \overline{F}\left(\frac{F\ (I_0^+).F^*(I_1)}{\left|F\ (I_0^+)\right|.\left|F^*(I_1)\right|} \right) \right) .L\left(\overline{F}(F(I_0).F^*(I_1)) \right) \right) \tag{5.18}$$

$$\text{where } L[x] = \begin{cases} 0 & \text{if } x \neq 0 \\ 1 & \text{if } x = 0 \end{cases} \tag{5.19}$$

When the inverse Fourier value in the second part of the formula is zero, the maximum values of the first part of the formula gives the ideal transformation parameters. By using of the linearity of Fourier operations, (5.18) can be written as:

$$S^2_{\text{real}} \equiv \text{imax} \left(\overline{F}\left(\frac{F\ (I_0).F^*(I_1^+)}{\left|F\ (I_0)\right|.\left|F^*(I_1^+)\right|} + \frac{F\ (I_0^+).F^*(I_1)}{\left|F\ (I_0^+)\right|.\left|F^*(I_1)\right|} \right) .L\left(\overline{F}(F(I_0).F^*(I_1)) \right) \right) \tag{5.20}$$

Figure 5.1 shows a two-pieces puzzle. In Figure 5.1-a,b, there are two original pieces. In Figure 5.1c,d, the expanded pieces are shown. The image in Figure 5.2-a represents the correlation matrix between these two expanded pieces. This is the general solution for the problem. If the constraint expression is applied, however, the image in Figure 5.2-d is found. This matrix is the output of the inner part of expression (5.20). The red circle in Figure 5.2-d indicates the maximum point in this matrix. The indices of this maximum point give the translation coefficients. The last image in Figure 5.2-e shows the solution if this translation is applied to the second piece.



(a) (b)

(c) (d)

(e)

Figure 5.2 : (a) The correlation matrix between the original regions and the expanded regions or the inner part of the $S_{general}$ (b) The correlation matrix of the original regions $C(I_0,I_1)$ (c) represented the impossible translations or signature of the $C(I_0,I_1)$, $L(C(I_0,I_1))$ (d) The possible correlations or the inner part of the S_{real} and the red circle shows the maximum point (e) The final solution the puzzle.

From another point of view, this solution is also similar to identifying the maximum overlapping area between expanded and original regions. This approach shows us that the method developed above is meaningful and valid, when the sizes of the pieces are not considerably different. As we mentioned before, we prefer this method, because FFT based method is very fast so it is reasonable to implement in a search algorithm.

Suppose that the size of the images is NxN. The computation of the correlation between these two images has a complexity of $O(C_1N^4)$ in spatial domain, and the computation of the constraint term has also a complexity $O(C_1N^4)$ (Totally $O(2C_1N^4)$). However, the computation in Fourier domain, as defined before, is easier. In order to use these easier operations, we assume that the size of the images or the board, N, is a power of two. As we know, the complexity of a 2D FFT is $O(CN^2\log(N))$. However, there is a difference between the primitive computations in spatial domain and in the Fourier domain. In Fourier domain, we use complex operation other than real operations. We can express this difference as $C\approx 5C_1$ (then, the complexity of the 2D FFT is $O(5C_1N^2\log(N))$.

In the expression (5.20) for the case 1, the proposed matching method requires computation of four FFT and two inverse FFT. Other than the FFT's, the remaining tasks (seven complex and one real multiplication, one complex summation and one sign operation) are all $O(C_1N^2)$ (Total $O(42\ C_1N^2)$). The total complexity for (5.20) becomes $O(30C_1N^2\log(N)+42C_1N^2)$. Note that the complexity for the calculations of the correlations in spatial domain is greater than the one in Fourier domain ($O(2C_1N^4)>>O(30C_1N^2\log(N)+42C_1N^2)$: Using the Fourier domain is efficient after $N\geq 16$).

Case 2 : Removing the "no rotation" assumption: The new assumptions are:

- There are only two pieces in the puzzle
- The puzzle pieces are two-dimensional
- The puzzle pieces do not contain texture, we will use only the shape information
- Only boundaries give the relation between pieces

Definition of the problem: Let us first consider the solution to a 2 pieces puzzle. The solution set consists of the piece I_0 and the transformed version of the piece I_1. These images are black and white images.

Solution: First step is to expand the images as it is described previously. This operation gives the expanded images, I_0^+ and I_1^+.

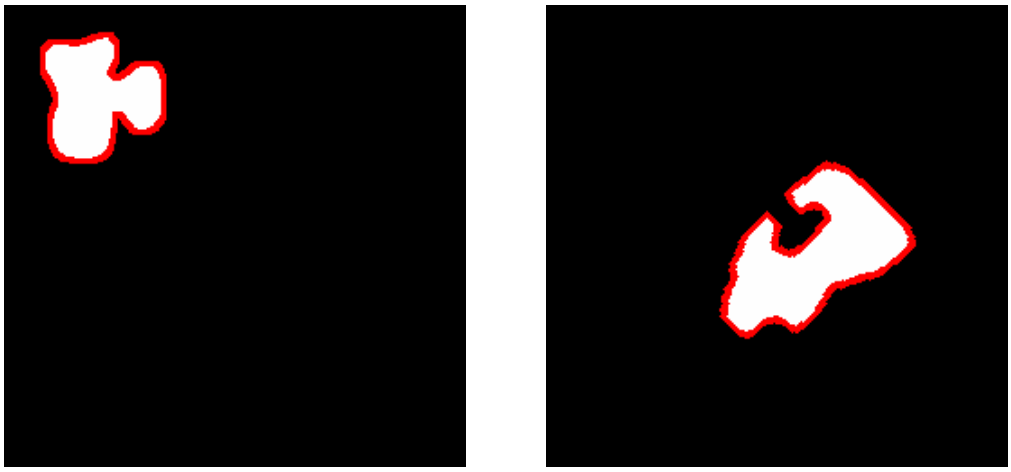


Figure 5.3 : Two pieces puzzle. The second piece (right) is rotated. Red lines represent the morphed regions.

The transformation consists of translation (T=(Δx,Δy,$\Delta\theta$)). The transformation that gives maximum correlation between I_1^+ and I_0 (and also I_0^+ and I_1) is the best match between the two pieces. All expressions are same as (5.15).

The main difference between the current problem and previous one is in the FFT phase. We can not find the translation and rotation with an operation as in (5.20). There are two possible ways to overcome this problem. First one is to try all possible discrete degrees for the best fitting. This method is slow but more accurate. This is an ordinary method, but it can be preferred in some cases. For example, the solution can be obtained more accurate and faster, if we use parallel computers. Because the formula (5.20) is run in each computer with different rotations, and then the best rotation and translation value is found in one step.

The second method is more formal and algorithmic. First polar transformation is used to generate new images that are suitable to implement to the formula (5.20). Then, use an iterative algorithm to find the translation and rotation at the same time. This method is faster than the previous exhaustive search method if it is worked in a sequential computation environment. The details of this iterative algorithm are described in the research [88]. Polar and log-polar transformation and determination of rotation and scaling are detailed in the Appendix B.

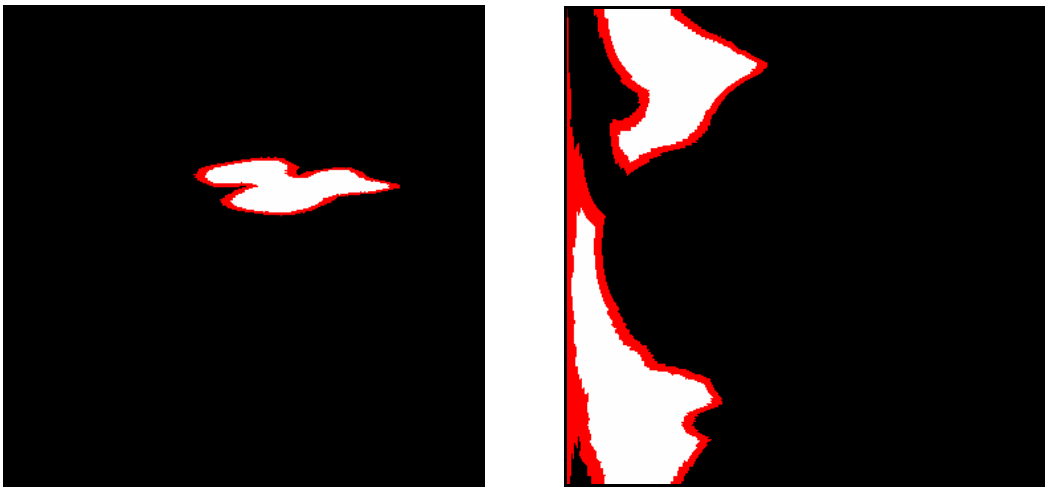
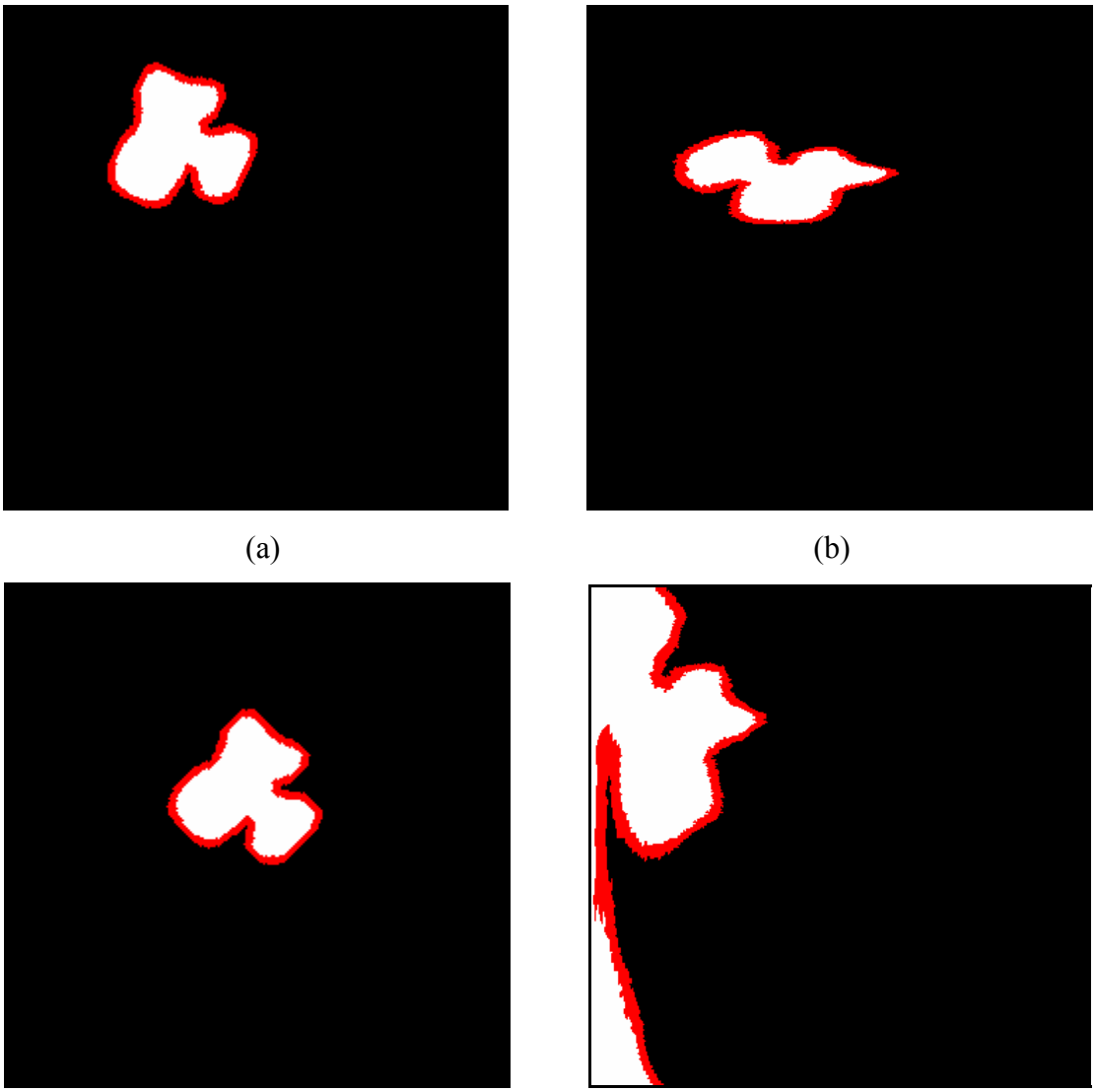


Figure 5.4 : The first (left) and second (right) image transformed to the polar coordinates.

Actually, the derivations in Appendix B cover the scaling and rotation. We can ideally acquire the fragments; hence the scaling is not a variable in a puzzle problem. So, we do not need to use log-polar coordinates. We use directly polar coordinates instead of log-polar coordinates.

$$\theta = \text{imax}\left(\overline{F_1}\left(\frac{F_1\ (I_0{}^T(r=r_0,\theta)).F_1^*(I_1{}^T(r=r_0,\theta))}{\left|F_1\ (I_0{}^T(r=r_0,\theta))\right|.\left|F_1^*(I_1{}^T(r=r_0,\theta))\right|} + \right.\right.$$
$$\frac{F_1\ (I_0{}^T(r=r_1,\theta)).F_1^*(I_1{}^T(r=r_1,\theta))}{\left|F_1\ (I_0{}^T(r=r_1,\theta))\right|.\left|F_1^*(I_1{}^T(r=r_1,\theta))\right|} + ... \qquad (5.21)$$
$$\left.\left. ..... + \frac{F_1\ (I_0{}^T(r=r_{n_r},\theta)).F_1^*(I_1{}^T(r=r_{n_r},\theta))}{\left|F_1\ (I_0{}^T(r=r_{n_r},\theta))\right|.\left|F_1^*(I_1{}^T(r=r_{n_r},\theta))\right|} \right)\right)$$

The above formula is derived in Appendix C. The $I_0{}^T$ represents the polar transformed of image I_0. The subscript of F_1 is used to symbolize the 1D Fourier operations. We can find the rotation value in an iteration of the algorithm, mentioned before, by using above formula.



(a) (b)

(c) (d)

Figure 5.5 : (a) shows an intermediate position of the first piece from the iteration. (b) represents its image in polar coordinates, (c) the final transformation of the first piece and (d) its image in polar coordinates

Even in the Polar Coordinates, the expanded regions of the second piece shown in the right image of the Figure 5.3 are correlated with the original regions of the first piece shown in the Figure 5.5-d.

Since the finding the rotation and translation is an iterative operation, the complexity depends on the number of the iterations, n_{iter}. In each iteration, we use two registrations (one in Cartesian, one in Polar coordinates), one transformation from Cartesian to Polar coordinates and one rotation operation. The complexity of the Cartesian to Polar transformation and rotation are $O(C_1N^2)$. In the Polar coordinates, the complexity of the expression (5.21) is $O(10C_1N^2\log(N)+5C_1N\log(N)+35C_1N^2)$ because the complexity of the 1D FFT's is $O(N\log(N))$. By adding the complexity of finding the

translation, the total complexity for each iteration is $O(40C_1N^2\log(N)+77C_1N^2+5C_1N\log(N))$. According to dominant terms and the number of iterations, it is $O(40C_1n_{iter}N^2\log(N))$. The complexity is proportional to $n_{iter}N^2$.

Case 3 : Removing the "no texture" assumption: The new assumptions are:

- There are only two pieces in the puzzle
- The puzzle pieces are two-dimensional

Definition of the problem: Let us first consider the solution to a two-piece puzzle. The solution set consists of the piece I_0 and the transformed version of the piece I_1. These images are textural images.

Solution: We used the morphing operations to expand the images in two previous problems, because the images are defined as black and white. Here, our pieces have pictorial images. These images may contain only drawing or lines as well as complex textural structures. The expanding operation for the pictorial pieces should be more sophisticated that the expanded region of a piece is seriously similar to original region of the right neighbor pieces. This expanding operation is defined in the Chapter 3.

The first step is to expand the images by using inpainting algorithms. This operation gives the expanded images, I_0^+ and I_1^+.

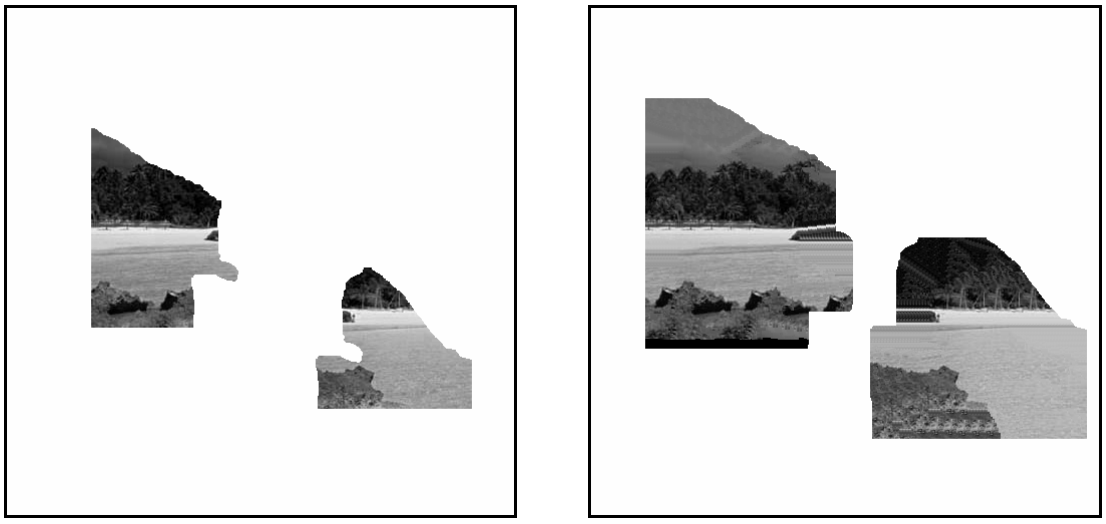


Figure 5.6 : (left) the two pieces with the texture (right) the expanded images.

The transformation consists of translation (T=(Δx,Δy,$\Delta\theta$)). Here, we have to define one more step. As we mention before in the Chapter 4, we can not directly use the pixel color values. We have to use the feature values of the textural structures.

