2016

Effective 3D Geometric Matching for Data Restoration and Its Forensic Application

Kang Zhang
Louisiana State University and Agricultural and Mechanical College

EFFECTIVE 3D GEOMETRIC MATCHING FOR DATA RESTORATION AND ITS FORENSIC APPLICATION

A Dissertation

Submitted to the Graduate Faculty of the
Louisiana State University and
Agricultural and Mechanical College
in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

in

The Department of Electrical and Computer Engineering

by
Kang Zhang
B.S., University of Science and Technology of China, 2010
August 2016

To my parents

ACKNOWLEDGEMENTS

First of all, I would like to sincerely thank my advisor, Prof. Dr.Xin Li. It has been an honor to be his Ph.D. student. I appreciate all his support for my Ph.D. study. I am also thankful for the excellent example he has made as a successful researcher.

I would like to thank my committee members, Prof. Jerry Trahan, Prof. David Koppelman, Prof. Hongchao Zhang and Prof. Shelly Meng for their time, interest, patience and constructive advices on my dissertation.

I would like to thank Prof. Mary Manhein from LSU FACES lab and Prof. Warren Waggenspack from LSU Mechanical Engineering Department for their collaboration in the forensic skull and face reconstruction project; and thank Prof. Jolene Zheng and Prof. Steven Heymsfield in Pennington Biomedical Center for their collaboration in the body scanning project. I have learned a lot from these collaborations.

I would also like to thank members in the Geometric and Visual Computing Group at LSU: Shenghua Wan, Huanhuan Xu, Wuyi Yu, Shuai Zheng, Celong Liu, and Kang Xie for the delightful collaborations and discussions we had together. Many thanks to their willingness and patience to share the ideas to help me finish my dissertation.

Last but not the least, I would like to thank my family for their love and trust. Without their support and encouragement during the past years, I would never have been able to keep up my Ph.D. study.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS iii

LIST OF TABLES vii

LIST OF FIGURES viii

ABSTRACT xii

1 INTRODUCTION 1
- 1.1 Background 1
- 1.2 Motivation 2
- 1.3 Overview 3
  - 1.3.1 2D Image Reassembly 4
  - 1.3.2 3D Fragment Reassembly 5
  - 1.3.3 Data Completion 5
  - 1.3.4 Ancestry Analysis 6
  - 1.3.5 Facial Reconstruction 7

2 CURVE MATCHING FOR 2D IMAGE REASSEMBLY 8
- 2.1 Related Work 11
  - 2.1.1 Local Pairwise Matching 11
  - 2.1.2 Global Groupwise Reassembly 12
  - 2.1.3 Reassembly of 3D Geometries 13
- 2.2 Problem Formulation and Experimental Settings 14
- 2.3 Pairwise Matching Between Fragments 15
  - 2.3.1 Curve Clustering 16
  - 2.3.2 Matching Clusters 17
  - 2.3.3 Evaluation of Pairwise Matching 18
  - 2.3.4 Matching Refinement using a Modified ICP 20
- 2.4 Global Image Reassembly 21
  - 2.4.1 Reassembly Graph 21
  - 2.4.2 Edge Compatibility 23
  - 2.4.3 Global Search Algorithm 24
  - 2.4.4 Discussion 26
- 2.5 Refinement of the Overall Reassembly 29
  - 2.5.1 Graph Optimization 30
  - 2.5.2 Refinement Based on Graph Optimization 30

2.6 Experimental Results . . . 31
2.6.1 Reassembly Results . . . 32
2.6.2 Running Time . . . 33
2.6.3 Error Evaluation . . . 34
2.6.4 Comparison with Previous Method . . . 37
2.7 Conclusion and Future Work . . . 39

3 CURVE NETWORK MATCHING FOR 3D FRAGMENT REASSEMBLY . . . 40
3.1 Background and Related Work . . . 42
3.1.1 3D Fragment Reassembly . . . 42
3.1.2 Geometric Partial Matching . . . 43
3.2 Template Matching . . . 44
3.3 Pairwise Fracture Region Matching . . . 47
3.3.1 Feature Curve Extraction . . . 47
3.3.2 Feature Curve Matching . . . 48
3.4 Multi-piece Matching . . . 51
3.4.1 Reassembly Graph Representation . . . 51
3.4.2 Matching Objective Function and Constraints . . . 52
3.4.3 Optimizing Multi-piece Matching . . . 54
3.4.4 Multi-piece Reassembly Refinement . . . 55
3.5 Experimental Results . . . 56
3.5.1 Reassembly Results . . . 57
3.5.2 Comparison With Existing Methods . . . 60
3.6 Conclusion and Future Work . . . 63

4 SURFACE MATCHING AND SSM FOR DATA COMPLETION . . . 64
4.1 Related Work . . . 66
4.1.1 Geometric Data Completion . . . 66
4.1.2 Statistical Shape Models . . . 68
4.2 Basic Ideas and Algorithm Overview . . . 69
4.3 Data Preparation . . . 70
4.3.1 Shape Representation . . . 71
4.3.2 Distribution Verification . . . 71
4.3.3 Data Partitioning . . . 72
4.3.4 Data Transformation . . . 74
4.3.5 Global versus Local SSM after Data Preparation . . . 75
4.4 SSM Construction and Data Completion . . . 78
4.4.1 Gaussian-distribution Based SSM . . . 78
4.4.2 T-distribution Based SSM . . . 79
4.4.3 Missing Data Computation . . . 82
4.5 Experimental Results . . . 84
4.5.1 Repairing Benchmark Datasets . . . 85
4.5.2 Completion of the Our Scanned Data . . . 89
4.6 Conclusion and Future Work . . . 90

5 CURVE MATCHING AND CLASSIFICATION FOR ANCESTRY ANALYSIS . 93
5.1 Related Work . . . 95
5.2 Algorithms . . . 96
5.2.1 Data Acquisition . . . 96
5.2.2 Preprocessing . . . 96
5.2.3 Extracting Geometric Features . . . 99
5.2.4 Classification . . . 105
5.3 Experimental Results . . . 105
5.3.1 Sample . . . 105
5.3.2 Data Collection . . . 106
5.3.3 Classification Accuracy . . . 106
5.4 Conclusion and Future Work . . . 108

6 NON-RIGID SURFACE MATCHING FOR FACIAL RECONSTRUCTION . . . 109
6.1 Related Work . . . 110
6.1.1 Facial Reconstruction. . . . 110
6.1.2 Surface Registration . . . 111
6.2 Skull and Template Face Preparation . . . 112
6.2.1 Acquisition of the Complete Skull . . . 112
6.2.2 Landmarks on Skull, Template Face and Corresponding Tissue Depth 113
6.3 Symmetry-guided Non-rigid Deformation . . . 115
6.3.1 Non-rigid Surface Registration . . . 115
6.3.2 Landmarks Fitting Error . . . 115
6.3.3 Smoothness Error . . . 116
6.3.4 Symmetry Error . . . 116
6.3.5 Combining Error . . . 117
6.3.6 Implementation . . . 117
6.4 Experimental Results . . . 118
6.5 Conclusion and Future Work . . . 120

7 CONCLUSIONS . . . 121

BIBLIOGRAPHY . . . 124

VITA . . . 135

LIST OF TABLES

2.1 Runtime table for our reassembly algorithm. 33

2.2 Error table for our reassembly algorithm. 36

2.3 Comparison of the Reassembly Generated by [1] and by Our Algorithm. Numerical Matching Error δ_M is compared. 38

3.1 Runtime Table: #V is the total vertex number of each fragmented model, #F is the number of fragments, T_t, T_p, and T_m are computational times (in seconds) for template-guided reassembly, pairwise matching, and multi-piece matching, respectively. 57

3.2 The Reference Error and Matching Error on different models. $\#F$ is the number of fragments. ε_R and ε_M are the absolute reference error and the absolute matching error (in millimeter). δ_R and δ_M are the reference and the matching error ratios. The error ratio is measured as a percentage of the bounding box diagonal length (i.e., average error / diagonal length $\times 100\%$). Intuitively, for a model with the bounding box diagonal is 1 decimeter, 1% error is 1 millimeter. Some reference errors are not available due to the lack of ground truth model. 62

5.1 Accuracy for a human observer 107

5.2 Accuracy for our algorithm 108

LIST OF FIGURES

1.1 Example Applications of Geometric Matching [2]: (a) partial 3D scans are registered to compose the complete geometry [3]; (b) the motion of deforming organs is modeled and predicted by matching a series of CT/MRI images [4] [5]. 2

1.2 Overview of our matching algorithms and problems. 4

2.1 The pipeline of our 3-step reassembly algorithm. 9

2.2 (a) The original scanned image fragment, (b) the extracted boundary contour, and (c) the approximated polygon. 17

2.3 Two clusters marked by green are matched. 18

2.4 The more the matched contour segments are curved, the higher matching score it is granted. The common sub-curve is marked as blue. 20

2.5 Five iterations on one example. 27

2.6 Two possible matching approaches: The compatible edge set (a) has a higher score and is favored. 28

2.7 (a) Before optimization, (b) After optimization. 29

2.8 An illustration of how the objective function is defined on a graph. 30

2.9 (a) House Image Fragments in 10 pieces, (b) Initial reassembly results, (c) Final reassembly result. 33

2.10 (a) GMP Image Fragments in 10 pieces, (b) Initial reassembly results, (c) Final reassembly results. 34

2.11 (a) Castle Image Fragments in 10 pieces, (b) Final reassembly results. . . . 35

2.12 (a) Flower Image Fragments in 26 pieces, (b) Final reassembly results. . . . 35

2.13 The comparison with existing method. (a) The results computed by the method in [1], (b) The results computed by our method. 37

