AUTOMATIC REASSEMBLY OF FRAGMENTS FOR RESTORATION OF HERITAGE SITE STRUCTURES

A Project Report

Submitted in partial fulfillment of the requirements

for the award of the degree of

BACHELOR OF TECHNOLOGY

in

CIVIL ENGINEERING

&

MASTER OF TECHNOLOGY

in

BUILDING TECHNOLOGY AND CONSTRUCTION MANAGEMENT

by

SIVAPRIYA V

CE13B046

BUILDING TECHNOLOGY AND CONSTRUCTION MANAGEMENT DIVISION

DEPARTMENT OF CIVIL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY MADRAS

MAY 2018

CERTIFICATE

This is to certify that the thesis entitled **" AUTOMATIC REASSEMBLY OF FRAGMENTS FOR RESTORATION OF HERITAGE SITE STRUCTURES "** submitted by **Sivapriya V (CE13B046)** in partial fulfillment of the requirements for the award of the degree **Master of Technology** in **Building Technology and Construction Management**, is a bonafide record of work carried out by him in the Department of Civil Engineering at Indian Institute of Technology Madras, during the academic year 2017-2018.

The contents of this thesis, in full or parts have not been submitted to any other institute or University for the award of any degree or diploma.

Dr. Koshy Varghese
Professor & Project Guide
Department of Civil Engineering
IIT Madras, Chennai-600036

Dr. K. Ramamurthy
Professor & Head of Department
Department of Civil Engineering
IIT Madras, Chennai-600036

Place: Chennai
Date: 08 June 2018

ACKNOWLEDGEMENT

It is my pleasure to express my deepest gratitude to my guide and well-wisher Prof. Koshy Varghese who trusted in my abilities and presented me this challenging opportunity. His encouragement gave me the strength to face the innumerable challenges that came on the way. He is a true inspiration who despite his hectic schedule as a Dean, was always there to brainstorm and guide me. This journey has made me more observant, analytical and most importantly confident in my abilities. Prof. Koshy Varghese has played a major role in my all round development by being a constant support throughout.

Ms. Lalitha of Department of Computer Science of IIT Madras has played a crucial role in development of the project. She was kind enough to explain many concepts with utmost patience. A smooth transfer of knowledge was essential for me to get a grip on the problem. Her ability to explain each method mathematically and physically is an absolute inspiration.

I thank Prof. Arun Menon for his inputs on how the problem was useful in the big picture which has made me realise how much of an impactful work this is. I thank Ms. Madhumitha who has given me very valuable suggestions while writing my paper for ISARC 2018. I am extremely thankful to Mr. Balaganesan of workshop and Prof. Benny Raphael whose timely help with my porotype brought my project back on track.

The journey would not have been a memorable one without my friends Vethathirri, Niranjhana, Surya Sarada, Shashi Sekhar, Bhavitha, Keshav, Hema, Samhita, Bindu, Akhila. My Autodesk labmates Anusha, Vijayalakshmi, Ramesh and Aparna were a continuous support throughout. Special mention to my friends Rakishma, Akash and Shubham who were in the same boat as me and were by my side during tough times.

Finally, I would like to thank my parents, my brother, my grandfather and my family for believing and supporting me with their unconditional love throughout.

-Sivapriya

ABSTRACT

A major bottleneck activity in the process of restoration of Heritage Structures is the reassembly of its fragments. Finding their relative position is a very time consuming task and sometimes needs expert knowledge. Computer-aided reassembly could assist in the process thereby reducing time, manpower and potential degradation to fragile fragments.

Using shape compatibility between adjacent fragments along the fracture surface as the central idea, a reassembly framework for a two dimensional fracture case is proposed. This has applicability in reassembling 3D objects with uniform thickness. This is further logically extended to a three dimensional scenario to help reassemble broken pieces of a hollow object.

The framework consists of steps like extracting the contour, smoothening the edge, deriving features, and finding suitable sub-matching parts. Edges are extracted as polygons and relevant features are computed at each of its vertices. This makes the whole problem translation and rotation invariant. This is especially advantageous because that is how the fragments are found in the sites. Mutually exclusive and collectively exhaustive features are found for each vertex. Sequences of the match for two fragments in the feature space are found using a modified version of Smith-Waterman Algorithm.

Each match is assessed using a connectivity score which is a function of the common length. The final choice of best match is left to the user by displaying the resultant assembled fragments of prospective candidates along with the score. After pairwise matching, the global reassembly is done through an iterative algorithm.

This framework can handle fragments with curved edges which can be reasonably approximated by a set of edges. The methodology was applied in the reassembly of a simulated dataset: A cracked 2D dataset and a shattered 3D surface object dataset. The challenges associated with performing the experiment for a real dataset is also presented.

TABLE OF CONTENTS

LIST OF FIGURES

Figure No. **Title** **Page No.**

LIST OF TABLES

Table No. **Title** **Page No.**

CHAPTER 1
INTRODUCTION

1.1 ASSEMBLING THE PAST

A large number of fragments are discovered during archaeological excavations and heritage site restoration tasks. Many of them have the required information within them which can enable one to put back the pieces together so as to obtain the original structure. Once the fragments are mended together, it can tell a lot about the past- the objects, the structures, the way of life etc. The classical approach involves the following steps:

1. Field Excavation: In this step, fragments are excavated carefully and brushed
2. Classification of fragments: Fragments originating from different parent objects which have different characteristics are separated
3. Hypothesis about sequence: Grouping of similar fragments together and creating basic hypothesis about their arrangement
4. Cataloging: Examining few precise features of the fragments.
5. Creation of support structures: Especially needed when dealing with heavy fragments.
6. Physical matching: On a trial and error basis within the accuracy of human vision and knowledge, a matching process is performed.

Figure 1.1 a) A reassembled structure b) Manual reassembly c) Reassembly at site

1.2 NEED FOR AUTOMATION

Reassembly is one of the most time consuming yet crucial step in the entire process of restoration.

Manual methods

I. are time-consuming

II. demand construction of special supporting structures

III. are labor intensive for heavy fragments

IV. can potentially damage the fragile pieces during the process

V. might require expert knowledge

Figure 1.2 a) Expert involvement in reassembly b) Creation of special Support structure

With advancements in technological developments to digitize objects, computers can aid to a great extent in automating the process. Automating this bottleneck activity can greatly enhance the time effectiveness and ease of the restoration process.

With the input of fragment details (mesh), goal is to achieve computer aided reassembly of the fragments to get the final structure thereby relieving archaeologist of the tedious work and boost the efficiency of the process.

1.3 CONCEPT

The problem to be solved here is similar to the problem of Jigsaw puzzle - to find relation between different fragments using the available information in the fragments. There are a variety of approaches to solve the problem. The fragmented pieces themselves contain the clue for solving it. The clue varies from colour compatibility in the case of paintings, incisions on the surface for stones, marble veining directions, hand impressions on pottery etc. The choice is dependent on the available information and its uniqueness.

Sometimes the clue also comes from knowing the end result like in the case of Skulls where similarity matching could be performed with the standard template to reassemble. For highly eroded fragments, reassembly can be performed with the objective of maximizing its packaging efficiency. Some fragments have texture variation as a feature while some other have thickness. The key is its variation. A feature needs to vary for it to be considered crucial in the reassembly process. The variation makes sure that adjacent fragment and especially the matching ends have the same value of the feature e.g., colour, shape, texture, thickness etc.

1.4 SCOPE

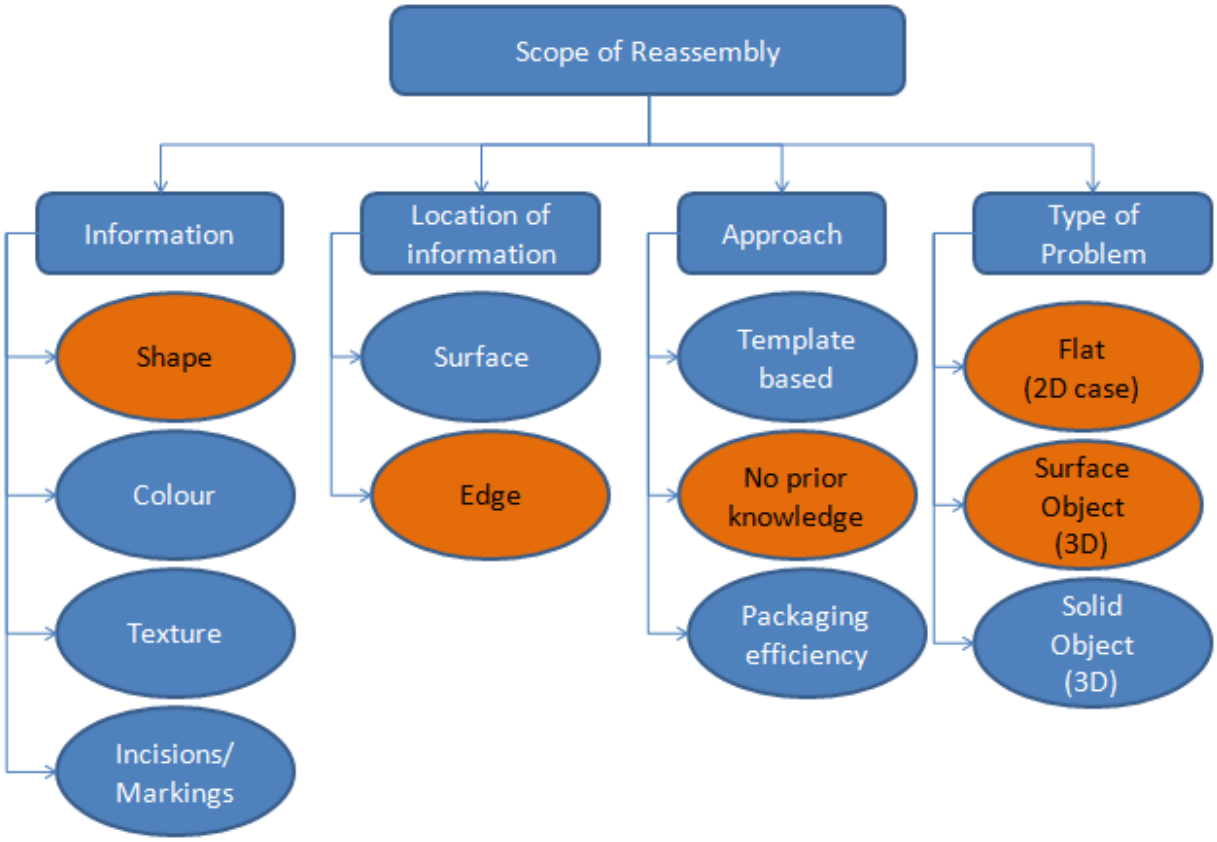


Figure 1.3 Flow Chart indicating scope of the project

Unlike paintings, unearthed and abandoned fragments of buildings do not have colour or colour variations. Neither do we have a clear idea of the end result of the structure after reassembly of the fragments.

The scope as shown in figure 1.3 is to assemble pieces which have retained their geometric information without making any assumptions about the reassembled piece. We follow a framework for assembling pieces in two dimensions making use of only its geometry by contour curve matching. This has applications in assembling flat fragments like the fresco, walls. We then, logically extend the framework to assemble a three-dimensional surface object. Applications include reassembly of surfaces like domes, thin shell structures etc. The aim is to help archaeologists reassemble interactively with user involvement only in the approval stage.

1.3 ORGANISATION OF THE THESIS

The thesis is organized into 7 chapters. A brief explanation of various chapters in this thesis is given below.

Chapter 2 discusses the relevant works done in two dimensional and three dimensional reassembly and a variety of features that were taken to reassemble the fragments.

Chapter 3 proposes the methodology and implementation of a framework proposed for reassembly of two dimensional fragments. It is also illustrated with the results of a simulated dataset of a cracked pentagon.

Chapter 4 extends the proposed framework in Chapter 3 with suitable modifications for the third dimension. This methodology is exclusively applicable for surface object or those whose thickness is very uniform. The framework is verified with a simulated data of shattered sphere using the software Autodesk Maya

Chapter 5 analyses the problem in detail by varying its parameters like number of pieces, jaggedness and reports the changes observed while reassembling them.

Chapter 6 presents the problems associated with performing real life dataset reassembly and the various methods attempted in detail

Chapter 7 summarizes, concludes and presents the scope for future work.

CHAPTER 2
BACKGROUND AND RELEVANT WORKS

2.1 FLAT IMAGE REASSEMBLY

A definition presented in the online encyclopedia Britannica, where jigsaw puzzles are “any set of varied, irregularly shaped pieces that, when properly assembled, form a picture or map”. Automated puzzle solving has great applications in archaeology and forensics. For instance, disasters like the collapse of the historical archive of cologne where more than 18 shelve kilometers have been destroyed, needs this kind of an automated reassembly solution. This has a huge potential to save objects of historic and cultural value. A forensic application could be to save a manually torn document. Demaine et al. in 2004, discusses the problem and calls it NP complete. Another consideration is to see when we have fragments of a different puzzle mixed with one another. In that case it is preferred to first clustered to their respective puzzles by some other means.

2.1.1 Apictorial Reconstruction Methods

The most obvious choice is to use the information of shape. Geometric reassembly dates back to 1964 where Freeman et al. who defined jigsaw puzzles as an “arrangement of a set of given pieces into a single, well-fitting structure, with no gaps left between adjacent pieces”. They also defined characteristics like Orientation, Connectedness, Exterior boundary, Uniqueness and Radiality. They encode the boundary as chains and partially matches chainlets through features for a commercially produced jigsaw puzzle.

Another heuristic based approach was used by Burdea et al. in 1989 where the end frame of the puzzle was identified using straight edge and the middle ones filled subsequently which match with the assembled frame. Most of the time when reassembling archaeological artefacts, prior knowledge of the reassembled object does not exist.

At a later stage, in 2002 Gama Leitao et al. developed a multiscale matching method and tested on unglazed ceramic tile. Initially, a fast search on coarse scale is done followed by matching on finer scale is done by only computing for shortlisted candidates from the previous case. However, it is more suitable for granulated material and less for sharp edged object like glass.

Here we seek to reassemble apictoral fragments which provide us with no other clue than its edge contour. A similar problem was attempted by Lalitha et al. where the fragments were converted

to invariant features and compared in the feature space. Different works explore different features, the way matches are detected and their global reassembly methods.

2.1.2 Pictorial Reconstruction Methods

In this category, fragments with shape as well as other information like writings on the surface or texture is also considered. Another variant is square jigsaw puzzle where every fragment is square in shape and only information other than shape should be used. For instance, reassembling a written torn document in forensics. One may argue that the torn sheet of paper is also a 3D puzzle since it tears in layers. If that is the case, boundary matching would not work.

The idea of a pictorial puzzle dates back to 1760 when John Spilsbury made a jigsaw puzzle out of a wooden map by cutting borders of countries using jigsaw to create an educational tool for children to learn geography. It is generally regarded as the oldest jigsaw puzzle following which puzzles started coming out in cardboard and later became a die-cut process.

Chung et al. in 1998 use shape as well as colours along the border of pieces. The jigsaw puzzle correctly assembles and they conclude that chromatic information leads to a significantly aid in solving the partial boundary matching problem. Nielsen et al use image features. In a rectangle puzzle, one pixel wide strip is extracted and compared with another fragments edge. Based on its similarity score, a match is decided. It has resulted in a very low error rate of 7.2% in a 320 piece puzzle. Hence square type jigsaw puzzles involve utilizing boundary pixel values as the parameter to search among other pieces since there is no variation of shape along the edges.

A generalized reassembly is different from the jigsaw puzzles because of the uncertainty in the endpoint, unlike jigsaw puzzles where the delimiting point is fixed. The combination of geometrical and colour matching is very suitable to handle fragments of paintings.

2.2 THREE DIMENSIONAL FRAGMENTS REASSEMBLY

2.2.1 Surface object

A common instance of a surface object would be a hollow object whose fracture does not result in a surface instead results in a curve. In practical cases, results in a curve of uniform thickness. Many an instances where specimen dependent techniques are developed. For example, for reassembling ceramic pottery, solutions are often based on broken surface, outline of shreds, or colors

and geometric characteristics like their axis of symmetry, corners of contour, visuals on the surface, concentric circular rills that are left during the base construction in the inner pottery side by the finger of the pottery artist etc. Some novel approaches like using information encapsulated in the inner part of the sherd i.e. thickness which are not heavily affected by the presence of environmental conditions have also been attempted.