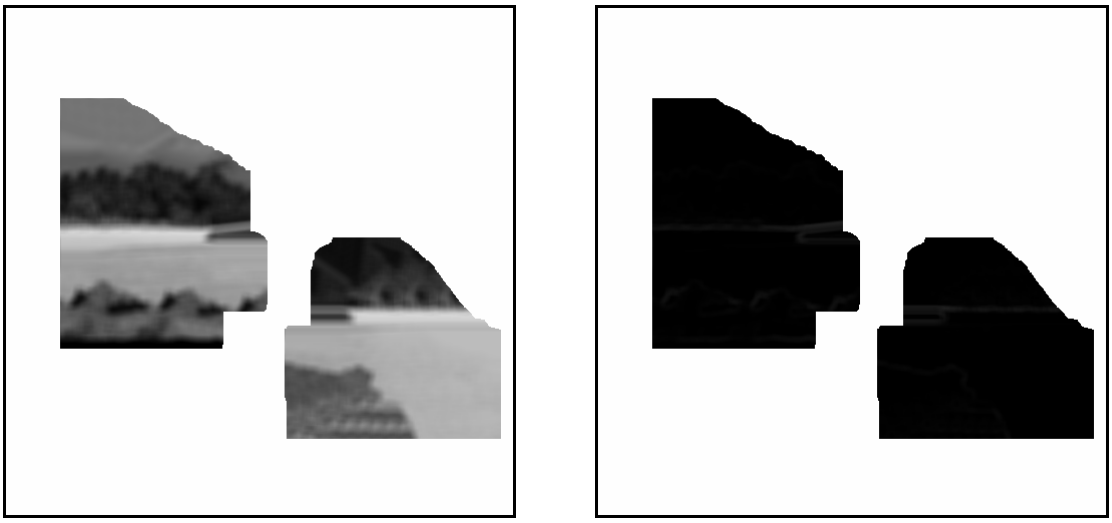


Figure 5.7 : Feature images of the original and expanded band. Mean (left) and variance (right) features.

When we define n_k (number of features) features, the new form of the formula (5.22), which represents the correlation between original and expanded regions, is;

$$S^2_{\text{general}} = \underset{T}{\text{argmax}} \sum_{\text{k}}^{\text{n}_\text{k}} w_k \left(C(f_{k0}, T(f^+_{k1})) + C(f^+_{k0}, T(f_{k1})) \right) \tag{5.22}$$

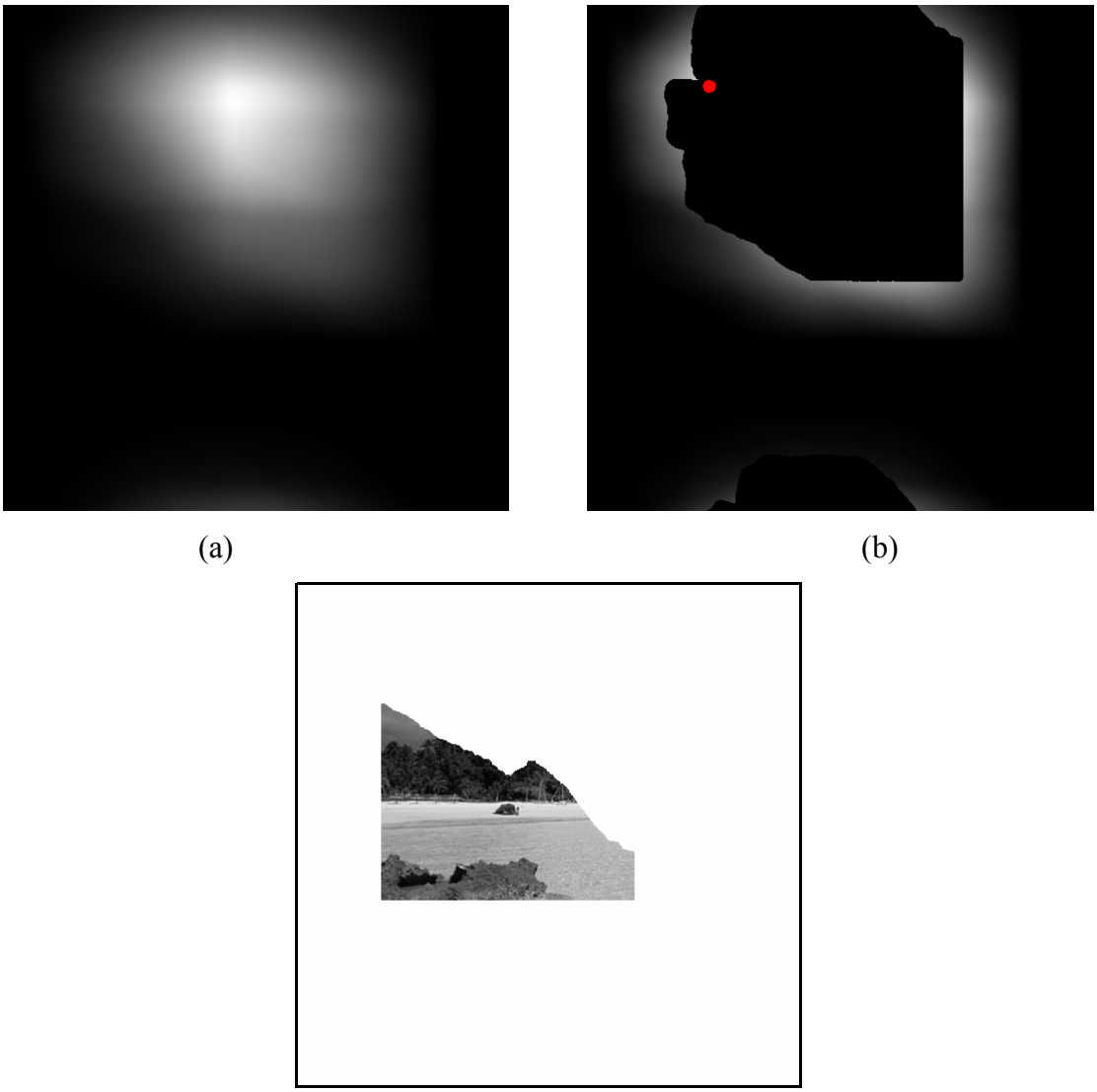
The constraint formulas (5.14)(5.17) do not differ in this solution. But the formula (5.15) defining the real solution of the two-piece puzzle differs as:

$$S^2_{real} = \left\{ \underset{T}{\text{argmax}} \sum_{\text{k}}^{\text{n}_\text{k}} w_k \left(C(f_{k0}, T(f^+_{k1})) + C(f^+_{k0}, T(f_{k1})) \right) \mid C(I_0, T(I_1)) = 0 \right\} \tag{5.23}$$

At the same time, the Fourier version of the above formula is changed as:

$$S^2_{\text{real}} \equiv \text{imax} \left(\overline{F} \left(\sum_{k}^{n_k} w_k \left(\frac{F\ (f_{k0}) . F^*(f^+_{k1})}{\left|F\ (f_{k0})\right| . \left|F^*(f^+_{k1})\right|} + \frac{F\ (f^+_{k0}) . F^*(f_{k1})}{\left|F\ (f^+_{k0})\right| . \left|F^*(f_{k1})\right|} \right) \right) . L\left(\overline{F}(F(I_0) . F^*(I_1)) \right) \right) \tag{5.24}$$

The formula used in polar coordinates is affected same as S^2_{real} formula. The images, $I_0{}^T$, terms will be feature, $f_{k0}{}^T$, terms and the weighted summation of features will also be added to expression. By using of mentioned iterative solution, the two-piece puzzle is solved.



(a) (b)

(c)

Figure 5.8 : (a) The correlation matrix between the original and the expanded regions or the inner part of the $S_{general}$. (b) The correlation matrix of the possible transformations or the inner part of the S_{real}. (c) The final assembly.

The red point has a maximum correlation value. The Figure 5.8-c shows the final replacement of the second image according to the transformation values found by S_{real}.

In the case 3, the complexity also depends on the number of the features, n_k. The complexity is then derived from that of the case 2. The new complexity for the (5.24) becomes $O(40C_1 n_{iter} n_k N^2 \log(N))$.

Case 4 : Removing the "two-piece puzzle" assumption:

The remaining last assumption is that the puzzle pieces are two-dimensional.

Definition of the problem: We have a puzzle including n_p pieces. The images of these pieces are textural. We have to find the right transformations of all pieces that generate the best assembly.

Solution: The solution outlined above can be generalized to the solution of more realistic puzzles with larger number of pieces. Let n_p be the number of pieces involved. The solution is the set of appropriate transformations of each piece:

$$S^{n_p} = \left\{ T_0, T_1, T_2, T_3, ..., T_{n_p} \right\} \tag{5.25}$$

We develop two different algorithms to solve the general problem. The first one is semi-automated algorithm. This algorithm is used to solve the puzzles containing large number of pieces. This algorithm generates faster solutions in each step, but they are generally partial solutions. These partial solutions are shown to user. The user indicates the pieces which may be a group. Then the same algorithm generates new possible solution. These interactive operations continue until the user think that the final shape is generated.

The second one is a full-automated algorithm. This algorithm is using the search tree methods. This method takes more time than the first one and the number of puzzle pieces that can manage to solve is less than the previous algorithm.

5.3 A Semi-Automated Algorithm

Assume that all the pieces are randomly dispensed on a big enough board (B). We randomly select a piece (I_t). For this piece, the transformation giving maximum correlation is obtained using the above technique.

$$T_t^{'} = \left\{ \underset{T_t}{\text{argmax}} \sum_{\text{k}}^{\text{n}_\text{k}} \left(C(f_{kB} - f_{kt}, T_t(f_{kt}^{+})) + C(f_{kB}^{+} - f_{kt}^{+}, T_t(f_{kt})) \right) \;\middle|\; C(B - I_t^0, T_t(I_t^0)) = 0 \right\} \tag{5.26}$$

$$\text{where} \quad B = \left\{ I_0, I_1, I_2, ..., I_{n_p} \right\} \tag{5.27}$$

Here $T_t^{'}$ symbolizes the best transformation when the other pieces in the board are fixed. The subtraction operation in the expression (like f_{kB}-f_{kt}) is used to represent the board without I_t image.

The algorithm used in solving the puzzle is outlined below:

1. Place the pieces on a board (B). This board should be big enough that all pieces can freely be placed.
2. Randomly select a t piece.
3. Find the best transform, $T_t^{'}$, by using the expression (5.26).

4. Continue to go to step 2 for all possible piece. Jump to next step if all pieces want to stay its original place. In another and formal words, if the transformation values calculated in step 3 give zero translation and rotation for all pieces, continue to step 5. ($T_t^{'} = (0,0,0) \quad \forall t$)
5. Select one or more pieces randomly and transform them any free place in the board. This step is likened to adding noise operation in a search algorithm. This step partially saves the search from being stuck in a local solution.
6. Go to step 2 until the step 4 generates same solutions (uniquely assembled) even if all pieces are tried in step 4.

The main drawback of this operation is that there may be multiple solutions to the problem. These multiple solutions depend on the initial placement of pieces on B and the random selections of the t^{th} piece. In these situations, the affinity measure developed in the previous chapter is used (Equation (4.6)). For a possible placement given by the proposed technique, the F_{cost} is calculated. If the algorithm reaches to a new solution that has a lower cost value than before, the new one becomes the best placement. These iterations continue until the last N possible placement cannot offer a better cost. This N value directly depends on the complexity of the puzzle. The main argument of the complexity for a puzzle is the number of the pieces. So the number N mainly depends on the n_p value. We assume $N \approx n_p^2$ in our experiments.

Hence the final version of the algorithm has the following two additional steps:

7. Find F_{cost} and if the new one is better then before, save the transformation as the best solution and clear the counter N.
8. Go to step 1, until N reaches to n_p^2.

5.3.1 Implementation of the Semi-Automated Algorithm

We implement the semi-automated algorithm into the same user interface. A hot key starts the algorithm to find a candidate placement. Then user decides to group the pieces to continue to assembly process. The grouped pieces combined together by using the expressions (4.10 and 4.11) mentioned in previous chapter. The algorithm may be triggered one more times to find a new solution. Figure 5.9 shows a scene from the program interface. It is captured when an algorithm just finished the search and offered a possible assembly of puzzle pieces.

The time for the semi-automated algorithm depends on the initial conditions. However, we may define the complexity by the use of coarse assumptions. In the first two steps, we can find the solution with a time linearly proportional to number of pieces, n_p. In the third step, we continue to do the same operations approximately n_p times. We can define a rough complexity for this algorithm as $O(40C_1 n_{iter} n_k n_p{}^2 N^2 \log(N))$. The time for the solution is proportional to the $n_k n_p{}^2 N^2 \log(N)$.

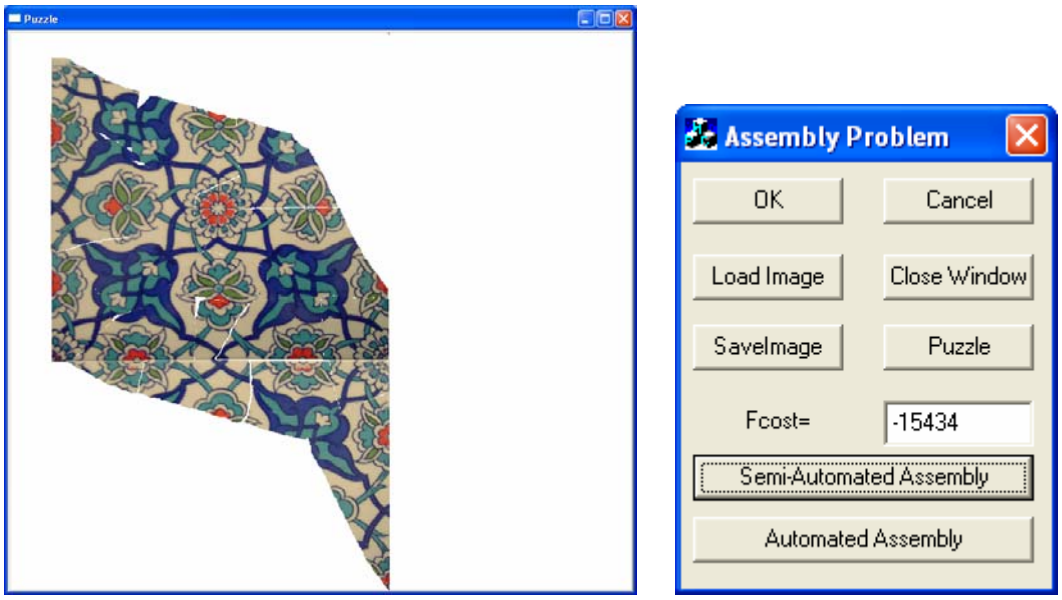
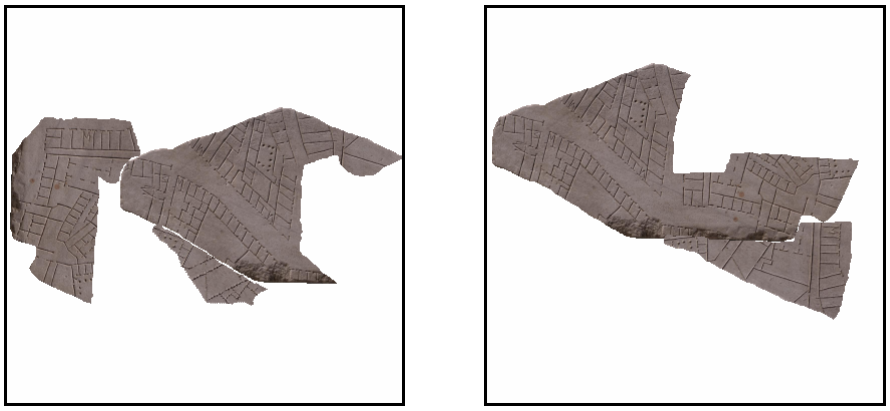


Figure 5.9 : A scene from the developed program.

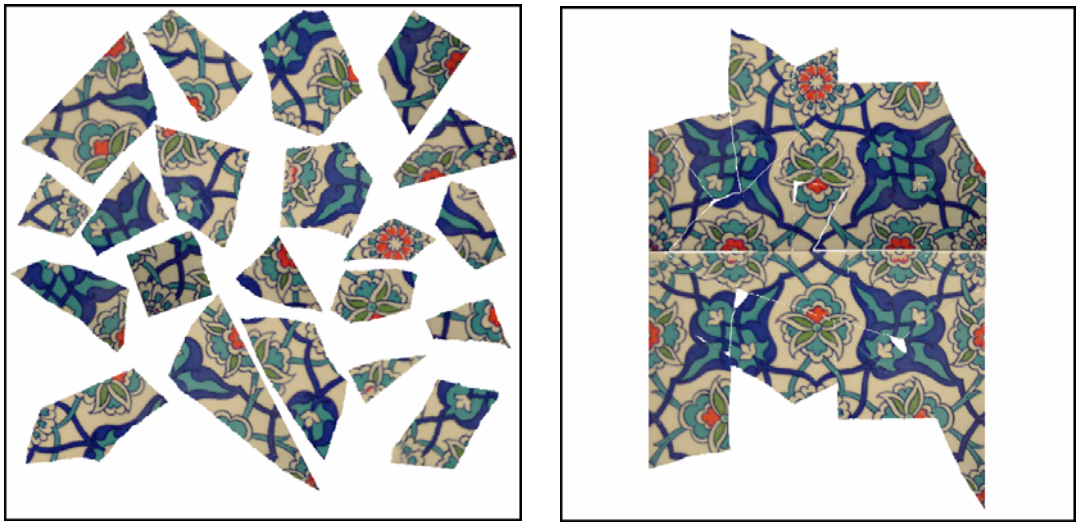
5.3.2 Results of the Semi-Automated Algorithm

We will demonstrate the results of the proposed algorithm on three different datasets. The first dataset (13 pieces) from Stanford University website is part of the Forma Urbis Romae dataset [47] which is a marble map of ancient Rome that has more than a thousand fragments. For this experiment, the image of a fragment from this dataset is broken artificially. Figure 4.7-a shows the pieces in the dataset and Figure 5.10-a,b show the different assembly obtained. The final assembly has shown in Figure 4.7-d

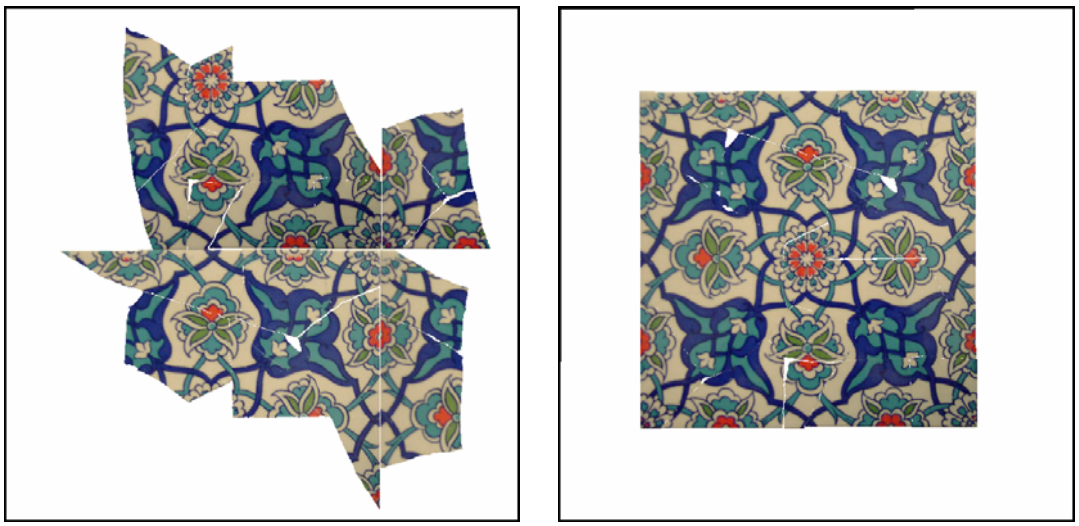


(a) Fcost = -18215 (b) Fcost = -18113

Figure 5.10 . Various intermediate solutions of the semi-automated algorithm.