2.14 The error estimation of our algorithm. (a) The ground truth image (b) The initial reassembly result before graph optimization, and $\delta_G = 0.25\%$, (c) Final reassembly result, and $\delta_G = 0.19\%$. We can see that the error is reduced about 20%. 38

3.1 A Fragmented Chicken Model. The complete model (a) is broken into pieces (b). (c,d) On small fragments, intact regions (blue) and fracture regions (red and gray) may not be easily differentiable. 41

3.2 Feature Extraction and Matching. (a) ISS keypoints on the template and a fragment, (b) refined feature correspondences. 45

3.3 Feature Curve Matching. (a) Extracted feature segments (orange) on a fragment, (b) pruned feature network, (c) sampled wide bases (red crosses). . . 48

3.4 Reassembling fragmented Child Buddha model and Chicken model. 58

3.5 (a) The ground truth model and fragments, (b) the gnome template, (c) a reassembly result ($\alpha = 1$), (d) another reassembly result ($\alpha = 0.01$). 59

3.6 Reassembling a 19-piece fragmented skull model. 60

3.7 Reassembling a fragmented skull obtained in a real law investigation case. . 61

3.8 Reassembling the Buddha Head. (b) The algorithm of [6] fails to reassemble small pieces (in red circles in (a)). (c) Our algorithm correctly reassemble all the small pieces. 62

4.1 Human Bodies with Different Poses. 70

4.2 The clustering on the human body data. (a) shows the center of three clusters (bones); (b) shows the color-coding of the distribution of the vertex weights with respect to each bone. Note that different subparts or clusters can overlap with each other. 74

4.3 Synthesized characters with different Gaussian distribution (the top three rows) and the incomplete characters (the last row). 77

4.4 The P-P plot of the dataset, and its Gaussian distribution fitting (dashed curves) and T-distribution fitting (solid curves). We can see that the data follow T-distribution better. 81

4.5 Color-encoded completion error. (a) Incomplete mesh; (b) completion result using LTSSM. (c) completion result using context-insensitive method [7]. . . 86

4.6 Color-encoded completion error. (a) Incomplete mesh; (b) Template; (c) completion result using LTSSM. (c) completion result using template-based method [8]. 87

4.7 Completion using a model with a different pose. Left column: The original model; Middle: The incomplete model; Right: The completion result. 88

4.8 Color-encoding of the completion error. The original model (a) is repaired using local SSMs (b) and a global SSM (c). 89

4.9 Color-encoding of the completion error. The original model (a) is repaired using T-distribution based SSMs (b) and Gaussian-based SSMs (c). 90

4.10 Our body scanning system. Left: the scanning device where 16 Kinects are mounted on a frame; Right: a completed scan with texture. 91

4.11 Some completion results on our digital scans. Left: Incomplete models; Right: completed models. 92

5.1 Photograph of how the points are recorded using the digitizer. 96

5.2 Rotation Normalization. (a) Original curve (b) Curve after rotation. 98

5.3 The issue when fitting ellipse. 100

5.4 The method to fit ellipse. 100

5.5 Fitting Rectangular Hyperbola. 101

5.6 Fitting Rectangular Hyperbola. 102

5.7 Front Shape for two ancestry. (a) Shape for white (b) Shape for black (c) Fitting for White. 103

5.8 Fitting Results. (a) Fitting for Asian (b) Fitting for Black (c) Fitting for White. 104

5.9 Photograph of how the points are recorded using the digitizer. 106

6.1 (Left) The skull and its landmarks, (Middle) The reconstructed face using clay, (Right) The reconstructed face with texture. 110

6.2 (Left) The fragments of the skull, (Right) The reassembled skull. 113

6.3 The Completion of the skull. 113

6.4 (Left) The landmarks on the skull, (Right) The corresponding landmarks on the face. 114

6.5 The facial Marker on the skull. The vectors from the landmarks on the skull to the facial landmarks should be orthogonal to the skull. 114

6.6 (a) The template face and skull with landmarks (b) The facial reconstruction result. 118

6.7 (a) The facial reconstruction result of asian male (b) The facial reconstruction result of white female. 119

6.8 (a) The reassembled fragmented skull (b) The facial reconstruction result. . 119

ABSTRACT

3D matching is a fundamental and important technique that detects and aligns repeated or similar patterns between geometric objects. It can facilitate many data processing and analysis tasks in computer graphics and vision, including shape comparison and retrieval, data fusion, scene understanding, object recognition, and data restoration. Despite its successful applications in a set of computer graphics, vision and medical image analysis tasks, 3D matching computation remains challenging when the corresponding regions/patterns of the matching data lack saliency (due to small feature size or ambiguous characteristics) or exhibit significant topological/geometric inconsistency (due to data damage, occlusion, or noise). Our research goal is to explore novel matching strategies that can tackle the above difficulties and develop reliable algorithms for practical data reconstruction and restoration applications. Data restoration of fragmented or damaged geometric or image data is to reconstruct their original complete shapes or figures. Its direct applications includes forensic evidence restoration and archaeological heritage preservation. In this dissertation, we systematically develop a set of matching algorithms, involving different geometric primitives (from curve to surface) and different number of matching objects (from pairwise matching to multi-object matching). We demonstrate these techniques' applications in a forensic data restoration pipeline of fragmented skull reassembly, skull damage repair, ancestry assessment/prediction, and facial reconstruction. This restoration pipeline is promising in migrating

the state-of-the-art forensic skull evidence processing and craniofacial reconstruction pipeline in law enforcement into a digital environment.

1. INTRODUCTION

1.1 Background

3D geometric matching is the technique to detect and correspond similar patterns among multiple objects. Specifically, given multiple objects $O_i (1 \leq i \leq n)$, we aim to detect the patterns $P_i \subset O_i$, such that for each P_i, there exists a transformation T_{ij} that brings P_i close to its similar pattern P_j. Here the object O_i can be points, curves, surfaces or volumes. 3D geometric matching is an important and fundamental problem and can facilitate many tasks in computer graphics and vision, including shape comparison and retrieval, data fusion, scene understanding and object recognition, and data restoration. For example, recent 3D scanners can provide us a set of partial scans from different angles while parts of the scans will share the same geometry. Then those scans are registered into a coherent coordinates frame using rigid transformations (Figure 1.1(a)). Another example is in medical image analysis, by computing the matching among the features of sequential CT images, one can construct a good nonrigid deformation and use it to predict the motion of the deforming organs(Figure 1.1(b)).

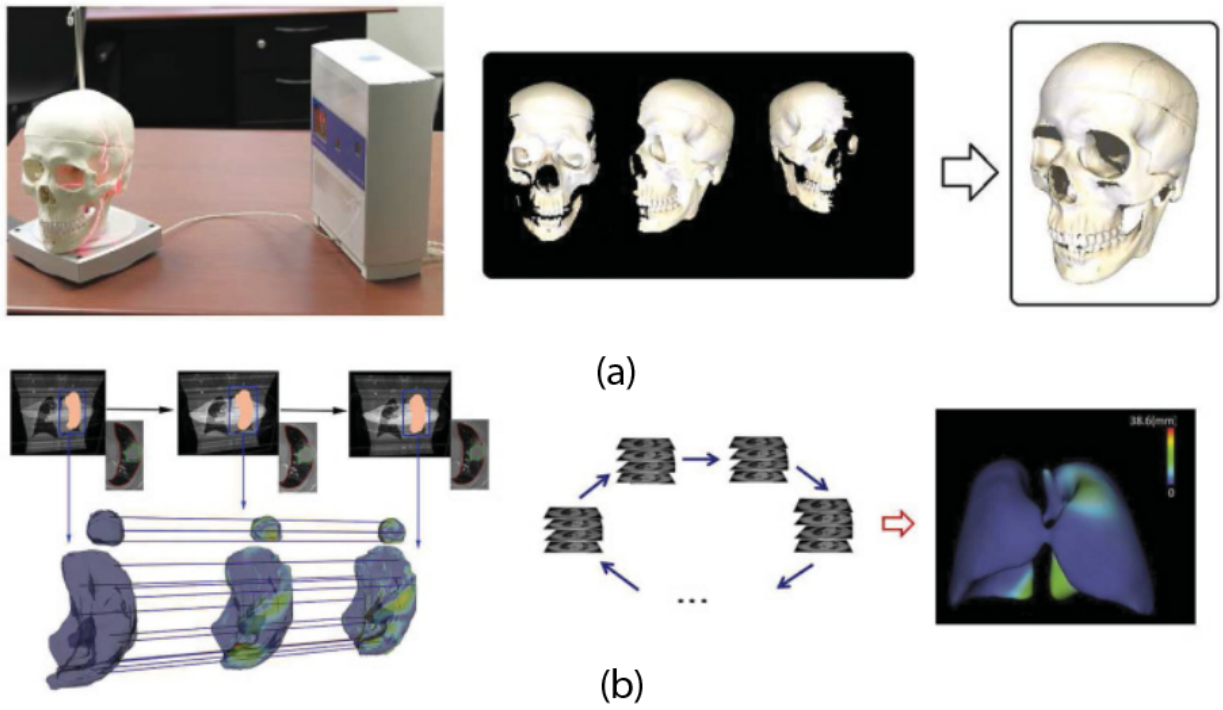


Figure 1.1: Example Applications of Geometric Matching [2]: (a) partial 3D scans are registered to compose the complete geometry [3]; (b) the motion of deforming organs is modeled and predicted by matching a series of CT/MRI images [4] [5].

1.2 Motivation

Despite its wide uses in computer vision and graphics, the 3D geometric matching problem still remains unsolved in literature. The main difficulties are as follows

1. The similar patterns between the two objects are small and lack the salient geometries, thus it is hard to detect the similar patterns.

2. When the data is incomplete due to the occlusions of the scanning process, or the data itself is fragmented or damaged, detecting similar patterns on the data is unreliable, which is the essential step in the geometric matching algorithm.