Anna Grim et al. in 2016 present three dimensional apictorial jigsaw puzzle by dividing a curve into finite number of interlocking pieces. An example of the situation would be a broken ostrich shell. They devise a method for constructing synthetic three-dimensional puzzles by randomly distributing points on a compact surface with respect to surface area measure, then determining the induced Voronoi tessellation, and finally curving the Voronoi edges by using Bezier curves with selected control points. The edges of the puzzle pieces are divided into bivertex arcs, whose signatures are directly compared. It is tested with a real world ostrich egg data.

2.2.2 Solid object reassembly

In 2013, Gregorio et al proposed a constraint-based methodology for computer aided reassembly to create a flexible interactive system. The target scenario of the work was the one in which reassembly was based on the experience and understanding of the Cultural Heritage operator and not on the properties of the fragments. Once a particular constraint like adjacency or iso-planarity is given by the user, the system implements and presents reassembled structure. This way it is a user interactive semi automated system. While this particularly has an advantage when the fragments are eroded, it requires a knowledge of expert and does not exploit the shape information.

Devi et al. propose a framework for 3D reassembly by using local features like key edges for a large database of fragments. They extract different key edges based on occupancy within a small sphere at each point on the surface. They use a spectral based scoring method to finalise the match by removing all other false matches. It is an interactive reassembly technique.

Huang et al. in their work present a structured way for a fully automated reassembly using surface matching. They adopt a graph cuts based segmentation algorithm to identify the fracture surface and analyse it. They develop the integral invariants for computing multi scale surface characteristics and use them as features for performing a surface match. Their pipeline consists of four important steps namely Data segmentation, Feature selection, Pairwise matching and multi-piece matching.

A significant project in this domain was The Digital Forma Urbis Romae project by Koller et al. in 2006 where reassembly of marble pieces belonging to a giant map of Rome was attempted. The clue, in this case is contained in the incisions of the surface. Prediction of the pattern of the adjacent fragment based on the boundary drawings and searching for it through a number of pieces had resulted in significant improvements in its reassembly. This had a particular advantage because the erosion of fragments has occurred majorly along the thickness of the fragments which is not utilised for the reassembly. Additional constraints come from the nature of the material - marble veining direction which is another information that can be utilised.

CHAPTER 3

FRAMEWORK FOR SOLVING TWO DIMENSIONAL REASSEMBLY

3.1 OVERVIEW OF THE FRAMEWORK

Every reassembly problem is essentially three dimensional but some problems can be converted to two dimensions. Instances like flat fragments where the third dimension is uniform and does not exhibit appreciable variation belong to this category as shown in figure 3.1.

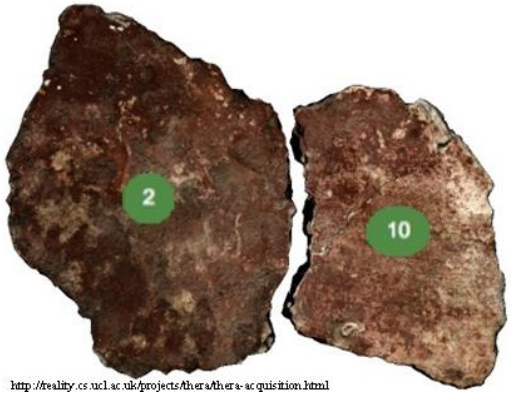


Figure 3.1 3D reassembly problem simplified into 2D problem

We create a framework for reassembling fragments generic enough to be implemented in either two or three dimensions. The framework as shown in figure 3.2 is created, applied for fragments in two dimensions and later extended by adding the third dimension which will be dealt in the next chapter. We assess the framework using a simulated dataset and real life dataset.

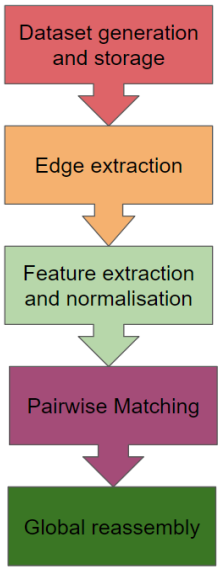


Figure 3.2 Overview of the framework

3.2 DATASET GENERATION AND STORAGE

The dataset is an image of cracked pentagon. It is stored and read in the .png format which is one of the most widely used formats for storing images.

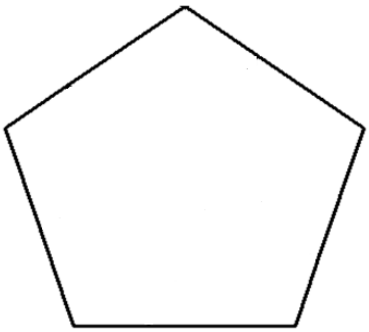


Figure 3.3 A simple polygon which is the parent object of the fragments to be created

Once a crack is generated, each fragment is extracted and stored as separate images. Further, it is masked as shown in Fig 3.4 which deletes all other information of colour and texture. This way, the image will carry only two colours (black and white) enabling easy extraction of contours. The proposed algorithm is translation and rotation invariant but is sensitive to scale changes and hence is important to maintain the scale throughout.

Figure 3.4 A masked form of the fragments stored

3.3 EXTRACTING THE CONTOUR

The only piece of information extracted from the fragment is its contour. The scope is to perform an apictoral reassembly.We read image from graphics file and find out the set of boundary points binding the image. This is accomplished using the 'imread' function of MATLAB.

We extract the boundaries in the form of polygon. It is necessary to reduce the polygon to a reasonable number of points i.e. smoothen it. We observe many consecutive 'almost' collinear points in the contour, which created a lot of edge jaggedness. This can be seen in Figure 3.5. We use Douglas-Peucker Algorithm to remove these points which do not contribute significantly to the matching process. This reduces the processing complexity significantly.

3.3.1 Douglas Peucker Algorithm

It is a line simplification algorithm which takes input of a polyline composed of several vertices, finds a subset of the vertices which form a similar polyline. Hence with lesser number of vertices we get a simplified line. This algorithm works for any number of dimensions. It takes a parameter called tolerance which decides the extent of simplification. Higher the value of tolerance, more simplified the line would be.

Input to the algorithm is a polyline of line of n points V_1 to $V_n \in$ A and the value for tolerance ε. The output is the line consisting of a subset of points$\in$ B.

Step 1: Select points V_1 and V_n into B.

Step 2: Find the farthest point from the line segment joining the selected point(s)

Step 3: If the point is at a distance lesser than ε, stop and display the output consisting of the selecting points. Else, that point becomes the splitting point of the originally straight line thereby increasing the number of line segments by one.

Repeat Step 2 and 3 iteratively for every line segment generated till there are no more points left over. Figure 3.4 demonstrates the step by step procedure of the algorithm with a polyline in two dimension but it can be applied to any m dimensional scenario with the only change in Step 2 where the distance would be calculated in the m dimensional plane.

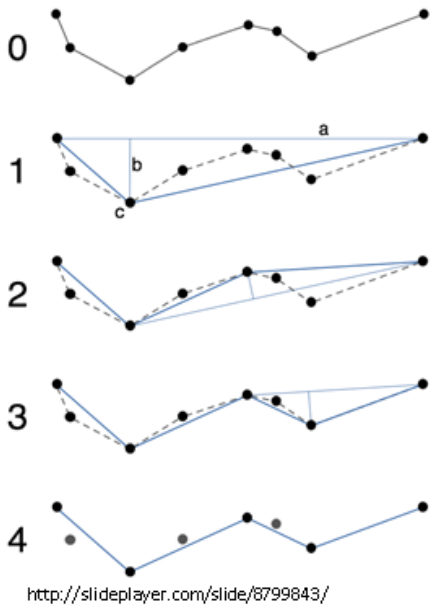


Figure 3.5 Demonstration of Douglas Peucker Algorithm stepwise

3.3.2 Applicability in a circular scenario

The original algorithm is meant for a polyline. The extracted points for each of the fragment is a polygon i.e. a closed structure. So, here the end point is same as the start point. Choice of a start point does not matter as long as it is the same as the end point. The rest of the algorithm remains the same except during the first iteration where Vnan V1 are the same and the starting line segment is actually a point. During the first iteration we get our first line segment. This is different from the polygon line simplification where the number of line segments is always one more than the number of iteration.

Fig 3.6 The original points on the contour of the fragment is shown in blue colour and the approximated line is shown in red

3.3.3 Choice of threshold

The choice of the threshold given as an input to the algorithm should be chosen wisely. Very small values can lead to retaining most of the points defeating its very purpose of reducing the computational complexity and removing the small variation caught probably because of noise. Higher values can alter the shape of the fragment itself making it less suitable for the further processing. Therefore along with an optimum choice in the value of the threshold, it is necessary to make sure every fragment that is believed to be a part of the same object is approximated with the same value of ε.

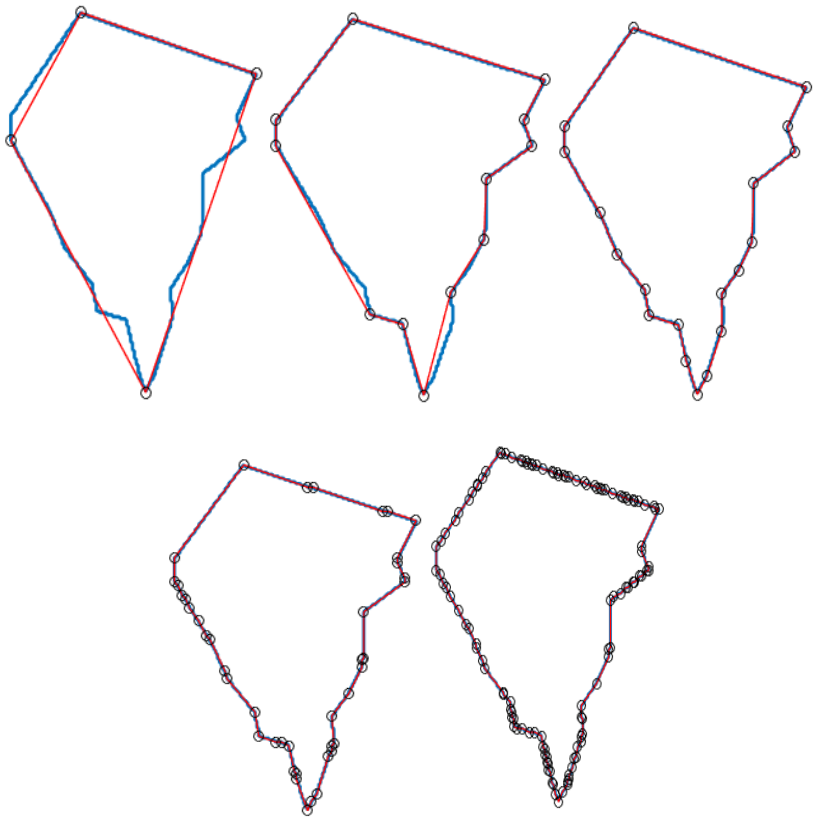


Figure 3.7 Output of DP Algorithm with thresholds a)10mm b)5mm c)1mm d)0.65mm e)0.5mm

For an initial fragment consisting of 452 points (shown in blue colour in Figure 3.6), the following are the number of points obtained when the threshold is varied. We see that as we decrease the tolerance, more number of points are retained. While too many points as shown in Fig 3.6e can be unnecessary and redundant, while Fig 3.6a does not capture the fragment details at all.

Number of points in the contour	Tolerance level
5	10
13	5
20	1
43	0.65
121	0.5
452	0

Table 3.1 Table showing the variation of number of points with the tolerance level.

3.4 FEATURE EXTRACTION AND NORMALIZATION

Each fragment may be translated and rotated at various angles. So, the information we have now is in the three dimensional coordinate system. The two matching sequence of points at different fragments cannot be found using this information. On changing to a coordinate system where the position of each point is dependent on its local features, a match can be easily performed. While now, the characteristic of each point is the three positional values it holds, in the feature space, its position would be determined by its local features. Also it becomes obvious that the corresponding points in two fragments will have complimentary local features.

With this reduced number of points in the contour which better capture the shape variation, we extract the following features at every vertex:

I. Log of mean of edge lengths (L)

II. Log of Absolute Value of signed curvature (C)

III. Internal angle (A)

Each vertex requires the information of its neighboring coordinates to calculate the features. For a point V_p (x_p,y_p), we have V_{p-1} (x_{p-1}, y_{p-1}) and V_{p+1} (x_{p+1}, y_{p+1}).

I. $$L_p = \log\left(\frac{\sqrt{(x_{p-1}-x_p)^2+(y_{p-1}-y_p)^2}+\sqrt{(x_{p+1}-x_p)^2+(y_{p+1}-y_p)^2}}{2} \right) \quad \text{.....(1)}$$

II. $$\text{Area} = 0.5 * det \begin{pmatrix} x_{p-1} & x_p & x_{p+1} \\ y_{p-1} & y_p & y_{p+1} \\ 1 & 1 & 1 \end{pmatrix} \quad \text{.....(2)}$$

$$C_p = \frac{4*Area}{\sqrt{(x_{p-1}-x_p)^2+(y_{p-1}-y_p)^2}+\sqrt{(x_{p+1}-x_p)^2+(y_{p+1}-y_p)^2}+\sqrt{(x_{p-1}-x_{p+1})^2+(y_{p-1}-y_{p+1})^2}} \quad \text{.....(3)}$$

III. $$A_p = \cos^{-1}\left(\frac{\sqrt{(x_{p-1}-x_p)^2+(y_{p-1}-y_p)^2}+\sqrt{(x_{p+1}-x_p)^2+(y_{p+1}-y_p)^2}-\sqrt{(x_{p-1}-x_{p+1})^2+(y_{p-1}-y_{p+1})^2}}{2*\sqrt{(x_{p-1}-x_p)^2+(y_{p-1}-y_p)^2}*\sqrt{(x_{p+1}-x_p)^2+(y_{p+1}-y_p)^2}} \right) \quad \text{.....(4)}$$

Based on the sign obtained for A_p, the angle is decided as A_p itself or 360 - A_p.

The features chosen should are mutually exclusive i.e. two features do not capture the same information and collectively exhaustive i.e. no information is left out and the point can be completely reconstructed using the features extracted.

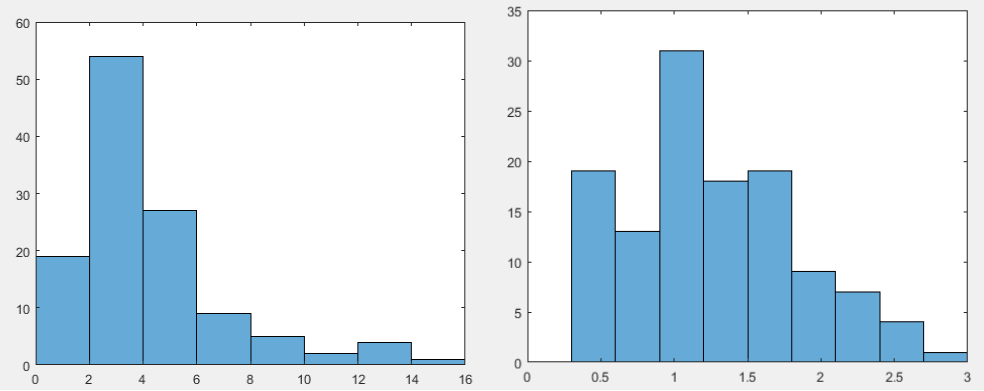


Fig 3.8(a) shows the skewness of the distribution of average lengths (b) shows a symmetric normal distribution once logarithm is taken

Same set of features were taken by Lalitha et al. and it has been replicated here after verifying it for the assumptions behind them. Figure 3.8 and 3.9 show how the distribution becomes more normal when logarithm is taken to the otherwise skewed distribution. This was done for the features average length of two sides and signed curvature.