(a) Fcost = 0 (b) Fcost = -17841



(c) Fcost = -18577 (d) Fcost = -20250

Figure 5.11 : Various solutions of the semi-automated algorithm for a ceramic tile with 21 pieces.

The second dataset consists of 21 pieces of an original ceramic tile. Figure 5.11-a shows the pieces to be assembled. Figures 5.11-b,c,d give three possible solutions. The corresponding cost functions are also given. It is noted that all the solutions are visually feasible solutions in terms of the texture and geometry information and the correct solution has the minimum cost function.

The last experiment has pieces from two different ceramic tiles. This experiment is important since in a real archaeological set-up, pieces may come from two or more objects. 10 pieces of the tile used in Experiment 2 are mixed with 9 pieces from another tile. Resulting assembly is given in Figure 5.12.

Figure 5.12 : Pieces from two different ceramic tiles.

5.4 An Automated Algorithm

The automated algorithm purposes to find a final assembly of the pieces. The developed method uses the arguments of the best first search algorithms.

First, we set a search buffer that will contain the best n_b assembly of the pieces in an intermediate stage of the algorithm. The size of the buffer depends on the complexity of the search space. As we mentioned before, the complexity of the puzzles is a function of only the number of the puzzle pieces if we consider the all possible puzzle types. In the experiments, we set the search buffer size, n_b, as $n_b \approx n_p^2$.

The second step is to place an initial piece to the search buffer. Then, a search procedure tries all possible pieces that are not used before. This procedure is applied for all elements of the search buffer. After this operation, we have a set of new candidate assemblies.

The FFT based method developed and described in previous stages is also used in the search procedure to fill the candidate set. The expression (5.24) can solve the best transformation when we have two pieces and want to place one close by the other. Consider that an assembly in a search buffer contains 3 pieces. These pieces can be merged together (by using the expressions (4.10 and 4.11)). After this operation, the problem becomes the similar with two pieces puzzle problem.

$$T_t^{'} = \left\{ \underset{T_t}{\text{argmax}} \sum_{\text{k}}^{\text{n}_\text{k}} \left(C(f_{kG}, T_t(f_{kt}^{+})) + C(f_{kG}^{+}, T_t(f_{kt})) \right) \;\middle|\; C(G, T_t(I_t^0)) = 0 \right\} \tag{5.28}$$

$$\text{where } G = \left\{ I \;\middle|\; (I \in Buffer_i) \right\} \tag{5.29}$$

Here $T_t^{'}$ symbolizes the best transformation between a group and t^{th} piece. This group consists of the pieces in an element of a search buffer.

Because the candidate set may contain the assemblies that have equal transformations, an extra procedure is called to clean such ambiguities. This procedure search for same solutions in the candidate set and cancels one of these solutions. Then, the affinity measures developed in the Chapter 4 is calculated for the remaining set consisting of unique solutions. The elements of the set are sorted according to the affinity measures. The assemblies that have the best n_b measures are stored for the next iteration. The iterations continue until all pieces are assembled in all n_b element of the search buffer.

After iterations end, the best affinity measure is selected as the solution of the algorithm. There is a fragility of this procedure. If adding one new piece does not improve the affinity measure for all possible trails, the search buffer remains same with previous one. In this situation, the algorithm stuck on fixed state. To overcome this problem, the algorithm ends when the current state and previous state are equal. The pseudo-code of the above algorithm is shown in Table (5.1).

1. Set the size of the search buffer.
2. Place the pieces into the search buffer one by one and sequentially.
3. For all search buffer elements, *i*;
    a. For all pieces that does not exist in the group of the i^{th} element;
        i. Find the best transformation by using the expression (5.28).
        ii. Calculate the affinity measure value.
        iii. Store transformations and affinity measure into the candidate buffer.
4. Test all candidate assemblies if two more candidates offer the same transformation. If they are, stay only one by removing the others.
5. Select the best n_b assemblies.
6. If the best assemblies are equal to previous search buffer, go to step 8.
7. Replace the previous buffer with new assemblies and continue to go to step 3.
8. Transform the pieces according to the best affinity measure in the search buffer and show it to the user.

Table 5.1 : The pseudo code of the fully-automated algorithm.

5.4.1 Implementation of the Automated Algorithm

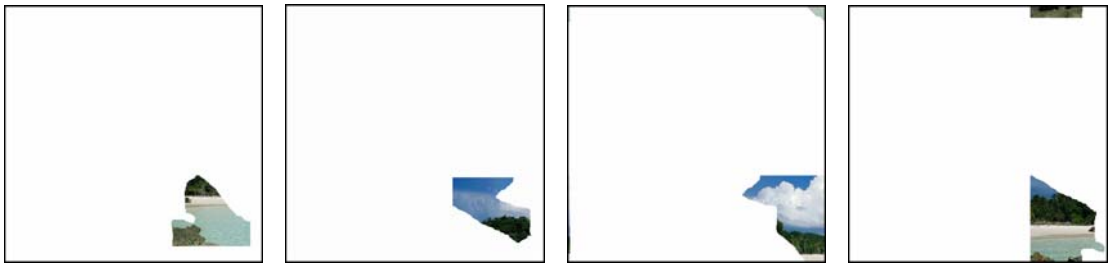
We implement the automated algorithm into the same user interface. A hot key starts the algorithm to find a possible assembly of the puzzle pieces. This algorithm is also used like the semi-automated algorithm. The user may also decide to group the pieces to continue to assembly process. The algorithm may be triggered one more time to find a new solution. But the execution time of this algorithm is considerably higher than the semi-automated algorithm. So it is not meaningful to use this method as part of an interactive assembly operation. Figure (5.9) shows a scene from the program interface.

Here, the succession of the algorithm is proportional to the size of the search buffer, n_{buf}. The time for the solution increases as the size of the buffer increases. In the first iteration (for the step 3), we compute the expression (5.28) (n_p-1) times for each n_{buf}. Then, (as a coarse assumption) we compute (n_p-2) times for each buffer element. Thus, we use this expression $n_{buf}.n_p.(n_p\text{-}1)/2 \approx n_{buf}.n_p^2/2$ times. As we define the complexity of the mentioned expression, total complexity becomes

$O(20C_1 n_{iter} n_k n_{buf} n_p^2 N^2 \log(N))$ for this algorithm. Also, suppose that the size of the buffer should, at least, be linearly proportional to the number of pieces. Thus, the complexity may be $O(20C_1 n_{iter} n_k n_p^3 N^2 \log(N))$. The complexity of the fully-automated algorithm is more dependent on the number of the pieces than the semi-automated algorithm.

5.4.2 Results of the Automated Algorithm

The example for automated algorithm contains four puzzle pieces. We choose the minimum number of pieces to be able to demonstrate the intermediate states of the algorithm. The search buffer is selected as four. Initially, the search buffer is stored with the pieces. The first piece is placed into the first element of the search buffer, and second goes second element, and so on.



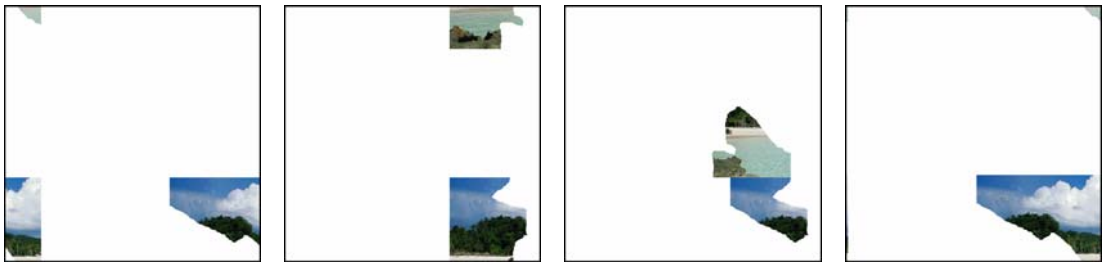
(a) $F_{cost} = 0$ (b) $F_{cost} = 0$ (c) $F_{cost} = 0$ (d) $F_{cost} = 0$

Figure 5.13 : Puzzle pieces

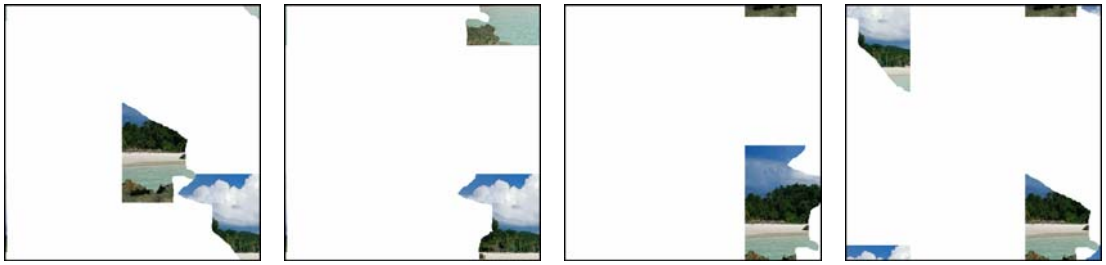
After first iteration, 12 new candidates is generated, and total candidates with the previous search buffer contains 16 possible assembly. All new candidates are shown in the Figure 5.14. The F_{cost} values are also attached to the images of the candidates.

We choose the best fourth assembly to continue the second iterations. The selected assemblies are illustrated in Figure 5.14. The second iteration generates new candidates and search buffer is updated until the all search buffer elements use all pieces. The all intermediate iterations and the final assemblies are presented in Figure 15.

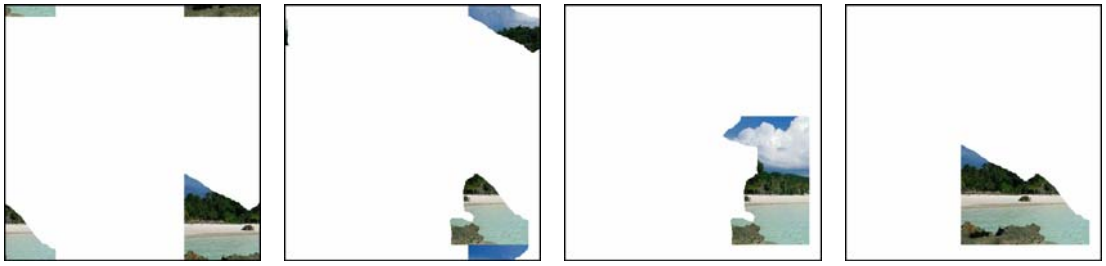
The second example belongs to same picture as the first example. The number of pieces is 16. The example is artificially prepared like first one. The pieces are shown in Figure 5.16.



(a) F_{cost} = -988 (b) F_{cost} = -3200 (c) F_{cost} = 289 (d) F_{cost} = -988



(e) F_{cost} = 259 (f) F_{cost} = -359 (g) F_{cost} = -3200 (h) F_{cost} = 259



(e) F_{cost} = -2564 (f) F_{cost} = 289 (g) F_{cost} = -359 (h) F_{cost} =--2564

Figure 5.14 : All new candidates in the second step of the automated algorithm. The candidates selected in first step are (b),(g),(f), and (h).

(a) F_{cost} = -3982 (b) F_{cost} = -5998 (c) F_{cost} = -3982 (d) F_{cost} = -5998

(e) F_{cost} = -8293 (f) F_{cost} = -8293 (g) F_{cost} = -8293 (h) F_{cost} = -5998

Figure 5.15 : (a),(b),(c), and (d) are the new elements of the search buffer in the second step. (e),(f),(g), and (h) are from third step.

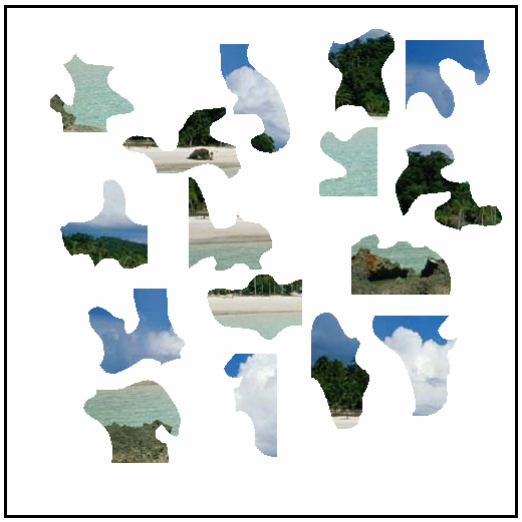
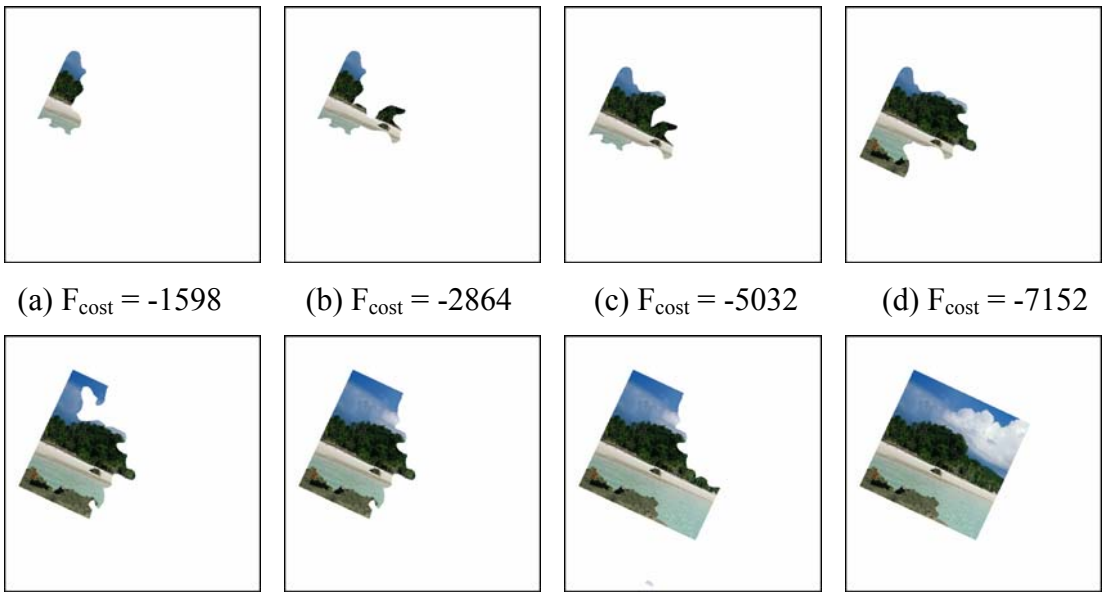


Figure 5.16 : An artificial puzzle with 16 pieces.

In this example, we present the assemblies that have the best affinity measure. Each assembly is taken from different iterations. These assemblies are shown in Figure 5.17.

The third example contains 21 pieces of real ceramic tiles. The erosion and the occlusion are tested. The intermediate assemblies and final result are demonstrated in Figure 5.18 and 5.11-d, respectively.



(a) F_{cost} = -1598 (b) F_{cost} = -2864 (c) F_{cost} = -5032 (d) F_{cost} = -7152

(e) F_{cost} = -10560 (f) F_{cost} = -12384 (g) F_{cost} = -17118 (h) F_{cost} = -22536

Figure 5.17 : The images represent the assemblies with best cost value from the various iterations

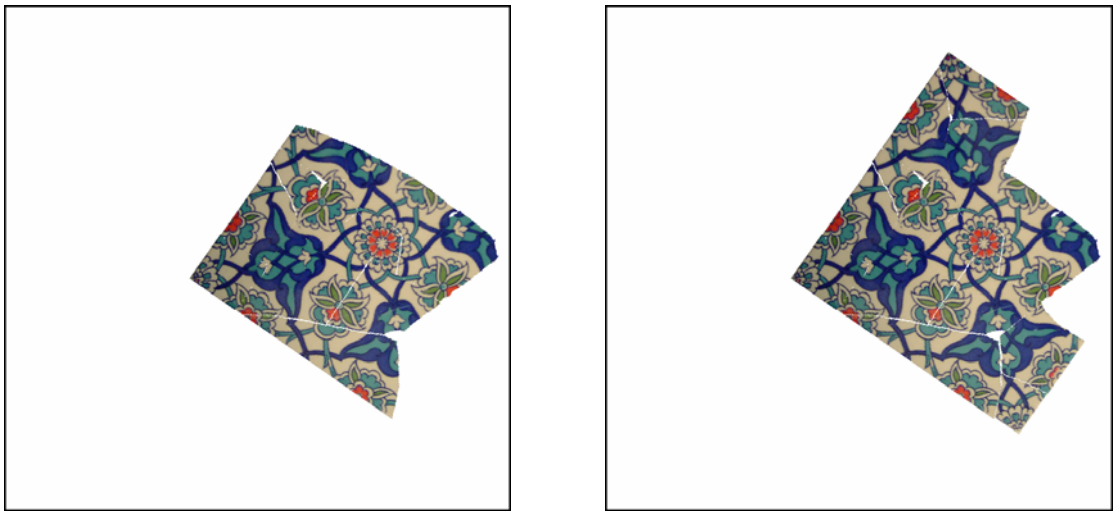


Figure 5.18 : Two different assemblies from the 12th and 15th iterations.

The last example contains more pieces. The solution can not generate final assembly. In this example, the pieces are grouped after the algorithm generates the intermediate solution. Then, the algorithm is triggered one more time. The next solution becomes the correct assembly. If it is observed that the F_{cost} belonging to final assembly is smaller than the first output of the algorithm, we can consider that the cost function responses to different placements accurately, but the optimization may fail as the number of the pieces increases [89][90][91].

We presented a method for the automated puzzle assembly problem using surface texture and picture. The approach is based on expanding the boundary of each piece using inpainting and texture synthesis methods and maximizing the correlation based on matching feature values obtained from these predicted regions. Initial experiments show that this approach is very promising for the automated puzzle assembly problem. The next chapter will concentrate on generalizing the presented algorithm to solve the matching problem for the 3D puzzle pieces.

Chapter 6

3D EXTENSION OF THE PURPOSED APPROACH

6.1 Introduction

We use the FFT based methods to solve the partial matching problem in previous chapter. The partial matching of the pieces is the main operation in a reconstruction or assembly problem. Because the puzzles used in preceding chapters consist of two-dimensional pieces, we have described and developed the 2D partial matching operations. In this chapter, we propose to apply the FFT based methods to the 3D partial curve and surface matching problems.

The main idea is similar to the previous one. First, the pieces are expanded and are attempt to find the best transformation that maximizes the correlation between the expanded region or surface of one piece and the other. To find the maximum correlation, the FFT shift theory is used. For the FFT operation, the only difference between 2D and 3D is the implementation. The 3D FFT calculation may be a memory critical operation although the 2D FFT operations are not. To overcome this problem, we rescale pieces so that all pieces can fit into the space with a size of 128x128x128. But this extra arrangement causes that the relatively small pieces can not be used in the solution.