3. There are multiple possible matches between the pair of objects, and the locally better match does not necessarily lead to the globally optimal matches between multiple objects.

In this dissertation, we aim to propose innovative geometric matching algorithms that can solve the above three difficulties. We demonstrate the effectiveness of our algorithms on data restoration problems. Data restoration is the problem to restore the fragmented or damaged model to its original complete state. It is a new area and has direct applications in many scientific fields such as Forensics and Archeology. For example, in Archeology, the excavated cultural relics are often highly fragmented. To further analyze them, the reassembly of the fragmented pieces is required. In 3D scanning, due to the occlusion and the damage of the object itself, to provide a watertight mesh which can be used for the 3D printing, we need to repair the raw scan. We study a set of matching algorithms, including **curve matching**, **surface matching**, **pairwise matching**, **multi-piece matching** and **template matching**, that can effectively tackle a series of data restoration problems, such as 2D/3D reassembly, data completion, ancestry analysis and facial reconstruction.

1.3 Overview

Fig 1.2 shows the pipeline of my dissertation. The left column shows the matching techniques we develop and the right column shows targeting problems to tackle. In the following chapters I will introduce our matching algorithms and demonstrate their effectiveness in each specific problem.

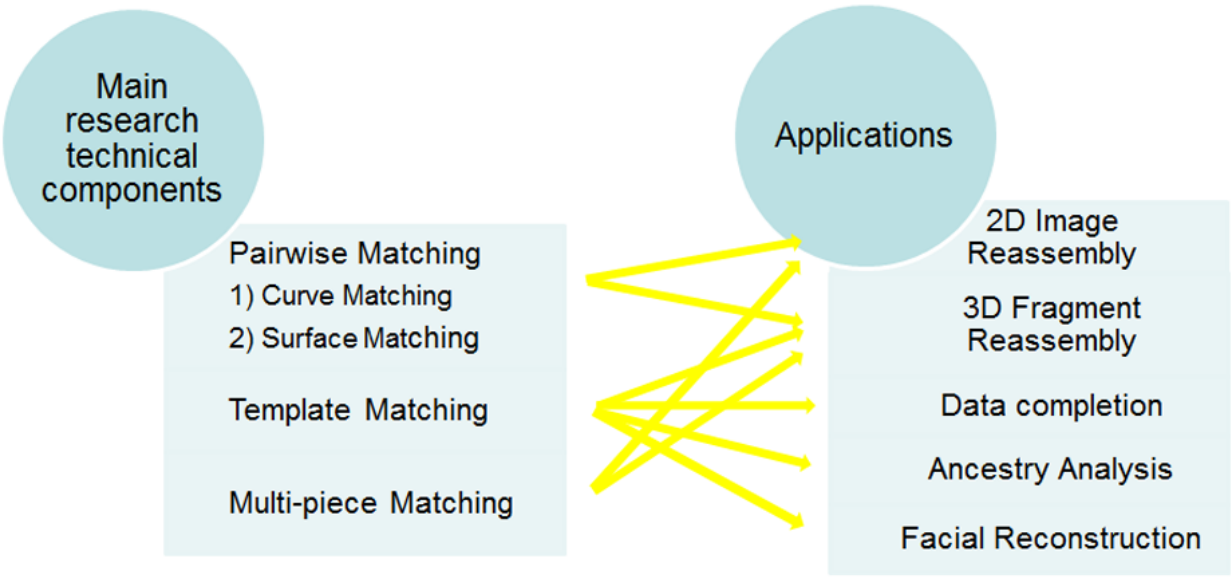


Figure 1.2: Overview of our matching algorithms and problems.

1.3.1 2D Image Reassembly

Fragmented image reassembly recomposes a group of picture fragments into the original complete image. It is a geometric processing problem that has important applications in many fields such as archaeology and forensics. For example, archaeologists need to spend a lot of effort to recompose fractured ancient paintings to restore their original appearances; forensic specialists manually compose damaged documents or pictures to recover the original evidence that was broken by humans. Developing robust geometric algorithms for automatic image reassembly can reduce expensive human labors and improve the restoration efficiency, and hence is highly desirable.

We propose a graph-based optimization framework for automatic 2D image fragment reassembly using **Curve Matching**, **Pairwise Matching** and **Multi-piece Matching**. First, we compute the potential matching between each pair of the image fragments based on

their geometry and color. After that, a novel multi-piece matching algorithm is proposed to reassemble the overall image fragments. Finally, the reassembly result is refined by applying the graph optimization algorithm. We perform experiments to evaluate our algorithm on multiple torn real-world images, and demonstrate that this new assembly framework outperforms the existing algorithms in both reassembly accuracy (in handling accumulated pairwise matching error) and robustness (in handling small image fragments).

1.3.2 3D Fragment Reassembly

Geometric restoration that composes 3D fragmented pieces into the original complete object is a geometric problem that has direct applications in archaeology and forensic investigation in the computer-aided restoration of damaged artifacts and evidence.

We develop a reassembly algorithm to effectively integrate both guidance from a template and from matching of adjacent pieces' fracture-regions. First, we compute partial matchings between fragments and a template using **Curve Matching** and **Pairwise Matching** among fragments. Many potential matches are obtained and then selected/refined in a **Multi-piece Matching** stage to maximize global groupwise matching consistency. This pipeline is effective in composing fragmented objects that are thin-shells and contain small pieces, whose pairwise matching is usually unreliable and ambiguous and hence their reassembly remains challenging to the existing algorithms.

1.3.3 Data Completion

The digitization of physical objects often results in missing regions on the geometric models, due to scanning occlusions, calibration errors, noise of hardware, or damage on the physical

objects themselves. Repairing these missing regions (holes) to get a complete digital model is desirable for various modeling and analysis purposes. This data completion problem is a long-standing and fundamentally challenging geometric modeling problem. For holes that are small or located in simple/smooth geometric regions, filling them using a smooth patch would result in satisfactory results. However, things become more challenging when the holes are big or located on regions with complex geometry or with rich details. Simple smoothing won't produce accurate enough repair in practical modeling tasks.

We propose a new geometric completion algorithm to repair the damaged or missing regions of 3D geometric data based on a novel Local T-distribution based Statistical Shape Model (LTSSM) which relies on the **Surface Matching** and **Template Matching** from the missing data to multiple templates. In this framework, geometric models will first be partitioned into subregions each of which satisfies a T-distribution. A statistical model is constructed to describe each subregion of these models. Then, a given incomplete model will be repaired by the geometry of its undamaged regions with respect to the example datasets. We conduct several experiments to demonstrate the effectiveness of this method.

1.3.4 Ancestry Analysis

Assessing ancestry from the skeleton is an important problem of the practice of forensic anthropology because it aids with the identification of unknown individuals in law investigations. For example, in computer-aided skull processing fragmented skull restoration, correctly identified ancestry can lead to more natural/accurate selection of skull template [9], in forensic facial reconstruction [10, 11], applying appropriate tissue depths also relies on the understanding of the subject skull's ancestry.

We study the palate shape of the skull, or as it is sometimes called, the dental arcade, as the indicator. For the three main ancestry groups: *white*, *black*, and *Asian/native American*, the dental arcades fall into three categories [12, 13]: the *parabolic* (or triangular), the *hyperbolic* (or rectangular), and the *elliptical* (or rounded), respectively. As three geometrically discrete shapes, it stands to reason that a parabola, hyperbola, and ellipse would lend themselves easily to metric assessment, and, thus, a more objective method for assessing ancestry from the palate could be formulated [14, 15]. However, practical computer-aided identification systems based on this idea do not exist. We model this as a **Curve Matching** and classification problem and use **Template Matching** to evaluate the effectiveness of palate shape analysis in ancestry determination.

1.3.5 Facial Reconstruction

Facial reconstruction for postmortem identifications of humans from their skeletal remains is a long-standing and challenging problem. Recently it is solely carried out by forensic artists using physical sculpting with clay, which is time-consuming and labor-intensive. We develop a automatic system based on **Surface Matching** and **Template Matching** to reconstruct the face from the skull. We pick the landmarks on the skull. Given the tissue depths on the skull landmarks, we deform the template face as rigid as possible while the distances between the corresponding skull landmarks and facial landmarks should be equal to the tissue depths. Meanwhile, we maintain the symmetry of the face. We demonstrate the effectiveness of our method on various skulls and template faces.

2. CURVE MATCHING FOR 2D IMAGE REASSEMBLY

We first study the data restoration problem of 2D image reassembly. Fragmented image reassembly recomposes a group of picture fragments into the original complete image. It is a geometric processing problem that has important applications in many fields such as archaeology and forensics. For example, archaeologists need to spend a lot of efforts to recompose fractured ancient paintings to restore their original appearances; forensic specialists manually compose damaged documents or pictures to recover the original evidence that was broken by humans. Developing robust geometric algorithms for automatic image reassembly can reduce expensive human labor and improve the restoration efficiency, and hence is highly desirable.

Existing automatic reassembly algorithms can be generally divided into two categories: color-based approaches and geometry-based approaches. Color-based methods mainly use the color information to predict the adjacency relationship of the fragments and guide the matching [1, 16, 17]. These algorithms are usually efficient, but they sometimes suffer from composition accuracy and may fail when the textures of different fragments are similar. Geometry-based methods reassemble the fragments by matching their boundary curves [18–21]. They can more accurately align adjacent fragments along their breaking regions, but they are sometimes slow and can easily get trapped in local optima then fail to reach the correct result. Therefore, incorporating both the geometry and the color information in the reassembly

computation could make process efficient, accurate, and reliable. In this paper, we present a novel 3-step composition algorithm, as illustrated in Fig. 2.1.