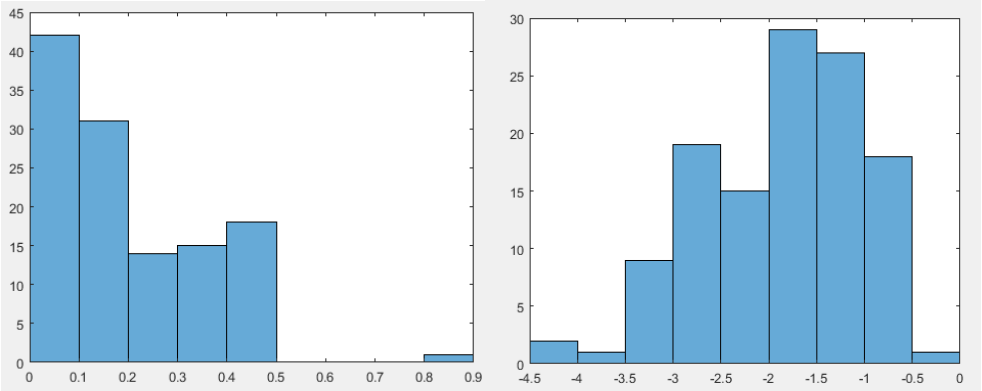


Fig 3.9 (a) shows the skewness of the distribution of signed curvature (b) shows a more symmetric normal distribution once logarithm is taken.

For fragment p, let F_c contain the features extracted from the clockwise and F_i anticlockwise. The perimeter of each fragment is also stored in PM.

Scale of each feature greatly influences the euclidean distance between points in the feature space. Therefore to maintain some uniformity, features are normalised using min-max normalisation. This is done by subtracting each feature from its minimum and then dividing it by its range.

3.5 PAIRWISE MATCHING

Now that we have transformed each vertices into a set of three values which capture its local characteristics, we can easily spot its position in the feature space. A feature space is the space created by having value of each feature plotted along mutually perpendicular axis. Logically we expect corresponding points of two different fragments to occupy the same space in the feature space because their local characteristics would be identical. But not only the corresponding points would occupy the same space, there could be other incorrect match points (could be same fragment or different) which occupy the same space just because of chance. We eliminate this possibility by choosing the longest string of continuous points in the cartesian coordinate system which occupy the same position in feature space.

3.5.1 Smith-Waterman Algorithm

The Smith-Waterman algorithm was originally conceived for finding similar substrings for matching nucleic acid and protein sequence. It is a dynamic programming algorithm which finds the optimal local alignment using the substitution matrix and the gap scoring scheme.

Let the two strings to be compared be $A = a_1a_2a_3a_4...a_n$ and $B = b_1b_2b_3...b_m$

Step 1: Create a scoring matrix Z of dimensions (m+1)*(n+1) with the column being named ‘0’ followed by each element of A and rows being named ‘0’ followed by each element of B.

	0	a1	a2	a3			an
0	0	0	0	0	0	0	0
b1	0						
b2	0						
b3	0						
	0						
	0						
	0						
	0						
	0						
bm	0						

Figure 3.10 Scoring matrix of the Smith-Waterman algorithm

Step 2: Compare a1 and b1. If they are the same, give them a positive similarity score s(a1, b1) else a score of 0. Let W be a term called penalty proportional to the gap. The score for each cell H(i,j) in the scoring table is filled based on choosing the maximum of

(i) $H(i-1,j-1) + s(a_i,b_j)$

(ii) $\max(H_{i-k,j} - W_k)$ for all $k>1$

(iii) $\max(H_{i,j-l} - W_l)$ for all $l>1$

(iv) 0

Each of the options indicate a direction i.e. (i)diagonal (ii)vertical and (iii) horizontal. It is necessary to keep track of that.

Step 3: Traceback - Once the entire matrix is filled, start with the cell having highest score and traceback along the route from which the score was obtained.

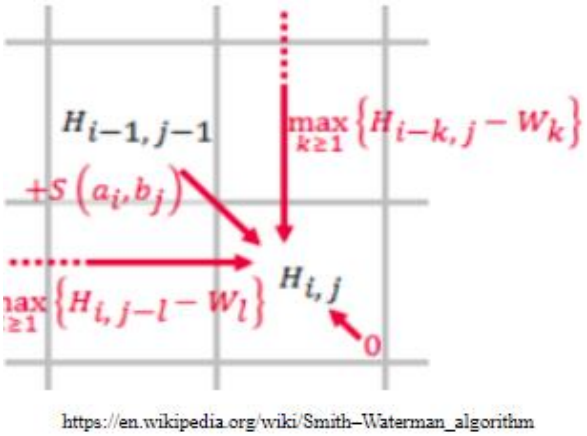


Figure 3.11 Scoring method of the Smith-Waterman algorithm

3.5.2 Suitability and Modifications to the algorithm

The algorithm is very similar to what we require - to compare a list of sequential points of one fragment with the sequential points of the other. To accomplish this, a modified version of Smith Waterman algorithm to match the two fragments i and j was used. A set of features extracted clockwise for one fragment is matched with the set of features extracted anticlockwise for the other fragment. Two vertices are said to be same if their Euclidean distance in the feature space is less than a particular threshold Φ. In ideal cases they would occupy the exact same spot and hence their distance between them is zero but we allow for some minor variation too. The similarity score is accordingly decided.

It is necessary to perform a cyclic matching in this case of a closed polygon because choosing a random point as the starting point with linear matching would break the longest string if the random point happens to be a part of the common length. Therefore the matching is cyclic i.e., we duplicate the sequence and append it to then end, and then match the sequence if found repetitions. The output of this is a list of matches between the fragments. The sequences greater than a minimum connectivity value C(i,j) is stored in G where connectivity is defined as in the equation 5.

$$C(i,j) = \frac{\text{Length of common boundary}}{\text{Minimum (PMi, PMj)}} \quad \text{.....(5)}$$

The concept of having a penalty for a gap in the similarity is especially advantageous because it handles sampling rate difference to a certain extent.

3.5.3 Experimental results

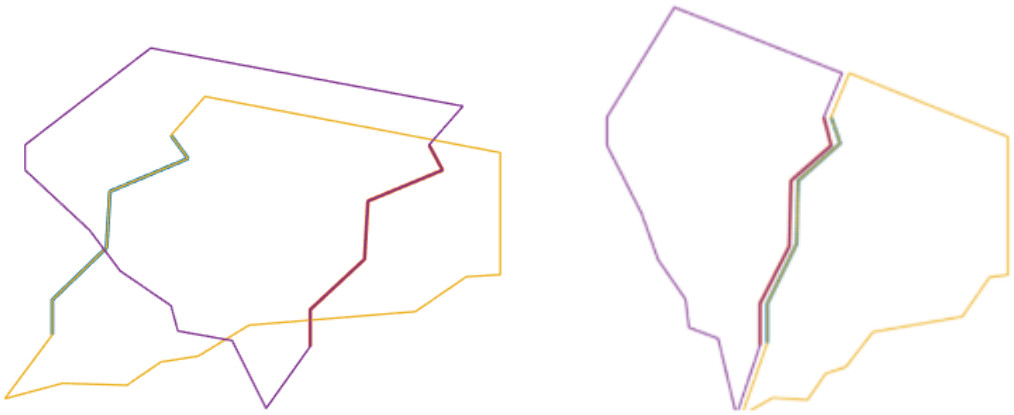


Figure 3.12 a) Output given by algorithm as the matching segments b) After their transformation

Fragment contours are fed into the algorithm as input and the sub-portion of it which matches each other is given as the output of the algorithm. We may get several substring matches. Of all of them, we display every match to the user and let the user decide the best one. The top few are displayed in decreasing order of their common length in the coordinate system. Once the common string is finalised, the transformation in the 2D space is estimated, checked for overlaps after transformation and displayed along with the connectivity value to the user to finalize the match.

3.6 GLOBAL REASSEMBLY

The order of assessment and reassembly of fragments is essential since wrong matches at initial stages potentially lead to error accumulation. Initially, connectivity score is found for all possible pairs of fragments (highest of all the matches for the pair of fragments is chosen). We use an agglomerative clustering algorithm. Pairwise matching is continuously performed for the pairs in decreasing order of their connectivity.

The scores are not calculated after each iteration significantly reducing the computational requirement. After the reassembly, the fragments merge and form a single piece. Now, if any fragment is left over, the connectivity is recalculated considering the assembled fragment as a single fragment because the newly formed fragment structure might pose a good match to the leftover fragments. For our test dataset, a single iteration is adequate to assemble all of the fragments when we follow the approach of following the decreasing connectivity order. The assembled structure is shown in Figure 3.13.

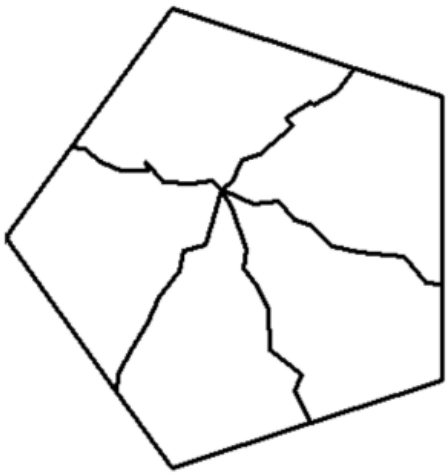


Figure 3.13 Reassembled fragments

Chapter 4

EXTENSION OF THE FRAMEWORK TO REASSEMBLE THREE DIMENSIONAL SURFACE FRAGMENTS

The framework is easily and directly extendible to a 3D scenario. The input model now contains information of fragments that occupy space in three dimensions. A simulated data of a surface object i.e. a hollow sphere is created. This is tested with the modifications made to the algorithm as discussed below.

4.1 CREATING THE DATASET AND STORAGE OPTIONS

A unit sphere was created and shattered into 10 pieces using the 3D computer graphics application Autodesk Maya using the following steps below.

4.1.1 Settings in Maya to shatter the fragments

The student version of Autodesk Maya was used. The procedure to shatter an object is

i) From the menu set, choose FX option.

ii) Choose the shape you want to scatter - here we choose sphere

iii) You can adjust the radius, number of latitudes and longitudes using the channel box/layer editor.

iv) Under edit option, make sure to choose the option delete by type and then delete history. Skipping this step produces error when we later shatter.

v) After selecting the sphere, freeze transformation available under modify section.

vi) Under the effects option, choose “Shatter” by clicking on the box

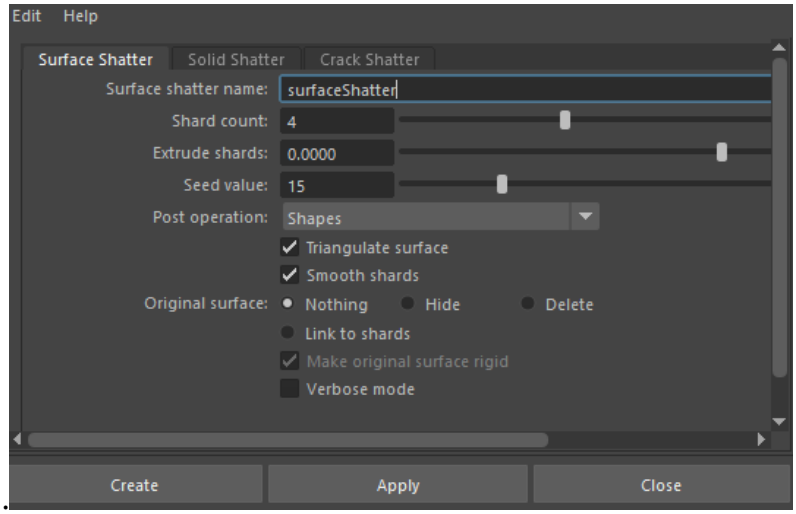


Figure 4.1 Autodesk Maya settings to shatter a surface or object

Surface shatter option is chosen because the framework applies for surface object. Additional attributes like the number of pieces to be shattered and their thickness are given in ‘Shard Count’ and ‘Extrude Shards’ respectively. Inward extrusion can also be done by giving negative numbers.Using the same number as seed value ensures that we get the same result every time we shatter. Once these parameters are decided, click on create to get the fragments in the GUI.

4.1.2 Storage Options

They were randomly translated along three axes and rotated at different angles. Each individual piece is stored separately and labeled for further processing.

There are different formats to store the surface information of the fragments. .obj format stores the surface as a set of triangles creating a mesh. The surface shape information is stored as a set of vertices with its coordinates and connectivity among the vertices indicated as the edges. For this particular application it is necessary to use .obj and not a similar storage format like .stl because of the difference in the way information is stored. In .stl format, the edges which belong to two different triangles are not repeated. We use the repetition concept to extract the edges later on.

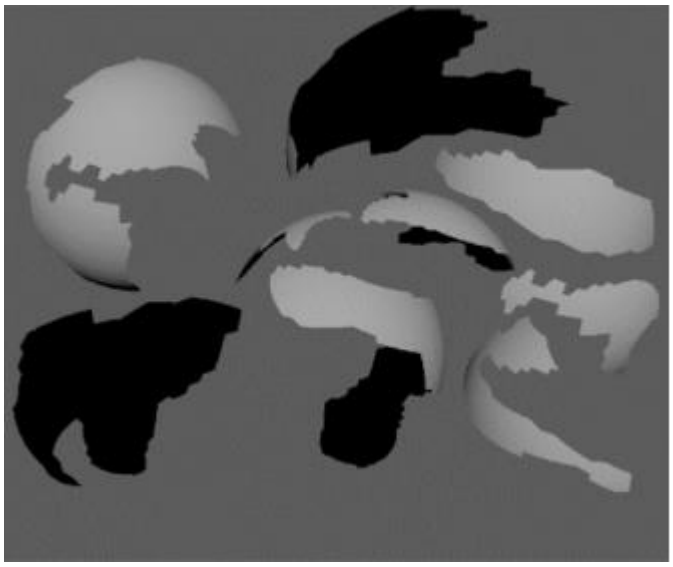


Figure 4.2 Scattered fragments produced using MAYA

4.2 EXTRACTING THE CONTOUR

The created fragments were stored as object files (.obj). It stores the mesh representation of the object as a set vertices with their coordinates, set of edges making up each triangle. Edges which are not common to two triangles indicate the end of the surface and are extracted. With the connectivity data, edges are aggregated to form a chain. Once a closed chain is obtained, that is declared the contour associated with that particular fragment.

Douglas Peucker algorithm is modified to be applied in three dimensions to smoothen the curve. As mentioned earlier, the threshold value decides the extent of smoothening and is necessary to maintain it constantly for all pieces. For our dataset threshold (th) was chosen as 0.005. It is necessary to maintain the same value of threshold of smoothening to all fragments. Since it is a simulated dataset, the contours are very accurate and only perfectly collinear points are removed. The number of vertices constituting the contour, drop drastically and hence makes the processing easier.

The case we take is a sphere and hence no other edges occur on the surface. However, if edges on the surface needs to be detected, edges having dihedral angle between the faces joining them greater than a particular value can be extracted to supplement the edge information to assess continuity. Also, a fragment can have two sets of boundaries like in the case where the cutting places are parallel. In that case both the fragments are used for matching.

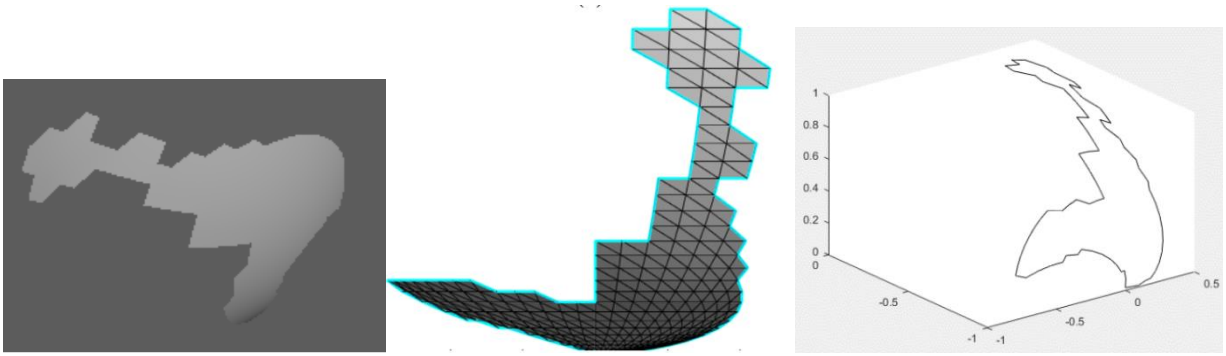


Figure 4.3 a) A scattered fragment b) highlighted contours in mesh format c) Boundary in 3D space

4.3 FEATURE EXTRACTION

Features extracted for a 3D polygon at its vertices are different from that of a 2D polygon owing to its non-planar nature. Every curve representing a fragment has a set of points P_1, P_2 … P_n. Each point has the location in the form of 3D coordinates associated with it. We use the numerical approximation for each of the feature defined for a smooth curve.