The other difference between the assembly of the 2D puzzles and the methods that will be described in the current chapter is to propose to solve only the best matching of two pieces and give a matching score for this operation. The matching of the expanded region with original pieces meets with more or less same difficulty with the operations in 2D. But the embedding of the constraints, which are preventing the pieces to overlap, is much more difficult than it is used in 2D images. The main reason of this situation is the freedom of the pieces in 3D space. For example, a shell can easily come closer to

another shell without touching each other. Figure 6.1 shows such a situation. An additional procedure is described in the later sections to overcome this problem.

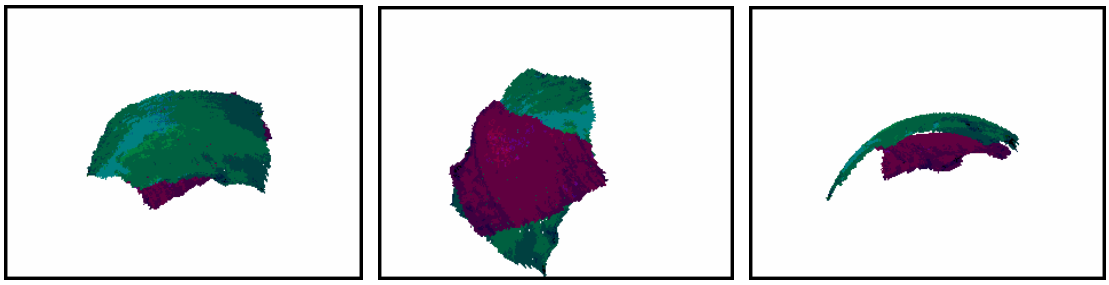


Figure 6.1 : The second piece comes to closer to the first piece without them touching one another.

In 3D, a piece has a freedom that can be expressed with six variables.

$$T = (\Delta x, \Delta y, \Delta z, \Delta \psi, \Delta \theta, \Delta \phi) \tag{6.1}$$

and the transformation operation in 3D is;

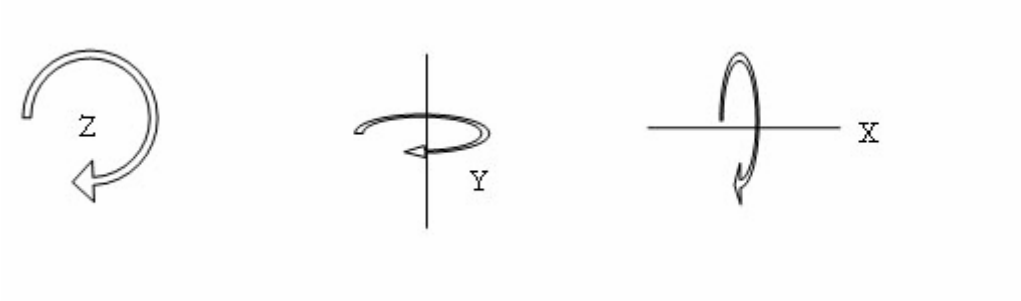


Figure 6.2 : Rotation in 3D

$$Rot = \begin{Bmatrix} \cos\psi\cos\phi - \cos\theta\sin\phi\sin\psi & \cos\psi\sin\phi + \cos\theta\cos\phi\sin\psi & \sin\psi\sin\theta \\ -\sin\psi\cos\phi - \cos\theta\sin\phi\cos\psi & -\sin\psi\sin\phi + \cos\theta\cos\phi\cos\psi & \cos\psi\sin\theta \\ \sin\theta\sin\phi & -\sin\theta\cos\phi & \cos\theta \end{Bmatrix} \tag{6.2}$$

$$T^3(I) = Rot.I + \begin{Bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{Bmatrix} \tag{6.3}$$

The FFT shift theory is applied translation and rotation iteratively as we implemented in 2D. In the iterations of the algorithm, we use the polar transformation methods to find the rotation. The Fourier calculations are relatively fast to find the transformation even in 3D, if the memory problem is overcame.

Another aspect in 3D puzzle assembly is the texture implementation. We use the finger holes in archeological fragments as a textural structure. Although we test some track on the fragment shell in experiments, the matching of the fully textural pieces in 3D is not examined in this study. The first open problem is to expand the textures. Some

results about 3D inpainting are reported in the work [63]. Another problem in the implementation is how we state the color values in a 3D volumetric space. This problem will be mentioned in other sections of this chapter.

The rest of the chapter is organized as follows: the next section describes the data acquisition and preprocessing of real 3D archeological data. The third section outlines the expanding the 3D surfaces outwards. The later sections present the study of FFT based partial matching in 3D, the implementation and the results of defined algorithms, respectively.

6.2 Data Acquisition

To test the experiments, we collect the fragments of a 3D pot. It consists of 19 pieces. The sizes of the pieces are various. We acquire the images of the pieces by using a camera system called ShapeSnatcher. The picture of the hard part of the system is shown in Figure 6.3. The System is a novel system to produce 3D-models of real objects. It consists of a grid projecting device, a camera, a computer and the ShapeSnatcher Slide that contains a fine grain pattern that is to be projected onto the object. All these components are assembled onto Eyetronics' ShapeCam, such that the whole setup is portable. The 3D-shape acquisition is fast and easy. The camera takes only one image and the deformation of the pattern yields the 3D-data. The ShapeCam takes two images, one with the grid and one as a texture, such that the final result is even mush better, since the texture is not extracted from the grid-image anymore.

Figure 6.3 : ShapeSnatcher scanner system

The advantages of this system may be summarized as: Complex hardware is replaced by standard hardware and sophisticated software. The resulting model is

delivered as a surface mesh in formats compatible with most of standard modeling software. Technology consists of portable components, so the system can be easily brought to the objects. Misalignments between shape and texture are not possible. Both are extracted from one and the same image, so they match perfectly onto each other. The objects captured with the ShapeSnatcher can be of any size, ranging from small toys to full bodies and large statues. And also, it has a lower cost than the other professional 3D acquisition systems.

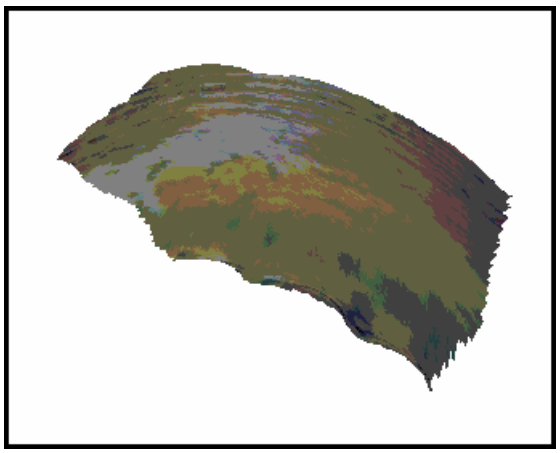
The Shapenatcher slide contains a very fine grid pattern that is produced by special lithographic techniques. Projecting the grid onto the object is easy to comprehend. The ShapeSnatcher Software will measure the deformations of the grid lines, and calculate the 3D structure of the object. This means the objects can be modeled in 3D from only one image. From that same image, the ShapeSnatcher Software will also extract the texture of the object, by artificially removing the grid lines.

In order to be able to use the ShapeSnatcher System, it has to be calibrated. Calibration is very crucial for all 3D acquisition systems. The calibration procedure is required to calculate the distances and angles between camera, projector and object. For most systems, the calibration procedure is very difficult, complex and time consuming. The ShapeSnatcher needs two kinds of information to calculate a 3D structure from the image. The first one is the relative position of the camera, slide projector and object. This information is obtained from the calibration file. The second is a picture of the object that will be modeled. This picture needs to be taken while the grid is projected, and has to same camera-projector setup as the calibration image. After these setups, the ShapeSnatcher software is ready to calculate the 3D structure of the model. The ShapeSnatcher Software also supports the extraction of the texture from the same image as used for the extraction of the 3D structure. By determining the 3D structure, the ShapeSnatcher knows how the grid looks like on the image. Then the program merely looks at the color intensities just beside the lines, and generates a new texture. The 3D structure and the texture match perfectly because all the information is rendered from one and the same image.

After we capture the 3D data, an additional process is applied. The mentioned system may produce undesired outputs especially on the edges of the pieces. We clean this data points in order not to cause the following processes to fail. We assume that the occluded 3D points are considered as eroded borders of the fragments. This process is

made manually. Because hand made preprocessing is a time consuming operation, an additional utility has to be developed to overcome this problem if the techniques described in this chapter is decided to be implemented automatically.

(a) (b)

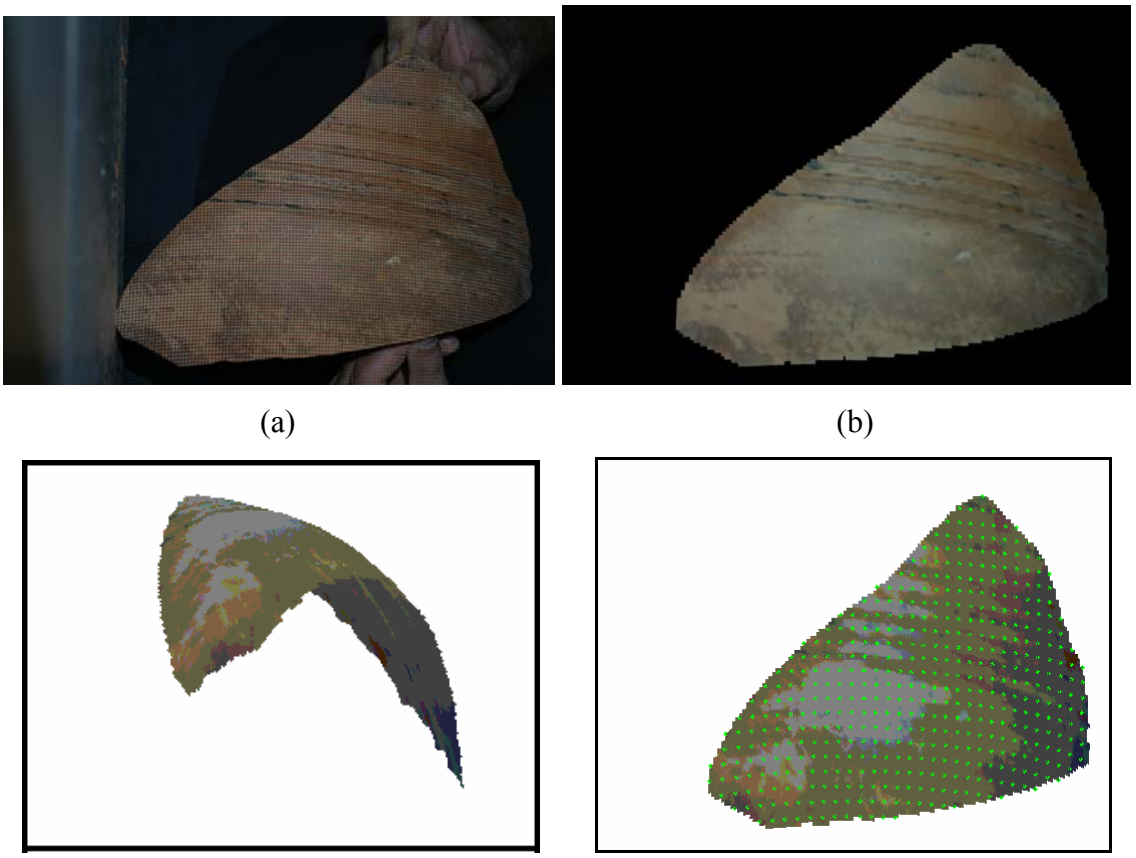


(c)

Figure 6.4 : (a) The captured 3D piece, (b) its 2D projection, (c) its 3D presentation.

6.3 Expanding 3D Pieces

The acquisition method mentioned above produces 3D coordinates of surface points and definitions of the surface patches, of which these points form. The points are scattered according to mesh grid pattern of acquisition method. Projected slide light structure produces the mesh grid pattern. A sample image and its mesh grid pattern are presented in Figure 6.5.



(a) (b)

(c) (d)

Figure 6.5 : (a) The captured image, (b) its textural view, (c) its 3D view, and (d) the grid structure of the piece.

To find the correlation of a 3D piece with a possible neighbor piece by using our technique, we have to expand the pieces outwards. In 3D, there are two aspects of expanding operation. The first is to expand the pieces geometrically, and the other is to fill this expanding surface with predicted texture. In this section, we will focus the geometrical expanding.

Currently, the expanding geometrically is assumed to find new mesh grid points and new surface patches, so that the surface formed by the new patches is reasonable continuation of original surface. In order to realize this assumption, we develop an algorithm to expand the pieces outwards. A pseudo code of this algorithm is shown in Table 6.1.

1. Load the image data.
2. Find the boundary grid points.
3. Calculate the neighborhood grid points for all boundary *i* points as:
    a. Find the all points that enter the sphere centered in i^{th} point.
    b. Find the coefficients of a 2^{nd} degree explicit function by fitting to point coordinates found in previous process.
    c. Find the coordinates of new points inside the defined expanding band.
4. Solve the ambiguity about multiple offering for one new point.
5. Find the surface patches formed by new points.

Table 6.1 : The pseudo code of the expansion algorithm.

First operation after loading image data is to detect the boundary points of the pieces. The points, which generate a patch from only one side, are accepted as boundary points. A sample piece and its boundary are presented in Figure 6.6.

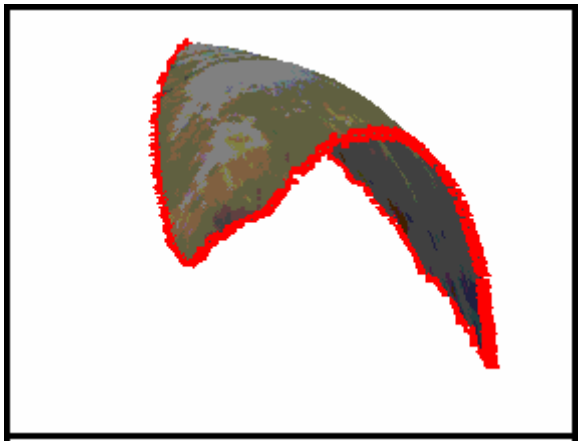


Figure 6.6 : The boundary grid points are shown with red points.

After determining the boundary points, they are processed in order to generate new points forming the expanding band. This process consists of three main steps. The first step is to find the closer points to current boundary point. These closer points belong to the surface of the piece. A predefined distance is used to determine the closers. In other words, the points entering a sphere centered in the same coordinate with the current point are determined for the next step.

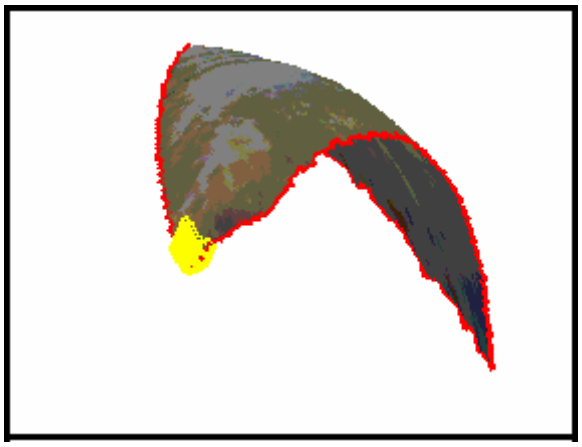


Figure 6.7 : The yellow points represent the closer points to the first boundary point.

The second step in the boundary point's loop is to fit an explicit function to the coordinates of the points determined previous step. The degree of the explicit function is set to 2. In experiments, it is seen that the higher degrees do not generates more accurate results, although they sometimes generates abnormal solutions because of their unstable structures. The explicit functions are expressed as:

$$(x, y, z) = (f_x(t,k), f_y(t,k), f_z(t,k)) \tag{6.4}$$

where

$$f(t,k) = a_5 t^2 + a_4 tk + a_3 k^2 + a_2 t + a_1 k + a_0 \tag{6.5}$$

t and k symbolize the sequence of the points in a mesh. As an example, a mesh grid with 12 points is shown in Figure 6.7. The red point represents the current boundary point. The other yellow points are the closer points to a chosen red point from all surface points.

The third step is to calculate the coefficients of the explicit functions. We use least square error techniques to fit the functions. Let consider that we use n_g number of closer points. The best coefficients according to least square method are found to be:

$$A = \left(\begin{bmatrix} t_0^2 & t_0 k_0 & k_0^2 & t_0 & k_0 & 1 \\ t_1^2 & t_1 k_1 & k_1^2 & k_1 & k_1 & 1 \\ \vdots & & \ddots & & & \vdots \\ t_{n_g}^2 & t_{n_g} k_{n_g} & k_{n_g}^2 & t_{n_g} & k_{n_g} & 1 \end{bmatrix} \cdot \begin{bmatrix} a_5 \\ a_4 \\ a_3 \\ a_2 \\ a_1 \\ a_0 \end{bmatrix} - \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{n_g} \end{bmatrix} \right) \tag{6.6}$$

$$error = AA^T \tag{6.7}$$

$$(a_5, a_4, a_3, a_2, a_1, a_0) = \arg\min_a (error) \tag{6.8}$$

The solution is be obtained by:

$$\begin{bmatrix} a_5 \\ a_4 \\ a_3 \\ a_2 \\ a_1 \\ a_0 \end{bmatrix} = \begin{bmatrix} t_0^2 & t_0 k_0 & k_0^2 & t_0 & k_0 & 1 \\ t_1^2 & t_1 k_1 & k_1^2 & k_1 & k_1 & 1 \\ \vdots & & \ddots & & & \vdots \\ t_{n_g}^2 & t_{n_g} k_{n_g} & k_{n_g}^2 & t_{n_g} & k_{n_g} & 1 \end{bmatrix}^{-1} \cdot \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{n_g} \end{bmatrix} \tag{6.9}$$

The above technique is very general method in all data fitting processes. Because the first matrix may not be a square matrix, the pseudo inverse operation is used. The described solution is obtained for only one dimension, x. So, we apply same procedure for the other 2 dimensions.

The next step is to create the new points by using of the generated functions. The first process in this step finds the unused (t,k) points in the mesh grid. These mesh points are searched in the region, which is not more distant than width of expanding band. The found mesh numbers is entered to functions to find the coordinates, (x,y,z).

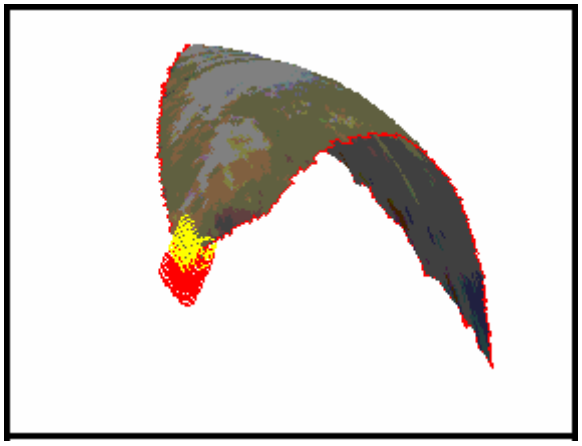


Figure 6.8 : The predicted points outside the surface are shown with red circles.