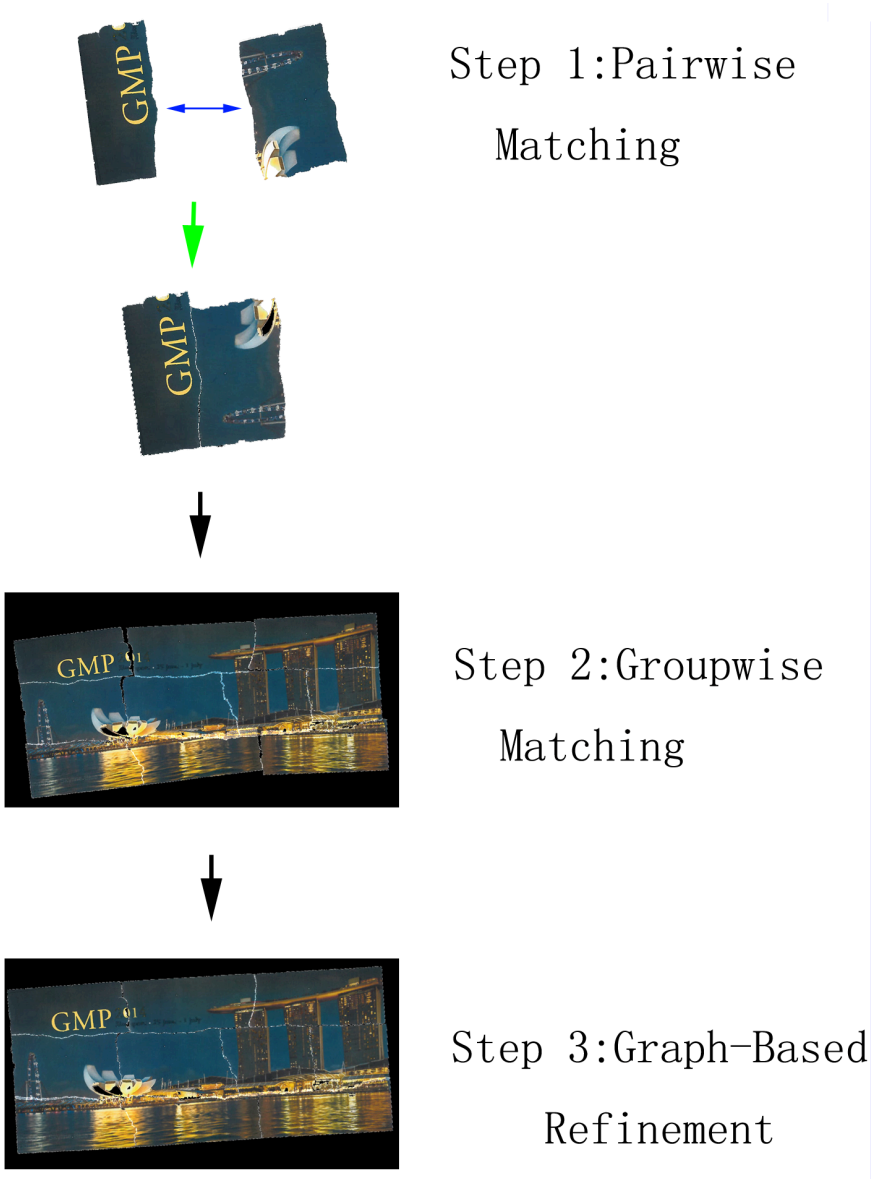


Figure 2.1: The pipeline of our 3-step reassembly algorithm.

Pairwise Matching. The first step is called the pairwise matching, which aims to identify adjacent pairs and compute their initial alignments. Geometry-based pairwise matching methods rely on analyzing the shape of the boundary curve contours; color-based pairwise

matching methods match fragments using their color information. Our integrated algorithm (1) extracts the border of each fragment and represent it as a curve contour, (2) clusters such a curve contour into multiple segments based on both color and geometry, and (3) matches contour segments to suggest the potential alignment between adjacent fragments. This *pairwise matching* algorithm will result in a set of possible matchings between identified pairs of fragments, some of which are correct while some are not.

Groupwise Matching. The pairwise matching produces redundant matches, which we intentionally generate in order to tolerate erroneous adjacency identification due to noise and local minima. A global groupwise matching can better filter out these false matches. Extensive examination of this is a combinatorial optimization problem and is NP-hard. Many previous works use a best-first search strategy (i.e., to explore a graph by expanding the most promising node) with backtracking to solve this problem. In our work, the global composition is formulated as a novel graph-based searching problem, solving which will give us a more reliable global reassembly of multiple fragments.

Composition Refinement via Graph Optimization. Groupwise matching can compose the fragments globally. However, the result is dependent on the composition sequence, and the alignment errors will be accumulated which may even cause a failure in the reassembly (i.e., fragment intersection, see Section 2.5 for details). We derive a graph optimization algorithm to reduce the global matching errors between adjacent fragments.

The **main contributions** of this work are as follows:

1. We integrate both geometry and color information in pairwise matching computation to obtain more reliable alignment between adjacent fragments.

2. We propose a graph-based algorithm that performs groupwise matching to better handle the errors resulted from pairwise alignments and obtain correct adjacency information of all the fragments.

3. We develop a variational graph optimization in the end to reduce the accumulated errors to refine the reassembly and achieve a global optimal result.

2.1 Related Work

2.1.1 Local Pairwise Matching

The essence of fragment reassembly is to find the relative transformations between adjacent fragments. In the procedure of 2D images composition, fragments can be modeled and aligned using their boundaries, which are 2D curve contours. Therefore, pairwise matching often reduces to a partial curve matching problem, which is mainly solved in existing literatures via either geometry-based or color-based approaches.

Wolfson [22] approximates the curves as the polygonal shapes and solves their matching by finding the longest common substring, through a geometric hashing. This algorithm is fast and effectively employed in many puzzle solving methods. However, this algorithm may fail in many cases such as when the boundaries of the image fragments are damaged, thus is inadequate for solving general image reassembly problem. Kong and Kimia [18] resample the boundary curves by a polygonal approximation, then compute a coarse alignment by dynamic programming, finally, use a dynamic programming again to obtain the fine-scale alignment. In [19], the authors calculate the curvatures of each curve and use a dynamic programming algorithm to match the curvature. However, since computing the curvatures

reduces to the second order derivative, this method is sensitive to noise. In [20], the local shape features of the curves are extracted and matched to introduce the local reconstruction. In [21], a shape feature, referred to as the turning function, is estimated for each boundary curve and used to discover the matching of fragment pairs.

The texture or color information of the image fragments is another useful clue in computing local pairwise matching. In [16], the authors try to calculate the pairwise matching by finding the largest common substring with the similar color and curvature information. Their algorithm mainly focuses on the pairwise matching. In [17], the color/texture features are extracted and an FFT-based registration algorithm is utilized to find the alignment of the fragment pieces. In [1], color histograms are calculated to identify the adjacency of the fragment pairs and a Smith-Waterman algorithm is proposed to discover the matching contour segments of the adjacent images.

After local pairwise matching, a matching score is assigned to each pair of matched fragments. Higher score indicates better alignment given certain matching criteria. A widely used registration technique called Iterative Closest Point (ICP) [23] can then be performed to further refine the alignment. Alternatively, many variants of ICP [24] can also be used to increase the robustness of this iterative refinement.

2.1.2 Global Groupwise Reassembly

Pairwise matching of fragments may result in wrong matching (usually due to either local minima, similar geometry/texture information of nonadjacent fragments, or erosion-caused geometry/texture dissimilarity of adjacent fragments). An examination of multiple pieces globally can better eliminate such error or ambiguity. In [25], the author proposed a global

method to solve jigsaw puzzles and the algorithm can solve a puzzle with a large number of pieces. But the algorithm requires the preliminary knowledge on the shapes of the puzzle and thus is not suitable for general image reassembly. In [26], all matching pairs are examined and the matching with the highest score is selected, the corresponding pair is merged into a new piece. This process is repeated until only one piece is left. Since the incorrect matching pair can easily be picked during this merging process, a backtracking strategy is incorporated in the process. But since the incorrect matching occurs very often, this merging process becomes computationally expensive and thus not practical in real applications. In [18], this best-first search strategy with backtracking is still used. They merge three pieces at a time based on the assumptions that generic breaks in puzzles only produce "T" and "Y" junctions. Similarly, in [27], the matching with triple junctions is granted with much higher score, thus is more likely to be selected in the global search process. Then in [20], the local reconstruction is resolved within the process of global search, which demonstrates better global results.

2.1.3 Reassembly of 3D Geometries

The reassembly of 3D objects is another closely related topic. One approach is to project 3D objects to 2D and match their 2D planar boundaries using 2D methods [28]. The boundaries are usually detected through the difference of surface texture or geometry on the intact surface and fracture surface or manually identified. These methods work well for thin shells whose break surface geometry can usually be ignored and treated as curves. Cooper et al. [29] assembled 3D pot fragments by matching the break-curves and shard normals. Willis et al. [30] improved this approach in pots assembly by applying Bayesian analysis through

a semi-automatic matching. Their experiments demonstrated to be very effective in pottery assembly. However, this approach is for geometric objects that are axially symmetric and geometrically simple/smooth, such as pots and vases. For more general 3D solid models, in [31, 32], fragments are segmented into multiple fracture surfaces and matched via depth maps; in [6], a 4-step framework was proposed, also relying on fracture region matching. Both of these methods require effective fracture region segmentation on each 3D fragment, which is usually highly nontrivial especially when the fragment is small. In [33] and [34], templates are used to guide initial reassembly of thin-shell fragments, then the initial compositions are refined through break-curve analysis. These algorithms were used to restore/repair fragmented human skulls [35,36], which have subtle and complex geometric details, but available template models with similar global structure.