The features extracted here are:

1. Curvature: It measures the amount of deviation of a curve from a straight line. Mathematically, it can be obtained by calculating the reciprocal of its radius. The numerical approximation for curvature

$$\alpha_i = 4 \cdot area_i / (a_i \cdot b_i \cdot c_i) \quad \text{.....(6)}$$

where

$area_i = \sqrt{(s_i \cdot (s_i - a_i) \cdot (s_i - b_i) \cdot (s_i - c_i))}$

a_i = Distance between P_i and P_{i-1}

b_i = Distance between P_i and P_{i+1}

c_i = Distance between P_{i-1} and P_{i+1}

$s_i = 0.5 \cdot (a_i + b_i + c_i)$

is

2. Derivative of Curvature: This measures the rate of change of curvature with respect to arc length. This can be numerically approximated by

$$\beta_i = \frac{3 \cdot (\alpha_{i+1} - \alpha_{i-1})}{2 \cdot a_i + 2 \cdot b_i + d_i + e_i} \quad \text{.....(7)}$$

where

d_i= Distance between P_{i-2} and P_{i-1}

e_i= Distance between P_{i+2} and P_{i+1}

3. Torsion: This is an indicator of the curve's non-planarity by measuring the speed at which the osculating plane rotates along the curve. A numerical approximation is given by

$$\gamma_i = 0.5 \cdot (\gamma'_i + \gamma''_i) \quad \text{.....(8)}$$

where

$\gamma'_i = (6 \cdot H'_i)/(\alpha_i \cdot e_i \cdot f_i \cdot g_i)$

f_i = Distance between P_{i+2} and P_i

g_i = Distance between P_{i+2} and P_{i-1}

$H'_i = 3 \cdot V'_i / area_i$

$$V'_i = \frac{\det(P_{i+2}-P_{i-1}, P_{i+2}-P_i, P_{i+2}-P_{i+1})}{6}$$

$\gamma''_i = ((6 \cdot H''_i))/((\alpha_i \cdot d_i \cdot h_i \cdot j_i))$

$\gamma''_i = (6 \cdot H''_i) / (\alpha_i \cdot d_i \cdot h_i \cdot j_i)$

h_i = Distance between P_{i-2} and P_i

j_i = Distance between P_{i-2} and P_{i+1}

$H''_i = 3 \cdot V''/area_i$

$$V''_i = \frac{\det(P_{i-2}-P_{i+1}, P_{i-2}-P_i, P_{i-2}-P_{i-1})}{6}$$

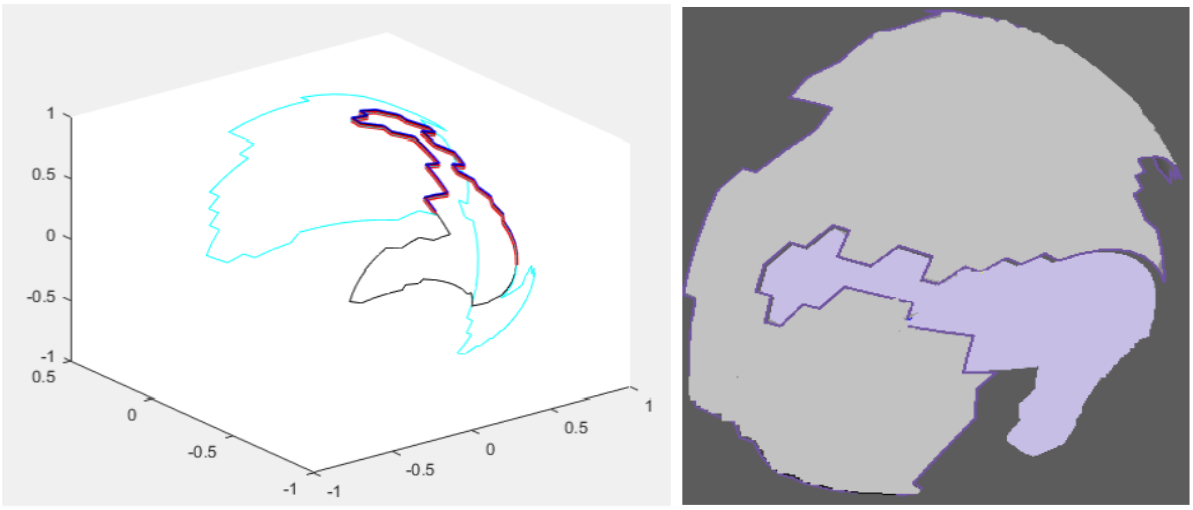
Here we take the average of γ' and γ'' to avoid introducing asymmetry in our calculations.

As Grim et al. in 2016 studies the features and states that only the above three metrics are enough to parameterize a Euclidean signature since they are the fundamental signature invariants. Feature extraction is done both clockwise and anticlockwise.

4.4 PAIRWISE MATCHING

We use the modified form of Smith Waterman algorithm like in the previous case except for the features matched. Two points A (α1,β1,γ1) and B (α2,β2,γ2) of two different fragments score a similarity score of 1 if the distance between them in the feature space is less than the threshold. For matching fragments i and j, features extracted clockwise for fragment i and counterclockwise for fragment j or vice versa is given as input to the algorithm.

A connectivity score as defined in (1) is evaluated for each pair and values below μ=0.1 are discarded. Once a good match (> 0.1) is found, the appropriate 3D transformation of one of the piece is estimated and made to merge with the second fragment and displayed (as shown in Figure X) for the user to approve the connection. Once approved, the connectivity score and the start and end vertices of the matching part is stored separately.



(a) (b)

Figure 4.4 Pairwise matching of Fragment 2 and 8. (a) shows Fragment 2 (cyan) and fragment 8 (black) with matching portions in blue and red colour respectively (b) shows the merged fragments

To compare the result with the ground truth, we compare their connectivity values which is as defined in the Equation 5.

Observed Connectivity values for pairs of fragments									
	1	2	3	4	5	6	7	8	9
1									
2	0.127								
3	0.129	0.138							
4	0	0.163	0						
5	0	0.25	0.209	0					
6	0.313	0	0	0	0.306				
7	0.142	0	0	0.26	0	0.336			
8	0	0.556	0.298	0	0	0	0		
9	0	0	0	0.472	0.253	0	0	0	

(a)

Actual Connectivity values for pairs of fragments								
1	2	3	4	5	6	7	8	9
0.161								
0.288	0.1865							
0	0.2498	0						
0	0.2917	0.2502	0					
0.369	0	0	0	0.377				
0.26	0	0	0.311	0	0.3995			
0	0.6161	0.3435	0	0	0	0		
0	0	0	0.524	0.318	0	0	0	

(b)

Percentage Error in connectivity values for pairs of fragmnets									
	1	2	3	4	5	6	7	8	9
1									
2	20.97								
3	55.13	26.01							
4	0	34.75	0						
5	0	14.3	16.47	0					
6	15.13	0	0	0	18.88				
7	45.41	0	0	16.37	0	15.89			
8	0	9.755	13.25	0	0	0	0		
9	0	0	0	9.992	20.44	0	0	0	

(c)

Table 4.1 Matrix showing the connectivity values of fragments (a) as estimated by the algorithm and (b) as calculated while creating the fragments. The rows and columns indicate fragment number. (c) shows the percentage error in connectivity value for each pair.

The maximum error in the connectivity value was 55% (for fragment 3 with fragment 1) and the minimum was 0% (for multiple fragments as shown in Table 4.1 c). This maximum value occurs when nearly half the original connection was undetected by the algorithm. The minimum value indicates a perfect fit.

The reason for the difference in connectivity values could be because of a variety of reasons including:

a) Different sampling rate at both fragments
b) The end points actually not having matching features because of their position

c) The algorithm takes a total of five neighboring points to calculate the set of features. This reduces a length towards the end of the common segment where the last three points will not have features similar though they are the corresponding points.

d) Less number of points or variation i.e., points located at large distances from each other and hence the leftover portions will be large.

4.5 GLOBAL REASSEMBLY

For each pair of fragments, the connectivity value calculated and approved by the user is recorded in the form of a matrix as shown below. The values less than 0.1 are reported as 0 as shown in Table 4.1. The connectivity values obtained for our dataset as shown in the figure indicate that each fragment has found a strong connection with atleast one other fragment implying if no overlaps detected, just one iteration is enough to assemble back all the nine fragments. Starting with the pair having the highest connectivity value, each pair is transformed and merged. At any point throughout the global reassembly process, the user can intervene and undo a merge if any sort of wrong overlaps is visually detected. A major advantage is that the connectivity value need not be calculated over and over again. Since there were no overlaps, in one iteration all of the pieces were assembled back to form the assembled object as shown in Figure 4.5.

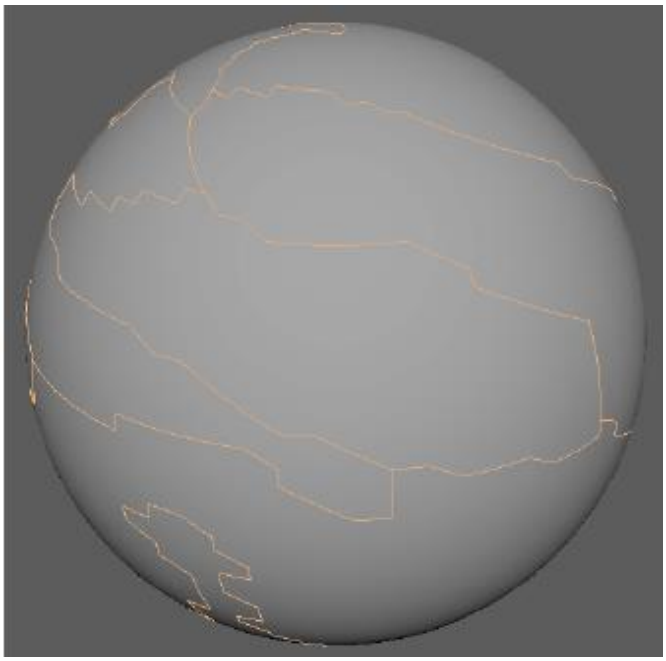


Figure 4.5 Assembled fragments

CHAPTER 5
EXPERIMENTATION WITH SIMULATED DATA

Now that the framework is verified for one particular case, it is needed to understand the algorithm's behavior by varying one parameter at a time and analyze the impact it has on the overall reassembly.

5.1 VARIATION OF JAGGEDNESS OF THE EDGES

As discussed earlier, the shape variation along the contour is a major input feature to our algorithm. Also non uniform shape variation is needed to get a unique solution. For example, if an object breaks into pieces whose fracture curves match in multiple ways, we will get infinite number of ways in which the puzzle is solved.

Figure 5.1 A solid object broken into half with an uniform fracture curve

Here we see if the algorithm is able to handle different levels of jaggedness. The experiment is designed as follows:

Method of Increasing jaggedness: Increasing the number of gridlines therefore giving it more choice of paths while breaking.

Number of levels: 5 levels where each of the level is a sphere of 50mm diameter

Number of fragments: Each sphere disintegrates into 4 fragments.

Details of each level:

Level No.	Number of horizontal & vertical line	Spacing between horizontal circles (mm)
1	20	2.5
2	40	1.25
3	60	0.83
4	100	0.5
5	150	0.33

Table 5.1 Design of each level

Level 1 Dataset

20x20 gridlines

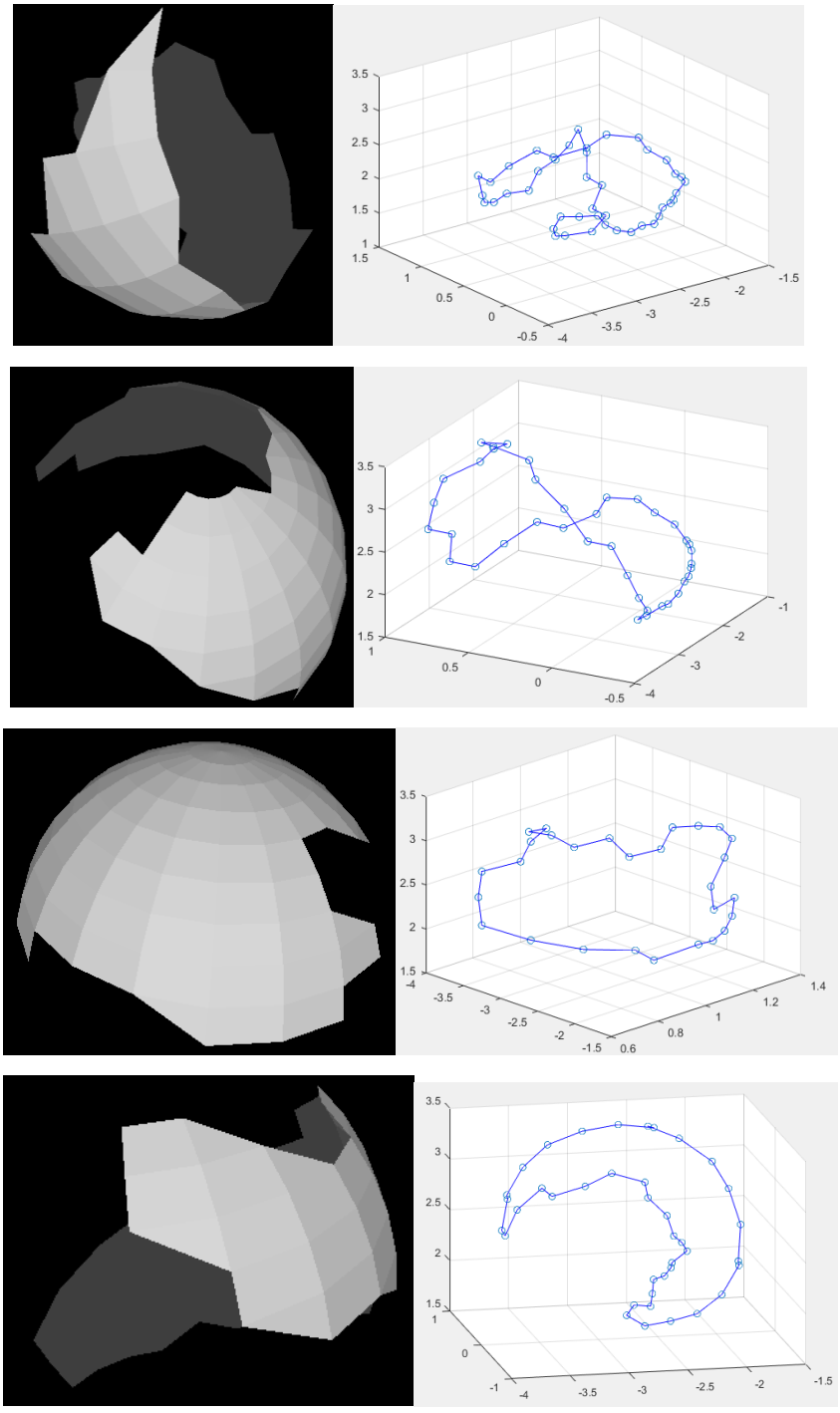
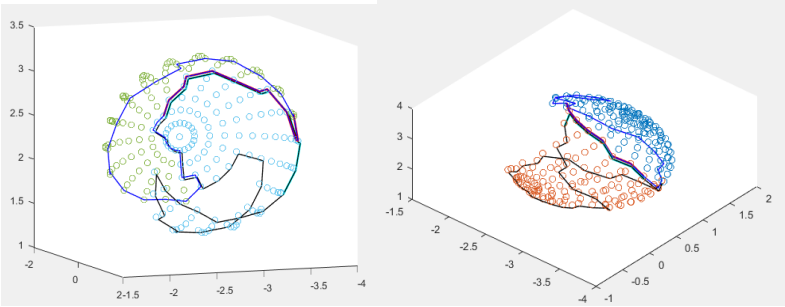
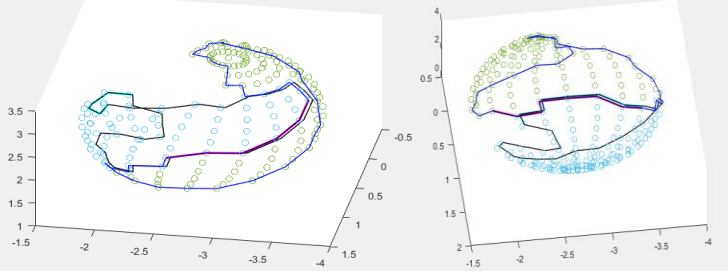