After applying steps described above for all boundary pieces, an extra problem is encountered. More than one boundary point may offer a coordinate value for a same mesh grid point. To overcome this ambiguity, we take the mean of all offers for same mesh grid point.

The new point cloud defines the expanding band, and it is sufficient to handle the matching problem. Here, we additionally generate the surface patches to examine the accuracy of the processes. Some samples are presented in Figure 6.10. The Figure contains the original images, expanded band and whole image of the samples. Also, the red line in the image, in which the expanded and original regions are combined, represents the boundary of original part of the pieces.

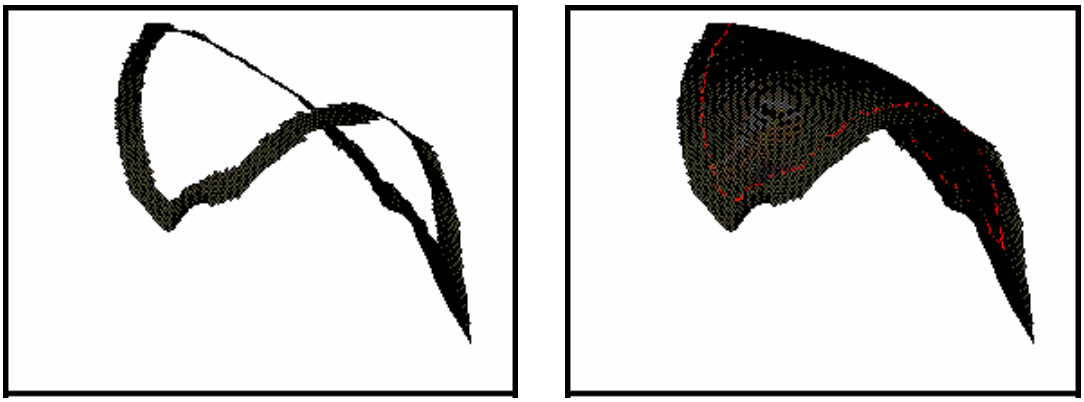


Figure 6.9 : (left) The expanded band and (right) the new expanded surface.

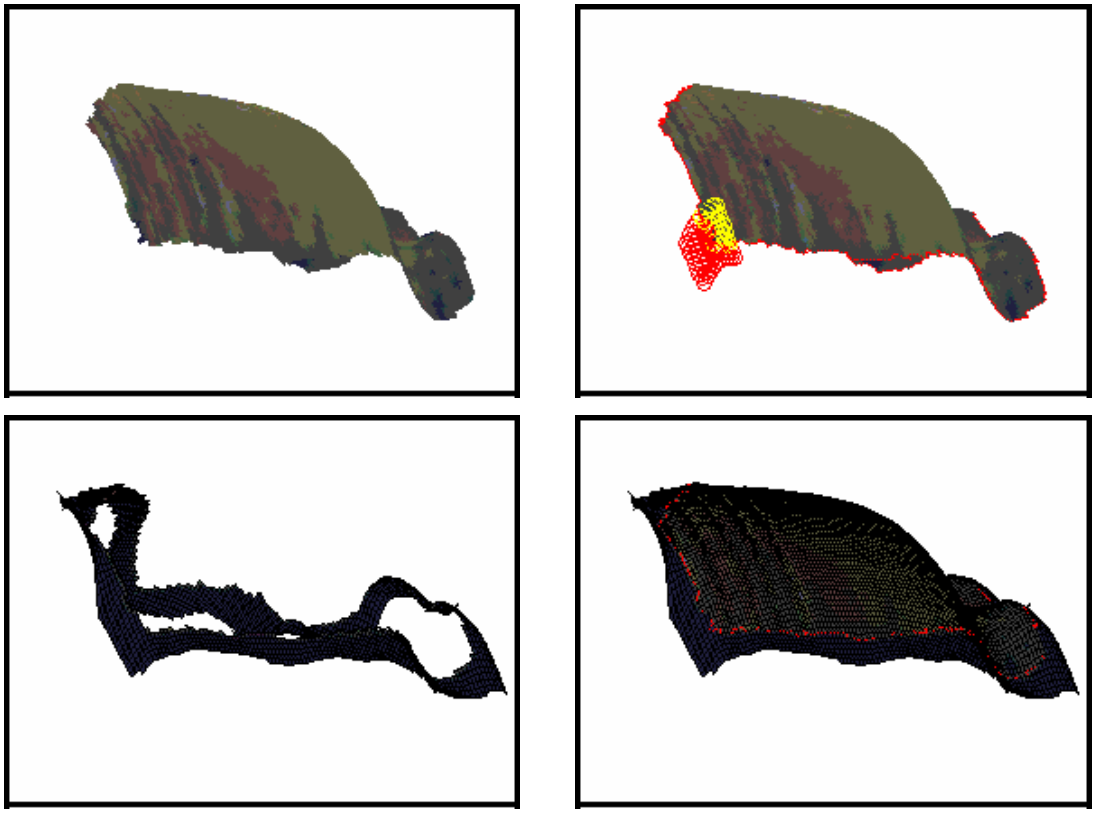


Figure 6.10 : The steps of the expansion operation for another piece.

6.4 Partial Matching Problem in 3D

In this section, we try to match two pieces. For the clarity of the explanations, we will firstly describe the matching problem based on the correlations of the surface. Then, we will define the problems that make the explained method unusable, and improve the method that can overcome the mentioned problems.

As we pointed before, 3D FFT computations are memory critical operations. We rescale the size of pieces to overcome this problem. In the experiments, the biggest piece is arranged that it is able to fit into a 64x64x64 discrete space. The dimension of the whole space is 128x128x128, so that two large pieces can be freely placed on it.

We have obtained the point clouds belonging to expanded regions in the previous section, and also we have the point clouds of the original regions. The next process is that these point clouds are placed on a 3D discrete space. When the coordinates of point cloud is rescaled to fit into finite and relatively small space, each point comes to a coordinate that is closed to a discrete point. Then, all discrete points which have a surface point around themselves are set to one.

By using above method, we generate two 3D scatter matrixes. One is for original part of first piece. The other is for expanded part of the second piece. Then, the maximum correlations are computed. The algorithm to find the correlations is similar to one used for 2D application. The formula defining the best correlation between the pieces can be written as;

$$S_{real} = \left\{ \underset{T}{\operatorname{argmax}} \left(C_3(I_0, T(I_1^+)) + C_3(I_0^+, T(I_1)) \right) \middle| \, C_3(I_0, T(I_1)) = 0 \right\} \tag{6.10}$$

and the FFT form of above equation is:

$$S_{real} \equiv \operatorname{imax} \left(\left(\overline{F_3} \left(\frac{F_3(I_0) . F_3^{\bullet}(I_1^+)}{\left| F_3(I_0) \right| . \left| F_3^{\bullet}(I_1^+) \right|} \right) + \overline{F} \left(\frac{F_3(I_0^+) . F_3^{\bullet}(I_1)}{\left| F_3(I_0^+) \right| . \left| F_3^{\bullet}(I_1) \right|} \right) \right) . L\left(\overline{F}_3 (F_3(I_0) . F_3^{\bullet}(I_1)) \right) \right) \tag{6.11}$$

$$\text{where } L[x] = \begin{cases} 0 & \text{if } x \neq 0 \\ 1 & \text{if } x = 0 \end{cases} \tag{6.12}$$

Subscript for Fourier operator, *F*, symbolizes the 3D FFT operations. Here, *I* is not an image or a surface. It is a matrix containing ones and zeros. It may be considered as a volumetric matrix that expresses the existence.

In 3D, another problem that does not exist in 2D is to calculate rotation. The rotation can be expressed with three variables in 3D. The polar transformation converts the Cartesian coordinates to Spherical coordinates. In Spherical coordinates, a point is expressed with two angles and a distance. However, two angles are not sufficient that each of three rotation angles is able to be used as a variable in a Spherical coordinates. Because of this situation, we can not use the iterative method using rotation and translation one by one. In order to overcome this problem, one of the rotation angles is extracted to be able to use the iterative algorithm. Then, this rotation angle is

exhaustively searched. To demonstrate the above processes, an artificial data is generated and shown in Figure 6.11.

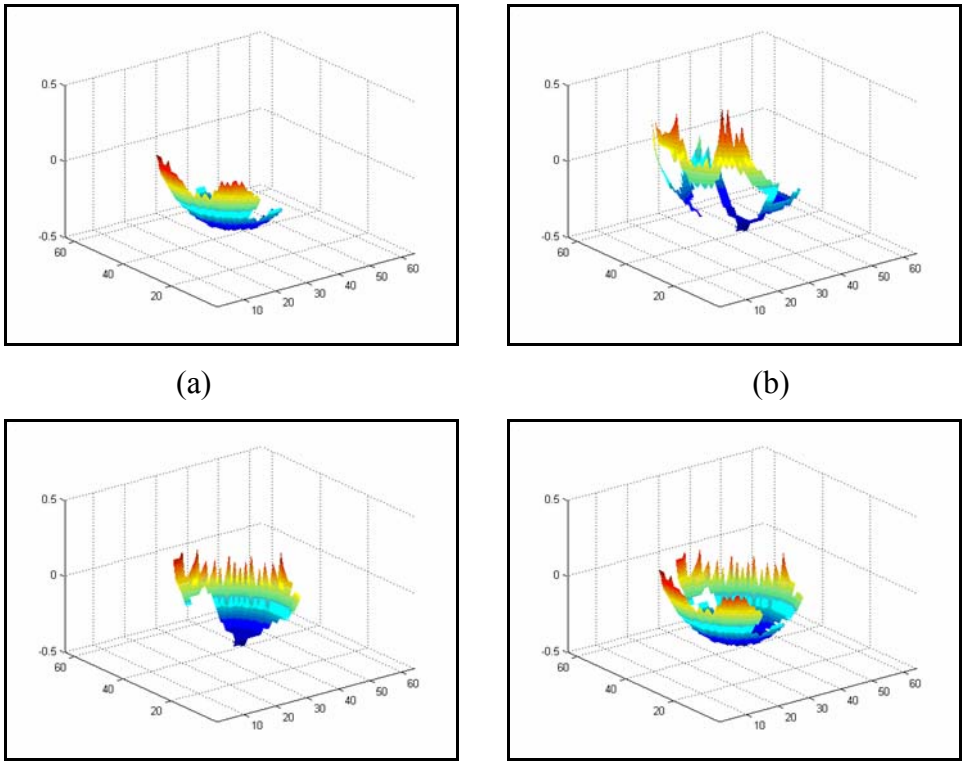


Figure 6.11 : (a) A piece created artificially, (b) The expanded band of the first image, (c) Another piece, and (d) the matching of two pieces.

The partial matching algorithm is described above. It is shown that the partial curve matching problem can be theoretically handled by using purposed method. Although the experiments give good results in artificial data, there are two open problems in order to apply for the real data. The first one is that the discrete points in expanded surface rarely intersect with the points of the original surface even if they are right positions. The second and more serious problem is in the constraint term. The constraint term is not capable to prevent the undesired matching of pieces. A surface can stealthily come closer to other ones without contacting. However, we want the pieces to come together head to head. Figure 6.12 is represents an undesired and desired samples.

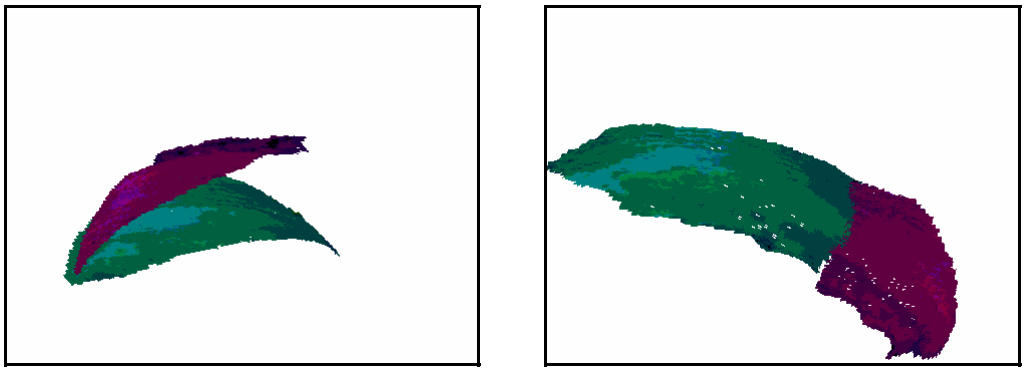


Figure 6.12 : (left) An undesired matching, and (right) a desired matching.

We will call the approaches described up to now as *primitive form of matching*. To improve the performance of this form of solution, we will add one more process for the final form of partial matching. The solution of these two problems is considered together. The surface is inflated towards its normal. In the primitive form of matching, the ones represent the existence of the surface and the zeros represent the empty space. We think that if we convert the surface to a volume, the undesired approaching of the pieces may be prevented. The matrix is filled according to the *probability of the existence*. The term '*probability of the existence*' means that a point of surface exists in a coordinate of the volume with a probability. This probability depends on the distance to the surface. The closer points to the surface have a greater probability than the farer ones. The mathematical relation with the distance and probability is calculated with Gaussian formulation.

$$P_{\text{existance}} = \frac{k}{\sigma} e^{-\frac{distance^2}{2\sigma^2}} \tag{6.13}$$

where σ is standard deviation. The value of the standard deviation set the spreading or inflation wideness of the surface. The distance is calculated by finding the perpendicular distance to the closer surface.

To test the effect of the texture information, we make an additional process in the matrix. The '*probability of the existence*' values are multiplied by minus in order to be able to define the differences between the texture segments. Although a texture approach using such a method is relatively poor, we consider that it may be used to test the usefulness of the texture information.

6.5 Implementation

MATLAB environment is used to implement the experiments. The data format containing acquired and preprocessed 3D surface data is *vrml*. Two kind of code is developed. The first is used to partially match the space curves. The other one is to match the two surfaces in the space. The theory described previous sections is used in the second implementation. The results of these experiments are presented in the next section.

In 3D, we used 3 dimensional FFT computations. The computational complexity of the 3D FFT is $O(N^3\log(N))$. The complexity of the partial matching operation is derived with similar methods defined in the previous chapter. Here, the time depends on the size of the 3D volume and the number of iterations for finding the translation and two angle of the rotation. The complexity for this operation can be defined as $O(C_2 n_{iter} N^3\log(N))$. However, we exhaustively search the third angle to find the final solution. By the use of the assumption that the number of the possible discrete angles is proportional to the size of the volume, the complexity becomes $O(C_2 n_{iter} N^4\log(N))$.

6.6 Results

The first results belong to partial space curve matching experiments. Two space curves taken from the boundary curve of pieces are applied the algorithm developed in this chapter. The curves are shown in Figure 6.13.

The first curve is inflated by using Gaussian function. Then, they are applied to FFT based algorithm. The second curve in the example is highly similar to a partition of the first curve. To test the erosion or deformation dependency, we deformed the second curve. The algorithm can match the curve even if they are distorted. The sigma, σ, value depends on the distortion level of the pieces. If the pieces are seriously distorted or eroded, the sigma value has to be taken higher. The value used in the samples without distortion may be relatively small, but it can not be as small as zero. The quantization or down-sampling in order to fit finite space always brings some amount of noise. This situation is mentioned in the previous theory section with a different view.

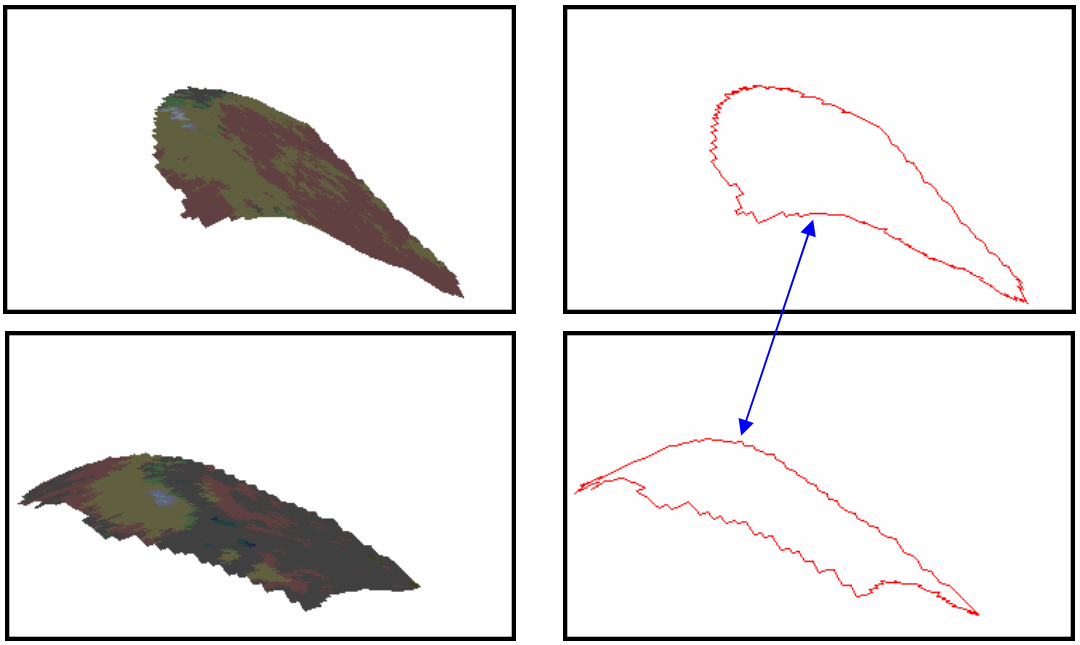


Figure 6.13 : The edge curves of two pieces from the puzzle.

The second group of experiments is related to the partial surface matching. Some images of the surfaces are shown in Figure 6.14. The expanded surfaces and the boundary lines are also presented in this figure.