2.2 Problem Formulation and Experimental Settings

Given a set of fragments $F = \{F_i\}, i = 1, \ldots, n, F_i \subset \mathcal{R}^2$, which are from a torn image I, we want to solve the transformation for each fragment F_i, i.e., a 3×3 matrix T_i, so that these transformations compose all the fragments back into the original image $I' = \bigcup T_i(F_i)$.

In our experiments, each image is torn into multiple fragmented pieces. We scan all the fragments. For each fragment F_i we obtain a digital image scan I_i which has a colorful region appeared on a white background. We can segment such a region F_i out from the white background by either using existing segmentation algorithms or by simply removing the exterior regions composed of connected white pixels. We use these extracted fragments as the input to the reassembly algorithm.

2.3 Pairwise Matching Between Fragments

This section elaborates our algorithm of computing the pairwise matching between two image fragments. First, the boundary of each fragment F_i is extracted from the scan image I_i; the boundary is a 2D curve loop. Then, the pairwise matching between two image fragments can reduce to a partial curve matching problem. Our pairwise matching algorithm utilizes both the color and geometry information. Unlike existing algorithms that directly use dynamic programming [19] or geometric hashing [22] for curve matching, we first cluster each boundary curve loop into several segments using both color and geometry, then match the clusters with the similar color and geometry. Intuitively, if two boundary curves are matched, then they should share a common sub-curve with the similar color. Therefore, by simply searching through all the matches between the similar segments we can evaluate all the possible matches between two curves.

The four-step pairwise matching pipeline is formulated as follows: first, cluster the boundary of each fragment into curve segments; second, align fragment pairs by computing transformations between their curve segments; third, refine the matchings by an accordingly modified ICP algorithm; finally, evaluate and sort the matching scores of all pairwise transformations.

Notations for Pairwise Matching. Given two boundary curves C_i and C_j, we aim to calculate a transformation $T_{ij}(C_i)$ that matches C_i to C_j. A desirable transformation T_{ij} should align a portion of C_i with that of C_j, indicating the two fragments are adjacent and share one or multiple common curve segments. A curve is partitioned into multiple curve segments, or also called clusters here. A cluster or a curve segment in C_i is denoted as S_{ik}. The length of S_{ik} is denoted as $l(S_{ik})$. The average color of the S_{ik} is denoted as $c(S_{ik})$.

2.3.1 Curve Clustering

First, we approximate each fragment F_i's boundary curve contour C_i using a polygon. Here we use a slightly modified Douglas-Peucker (DP) algorithm [37] to partition the boundary curve contour C_i into a set of curve segments. For a curve segment, denoted as $S = \widetilde{v_1 v_2}, S \subset \mathcal{R}^2$, where v_1 and v_2 are the starting and ending points, the DP algorithm first uses a line segment connecting v_1 and v_2, denoted as (v_1, v_2), then, finds a point $v_3 \in S$ that is farthest from the line (v_1, v_2). If the distance d from v_3 to (v_1, v_2) exceeds a predefined threshold, then $\widetilde{v_1 v_2}$ is divided into two shorter segments $\widetilde{v_1 v_3}$ and $\widetilde{v_3 v_2}$. Then the DP algorithm recursively runs on $\widetilde{v_1 v_3}$ and $\widetilde{v_3 v_2}$, until no segment can be further divided. After performing the DP algorithm, the curve $S = \widetilde{v_1 v_2}$ is divided and approximated by a set of segments $\{S_i\}, i = 1, \ldots, N_C$. Next, we further partition each segment S_i into several segments S_{ik} such that points on each segment S_{ik} should have similar color. For the curve contour C_i, we first extract a pair of points v_1, v_2 that are farthest from each other, then we run the DP algorithm on $\widetilde{v_1 v_2}$ and $\widetilde{v_2 v_1}$ respectively. After this clustering, the curve contour C_i is divided into a set of curve segments $\{S_{ik}\}$ such that: (1) each curve segment can be approximated by a line segment, and (2) each curve segment has a uniform color. We call each such curve segment a *cluster*. Fig. 2.2 shows an example of boundary extraction and its polygon approximation.

Note that the polygon approximation of the contour C_i can be affected by the selection of starting and ending points v_1 and v_2. However, when the threshold is small and the curve subdivision is fine, the approximation result is still mainly decided by the intrinsic geometry of C_i and is not sensitive against the selection of v_1 and v_2.

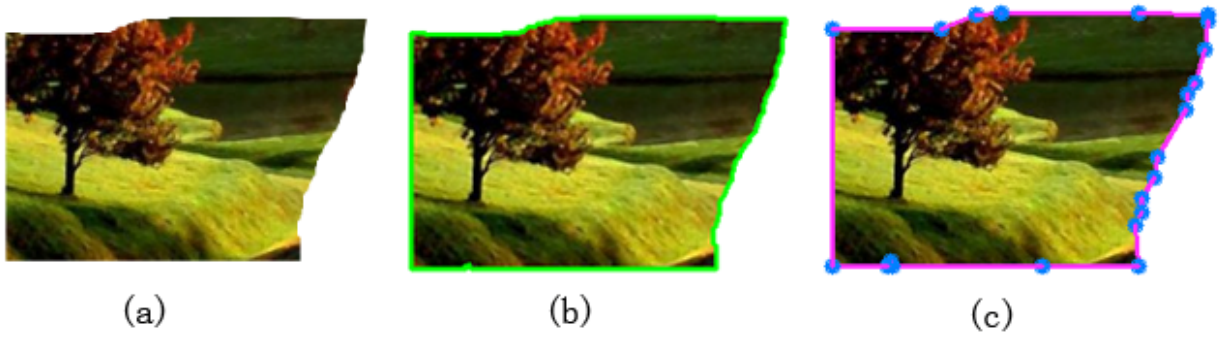


Figure 2.2: (a) The original scanned image fragment, (b) the extracted boundary contour, and (c) the approximated polygon.

2.3.2 Matching Clusters

If two fragments F_i and F_j are adjacent, their boundaries C_i and C_j should share one (or multiple) common curve segment $S_i \subset C_i, S_j \subset C_j$ where S_i and S_j have the same geometry and color. We match C_i and C_j by matching the clusters extracted in Sec. 2.3.1. This cluster matching will provide us a good estimation of the matching between two fragments.

We say two clusters S_i and S_j are *potentially corresponded* if their average colors and lengths are close enough. First, we calculate the barycenter for each cluster. Also, since the cluster is like a line segment, the direction of the cluster is defined by its two endpoints. For each cluster S_i, it is assigned with its barycenter and direction. Given two contours C_i and C_j, we pair clusters $S_{ik} \subset C_i$ and $S_{jh} \subset C_j$ such that S_{ik} and S_{jh} are potentially corresponded. For each pair of corresponded segments S_{ik} and S_{jh}, we can compute a transformation (rotation + translation) using their directions and barycenters (Fig. 2.3). Each transformation between two clusters indicates a potential matching between these two contours. By efficiently calculating the matching between the clusters, we can obtain a set of potential matches between two fragments.

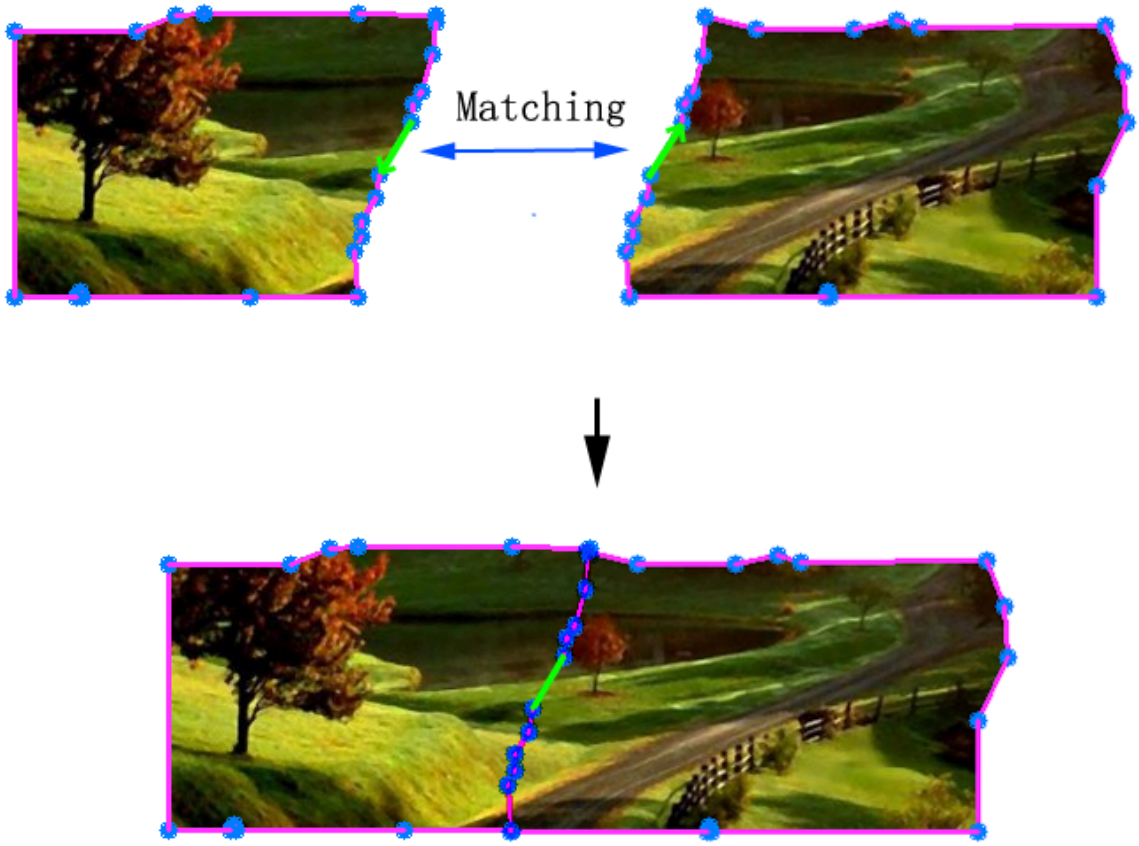


Figure 2.3: Two clusters marked by green are matched.