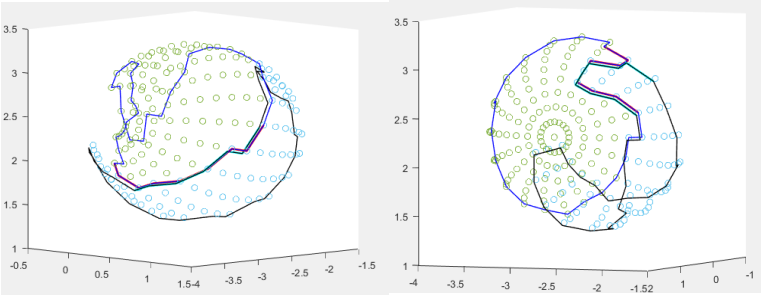
Figure 5.2 Fragments with their boundaries extracted for Level1



a) Fragment1 and 2 b) Fragment 1 and 3



c) Fragment 1 and 4 d) Fragment 2 and 3



e) Fragment 2 and 4 f) Fragment 3 and 4

Figure 5.3 Pairwise reassembly of Level 1 fragments of all possible combination with the output matching string given by the algorithm highlighted in red and blue colour

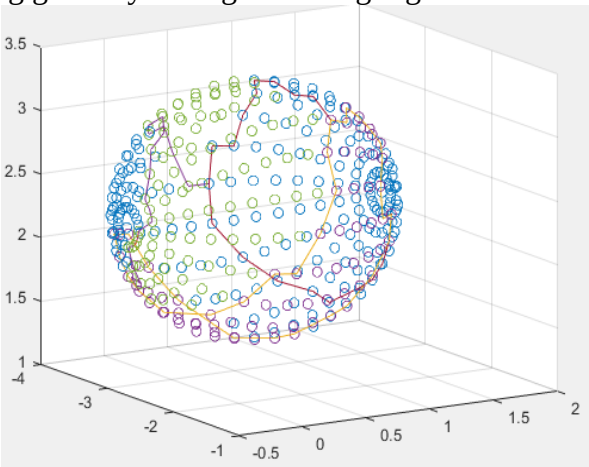


Figure 5.4 Reassembly of Level 1 fragments

Level 2 Dataset

40x40 gridlines

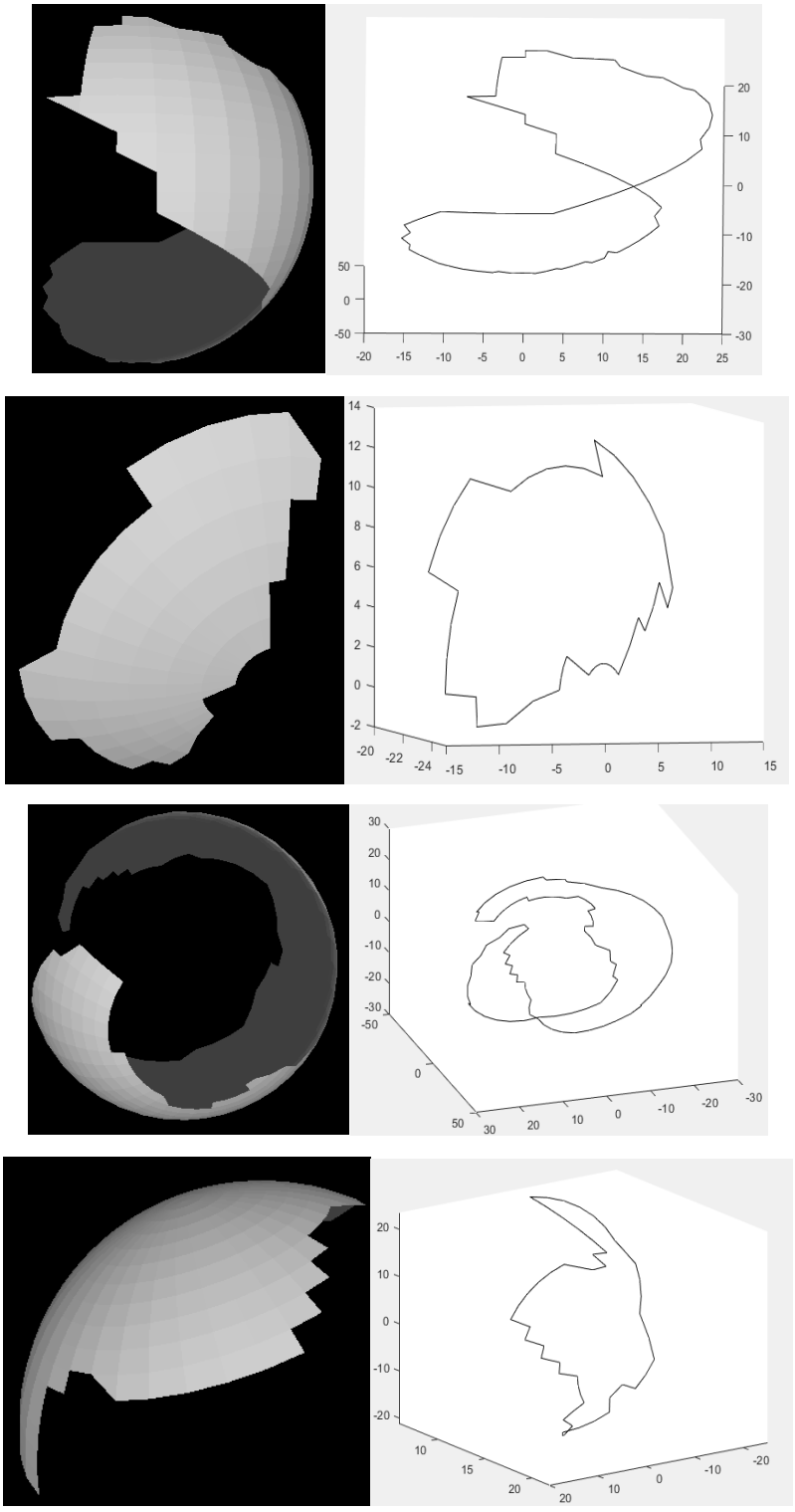
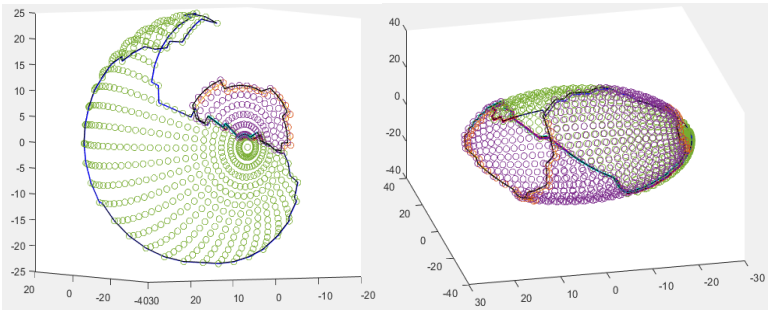
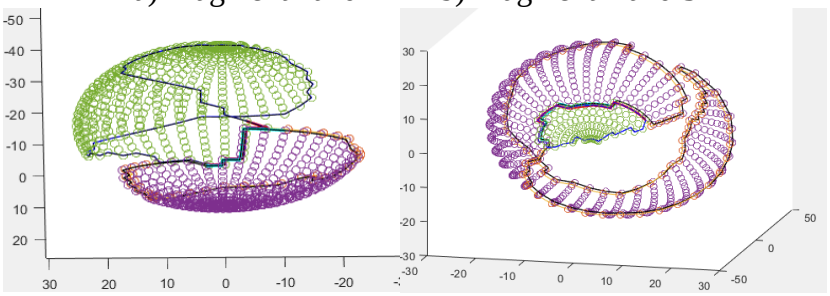


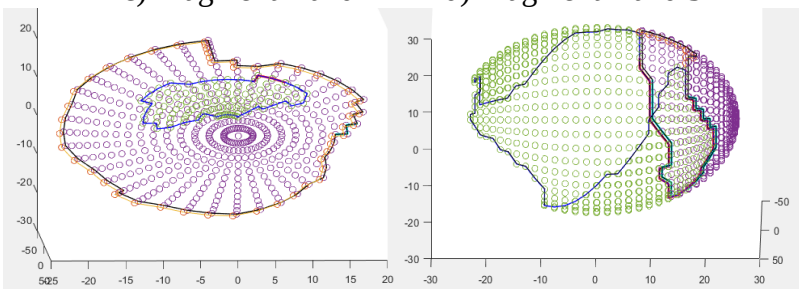
Figure 5.5 Fragments with their boundaries extracted for Level2



a) Fragment1 and 2 b) Fragment 1 and 3



c) Fragment 1 and 4 d) Fragment 2 and 3



e) Fragment 2 and 4 f) Fragment 3 and 4

Figure 5.6 Pairwise reassembly of Level 2 fragments of all possible combination with the output matching string given by the algorithm highlighted in red and blue colour

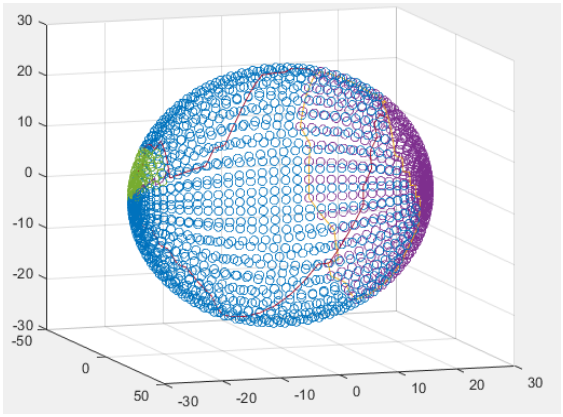


Figure 5.7 Reassembly of Level 2 fragments

Level 3 Dataset

60x60 gridlines

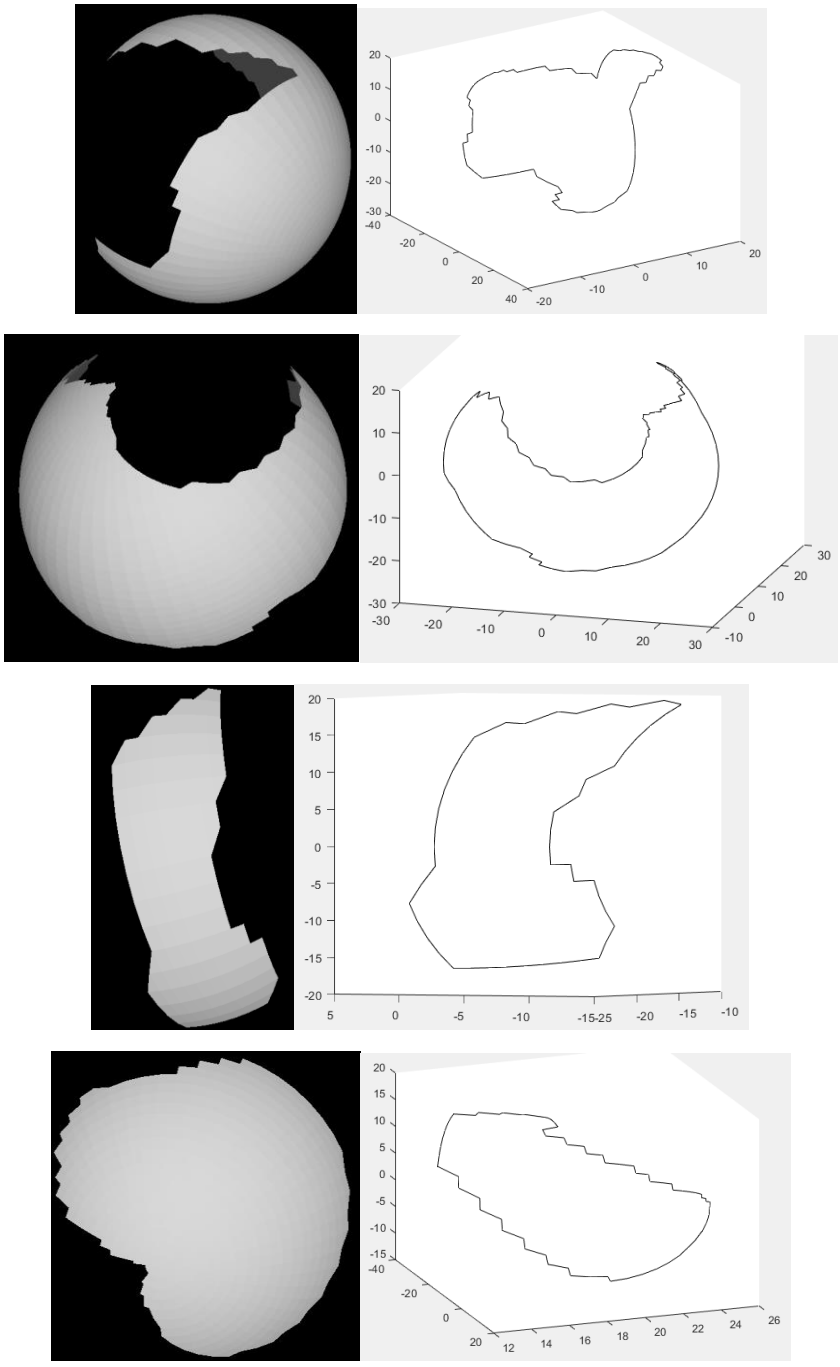
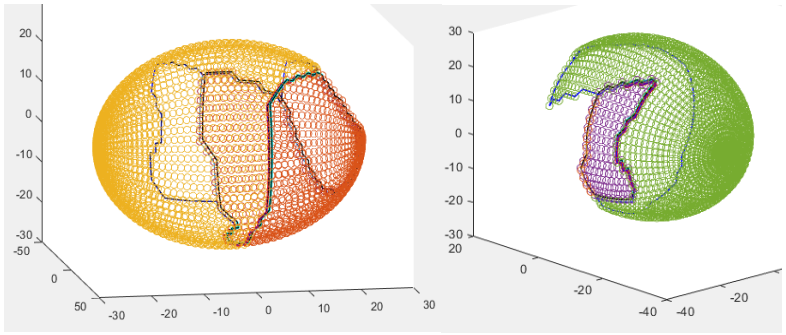
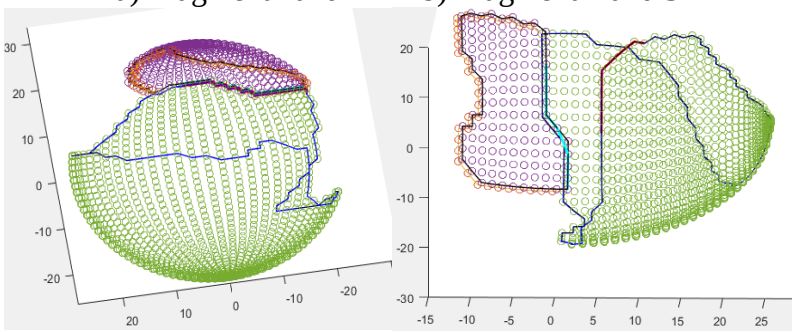