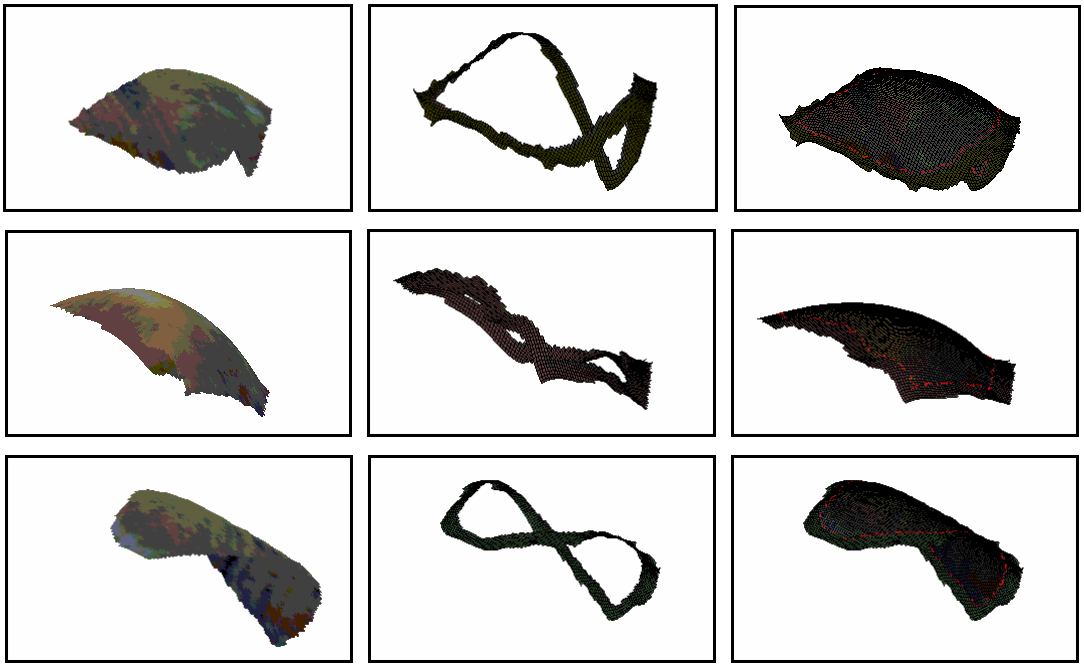


Figure 6.14 : Three sample pieces are shown in first column. The second column shows the expanded bands and last column represents the corresponding expanded pieces.

The primitive form of the matching algorithm is applied to some selected pieces. Two results are presented in Figure 6.15. The pieces mostly can not be placed to their true position, although the others can. The performance of the primitive form is not

sufficient to implement the algorithm for real data. The overall performance calculated for all possible pairs is 16%. Only 3 of 36 pairs can meaningfully reconstructed.

The final form of the matching algorithm is applied to the same pairings. The performance of the matching is improved. A sample result is shown in Figure 6.16. This improvement is obtained by using the possibility of existence approach. Although the results are better and the method may be used, we think that the method have to be improved in order to be able to apply to a real life problem.

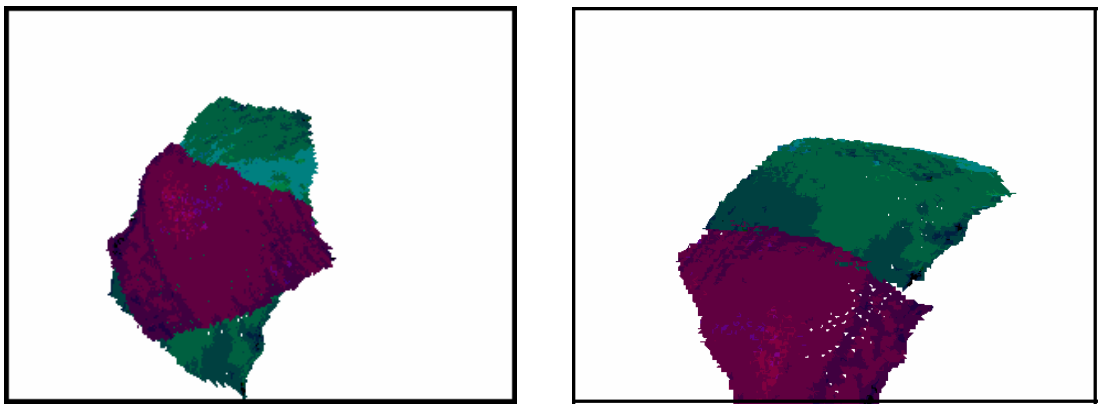


Figure 6.15 . (left) An undesired matching, and (right) a desired matching.

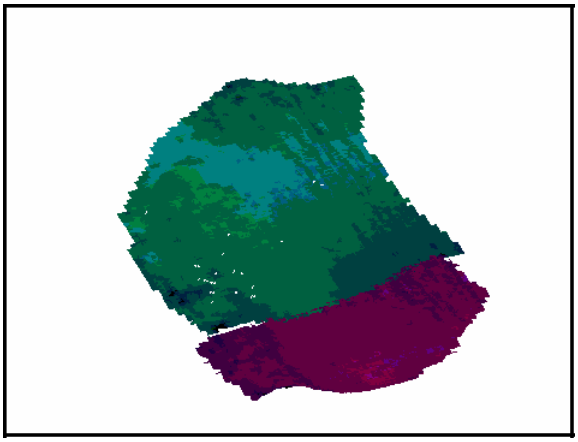


Figure 6.16 : A true matching.

The sigma value set a constant value in all experiments. We try different sigma values. In very small sigma values, the possibility of existence becomes unaffected. After a threshold, the above results are obtained. The solutions are not sensitive to the sigma values greater than mentioned threshold. It is also noted that very high sigma values remove the information of the existence of the surface, so the solution becomes impossible.

The last experiments are made to test the effect of the footprints on the surface pieces. The pieces with lines are processed as we described in previous section. The '*probability of the existence*' values in marked regions are arithmetically inverted. In some cases, the algorithm can manage to match the pairs, whereas they are not regularly assembled without this process.

In this chapter, we try to apply the partial matching algorithm to 3D pieces. This algorithm is first developed for the matching of 2D textural pieces. Here, we claim that the same algorithm may work for 3D applications. The experimental results show that the partial matching approach based on FFT methods can be used in 3D. However, we consider that the performance of the method is not sufficient. It has to be improved for the real life implementations in 3D. Furthermore, it has to be worked on the more complicated surface texture. The question "how can we embed both the 3D texture information and the probability of existence to the solution" has to be examined.

The implementation of the FFT may also be problem depending on environment. 3D FFT operations are memory critical and may also be computationally costly, nevertheless it is very fast method to implement.

Chapter 7

SUMMARY AND CONCLUSIONS

Our purpose in this research is to develop a method for the automated assembly of objects from their pieces. The task of reassembling has great importance in the fields of anthropology, failure analysis, forensics, art restoration and reconstructive surgery, restoration and reconstruction of archeological findings, repairing of broken objects, solving jigsaw type puzzles, molecular docking problem, etc.

Reassembling contains many problems endemic to pattern recognition, computer vision, feature extraction, boundary matching and optimization fields. The one of the main subtitle for the assembly problem is the partial matching of pieces. Previous works focus mainly on this partial matching problem by using geometrical properties of pieces. The puzzle pieces are represented by their boundary curves. As the fractions of boundaries are adjacent and thus similar, a pairwise affinity measure is computed by partial curve matching. The puzzle pieces usually include not only geometrical shape information but also visual information of texture, color, continuity of lines and so on. Moreover, textural information is mainly used to assembly pieces in some cases like classic jigsaw puzzles. This research presents a new approach that pictorial assembly, in contrast to previous curve matching methods, uses texture information as well as geometric shape. The assembly is performed using textural features and geometrical constraints.

In this research, the main idea used to solve the assembly problem can be explained in that the correlation between the features of the predicted region and the right neighbor is significantly higher than the alternative pairings. In order to realize this idea, we study four main topics. The first one is to develop a prediction algorithm to expand the pieces outwards. The second defines the performance measure that represents the appropriateness of the assembly based on textural features and geometrical shape. In the next study, we try to find the best transformations of pieces

that maximize harmony of textures of fragments while the geometrical constraints are being satisfied. Finally, the methods developed for the 2D pieces are applied to 3D problems.

The prediction algorithm automatically fills in this expanding region with information diffusing from the central part. We use the mixture of inpainting and the texture synthesis methods for prediction. While extending the fragment image, we introduce confidence of expanding as a new parameter in the prediction phase of assembly problem. This parameter represents the reliability of expanded values. Then, we derive the feature values of both original fragment and expanded region. The results of the expanding process are presented in the research. Because the expanded regions are not used as an output of the assembly problem, actually, the quality of the pictures on the expanded band is not important. It is sufficient that the expanded band have similar features with possible right neighbors. Thus, we think that the performance of this process is sufficient for such an application.

Actually, the assembly of fragments is to find the right transformations of pieces. To manage the assembly, we have to able to sense whether any arrangement of pieces becomes better or worse. We develop a measure called total affinity to be able to decide. According to the results of the experiments on artificial and real data, we determine that the harmony of pieces and the achievement of assembly improve while optimizing this affinity measure.

Although the puzzle assembly problem can be stated as the optimization of the above cost function, the optimization is too computationally costly. So, we use an assumption that the minimizing distance function is equivalent to maximizing the correlation between the pieces. Therefore, we use the FFT based methods to maximize the correlation between the predicted parts of a piece and other pieces. Two new assembly algorithms using mentioned FFT approach are introduced in this research. The first one is the semi-automated algorithm. This is developed for the interactive usage. This method is relatively fast but produces uncompleted or intermediate solutions. And also, there is not any theoretical limitation for the number of pieces according to semi-automated algorithm. We consider that this method can be implemented in the real life puzzle problems. The second one is a fully-automated assembly method. The algorithm for automated one has some limitations when the number of pieces, n_p, is greater than a threshold. The time for the solution is geometrically increasing while n_p is increasing. Another problem is that the final assembly may not be absolutely guaranteed in large n_p

values. It is useful for the puzzles that have a smaller n_p value than a threshold depending on the structure of pieces.

The last study regarding the assembly problem involves testing the developed methods in 3D. We collect and preprocess 3D real data for the experiments and future research on the same topic. The experimental results show that the partial matching approach based on FFT methods can be used in 3D. However, we consider that the performance of the method has to be improved for the real life implementations in 3D.

During our research, we produce 2 programs and a MATLAB implementation. The first program is used for generating expanded images from the acquired pictures. The same program generates feature values of the pieces for the next assembly process. The second program is the main one to assemble the pieces. It has a user interface to serve the user to open the data and run the assembly operations. The hot keys are implemented in order to move, rotate, merge, group or trigger the semi-automated or automated assembly algorithms.

As a summary:

- We used the texture information in classical puzzle problem. Textural information is very important to assemble the pieces. Using this information dramatically increases the assembly performance.
    - It can be used to order the related images with soft transitions.
    - Up to now, embedding the texture information to the assembly problem is supposed to be very hard operation in the previous research.
- We developed a measure that expresses the harmony of the assembly.
    - It can be used to order the related images with soft transitions.
    - The future research about the puzzle problem may accept this affinity as a common metric to compare the different assemblies.
- We developed a texture based partial matching method using the Fourier techniques. The FFT operations have many advantages for such applications. For the image registration, there may be more efficient techniques, but the nature of the assembly in some points differs from the image registration. We need more robust technique, and the registrations should not directly depend on the pixel's locations. In another words, we need an area based techniques. The FFT processes have all these properties. The partial image registration by FFT methods can be used as partial matching in proper applications, as ours.

- The predicting outwards (also expanding or morphing) for the images is practical for registration or matching. Especially, if we work on partial matching, the advantage of expanding increases more.
- The developed technique can be applied to the space curve matching problem.
- In all kinds of recognition problems using the silhouette (or boundary curves), the method may be tried.

- We show that image registration techniques can also be used in 3D matching. The FFT can also be used in 3D if the memory problems are overcome in the implementations. Even if there is a large amount of data, it is a fast process.
  - The data matching in 3D may be solved by the Fourier based image registration techniques. (Espacially in the noisy data)
  - For example, we may use the mentioned method for fitting of a candidate face data points cloud to a template data instead of classical ICP algorithms.

During the thesis, we emphasize some open problems in the result sections of the related chapters. These problems may be worked on by using the outcomes of this study. Some are:

- The assembly problem can be solved by directly optimizing the affinity measure depending on the translations of the pieces. For example, it may be assumed that the problem of obtaining the translations is similar to the *traveling salesman* problem, and the self organizing maps may be used to assemble the pieces.
- The developed partial curve matching method based on FFT theory may be applied to other applications.
- The texture expanding method may be implemented into the new research fields such as medical image registration and ours.
- The embedding the texture information into the 3D assembly method explained in the previous chapter may be improved.

Bibliography

[1] R. Chen, and Z. Weng, "Docking unbound proteins using shape complementarity desolvation and electrostatics", *Proteins*, vol. 47(3), May 2002, pp. 281-94

[2] A. Freeman, and L. Garder, "A pictorial jigsaw puzzles the computer solution of a problem in pattern recognition", *IEEE Transactions Electron. Comput.*, 13(1964), pp. 118-127

[3] D. Goldberg, C. Malon, M. Bern, "A global approach to automatic solution of jigsaw puzzles", In *Proc. of Conference on Computational Geometry*, 2002, pp. 82-87

[4] D. A. Kosiba, P. M. Devaux and S. Balasubramanian, "An automatic jigsaw puzzle solver", In *Proc. of ICPR*, 2001

[5] A. Glassner, "Putting pieces together", *IEEE Computer Graphics and Applications*, vol. 22(3), pp. 76-86, May/June 2002

[6] Y. Murai, H. Tatsumi, and S. Tokumasu, "Game theoritic approarch to placement problems of rectilinear blocks solution of rectilinear jigsaw puzzle", *IEEE NANO*, October 2001, pp. 128-133

[7] H. J. Wolfson, "On curve matching", *IEEE Transactions on Pattern Analysis and Machine Intelligence* , Vol. 12, May 1990

[8] W. Kong and B.B.Kimia, "On solving 2D and 3D puzzles using curve matching", In *Proc. of CVPR*, Hawaii, USA, 2001

[9] A. M. Radack, and N. I. Badler, "Jigsaw puzzle matching using a boundary centered polar encoding", In *CGIP*, 19(1982), pp. 1-17

[10] G. Üçoluk and ?. H. Toroslu, "Automatic reconstruction of broken 3D surface objects", *Computers and Graphics*, Vol. 23, Agust 1999, pp. 573-582

[11] H. C. Da Gama Letio, and J. Stolfi, "A multiscale technique for computer assisted reassembly of fragmented objects", *IEEE Transactions on Pattern Analysis and Machine Intelligence* , Vol. 24, No. 9, September 2002, pp. 1239-1251

[12] M. Levoy, K. Pulli, B. Curless et al., "The digital michelangelo project 3D scanning of large statues", In *Proc. of Computer Graphics (SIGGRAPH)*, 2000

[13] G. Papaioannou, E.A. Karabassi, and T. Theoharis , "Virtual archaeologist assembling the past", *IEEE Computer Graphics and Applications*, 21(2)(2001), pp. 53-59

[14] A. R. Willis and D. B. Cooper, "Bayesian assembly of 3D axially symmetric shapes from fragments", In *Proc. of CVPR*, 2004, pp. 1063-1069

[15] D.B. Cooper, A. Willis, S. Andrews et al., "Bayesian pot-assembly from fragments as problems in perceptual grouping and geometric-learning", In *Proc. of ICPR*, 2002, p. 30297

[16] A. Willis, X. Orriols, D. B. Cooper, "Accurately estimating sherd 3D surface geometry with application to pot reconstruction", In *CVPR workshop on applications of computer vision in archaeology*, 2003, p. 5

[17] M. Fornasier, and D. Toniolo, "Fast robust and efficient 2D pattern recognition for reassembling fragmented images", *Journal of Pattern Recognition*, vol. 38(11), 2005, pp. 2074-2087

[18] M. Kampel, R. Sablatnig, "Profile based pottery reconstruction", In *CVPR workshop on applications of computer vision in archaeology*, 2003, p. 4

[19] A. Kulak, "Analysis of textural image features for content based retrieval", *MSc Thesis*, Sabancı University, October 2002

[20] F. Ritter, T. Srothotte, O. Dresden, and B. Preim, "Virtual 3D puzzles a new method for exploring geometric models in VR", *IEEE Computer Graphics and Applications*, vol. 21(5), September/October, 2001, pp. 11-13

[21] B. Wolfson, E. Schonberg, A. Kalvin, and Y. Lambdan , "Solving jigsaw puzzles by computer", *Annals of Operations Research*, 12(1998), pp. 51-64

[22] A. Yamamoto, H. Ninomiya, and H. Asai, "Application of neuro based optimization algorithm of three dimensional cylindric puzzles", In *Proc. of IEICE/ITC-CSCC*, vol.1, June 1996, pp.377-380

[23] M. Kajiura, Y. Akiyama, and Y. Anzai, "Solving large scale puzzles with neural networks", *Proceedings Tools for AI Conference*, Fairfax, 1990, 562-569

[24] H. Yamamoto, H. Ninomiya, and H. Asai, "Application of neuro based optimization to 3D rectangular puzzles", In *Proc. of IEEE&INNS/IJCNN'98* , May 1998

[25] M. G. Chung, M. M. Fleck, and D. A. Forsyth, "Jigsaw puzzle solver using shape and color", In *Proc. of ICSP*, 1998, pp. 877-880

[26] F. Toyama, Y. Fujiki, K. Shoji, and J. Miyamichi, "Assembly of puzzles using a genetic algorithm", In *Proc. of ICPR*, 2002, p. 40389

[27] T. Kennedy, "Jigsaw puzzle solver using computervision", *BSc Thesis*, University of Queensland, October 1998

[28] P. N. Suganthan, "Solving jigsaw puzzles using hopfield neural networks", *International Joint Conference on Neural Networks, (IJCNN)*, vol. 6, July 1999, pp. 3745-3749

[29] R. W. Webster, P. S. LaFollete, and R. L. Stafford, "Isthmus critical points for solving jigsaw puzzles in computer vision", *IEEE Transactions on Sytems, Man, and Cybernetics*, Vol. 21, No. 5, September 1991, pp. 1271-1278

[30] C.A. Rothwell, A. Zisserman, D.A. Forsyth, and J.L. Mundy, "Canonical frames for planar object recognition", In *Proc. of the 2nd European Conference on Computer Vision*, Italy, May 1992, pp. 757-772

[31] G. C. Burdea, and H. J. Wolfson, "Solving jigsaw puzzles by a robot", *IEEE Transactions on Robotics and Automation* , Vol. 5, December 1989, pp. 752-764

[32] T. B. Sebastian, J. J. Crisco, P. N. Klein, and B. B. Kimia, "Constructing 2D curve atlases", In *Proc. Of Mathematical Methods in Biomedical Image Analysis*, 2000, pp. 70-77

[33] H. C. Da Gama Letio, and J. Stolfi, "Automatic reassembly of irregular fragments", *Technical report IC-98-06*, Univ. Of Campinas, 1998

[34] H. C. Da Gama Letio, and J. Stolfi, "Information contents of fracture lines", *Technical report IC-99-24*, Univ. Of Campinas, 1999

[35] T. B. Sebastian, P. N. Klein and B. B. Kimia, "Aligment-based recognition of shape outlines", *IWVF* , 2001, pp. 606-618

[36] G. Baraquet, and M. Sharir, "Partial surface and volume matching in three dimensions", *IEEE Transactions on Pattern Analysis and Machine Intelligence* , Vol. 19, No. 9, September 1997, pp. 929-948

[37] G. Papaioannou, T. Theoharis, "Fast fragment assemblage using boundary line and surface matching", In *CVPR workshop on applications of computer vision in archaeology*, 2003, p. 2

[38] D.B. Cooper, A. Willis, S. Andrews et al., "Assembling virtual pots from 3D Measurements of their fragments", In *Proc. of Virtual reality, archeology, and cultural heritage*, 2001, pp. 241-254