2.3.3 Evaluation of Pairwise Matching

We need a qualitative score to evaluate the probability of each matching between two fragments: a higher matching score indicates that a match is more likely to be correct. Given two curves C_i and C_j and their matching T, a portion of the first curve, after transformation, denoted as $T(C_i)$, will (approximately) overlap with that of the second curve C_j. We call such shared common portion as the *common curve segment* on C_i and C_j. After being identified as potential adjacent fragments, the two fragments F_i and F_j will have higher matching score if:

1. The arc-length of the common curve segment they share is large.

2. The color difference between the common curve segments on C_i and C_j is small.

3. The common curve segments have salient geometry (see Figure 2.4), rather than being simple straight lines.

Consider the clusters $S_{ik} \subset T(C_i)$ and $S_{jh} \subset C_j$, where the arc length of S_{ik} is $l(S_{ik})$ and its average color is denoted as $c(S_{ik})$. We define the matching score as follows. We find all pairs of the clusters (S_{ik}, S_{jh}) such that their barycenters are close enough. Then, according to the three criteria:

1. $w_1 = \sum (l(S_{ik}) + l(S_{jh}))/2$;

2. $w_2 = \sum (l(S_{ik}) + l(S_{jh}))/2 * d(c(S_{ik}), c(S_{jh}))$, here $d(c(S_{ik}), c(S_{jh}))$ measures the difference of their average colors;

3. $w_3 =$ the number of the pairs of the clusters (S_{ik}, S_{jh}).

The final total score is $w = (w_1 + \alpha * w_3)/w_2$. The parameter α weights the importance of w_3. In our experiments, we found $\alpha = 5$ is usually reliable and leads to good reassembly results, and set it this way empirically.

Once the evaluation of the matches is done, we pick the k ($k = 50$ in our experiment) top matches with the highest scores for each pair of adjacent fragments. Further more, (1) if two matches have similar transformations, then only one copy (the match with the higher score) is kept; (2) the intersection test will be performed for each match: transformations that lead to obvious intersections of fragments will be detected and discarded.

(a) (b)

Figure 2.4: The more the matched contour segments are curved, the higher matching score it is granted. The common sub-curve is marked as blue.

2.3.4 Matching Refinement using a Modified ICP

Due to the noise of the device or the damage of the image, the boundary curves of adjacent images often fail to be matched perfectly. This pairwise matching of clusters usually only provides good initial alignments, and we further refine these matches. The Iterative Closest Points (ICP) algorithm is a well-known effective method to rigidly align two geometries. Many of its variants have been discussed and evaluated in [24]. Integrating several ICP variants introduced in [24], we design a *modified ICP* algorithm that works more robustly against outliers in this partial contour matching. Specifically, for each point on the boundary of a fragment F_i, we find its nearest point on the other fragment F_j as its correspondence point; then iteratively we solve a rigid transformation based on the point correspondence on F_i to better align with F_j. In our algorithm, the correspondence between a pair of points is identified as an outlier if one of the following three conditions is met:

(1) The distance between two corresponded points is larger than a threshold;

(2) The normal vectors of two corresponded points are not pointing towards opposite directions, i.e., their dot product is far from -1;

(3) The color difference between two corresponded points is large.

2.4 Global Image Reassembly

The pairwise matching step provides us a set of possible matchings between each pair of the image fragments; each matching is associated with a matching score. This score is usually a good indicator to whether this alignment is correct. A straightforward strategy is to iteratively pick the matching following the scores from high to low. However, this may not always work as the highest score is not a guarantee to the correct match, especially when there are many small fragments: small fragment's alignment with large fragment, even if it is not perfect (i.e. not correct), could outscore its correct alignment with another small fragment. Hence we propose a more reliable graph-based global search algorithm to extract correct matches globally from the initial corresponding pairs extracted in pairwise matching step.

2.4.1 Reassembly Graph

We define a reassembly graph $G = (V, E)$, where each node $v_i \in V$ represents one image fragment F_i. The matching from F_i to F_j are denoted by an edge set E_{ij}, each edge $e_{ij}^k \in E_{ij}$ represents a potential matching we compute. $T(e_{ij}^k)$ denotes the *relative transformation* of the edge e_{ij}^k and $w(e_{ij}^k)$ denotes its associated *matching score*. Note that, with multiple

pairwise matching between a pair of fragments computed, in G there may be multiple edges connecting two nodes, namely, G is not a simple graph.

We define a transformation matrix $T(v_i)$ on each node v_i. $T(v_i)$ represents a 3×3 matrix that transforms the image fragment F_i from its original position to the final position in the composed image. In the global reassembly step, we fix the relative transformations $T(e_{ij})$ computed during the pairwise matching step and solve each fragment's final transformation matrix $T(v_i)$.

On a reassembly graph, the global reassembly is formulated as follows. We want to find a simple subgraph $G' = (V, E'), E' \subset E$, and compute the transformation matrix $T(v_i)$ for each fragment F_i, subject to three constraints:

(1) G' is a *simple graph*, namely, between each pair of nodes v_i and v_j, there is at most one selected edge e_{ij}^k from E_{ij};

(2) $\forall F_i, F_j$, after applying $T(v_i)$ and $T(v_j)$, they should not intersect with each other;

(3) For any edge $e' = (v_i, v_j) \in E'$, the relative transformation relationship is satisfied: $T(v_i) = T(e') * T(v_j)$.

A subgraph satisfying constraints (1)-(3) introduces a valid reassembly of the given image fragments. In practice, it is difficult to fully enforce constraint (3) due to noise and numerical issues, we use a $2D$ threshold vector, $\epsilon = (\epsilon_L, \epsilon_\theta)$, where ϵ_L and ϵ_θ indicate the magnitude of translation and rotation angle, respectively. If the difference between $T(v_i)$ and $T(e') * T(v_j)$ is smaller than ϵ, then we consider constraint (3) is satisfied.

2.4.2 Edge Compatibility

To model the constraints described in Section 2.4.1, we define the concept of *edge compatibility*, which describes whether the relative transformations defined on different edges incident to a node (fragment) are consistent. Given two edges e_{ij}^h and e_{ik}^l, if they are both in the final subgraph G', then we shall have $T(v_i) = T(e_{ij}^h) * T(v_j)$, and also $T(v_i) = T(e_{ik}^l) * T(v_k)$. Therefore, on v_i the following *compatibility equation*

$$T(e_{ij}^h) * T(v_j) = T(e_{ik}^l) * T(v_k). \tag{2.1}$$

should be satisfied. We say two edges e_{ij}^h and e_{ik}^l are *compatible* if equation (2.1) is satisfied. Recall that the subscripts i, j, k indicate the nodes that the edges are connecting, and the superscripts h, l indicate different pairwise transformations computed between fragment pairs. If two edges are not incident to a common node, they are considered to be compatible to each other. When a node is incident to several edges, among these edges some may not be compatible to some others; non-compatible edges should not exist together in the resultant simple graph G'. A set of edges is said to be *compatible* if any two edges in it are compatible to each other.

Based on this definition, we have the following properties of a compatible edge set.

- For two different edges, e_{ij}^h and e_{ij}^l, connecting a same pair of nodes v_i and v_j, since each edge is associated with a different relative transformation, we have $T(e_{ij}^h) * T(v_j) \neq T(e_{ij}^l) * T(v_j)$. Therefore, they are non-compatible. If the edge set of a graph is compatible, then this graph must be a simple graph.

- For two edges that are connecting three nodes, e_{ij}^h and e_{ik}^l, if $T(e_{ij}^h) * T(v_j)$ and $T(e_{ik}^l) *$

$T(v_k)$ indicate two different $T(v_i)$, namely, two different manners to position fragment F_i, then they are non-compatible.

- If a node v_i has only one incident edge e_{ij}, then $T(v_i)$ can be unambiguously determined by $T(v_j)$ and the relative transformation $T(e_{ij})$. Hence, an edge e_{ij} is compatible to all other edges if no other edge is incident to node v_i.

The *score* of a compatible edge set $w(E_{comp})$ is defined as $w(E_{comp}) = \sum_{e \in E_{comp}} w(e)$, namely, the sum of weights of all the edges in this set. Fig. 2.6(a) shows an example of the compatible edge set. If all the edges in the subgraph G' are compatible, then this subgraph G' satisfies constraints (1) and (3) described in Section 2.4.1. Therefore, we want to search for a compatible edge set with a largest total score.

2.4.3 Global Search Algorithm

Suppose we have a set of partially reassembled images $I = \{F_1, F_2, F_3, \ldots, F_n\}$. The widely-used best-first search algorithm [26] tries to find the matching with the highest score between the fragment $F_i \in I$ and a fragment $F_j \notin I$. This step is recursively called until all the fragments are reassembled. However, this greedy strategy may fail, because as we mentioned before, higher score may not always guarantee a correct matching. In our approach, instead of simply selecting the edge with the highest weight, we try to find the compatible edge set with the highest weight. The intuition behind this is that if one fragment is matched with multiple fragments, then it is more favored. Our algorithm is given as follows

1. Generate graph G with the pairwise matching input. Initiate a new graph $G' = (V', E')$ with $V' = \emptyset$ and $E' = \emptyset$.

2. Find out the edge with the highest weight. Suppose it is e_{ij}. Then set $E' = E' \cup \{e_{ij}\}$. $V' = V' \cup \{v_i, v_j\}$. Set $T(v_j)$ as identity matrix and $T(v_i) = T(e_{ij})$.