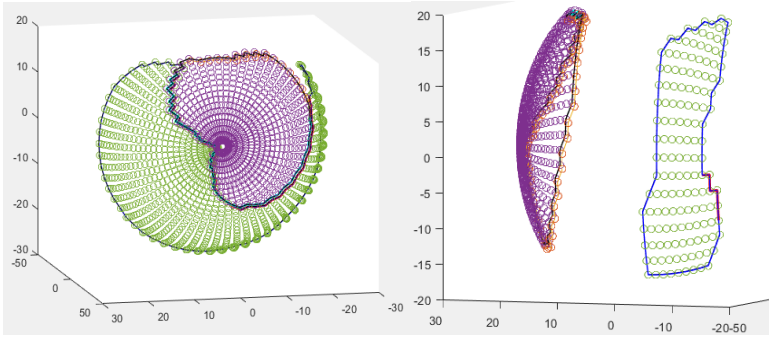
Figure 5.8 Fragments with their boundaries extracted for Level 3



a) Fragment1 and 2 b) Fragment 1 and 3



c) Fragment 1 and 4 d) Fragment 2 and 3



e) Fragment 2 and 4 f) Fragment 3 and 4

Figure 5.9 Pairwise reassembly of Level 3 fragments of all possible combination with the output matching string given by the algorithm highlighted in red and blue colour

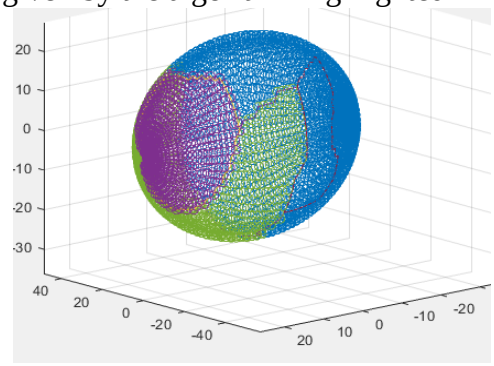


Figure 5.10 Reassembly of Level 3 fragments

Level 4 Dataset

100x100 gridlines

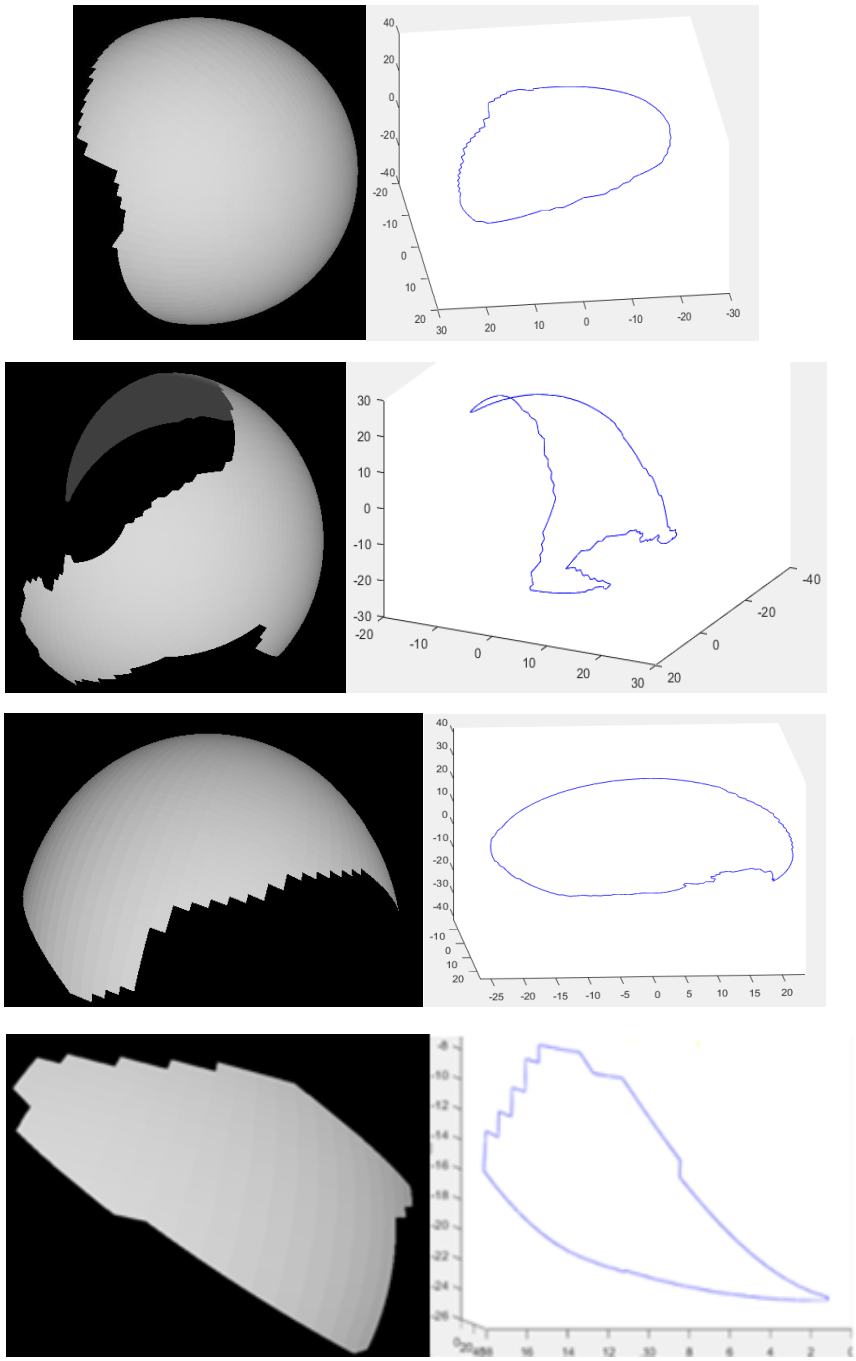
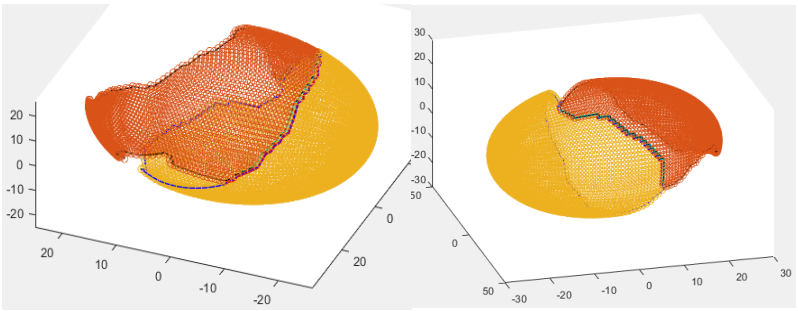
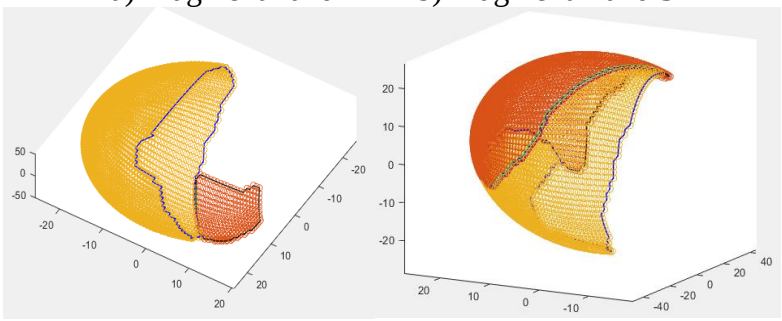


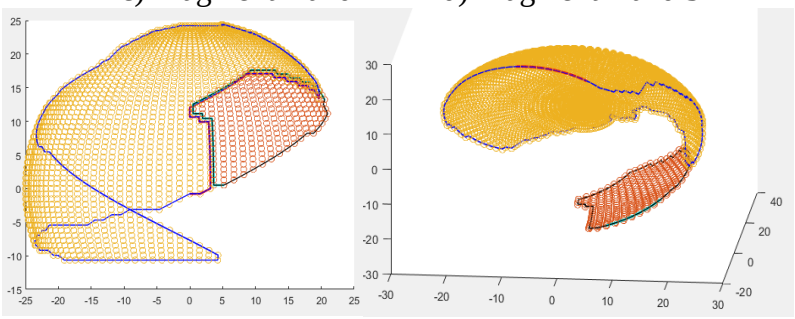
Figure 5.11 Fragments with their boundaries extracted for Level 4



a) Fragment1 and 2 b) Fragment 1 and 3



c) Fragment 1 and 4 d) Fragment 2 and 3



e) Fragment 2 and 4 f) Fragment 3 and 4

Figure 5.12 Pairwise reassembly of Level 4 fragments of all possible combination with the output matching string given by the algorithm highlighted in red and blue colour

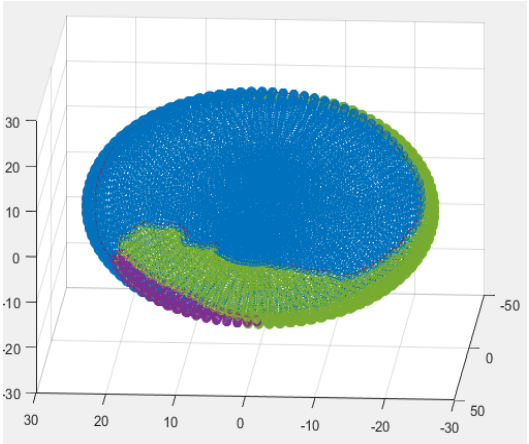


Figure 5.13 Reassembly of Level 4 fragments

Level 5 Dataset

150x150 gridlines

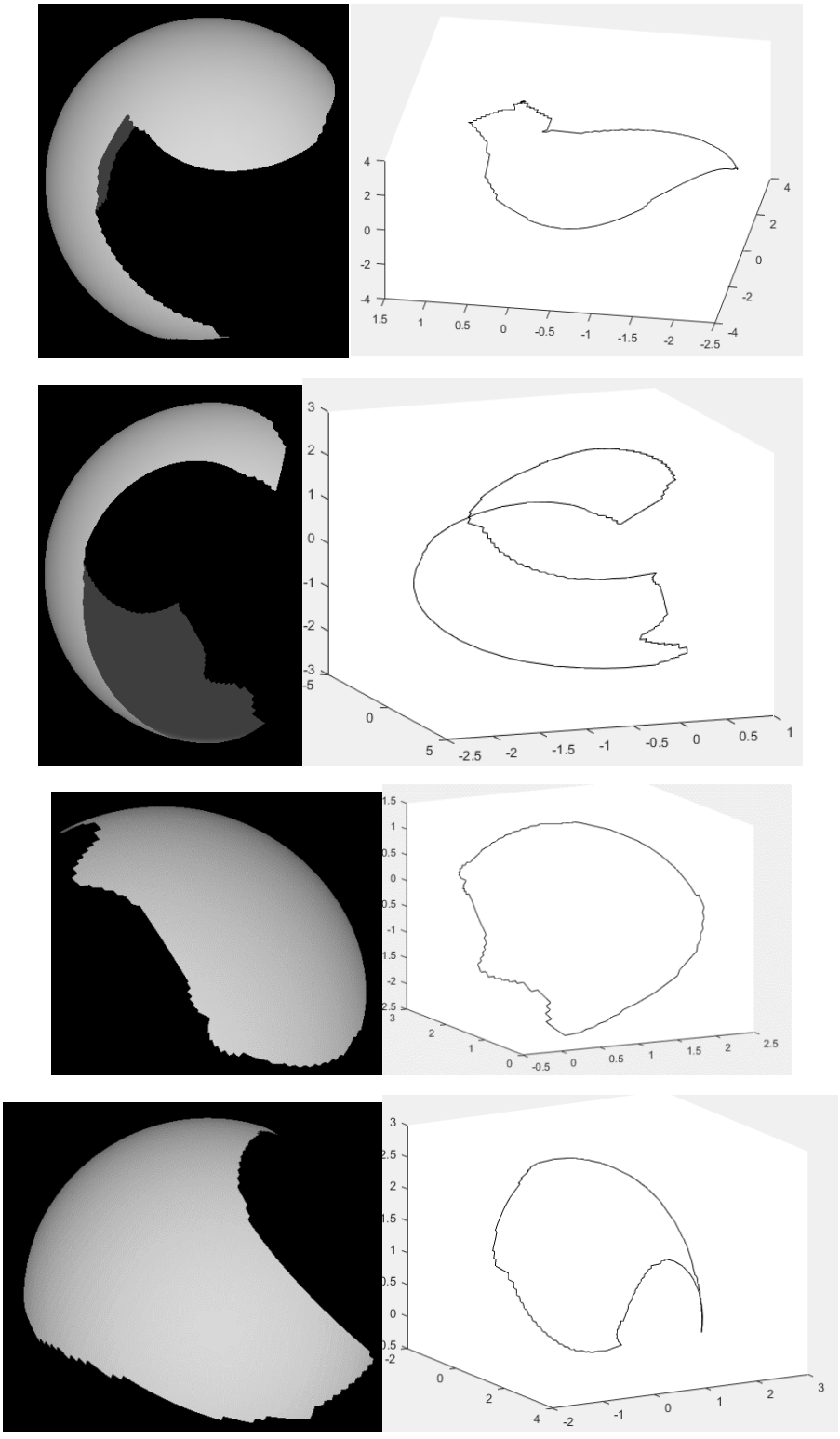
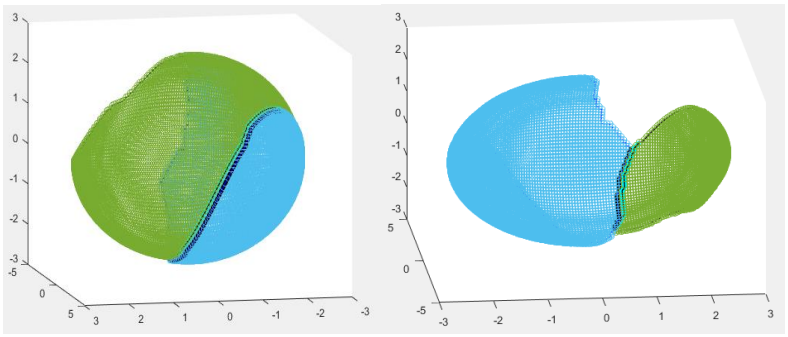
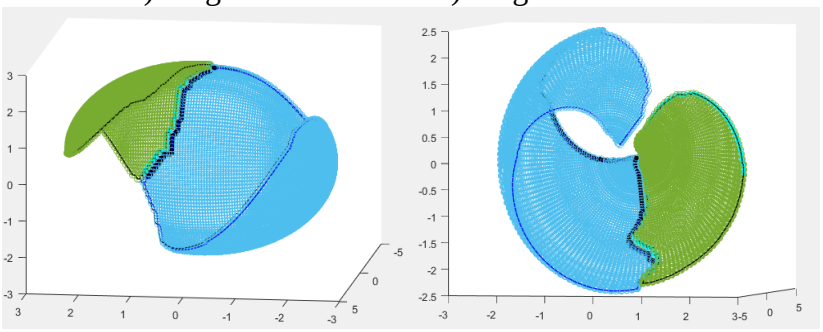


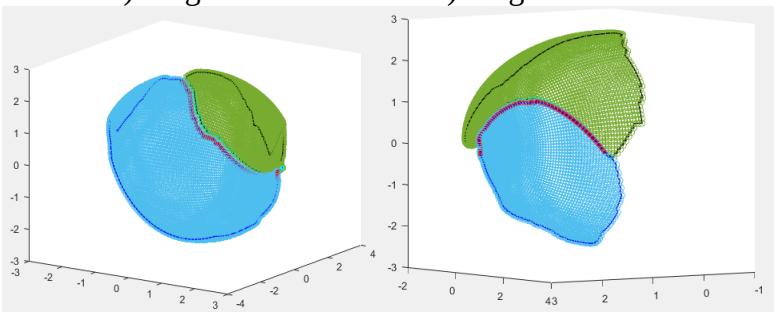
Figure 5.14 Fragments with their boundaries extracted for Level 5



a) Fragment1 and 2 b) Fragment 1 and 3



c) Fragment 1 and 4 d) Fragment 2 and 3



e) Fragment 2 and 4 f) Fragment 3 and 4

Figure 5.15 Pairwise reassembly of Level 5 fragments of all possible combination with the output matching string given by the algorithm highlighted in red and blue colour

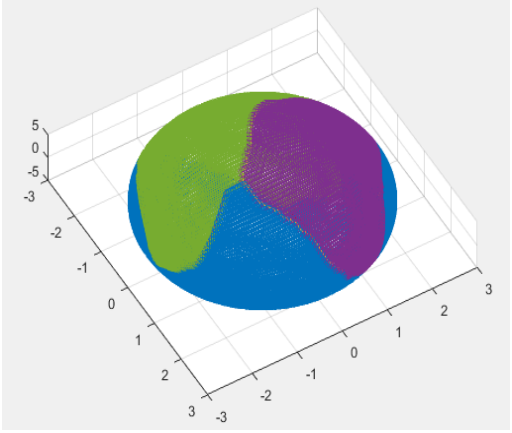


Figure 5.16 Reassembly of Level 5 fragments

We notice that the algorithm successfully reassembles all the levels of jaggedness and correctly identifies its pairwise match. However there are a few interesting observations:

a) Higher connectivity value as the level of complexity increases.

b) More number of matches as the complexity increases.

c) Extra length of matches with higher levels

d)All of them reassemble in one iteration

These all are because of the nature of the algorithm we use to match fragments. For matching one point, it takes five neighboring points in three dimensions and three in two dimensions. As the levels increase, more are the number of points and hence the length forming the few number of leftover points in the end decrease, increasing the connectivity value. More matches and extra length of the matches are explained by the regularity of the curve in lot many places.