[39] A. Willis, X. Orriols, S. Velipasalar et al., "Extracting axially symmetric geometry from limited 3D range data", *Technical Report of Shape Lab*, Brown University, 2001

[40] A. R. Willis, "Stochastic 3D geometric models for classification deformation and estimation", *PhD Thesis*, Brown University, May 2004

[41] J. McBride, "Archaeological fragment reassembly using curve matching", *MSc Thesis*, Brown University, May 2003

[42] K. Hlavackova-Schindler, M. Kampel, and R. Sablatnig, "Fitting of a closed planar curve representing a profile of an archaeological fragment", In *Proc. of Virtual Reality, Archaeology and Cultural Heritage*, Athens, Greece, November 2001

[43] M. Kampel and R. Sablating, "Automated segmentation of archeological profiles for classification", In *Proc. of ICPR*, 2002, p. 10057

[44] R. Sablatnig, and C. Menard, "3D reconstruction of archeaological pottery using profile primitives", In *Proc. of International Workshop on Synthetic-Natural Hybrid Coding and Three-Dimensional Imaging*, pp. 93-96, 1997

[45] S. Tosovic, and R. Sablatnig, "3D modeling of archealogical vessels using shape from silhouette", In *3DIM, 2001*, p. 51

[46] R. Sablatnig, S. Tosovic, and M. Kampel, "Combining shape from silhouette and shape from structured light for volume estimation of archaeological vessels", In *Proc. of 16th International Conference on Pattern Recognition*, vol. 1, 2002, pp. 364-367

[47] M. Levoy, "Digitizing the forma urbis romea", http://graphics.stanford.edu/projects/forma-urbis/forma-urbis.html

[48] 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, January 2002, pp. 114-124

[49] K. Hori, M. ?mai, and T. Ogasawara, "Joint detection for pot sherds of broken earthenware", In *Proc. of CVPR*, 1999, p. 2440

[50] M. Kampel and R. Sablating, "Color classification of archaeological fragments", In *Proc. of ICPR*, 2000, p. 4771

[51] "3D murale:A multimedia system for archaeology", http://users.monash.edu.au/~konrads/papers/fim02.pdf

[52] J. Y. Zheng, Z. L. Zhang, and N. Abe, "Virtual recovery of excavated archaeological finds", In *Proc. of Multimedia Computing and Systems* , July 1998, pp. 348-357

[53] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, "Image inpainting", In *Proc. of ACM Conf. Comp. Graphics (SIGGRAPH)*, New Orleans, July 2000, pp. 417-424

[54] T. Chan, and J. Shen, "Mathematical models for local non-texture inpaintings", *SIAM Journal on Applied Mathematics*, 62(3), pp. 1019-1043, 2002

[55] S. D. Rane, G. Sapiro, and M. Bertalmio, "Structure and texture filling-in of missing image blocks in wireless transmission and comression applications", *IEEE Transactions on Image Processing*, vol. 12(3), march 2003, pp. 296-303

[56] A. Levin, A. Zomet, and Y. Weiss, "Learning how to inpaint from global image statistics", In *Proc. of ninth International Conference on Computer Vision*, vol. 2, 2003, pp. 305

[57] M. M. Oliveira, B. Bowen, R. McKenna, and Y. S. Chang, "Fast digital image inpainting", In *Proc. of Visualization, Imaging, and Image Processing*, Marbella, Spain, September 2001, pp. 261-266

[58] M. Bertalmio, A. L. Bertozzi, and G. Sapiro, "Navier stokes fluid dynamics and image and video inpainting", In *Proc.of Conf. Computer Vision Pattern Recognition*, Hawai, December 2001, pp. I:355-365

[59] A. Ballester, M. Bertalmio, V. Caselles et al., "Filling in by joint interpolation of vector fields and gray levels", *IEEE Transactions on Image Processing*, 10(1), August 2001

[60] A. Ballester, M. Bertalmio, V. Caselles et al., "A variational model for filling-in gray level and color images", In *Proc. of ICCV*, 2001, p. 10

[61] G. Sapiro, "Image inpainting", from *SIAM News*, Vol. 35, No 4

[62] J. Davis, S. R. Marschner, M Garr, and M. Levoy, "Filling holes in complex surfaces using volumetric diffusion", *First International Symposium on 3D Data Processing, Visualization, and Transmission*, June 2002

[63] J. Verdera, V. Caselles, M. Bertalmio, and G. Sapiro, "Inpainting surface holes", In *Proc. of ICIP*, September 2003, pp. II:903-906

[64] T. F. Chan, and J. Shen, "Non-texture inpainting by curvature driven diffusions", *Journal of Visual Communication and Image Representation*, vol. 12(4), 2001, pp. 436-449

[65] T. F. Chan, S. H. Kang, and J. Shen, "Euler's elastica and curvature based inpaintings", *SIAM Journal on Applied Mathematics*, 2002

[66] T. F. Chan, and J. Shen, "Morphologically invariant PDE inpaintings", *UCLA CAM Report*, 2001

[67] S. Esedoglu, and J. Shen, "Digital inpainting based on the mumford-shah-euler image model", *European J. Appl. Math.*, Vol. 13, 2002, pp. 353-370

[68] "Image inpainting", http://www.math.ucla.edu/~imagers/htmls/inp.html

[69] A. A. Efros, and T. K. Leung, "Texture synthesis by non-parametric sampling", *IEEE Internation conference on Computer Vision*, Greece, September 1999, pp. 1033-1038

[70] Drori,D. Cohen-Or,H. Yeshurun , "Fragment based image completion", In *Proc. of ACM SIGGRAPH*, Vol. 22(3), 2003, pp. 303-312

[71] A. Zalensny, V. Ferrari, G. Caenen, and L. V. Gool, "Parallel composite texture synthesis", In *Texture 2002 Workshop-(with ECCV02),* Denmark, June 2002

[72] M. Bertalmio, L. Vese, G. Sapiro, and S. Osher, "Simultaneous structure and texture image inpainting", *IEEE Transactions on Image Processing*, vol. 12(8), August 2003, pp.882-889

[73] L. A. Vese, and S. J. Osher, "Modelling textures with total variation minimization and oscillating patterns in image processing", *Journal of Scientific Computing*, Vol. 19, December 2003

[74] P. Harrison, "A nonhierarchical procedure for resynthesis of complex textures", In *Proc. Int. Conf. Central Europe Comp. Graphics*, Visua. and Comp. Vision, Czech Republic, February 2001

[75] A. Criminisi, P. Perez, and K. Toyama, "Region filling and object removal by exemplar based image inpainting", *IEEE Transactions on Image Processing*, Vol. 13, No.9, September 2004, pp. 1-13

[76] M. Ş. Sağıroğlu, A. Erçil, C. Özmen, and S. Balcısoy, "Texture based approach to puzzle assembly", *Virtual Retrospect*, Biarritz, France, November 2005

[77] S. D. Rane, J. Remus, G. Sapiro, "Wavelet domain reconstruction of lost blocks in wireless image transmission and packet switched networks", In *Proc. of Image Processing*, vol. 1, 2002, pp. I:309-312

[78] M. Ş. Sağıroğlu, and A. Erçil, "A texture based approach to reconstruction of archaeological finds", In *Proc. of Virtual Reality, Archaeology and Cultural Heritage*, Athens, Italy, November 2005

[79] M. Ş. Sağıroğlu, and A. Erçil, "Dizilim problemine dokusal tabanlı bir yaklaşım", *Sinyal İşleme Kurultayı*, Turkey, April 2005

[80] L. G. Brown, "A survey of image registration techniques", *ACM Computing Surveys*, vol. 24(2), pp. 325-376, 1992

[81] A. Sticker, "Tracking with reference images a real time and markerless tracking solution for out-door augmented reality applications", In *Proc. of Virtual reality, archeology, and cultural heritage*, Greece, 2001, pp. 77-82

[82] B. Zitova, and J. Flusser, "Image registration methods: a survey", *Image and Vision Computing*, 21(2003), pp. 977-100

[83] M. McGuire, "An image registration technique for recovering rotation scale and translation parameters", *Technical Report NEC Res. Inst.*, TR 98-018, February 1998

[84] W. S. Hoge, and C. F. Westin, "Identification of translational displacements between N-dimensional data sets using the high order SVD and phase correlation", *IEEE Transactions on Image Processing*, vol. 14(7), July 2005, pp. 884-889

[85] A. Makadia, and K. Daniilidis, "Direct 3D rotation estimation from spherical images via a generalized shift theorem", In *Proc. of CVPR*, vol. 2, June 2003, pp. II:217-224

[86] S. Krüger, and An Calway, "Image registration using multiresolution frequency domain correlation", In *Proc. of British Machine Vision Conf*, 1998

[87] E. D. Castro, and C. Morandi, "Registration of translated and rotated images using finite fourier transforms", *IEEE Transactions on Pattern Analysis and Machine Intelligence* , Vol. 9, No. 5, Semptember 1987, pp. 700-703

[88] G. Wolberg, and S. Zokai, "Robust image registration using log-polar transform", In *Proc. of IEEE ICIP*, 2000

[89] M. Ş. Sağıroğlu, and A. Erçil, "Otomatik dizilim problemi için FFT tabanlı uyumlama yaklaşımı", *Sinyal İşleme Kurultayı*, Turkey, April 2006

[90] M. Ş. Sağıroğlu, and A. Erçil, "A texture based matching approach for automated assembly of puzzles", accepted in *ICPR*, 2006

[91] M. Ş. Sağıroğlu, A. Erçil, and S. Balcısoy, "Automated reconstruction of archaeological finds", Presented in *Computer Applications and Quantitative Methods in Archaeology*, April 2006

[92] "Appendix F: determination of translation of the translation rotation and scaling" http://etd.lsu.edu/docs/available/etd-10132004-211301/unrestricted/17AppendixF.pdf

Appendix A

Determination of the Translation

Below descriptions are collected by using of the source [92]. All variables and the operations are arranged in accordance with the terminology used previously. We will assume that:

$$I(\vec{x}) = I(x, y) \quad \text{and} \quad F(\vec{w}) = H(u, v) \tag{A.1}$$

It is known that if two images, $I_1(\vec{x})$ and $I_2(\vec{x})$ differ only by shift (translation), i.e. if $I_2(\vec{x}) = I_1(\vec{x} + \vec{a})$, where $\vec{a}$ is some unknown shift, then their Fourier transforms are:

$$F_1(\vec{w}) = \frac{1}{2\pi} \iint I_1(\vec{x}) e^{-2\pi i(\vec{x}\vec{w})} dxdy \tag{A.2}$$

$$F_2(\vec{w}) = \frac{1}{2\pi} \iint I_2(\vec{x}) e^{-2\pi i(\vec{x}\vec{w})} dxdy \tag{A.3}$$

Their Fourier transforms are then related by the following equations:

$$F_2(\vec{w}) = e^{-2\pi i(\vec{w}\vec{a})} F_1(\vec{w}) \tag{A.4}$$

and also

$$\left|F_2(\vec{w})\right| = \left|F_1(\vec{w})\right| \tag{A.5}$$

To obtain shift $\vec{a}$, equation (A.4) is used to compute the value of the following ratio:

$$R(\vec{w}) = \frac{F_2(\vec{w})}{F_1(\vec{w})} = e^{-2\pi i(\vec{w}\vec{a})} \tag{A.6}$$

Inverse Fourier transform of this ratio is equal to the delta function $\delta(\vec{x} - \vec{a})$. This inverse Fourier transform is equal to zero everywhere except for the point $\vec{x} = \vec{a}$. So the desired shift $\vec{a}$ can be determined from the fact that it represents only value for which inverse Fourier of the ratio is not equal to zero.

In the ideal case the absolute value of the ratio is equal to 1. In real life images have some noise in them. In the presence of noise observed values of the intensities may

differ from the actual values, and as a result the absolute value of the ratio $R(\vec{w})$ may be different from 1. To find out exact value of the ratio in the presence of noise following can be done. Let

$$e^{-2\pi i(\vec{w}\vec{a})} = b \tag{A.7}$$

then in absence of noise:

$$F_2(\vec{w}) = bF_1(\vec{w}) \tag{A.8}$$

In the presence of noise Fourier transforms, $F_1(\vec{w})$ and $F_2(\vec{w})$, can be different from the actual values so the equation (A.8) changes to

$$F_2(\vec{w}) \approx bF_1(\vec{w}) \tag{A.9}$$

As stated before it is also known that absolute value of b is equal to 1:

$$|b| = 1 \quad \text{i.e.} \quad |b|^2 = b.b^* = 1 \tag{A.10}$$

where b^* is complex conjugate to b.

Now the best estimate for b that satisfies condition in equation (A.10) and approximation equation (A.9) has to be found. For this task the Least Square Method can be used. This method assumes that the best curve fitting of a given type is the curve that has the minimal sum of the deviations squared (least square error) from a given set of data. According to this method for each estimation of *b*, error (*E*) can be defined as follows:

$$E = F_2(\vec{w}) - bF_1(\vec{w}) \tag{A.11}$$

Then among all estimates that satisfy the additional condition in equation (A.10), a value of *b* for which the square of error, $|E|^2 = E.E^*$, is minimum is found. Knowing

$$\begin{aligned} |E|^2 = E.E^* &= \left[F_2(\vec{w}) - bF_1(\vec{w})\right]\left[F_2{}^*(\vec{w}) - b^*F_1{}^*(\vec{w})\right] \\ = F_2(\vec{w}).F_2{}^*(\vec{w}) - b^*F_2(\vec{w}).F_1{}^*(\vec{w}) &- bF_1(\vec{w}).F_2{}^*(\vec{w}) + |b|^2 F_1(\vec{w}).F_1{}^*(\vec{w}) \end{aligned} \tag{A.12}$$

Resulting expression has to be minimized under the constraint in Equation (A.10). Constraint optimization is the minimization of an objective function subject to constraints on the possible values of the independent variable. The typical constrained optimization problem has the following form:

$$\underset{x}{\text{argmin}}\ f(x) \quad \text{subject to g(x)} = 0 \tag{A.13}$$

where $f(x)$ is the scalar-valued objective function and $g(x)$ is the vector-valued constraint function.

The classical approach to solving constraint optimization problems is the method of Lagrange Multipliers. This approach converts the constrained optimization problem into an unconstrained one. The Lagrangian of a constraint optimization problem is defined to be the scalar-valued function

$$L(x,\lambda) = f(x) + \lambda^T \mathrm{g(x)} \tag{A.14}$$

where λ is Lagrange multiplier.

Stationary points of the Lagrangian are potential solutions of the constrained optimization problem, as always each candidate solution must be tested to determine which one minimizes the objective function. As shown in Figure A.1, the constraint corresponds to a contour in the *x* plane.

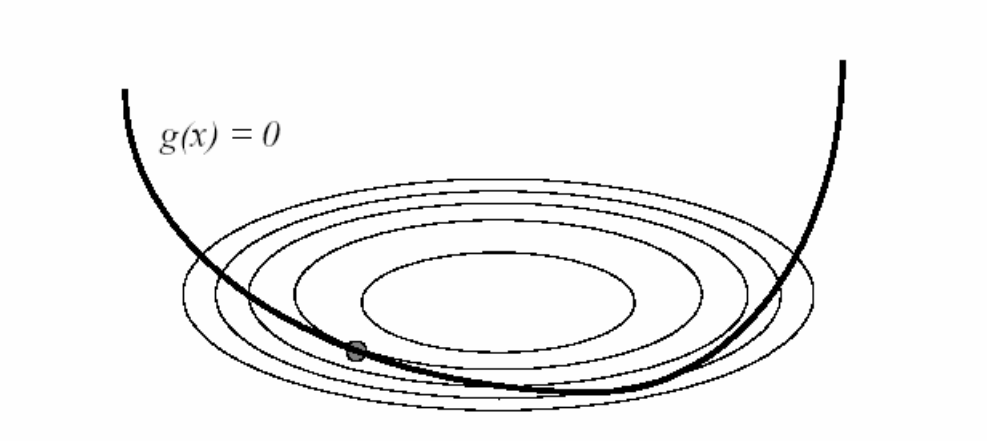


Figure A.1 : Geometric interpolation of Lagrange multipliers (taken from [92])

In the above figure thick line corresponds to the contour of the values of *x* satisfying constraint equation $g(x) = 0$. The thinner lines are contours of constant values of the objective function $f(x)$. The contour corresponding to the smallest value of the objective function just tangent to the constraint contour is the solution to the optimization problem with equality constraints.

In the other words minimum of the function $f(x)$ under the constraint $g(x) = 0$ is attained when for some real number λ function $L(x,\lambda)$ attains its unconstraint minimum.

This leads to finding the value of the complex variable *b* for which the expression

$$F_2(\vec{w}).F_2^*(\vec{w}) - b^* F_2(\vec{w}).F_1^*(\vec{w}) - bF_1(\vec{w}).F_2^*(\vec{w}) + |b|^2 F_1(\vec{w}).F_1^*(\vec{w}) + \lambda(b.b^* - 1) \tag{A.15}$$

takes the smallest possible value.

Since complex variable is in effect a pair of two real variables minimum can be found as a point at which the partial derivatives with respect of each of these variables

are both equal to zero. If equation (A.15) is differentiated relative to b^* following linear equation is obtained:

$$-F_2(\vec{w}).F_1^*(\vec{w}) + bF_1(\vec{w}).F_1^*(\vec{w}) + \lambda b \tag{A.16}$$

From this equation b can be found as

$$b = \frac{F_2(\vec{w}).F_1^*(\vec{w})}{F_1(\vec{w}).F_1^*(\vec{w}) + \lambda} \tag{A.17}$$

From the condition in equation (A.10) that value b should satisfy, the coefficient λ can be determined. Knowing that denominator in equation (A.17) is a real number, it is sufficient to find a value of this denominator for which $|b|^2 = b.b^* = 1$. To achieve this, the absolute value of the numerator should be taken as denominator, i.e.

$$\left|F_2(\vec{w}).F_1^*(\vec{w})\right| = \left|F_2(\vec{w})\right|.\left|F_1^*(\vec{w})\right| \tag{A.18}$$

and now expression for b, equation (A.17), has a following form

$$b = \frac{F_2(\vec{w}).F_1^*(\vec{w})}{\left|F_2(\vec{w})\right|.\left|F_1^*(\vec{w})\right|} \tag{A.19}$$

The ratio $R(\vec{w}) = \frac{F_2(\vec{w})}{F_1(\vec{w})}$ from the ideal non-noise case becomes

$$R(\vec{w}) = \frac{F_2(\vec{w}).F_1^*(\vec{w})}{\left|F_2(\vec{w})\right|.\left|F_1^*(\vec{w})\right|} \tag{A.20}$$

in the presence of noise.

From the equation above it is obvious that inverse Fourier transform of $R(\vec{w})$ in the presence of noise has values that are slightly different from the delta function, but still absolute value of the inverse Fourier transform should be much larger at point $\vec{x} = \vec{a}$ than all the other values of this function. In this case desired shift $\vec{a}$ represents the point for which the absolute value of the inverse Fourier transform takes the largest possible value.