3. For any node $v_i \notin V'$, we establish a set of edges $E^i_{new} = \{e_{ik} | v_k \in V'\}$. Here E^i_{new} is simply a collection of edges connecting v_i and one of the nodes in V'. Find all the subsets $E^i_{comp} \subseteq E^i_{new}$ such that the edge set E^i_{comp} is compatible.

4. Let $E_{comp} = \{E^i_{comp} | v_i \notin V'\}$ denote a collection of all the compatible edge sets computed in step 3. We compute the weight for each set in E_{comp}. E_{comp} is sorted by the weight of each set.

5. First we choose the compatible edge set with the largest weight produced in step 4, for example, E^h_{comp}. The transformation matrix for v_h is computed by $T(v_h) = T(e_{hk}) * T(v_k)$ for one arbitrary edge $e_{hk} \in E^h_{comp}$. Then we apply $T(v_h)$ on the fragment F_h and check whether it is intersected with the existing reassembled image fragments. If not, then $V' = V' \cup \{v_h\}$ and $E' = E' \cup E^h_{comp}$. If yes, we discard this edge set, and go to the edge set with the second largest weight. This step is repeated until one compatible edge set is found.

6. If $V' \neq V$, which means that not all the fragments are reassembled, go back to Step 3.

The illustration of our algorithm is shown in Fig. 2.5.

After the global search algorithm terminates, the transformations $T(v_i)$ for each fragment F_i are computed. By applying the transformation matrix on each fragment, the image is reconstructed.

2.4.4 Discussion

Instead of searching for one single edge, the matching that brings one fragment to align with multiple fragments is more favored in our process. For example, in Fig. 2.6(a), the fragment is aligned with multiple fragments, thus is more favored in our global search process.

The previous works also consider the problem of original best-first global search algorithm. For example, in [18], the best-first search with backtracking is still used. Instead they merge three pieces at a time based on the assumptions that generic breaks in puzzles only produce T and Y junctions. Similarly, in [27], the matching with triple junctions is granted with much higher score, thus is more likely to be selected in the global search process. Compared with their work, our global reassembly algorithm solves this problem in a more general form. In [20], the local reconstruction is resolved within the process of global search. In comparison, our algorithm does not require to resolve the local reconstruction within the process, thus is more efficient. The authors in [1] also consider the compatibility of matching pairs. However, they only consider compatibility of the rotation angle which is not sufficient for determining the reassembly is valid. In our algorithm, not only rotation but also translation is considered, thus our method is more reliable.

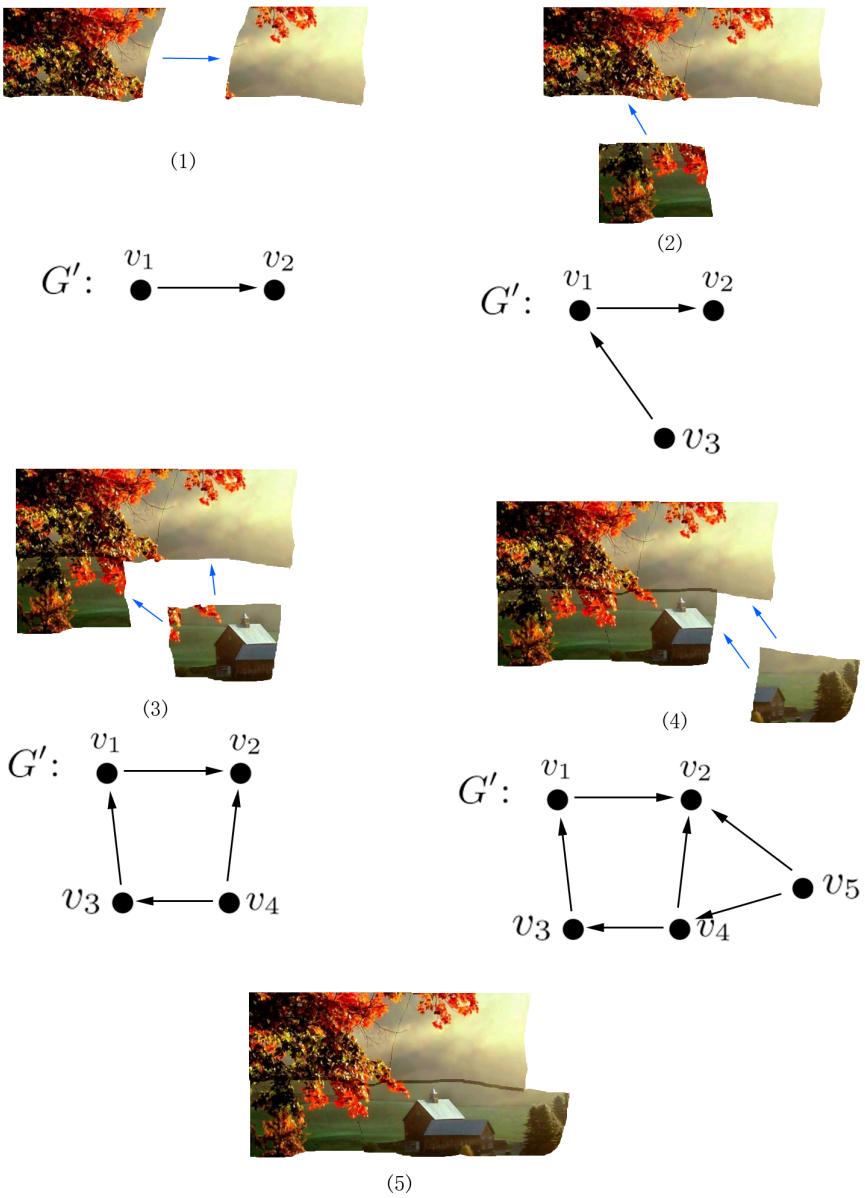


Figure 2.5: Five iterations on one example.

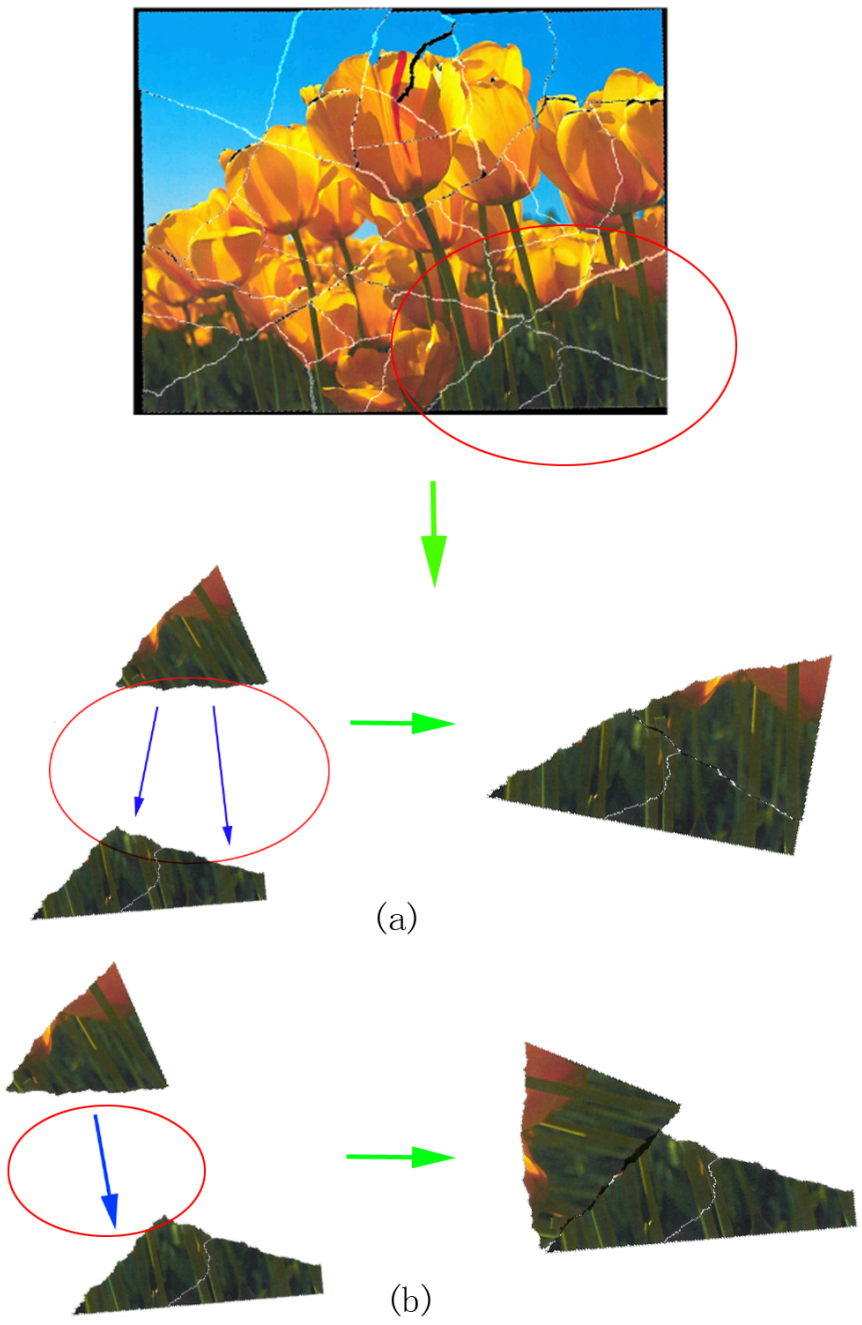


Figure 2.6: Two possible matching approaches: The compatible edge set (a) has a higher score and is favored.