5.2 EFFECT OF COMMON LENGTH

All the levels are reassembled in just the first iteration. Hence it is implied that number of iteration is not much sensitive to the jaggedness at the level in which the experiment was performed. Another factor which could possibly influence the number of iteration is the common length. So we specifically design a problem where just a small part is left out and see if that influences the number of iteration. It is compared with the case where two fragments form the original sphere.

Here we see that the base case where two fragments forming a sphere is reassembled, it finishes reassembly in one iteration as shown in the figure 5.17 whereas the second case with three fragments where the third has a smaller common boundary overall and that is again split with the two other fragments. We observe that in the first iteration, only the first two fragments designed to have a longer boundary match as shown in the figure 5.18 and the third fragment does not match with either of the two. After the first iteration when the two fragments have merged into one fragment, the boundary matches with the smaller fragment. This way, the reassembly is done in two iterations.

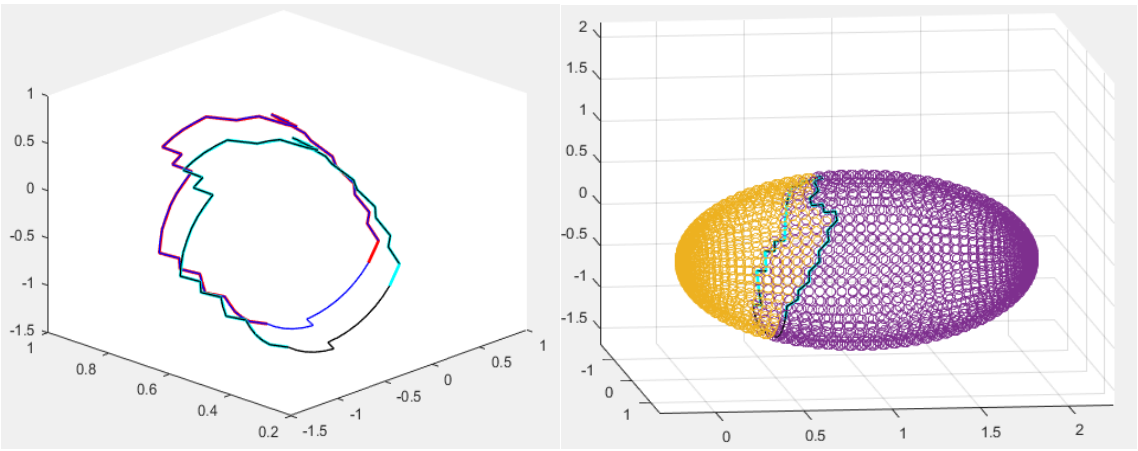


Figure 5.17 Two pieces assemble in one iteration in the base case

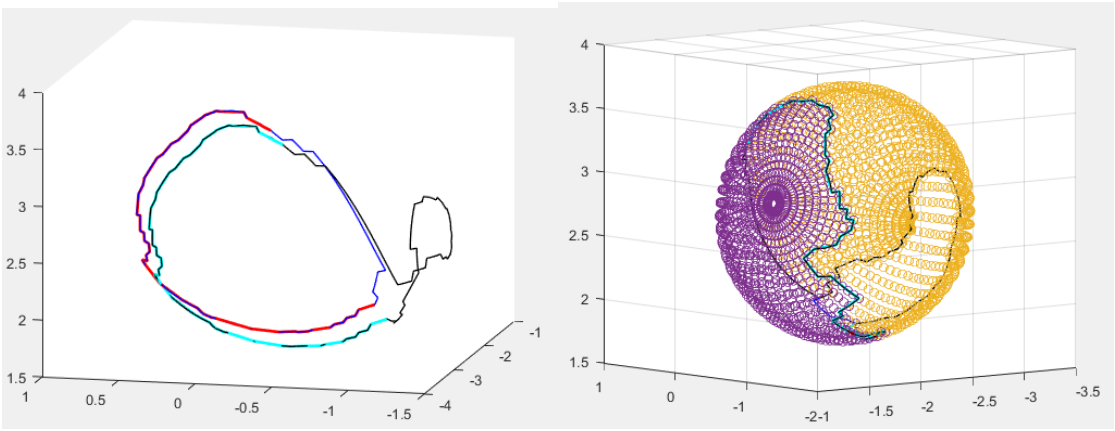


Figure 5.18 Three piece reassembly with the matched fragments in the first iteration

CHAPTER 6
REAL LIFE DATASETS: RESULTS & CHALLENGES

6.1 CHOICE OF MATERIAL

The choice of material is extremely crucial to perform this experiment.

6.1.1 Qualities expected

There are several requirements:

i) Material needs to be a hollow solid

ii) Needs to have very little thickness

iii) Should be robust after breaking

iv) We should have control on the way it breaks (for experiment purpose only)

v) Should not warp

vi) Edges should be steady i.e. not crumble

vi) Should break into sizeable pieces i.e. the edge pieces should not go missing

6.1.2 Study of the material- Utility and Challenges

A comprehensive study of materials tried and how well they are useful in the scenario and the challenge they present are discussed in this section.

1. EGG SHELL

It is a very representative case of the problem at hand. A major issue with this is that he edges crumble once broken. This is because once it is broken, the surface near the fracture edge lose strength as shown in figure 6.1. It has to be preserved carefully till scanning is done.

Figure 6.1 Egg shells crumbling near its edge

2. PLASTIC BALL

Plastic balls are easily available and they have minimum thickness making it a very good fit for the case. However once the air inside is released the fragments are free to bend, twist and warp making it change its shape in space. So, if the motion mentioned above can be arrested, it forms a good material to test the framework on. Here we have control over how the fracture edge shape has to vary also which is an added advantage for the experimental setup.

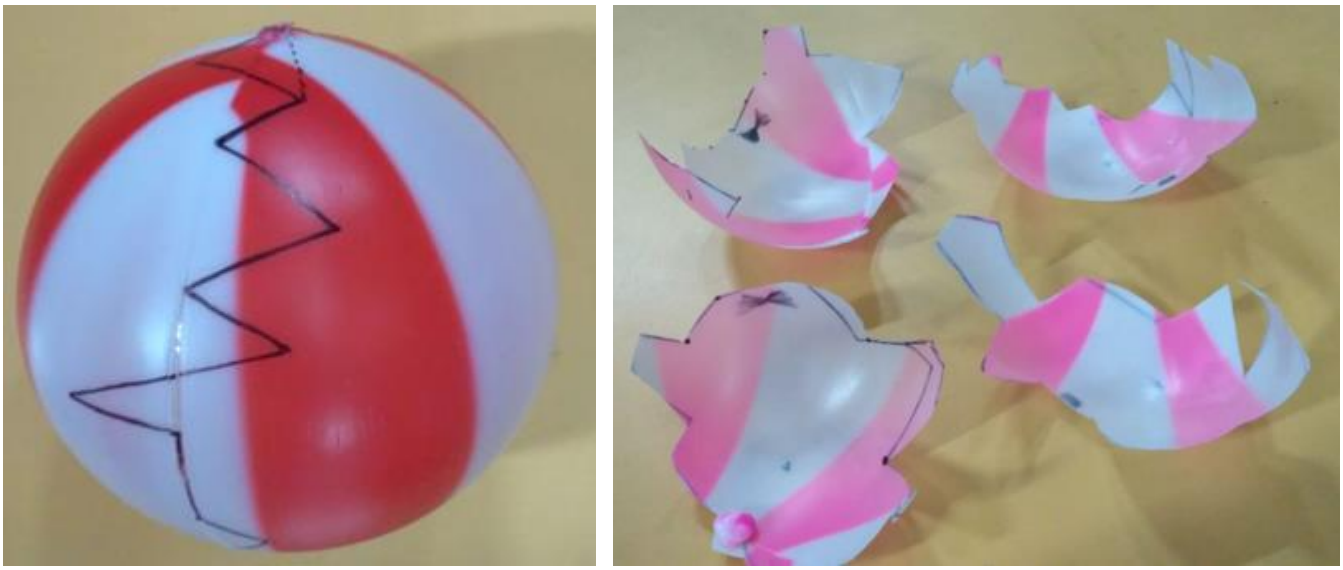


Figure 6.2 a) A plastic ball with markings before cutting b) Fragments after cutting them

To cut the plastic ball, an initial attempt to cut it with normal cutter was found to be extremely tedious, followed by trying to cut it after keeping the blade in the flame for some time. This was better than the previous. So for a blade to get continuous heat, it was attached to the end of a solder rod.

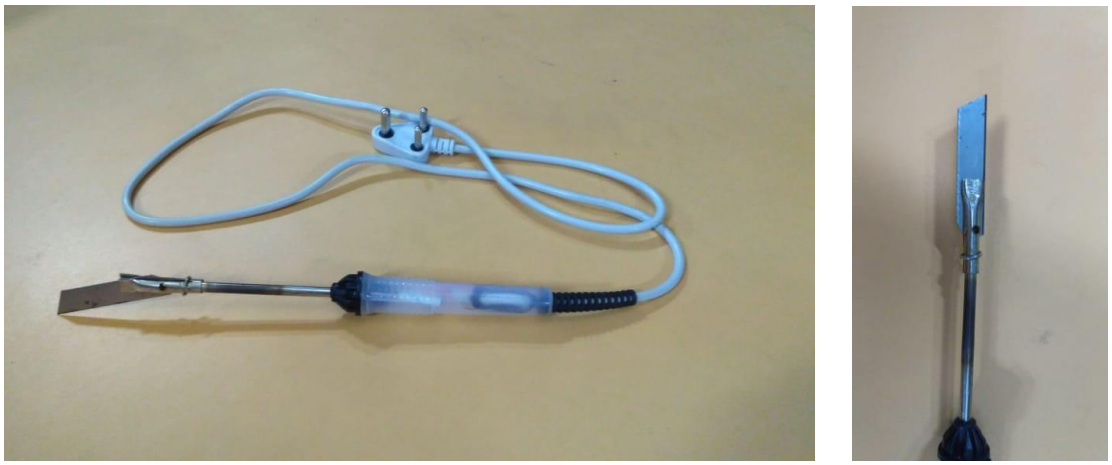


Figure 6.3 a) Solder rod with blade attached to the tip b) A closer view of the tip

While this made the cutting process easier, the results as will be discussed in the next section, had some problem because the material gets melted in the edges and could be one of the possible reasons for why the algorithm did not give good results.

3. PAPER MACHE

Preparation of a paper-mache

A bowl of gum is mixed with water and applied over a blown balloon. Stripes of paper are placed over it and left to dry. At least three such layers are made preferable the second and third layer with softer tissues and the first one with newspaper as shown in Figure 6.4 Leave it to harden for two days. Later burst the balloon and remove it to get the hard shell.

Figure 6.4 Steps to create a balloon paper mache

Figure 6.5 Paper mache after removing the balloon

We cut it to break it since this is not a brittle material. This material gives us flexibility in deciding the shape of the joining crack. When you cut it using scissor, the twisting and bending are inevitable especially while cutting at the vertices where it has to change direction. The conservation of shape of vertex is crucial to the matching because of the feature extraction process taking place at the vertices.

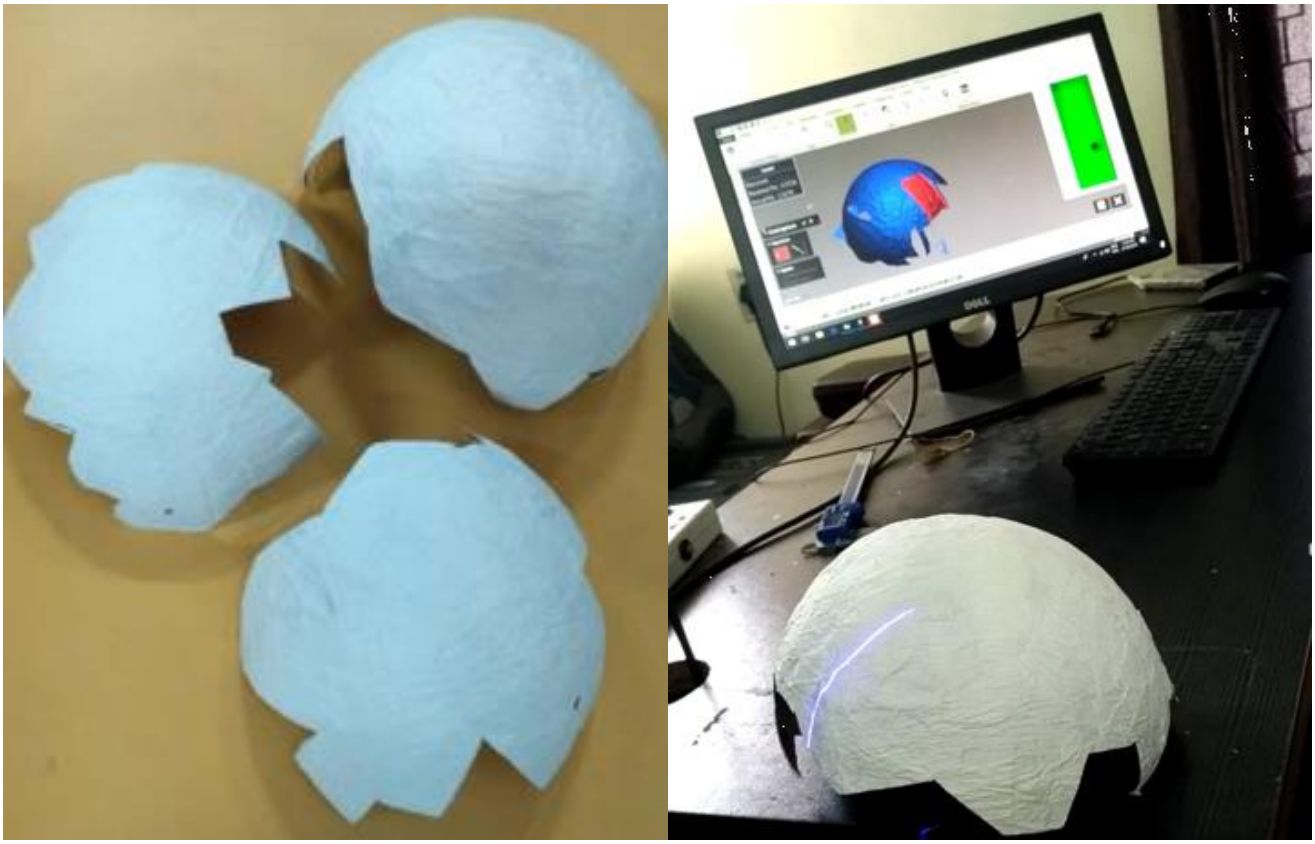


Figure 6.6 Scanning of Paper Mache fragments

4. PLASTER OF PARIS BALLS

This method is similar to the paper mache except that the balloon is painted externally or internally with a mix of Plaster of Paris (PoP) and water. This mix takes a while to harden on the surface. The advantage is that with this we create a realistic crack on a thin surface but the problem turns out that the mix of PoP has to be determined accurately maintaining balance between flow ability and strength and that it crumble when attempting to peel of the layer later. If done properly, it has enormous applicability to this particular scenario.