Appendix B

Using Log-Polar Coordinates for the Determination of Rotation & Scaling

Below descriptions are collected by using of the source [92]. All variables and the operations are arranged in accordance with the terminology used previously.

If one image differs from another not only by shift, but also by the rotation and scaling then the absolute values (magnitudes) of their Fourier transforms are not equal, but also differ by the corresponding rotation and scale.

By using transformation to go from Cartesian to polar coordinates (r,θ) in the $\vec{w}$ plane, rotation by an angle θ_0 is described by simple shift $\theta \to \theta - \theta_0$. In the polar coordinates scaling is also simple but cannot be described by simple shift $r \to \lambda r$. On the other hand if transformation is made form Cartesian to log-polar coordinates (ξ,η) where $\xi = \log(r)$ and $\eta = \theta$ then scaling can be also represented by the shift $\xi \to \xi - c$, where $c = \log(\lambda)$. From this is obvious that log-polar transformation can be used to describe both rotation and scaling as a shift.

Log-Polar Transformation of an Image

The log-polar transformation is a conformal mapping from the points on the Cartesian plane (x,y) to points in the log-polar plane (ξ,η) as shown in Figure B.1.

If $I(x,y)$ is an image with support on a rectangular set in the Euclidean plane, then the log-polar transform with origin (x_0,y_0) is described by following mapping:

$$\xi = M\log(r+\alpha) \qquad \text{where} \quad r = \sqrt{(x-x_0)^2+(y-y_0)^2} \tag{B.1}$$

$$\eta = \tan^{-1}\left(\frac{y-y_0}{x-x_0}\right) \tag{B.2}$$

The transform maps a 2D image onto the surface of a cylinder. The cylinder is indexed by ξ and η. The ξ-axis is parallel to the axis of the cylinder. The η-axis forms a circle around the cylinder.

As shown above log-polar image is produced through a projection onto an image plane, which is not sampled in a rectangular (x,y)-grid, but in the following way: the pixels are arranged in concentric circular rings around the focus of attention. On each

ring the same number of pixels is sampled, and the pixel size increases exponentially with growing radius of the rings in such a way, that all pixels are approximately square (see Figure B.1).

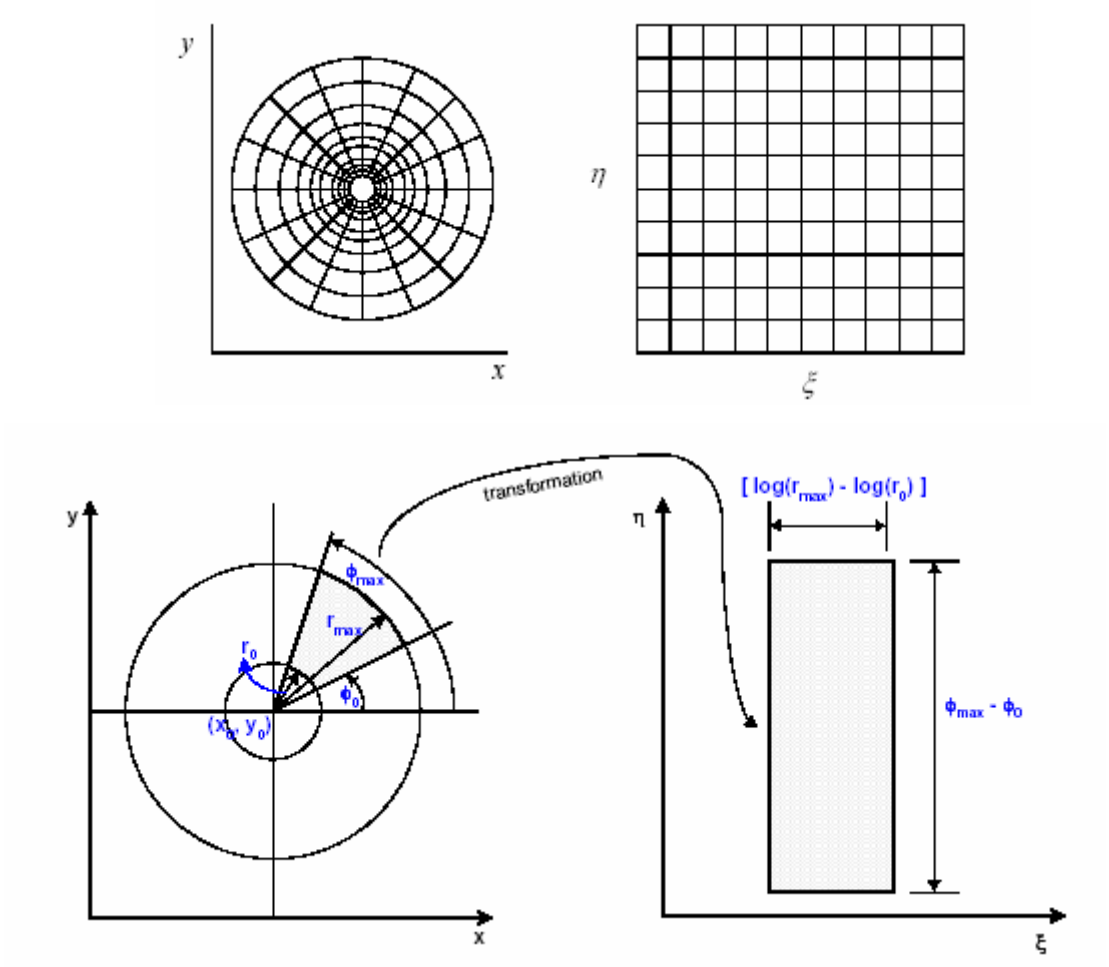


Figure B.1 : Cartesian and log-polar planes (taken from [92])

After log-polar image is mapped back to Cartesian coordinates the decrease in resolution with increasing radius can be observed.

Determining Rotation and Scaling

Log-polar transformation can be used to describe both rotation and scaling as a shift. To determine rotation and scaling needed for the image registration both images have to be transformed from the original Cartesian coordinates to log-polar coordinates. Next same Fourier transforms as described for the translation determination are used to determine the corresponding shift $(\theta_0, \log(\lambda))$, from these values rotation angle θ_θ and scaling factor λ are reconstructed.

However, computing (ξ, η) from the original rectangular grid leads to points that are not located exactly at points in the original grid. Thus, interpolation is needed to find

a value of $abs(F(\vec{w}))$ on the desired grid. A bilinear interpolation is used for resampling. Knowing the transformation relationship between the log-polar plane and Cartesian plane, point (x, y) in Cartesian plane is related to the desired grid point (ξ, η) by:

$$\begin{aligned} x &= r\cos(\eta) = 10^{\xi}\cos(\eta) \\ y &= r\sin(\eta) = 10^{\xi}\sin(\eta) \end{aligned} \tag{B.3}$$

To find value of $A(\vec{w}) = abs(F(\vec{w}))$ i.e. $A(x, y)$ using bilinear interpolation, intensities $A_{i,j}$, $A_{i+1,j+1}$, $A_{i,j+1}$, and $A_{i+1,j+1}$ of four original grid points $(i, j), (i+1, j), (i, j+1),$ and $(i+1, j+1)$ that surround point are used:

$$A(x, y) = (1-u)(1-v)A_{i,j} + u(1-v)A_{i+1,j} + (1-u)vA_{i,j+1} + uvA_{i+1,j+1} \tag{B.4}$$

where u is fractional part of x and v is fractional part of y, see Figure B.2.

Relations used:

$$\begin{aligned} x_0 &= \text{integer}(x) \quad \text{and} \quad x_1 = x_0 + 1 \\ y_0 &= \text{integer}(y) \quad \text{and} \quad y_1 = y_0 + 1 \end{aligned} \tag{B.5}$$

So

$$\begin{aligned} A(x, y_0) &= (x_1 - x)A(x_0, y_0) + (x - x_0)A(x_1, y_0) \\ A(x, y_1) &= (x_1 - x)A(x_0, y_1) + (x - x_0)A(x_1, y_1) \end{aligned} \tag{B.6}$$

Finally

$$\begin{aligned} A(x, y) &= (y_1 - y)A(x, y_0) + (y - y_0)A(x, y_1) \\ A(x, y) &= (y_1 - y)(x_1 - x)A(x_0, y_0) + (y_1 - y)(x - x_0)A(x_1, y_0) + \\ &\quad (y - y_0)(x_1 - x)A(x_0, y_1) + (y - y_0)(x - x_0)A(x_1, y_1) \end{aligned} \tag{B.7}$$

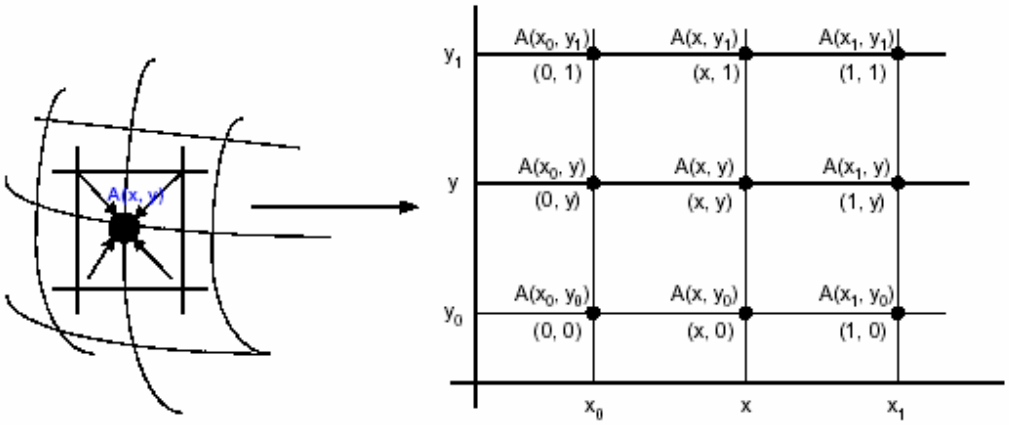


Figure B.2 : Bilinear interpolation (taken from [92])

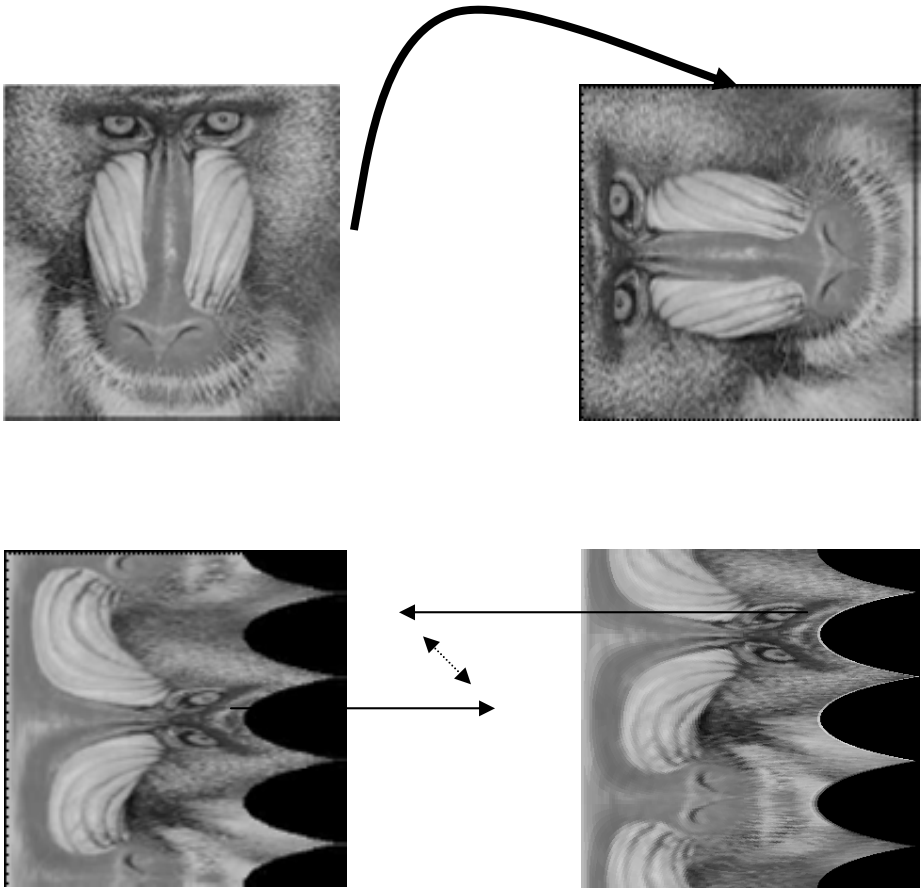


Figure B.3 : A sample image shows the relation between Cartesian and Polar coordinates.

Appendix C

Using Polar Coordinates for the Determination of only the Rotation

As we stated in chapter 4 and defined in Appendix A, the max phase correlation gives the best translation as:

$$(x_0, y_0) = \text{imax}\left(\overline{F}\left(\frac{F\ (I_0).F^*(I_1)}{\left|F\ (I_0)\right|.\left|F^*(I_1)\right|} \right) \right) \tag{C.1}$$

where imax is a function that returns the index of the max value of input. $\overline{F}$ is the inverse Fourier operator and *I* is represents the images.

Additionally, the phase correlation operation can also give the best scaling and rotation by using log-polar coordinates (See Appendix B). But (as we described in chapter 4) the scaling is not a variable in a puzzle problems. So, we do not need to use log-polar coordinates. We will use directly polar coordinates instead of log-polar coordinates.

When we transform the image to polar coordinates, we obtain another 2D image (Figure B.3). Using this image, we can find two parameters by applying a phase correlation method. These parameters refer to scaling and rotation, as we know. Here, we will deform this operation to find the ideal rotation with a constraint of constant scaling.

In ideal case, the solution can be obtained using any $r = r_c$ value as:

$$\theta = \text{imax}\left(\overline{F_1}\left(\frac{F_1\ ({I_0}^T(r = r_c, \theta)).{F_1}^*({I_1}^T(r = r_c, \theta))}{\left|F_1\ ({I_0}^T(r = r_c, \theta))\right|.\left|{F_1}^*({I_1}^T(r = r_c, \theta))\right|} \right) \right) \tag{C.2}$$

Here, F_1 represents the 1D Fourier operation and the polar transformed form of image I_0 is shown by I_0^T. The notation $I_0^T(r = r_c, \theta)$ is used to symbolize the 1D sequence that consists of the values in $r = r_c$ for all degrees.

(Note that this assumption is valid only for sufficiently complex images. In another word, there should be no ambiguous rotations for $r = r_c$.)

But this operation may fail for non-ideal cases. So we have to use all information to find rotation (instead of calculating using only $r = r_c$).

$$\theta = \text{imax}\left(\overline{F_1}\left(\frac{F_1\ (I_0{}^T(r=r_0,\theta)).F_1^*(I_1{}^T(r=r_0,\theta))}{\left|F_1\ (I_0{}^T(r=r_0,\theta))\right|.\left|F_1^*(I_1{}^T(r=r_0,\theta))\right|} \right) + \right.$$

$$\overline{F_1}\left(\frac{F_1\ (I_0{}^T(r=r_1,\theta)).F_1^*(I_1{}^T(r=r_1,\theta))}{\left|F_1\ (I_0{}^T(r=r_1,\theta))\right|.\left|F_1^*(I_1{}^T(r=r_1,\theta))\right|} \right) + \ldots\ldots \qquad \text{(C.3)}$$

$$\left. \ldots\ldots + \overline{F_1}\left(\frac{F_1\ (I_0{}^T(r=r_{n_r},\theta)).F_1^*(I_1{}^T(r=r_{n_r},\theta))}{\left|F_1\ (I_0{}^T(r=r_{n_r},\theta))\right|.\left|F_1^*(I_1{}^T(r=r_{n_r},\theta))\right|} \right) \right)$$

Here n_r is the number of different r-values or we can say the width of the images in polar coordinates. The summation of the phase correlations for all radius give the over all phase correlation value for a fix scale. By using of linearity of Fourier operation, we will reorganize the equation (C.3) as:

$$\theta = \text{imax}\left(\overline{F_1}\left(\frac{F_1\ (I_0{}^T(r=r_0,\theta)).F_1^*(I_1{}^T(r=r_0,\theta))}{\left|F_1\ (I_0{}^T(r=r_0,\theta))\right|.\left|F_1^*(I_1{}^T(r=r_0,\theta))\right|} + \right.\right.$$

$$\frac{F_1\ (I_0{}^T(r=r_1,\theta)).F_1^*(I_1{}^T(r=r_1,\theta))}{\left|F_1\ (I_0{}^T(r=r_1,\theta))\right|.\left|F_1^*(I_1{}^T(r=r_1,\theta))\right|} + \ldots\ldots \qquad \text{(C.4)}$$

$$\left.\left. \ldots\ldots + \frac{F_1\ (I_0{}^T(r=r_{n_r},\theta)).F_1^*(I_1{}^T(r=r_{n_r},\theta))}{\left|F_1\ (I_0{}^T(r=r_{n_r},\theta))\right|.\left|F_1^*(I_1{}^T(r=r_{n_r},\theta))\right|} \right) \right)$$

Equation (C.4) gives the best rotation with a fix scale according to phase correlation methods.

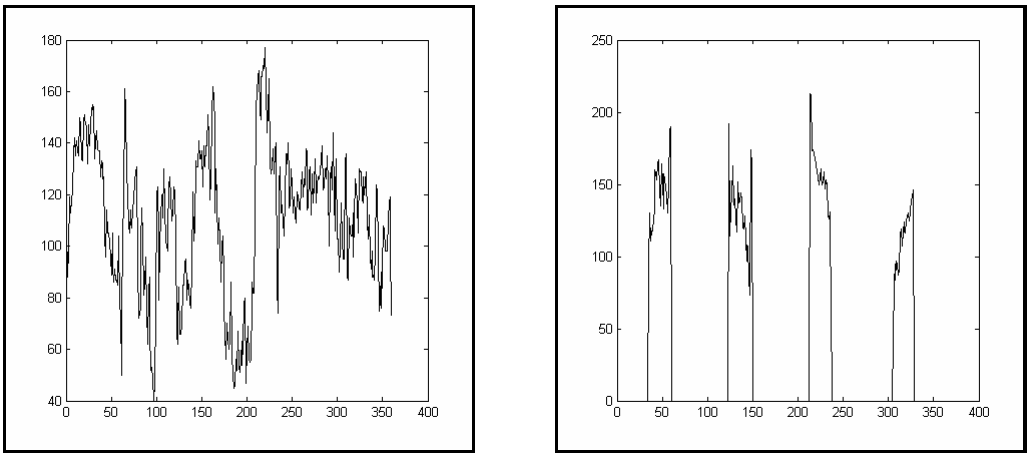


Figure C.1 : The 1D plotting of the image presented in Figure B.3 for r = 180 (left) and r = 300 (right). The horizontal and vertical axes show the angles and the gray scale intensity values, respectively. The second image has zeros because the r = 300 exceed the rectangle image in some regions.