Caution: Make sure the balloon is thick enough to withstand the heat released during this exothermic process.

5. CERAMIC MUG

This is not spherical in shape but the concept at hand universally applies to every shape with a thin surface. These mugs have a small thickness which if gets scanned can create problems while feeding to the algorithm. They are the easiest dataset to create but will have to spend some time to get the results because of the trimming process.

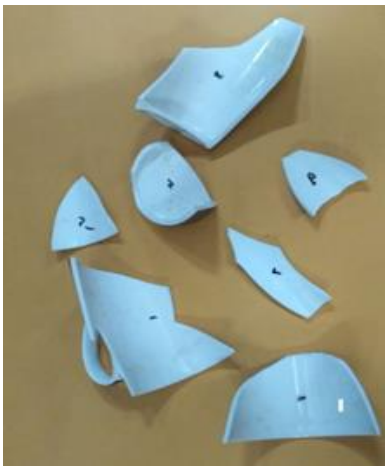


Figure 6.7 Broken fragments of a ceramic mug

On the whole, every material has its own challenge. The list of advantages and disadvantages has been identified. Once the problems associated with the material has been addressed, every material above is very suitable for 3D surface reassembly.

6.1.3 Creating the dataset

The central idea is to make sure that there is some variation along the boundary. We can accomplish either by cutting in a zig zag fashion in a plane of an object or choose objects which are naturally having a lot of bends and curves as shown in the figure 6.8 and cut it straight.

Figure 6.8 A hollow curvy surface toy

6.2 REASSEMBLING A PHYSICAL PROTOTYPE

6.2.1 Plastic balls

While scanning, if it is not done well, some part of the surface gets left out. In that case, it would appear as if there are several holes on the surface in addition to one big open curve indicating the end of surface. The algorithm treats that as end of surfaces too and extracts them. One way is to display to the user to choose which of the following contour the actual open boundary is and which one is a scanning error as shown in figure 6.9. Other way is to go with the assumption that the longest closed curve is the one we actually need.

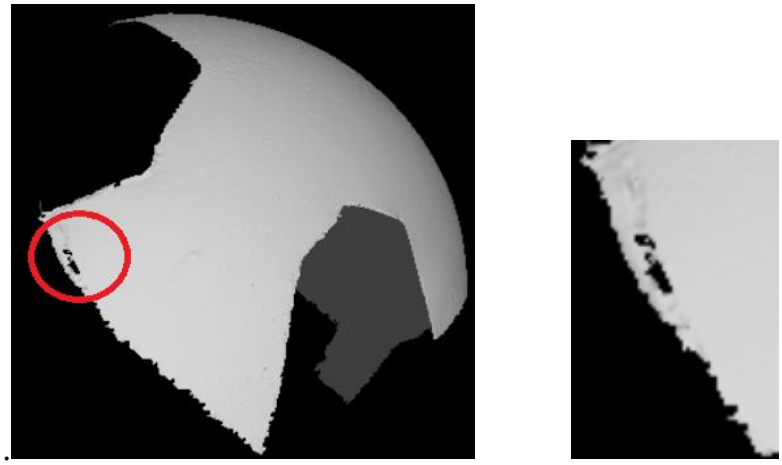


Figure 6.9 Laser scanning of plastic ball prototype highlighting the errors in scanning

While this a problem that could be easily overcome, the warping and bending was a serious challenge because it alters the shape of the contour in three dimensions that is the major input used for comparison. It indicates a lot of false matches of shorter segments owing to the regularity of the curve and plane.

6.2.2 Paper Mache

A major challenge with paper mache was the texture. It was very rough and the laser captures all of it making the size of the mesh very small and the overall size of the file large. The software used hangs after 45 minutes of reading the file. A faster computer or some way to smoothen out the surface would help in this case. The surface for now is not a serious concern here and hence smoothening would be a more economical choice.

6.2.3 3D Printed fragments and Mug fragments

The fragments which were used to study the variation of jaggedness were 3D printed using ABS material each with 1 mm thickness. The same 5 levels of 4 fragment each were taken for printing. These fragments reassemble to form a hollow sphere of diameter 5 cm.

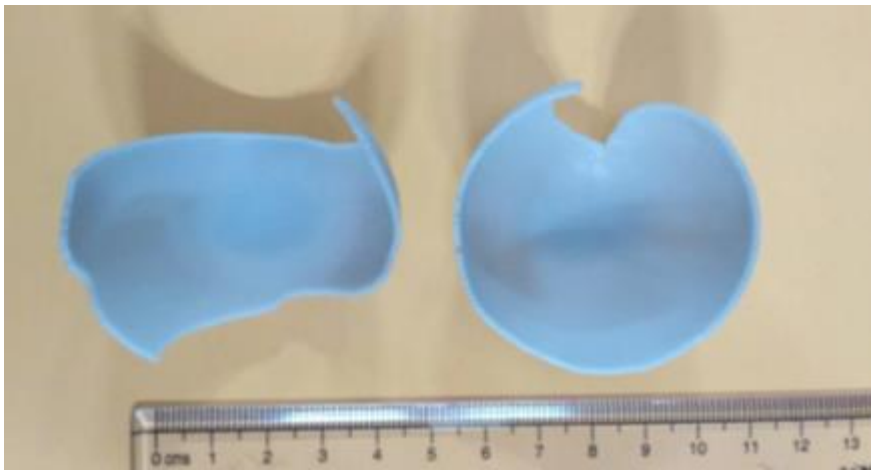


Figure 6.10 3D printed fragments

A major challenge with this is the non-uniform removal of thickness as a result, if this is fed, the entire contour is not what is expected to match with the corresponding fragment.

Figure 6.11 Non uniform removal of thickness

CHAPTER 7
SUMMARY AND FUTURE SCOPE

7.1 SUMMARY AND CONCLUSION

Initially a framework is developed for reassembling two dimensional fragments which consists of several steps including extraction of contours, smoothening them to an optimum level, converting them into points in the feature space, matching them using a modified version of Smith Waterman algorithm, finding the correct match based on its connectivity values and deciding the order in which they should be merged to form a single fragment.

This is verified using a simulated dataset of a cracked pentagon and is found to reassemble the fragments successfully in a single iteration. Later this framework is extended with suitable modifications to the features extracted making it more relevant to a three dimensional object. This was again verified with a simulated dataset of a sphere shattered into 10 fragments.

Further, an analysis was carried out on the algorithm and found that it is robust enough to handle various levels of complexity in terms of jaggedness. Errors during the reassembly and their possible cause are discussed. Relation between various factors like the number of iterations and common boundary length is proved using an experiment with simulated data.

The set of challenges that arise because of the choice of material and the scanning process is also discussed. A detailed analysis on choice of material considering various examples, creation of datasets and its challenges are discussed.

7.2 FUTURE SCOPE OF WORK

The following are the key issues which need to be considered while considering extending the work presented in the thesis:

i) The proposed method does not handle loss of material along the edges which could be issue in case of erosion. Taking account of erosion can help the reassembly process better.

ii) Other factors like colour and texture can be incorporated to make the process more robust.

iii) A predictive mechanism in case of a wrong pairwise match can be displayed instead of a simple connectivity measure. This will help in a better informed decision making than the present scenario.

iv) A better measure of error and compatibility metric can be proposed in place of the present metric "Connectivity Value" which takes into account only the common boundary length and the perimeter of fragments

v) Handling different rates of sampling of points along the edge for different fragments can be studied to make the match more error free.

REFERENCES

1. Willis, A. R., & Cooper, D. B. Computational reconstruction of ancient artifacts. IEEE Signal Processing Magazine, 25(4), 2008.
2. Pavlidis, G., Koutsoudis, A., Arnaoutoglou, F., Tsioukas, V., &Chamzas, C. Methods for 3D digitization of cultural heritage. Journal of cultural heritage, 8(1), 93-98, 2007.
3. Li, R., Luo, T., &Zha, H. 3D digitization and its applications in cultural heritage. Digital Heritage, 381-388, 2010.
4. Tsamoura, E., & Pitas, I. Automatic color based reassembly of fragmented images and paintings. IEEE Transactions on Image Processing, 19(3), 680-690, 2010.
5. Stamatopoulos, M. I., & Anagnostopoulos, C. N. 3D digital reassembling of archaeological ceramic pottery fragments based on their thickness profile. arXiv preprint arXiv:1601.05824, 2016.
6. Koller, D., Trimble, J., Najbjerg, T., Gelfand, N., & Levoy, M., Fragments of the city: Stanford's digital forma urbis romae project. In Proc. Third Williams Symposium on Classical Architecture, Journal of Roman Archaeology Suppl Vol. 61, 237-252, 2006.
7. Kampel, M., H. Mara . Robust 3D reconstruction of archaeological pottery based on concentric circular rills. In Proc. of the 6th International Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS 05), pages 14–20, 2005.
8. Yin, Z., Wei, L., Li, X., &Manhein, M. An automatic assembly and completion framework for fragmented skulls. In Int'l Conf. on Computer Vision, pages 2532–2539, 2011.
9. Freeman, H., & Garder, L. Apictorial jigsaw puzzles: The computer solution of a problem in pattern recognition. IEEE Transactions on Electronic Computers, (2), 118-127, 1964.
10. Alajlan, N. Solving square jigsaw puzzles using dynamic programming and the hungarian procedure. American Journal of Applied Sciences, 6(11), 2009.
11. McBride, J. C., & Kimia, B. B. Archaeological fragment reconstruction using curve-matching. In Computer Vision and Pattern Recognition Workshop, 2003. CVPRW 03. Conference on (Vol. 1, pp. 3-3). IEEE, 2003.
12. Lalitha, K. S., Das, S., Menon, A., & Varghese, K. Graph-Based Clustering for Apictorial Jigsaw Puzzles of Hand Shredded Content-less Pages. In International Conference on Intelligent Human Computer Interaction, pages 135-147. Springer, Cham, 2016.

13. Üçoluk, G., & Toroslu, I. H. Automatic reconstruction of broken 3-D surface objects. Computers & Graphics, 23(4): 573-582, 1999.
14. Palmas, G., Pietroni, N., Cignoni, P., & Scopigno, R. A computer-assisted constraint-based system for assembling fragmented objects. In Digital Heritage International Congress (Digital Heritage), pages 529-536, 2013.
15. Smith, T.F., Waterman, M.S.: Identification of Common Molecular Subsequences. Journal of Molecular Biology 147, 195–197, 1981.
16. Grim, A., O'Connor, T., Olver, P. J., Shakiban, C., Slechta, R., & Thompson, R. Automatic reassembly of three-dimensional jigsaw puzzles. International Journal of Image and Graphics, 16(02), 1650009, 2016.
17. Douglas, D. H., & Peucker, T. K. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. Cartographica: The International Journal for Geographic Information and Geovisualization, 10(2), 112-1221, 1973.
18. Boutin, M., Numerically invariant signature curves, Int. J. Computer Vision 40 pages 2
19. Britannica Encyclopedia, Jigsaw Puzzle (search term). http://www.britannica.com/, accessed January 2009.
20. B. Burdea and H. Wolfson. Solving jigsaw puzzles by a robot. Robotics and Automation, IEEE Transactions on, 5(6):752–764, Dec 1989.
21. M. G. Chung, M. Fleck, and D. Forsyth. Jigsaw puzzle solver using shape and color. Proc. of 4th Int. Conf. on Signal Processing (ICSP '98), 2:877–880, 1998.
22. A. Curry. Archive collapse disaster for historians. Spiegel online international, 04th march 2009. http://www.spiegel.de/international/ germany/0,1518,611311,00.html.
23. H. C. da Gama Leitao and J. Stolfi. A multiscale method ˜ for the reassembly of two-dimensional fragmented objects. IEEE Trans. Pattern Anal. Mach. Intell., 24(9):1239–1251, 2002.
24. H. C. da Gama Leitao and J. Stolfi. Measuring the information content of fracture lines. Int. J. Comput. Vision, 65(3):163–174, 2005.
25. E. D. Demaine and M. L. Demaine. Jigsaw puzzles, edge matching, and polyomino packing: Connections and complexity. Graphs and Combinatorics, 23(1):195–208, 2007.
26. R. Fuchs, C. Meinert, and J. Schrempf. Pergament: Geschichte, Material, Konservierung, Restaurierung. Anton Siegl, Fachhochschule Koln, 2001.

27. M. Kampel and R. Sablatnig. Profile based pottery reconstruction. In D. Martin, editor, Proc. of IEEE/CVPR Workshop on Applications of Computer Vision in Archaeology, pages CD–ROM, Madison, Wisconsin, USA, 2003.
28. M. Kampel and R. Sablatnig. On 3d mosaicing of rotationally symmetric ceramic fragments. In J. Kittler, M. Petrou, and M. Nixon, editors, Proc. of 17th International Conference on Pattern Recognition, Cambridge, UK, volume 2, pages 265–268. IEEE Computer Society, 2004.
29. F. Kleber, M. Lettner, M. Diem, M. Vill, R. Sablatnig, H. Miklas, and M. Gau. Multispectral Acquisition and Analysis of Ancient Documents. In M. Ioannides, A. Addison, A. Georgopoulos, and L. Kalisperis, editors, Proc. of the 14th Int. Conf. on Virtual Systems and MultiMedia (VSMM ’08), Project Papers, pages 184–191, Limassol, Cyprus, 2008. Archaeolingua.
30. Merriam Webster Online Dictionary, Puzzle. http: //www.merriam-webster.com/dictionary/ puzzle[2], accessed January 2009.
31. T. R. Nielsen, P. Drewsen, and K. Hansen. Solving jigsaw puzzles using image features. Pattern Recogn. Lett., 29(14):1924–1933, 2008.
32. C. Papaodysseus, T. Panagopoulos, M. Exarhos, C. Triantafillou, D. Fragoulis, and C. Doumas. Contour-shape based reconstruction of fragmented, 1600 bc wall paintings. Signal Processing, IEEE Trans. on, 50(6):1277–1288, Jun 2002.
33. M. Sagiroglu and A. Ercil. A texture based matching approach for automated assembly of puzzles. 18th Int. Conf. on Pattern Recognition (ICPR ’06), 3:1036–1041, 2006.
34. R. Tybon. Generating Solutions to the Jigsaw Puzzle Problem. PhD thesis, Griffith University, Australia, 2004.
35. A. Ukovich and G. Ramponi. Features for the reconstruction of shredded notebook paper. IEEE Int. Conf. on Image Processing (ICIP ’05), 3:III–93–6, Sept. 2005.
36. J.-M. Viglino and L. Guigues. Cadastre map assembling: a puzzle game resolution. Proc. of 6th Int. Conf. on Document Analysis and Recognition, 2001, pages 1235–1239, 2001.
37. J. Weberling and G. Spitzer, editors. Virtuelle Rekonstruktion “vorvernichteter” Stasi-Unterlagen. Technologische Machbarkeit und Finanzierbarkeit - Folgerungen für Wissenschaft, Kriminaltechnik und Publizistik, volume 21. Schriftenreihe des Berliner

Landesbeauftragten für die Unterlagen des Staatssicherheitsdienstes der ehemaligen DDR (German), second edition, Berlin, 2007.

38. F.-H. Yao and G.-F. Shao. A shape and image merging technique to solve jigsaw puzzles. Pattern Recogn. Lett., 24(12):1819–1835, 2003.

CONFERENCE PAPER BASED ON THE WORK

Sivapriya V, Madhumitha S, & Varghese K Automatic Reassembly of Fragments for Restoration of Heritage Site Structures, In International Symposium for Automation and Robotics in Construction 2018