2.5 Refinement of the Overall Reassembly

In the previous section, we obtain a global reassembly of all the fragmented images. But small error may be inevitable in the transformations computed in the previous two steps, caused during e.g. polygon approximation, the inconsistency of the pixelization in image scan. In Step 2, we start with one fragment and iteratively glue other fragments. Transformation errors will be accumulated and sometimes this could even cause a failure in the reassembly (Fig. 2.7(a)). The error becomes especially big when the number of the fragments is large. In this section we take the global reassembly obtained in first two steps and apply a graph optimization algorithm to globally refine the reassembly, by minimizing the matching errors caused by the global reassembly process.

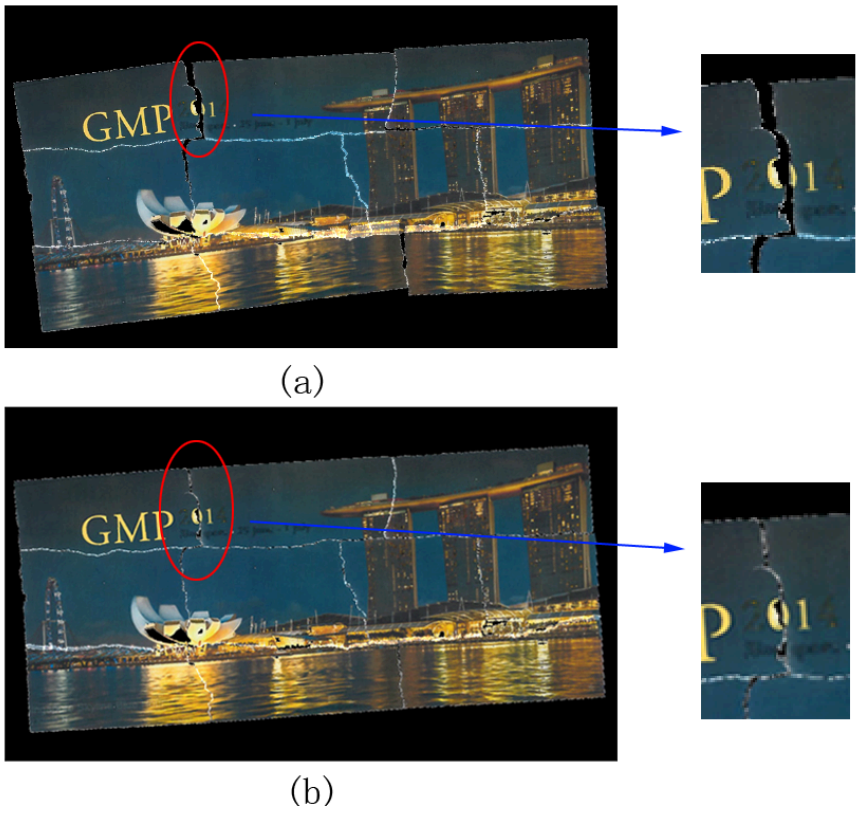


Figure 2.7: (a) Before optimization, (b) After optimization.

2.5.1 Graph Optimization

The general graph optimization problem can be formulated as the minimization of the following nonlinear least square function:

$$F(x) = \sum_{e_{ij} \in E} f(x_i, x_j, z_{i,j})^T \Omega_{ij} f(x_i, x_j, z_{i,j}) \tag{2.2}$$

Here, $x = (x_1, ..., x_n)$ is a vector of parameters we need to solve, where each x_i represents a variable defined on a node of the graph, z_{ij} and Ω_{ij} represent respectively the constraint and the weight matrix relating the parameters x_j and x_i which are defined on each edge of the graph. $f(x_i, x_j, z_{ij})$ is a vector error function that measures how well x_i and x_j satisfy the constraint z_{ij}. The error is 0 when x_i and x_j perfectly match the constraint. This minimization problem can be solved by the famous Gauss-Newton method or the Levenberg-Marquardt algorithm. Fig. 2.8 shows an example of an objective function on a graph.

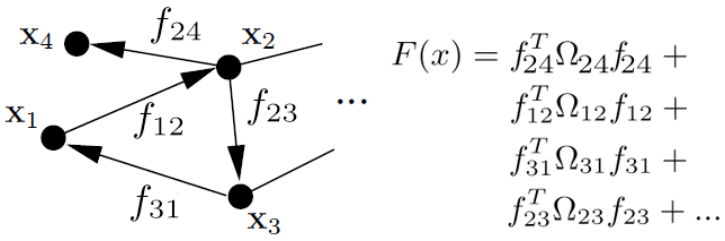


Figure 2.8: An illustration of how the objective function is defined on a graph.

2.5.2 Refinement Based on Graph Optimization

In our work, following the general framework of the graph optimization algorithm, our problem can be similarly formulated as:

$$F(x) = \sum_{e_{ij} \in E'} ||T(e_{ij}) - T(v_i) * T(v_j)^{-1}||^2 * w(e_{ij}) \tag{2.3}$$

Here, $T(v_i)$ denotes the transformation matrix on each node v_i, representing the transformation matrix that transforms the image fragment F_i to its position in the assembled image. $T(e_{ij})$ denotes the relative transformation of the edge e_{ij} and $w(e_{ij})$ denotes its associated matching score. It is clear that when the relative transformation relationship is satisfied, that is, equation $T(v_i) = T(e_{ij}) * T(v_j)$ is satisfied, then the error of $||T(e_{ij}) - T(v_i) * T(v_j)^{-1}||$ is 0.

According to the global search algorithm, we compute the transformation matrix for each node by the relative transformation on one edge at a time. Thus, the calculation of the transformation matrix for each fragment is based on a spanning tree T in the subgraph G'. For those edges not in T. The equation $T(v_i) = T(e_{ij}) * T(v_j)$ can not be satisfied, and the errors occur on those edges. Therefore, we take the subgraph $G' = (V, E')$ obtained in Sec. 2.4 as the input. Then by minimizing $F(x)$, the errors on the edges are globally minimized in the least square sense, and the transformation matrices for the fragments are refined. We implement this graph optimization algorithm using the framework of [38]. With the derived analytic derivatives, this optimization converges to a local minimum efficiently.

2.6 Experimental Results

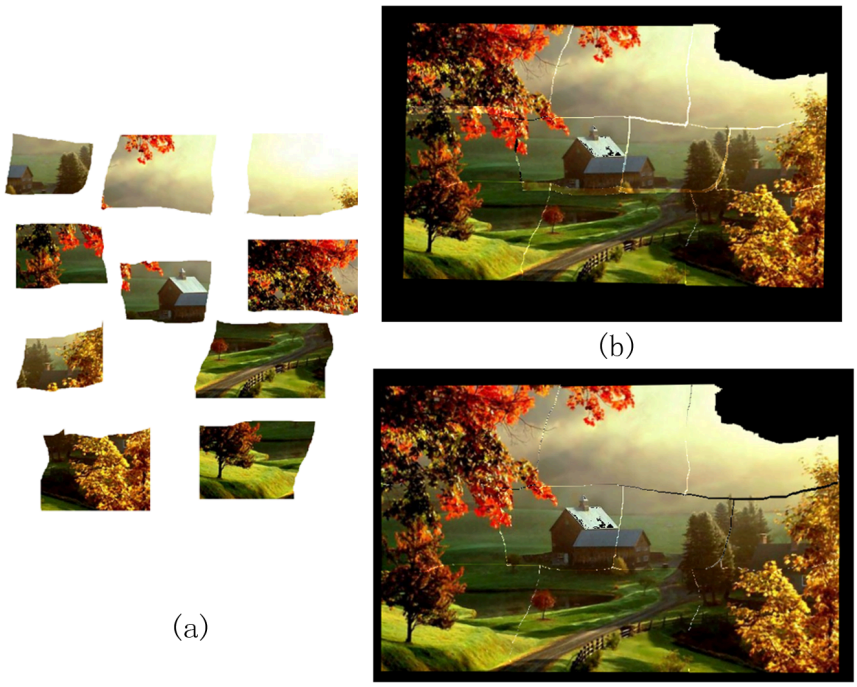
We perform extensive experiments to evaluate our reassembly algorithm. We found a few images online, and print each image out with the size of about $20cm \times 25cm$, and we then randomly tore an image into multiple pieces. The size of the fragments is ranging from $4cm \times 4cm - 7cm \times 7cm$. Then each image fragment is scanned into the computer with 150 dpi resolution. Our algorithm is implemented in C++ and performed on a desktop with 2.27GHz Xeon(R) CPU and 12 GB RAM.

2.6.1 Reassembly Results

Fig. 2.9(a) shows the 10 image fragments. Fig. 2.9(b) shows the initial reassembly result after the global reassembly and Fig. 2.9(c) shows the final result after graph optimization. From the result we can see that the initial reassembly result contains overlapping between the neighboring fragments and is eliminated after optimization.

A challenging example is shown in Fig. 2.10. In this example, the fragments have similar boundary geometry (the shape is like a square) and color. Therefore, in the pairwise matching stage, a large number of the ambiguous matchings with similar scores are produced. Many best-first search methods [18,20,26] will be trapped in the local minima due to the ambiguity of the local pairwise matches. However, the correct matchings are still obtained during our global search process and a successful reassembly is achieved. This demonstrates the effectiveness of the Steps 2 and 3 of our reassembly algorithm.

Fig. 2.11 shows another example of the images torn into 10 pieces. Fig 2.12 shows an example of the images torn into 26 pieces. In this example, the number of pieces becomes large and the size of the fragments becomes small. It should be noted that for those small fragments, they are very likely to have similar colors. Therefore, for those color based method like [1, 16, 17], it is hard to provide the accurate matching between those fragments. Also, for small fragments, due to the lack of features on their boundary, it becomes difficult for the existing method to get the correct matches. However, we can still get the correct global reassembly, which demonstrates the robustness of our algorithm.



(a) (b) (c)

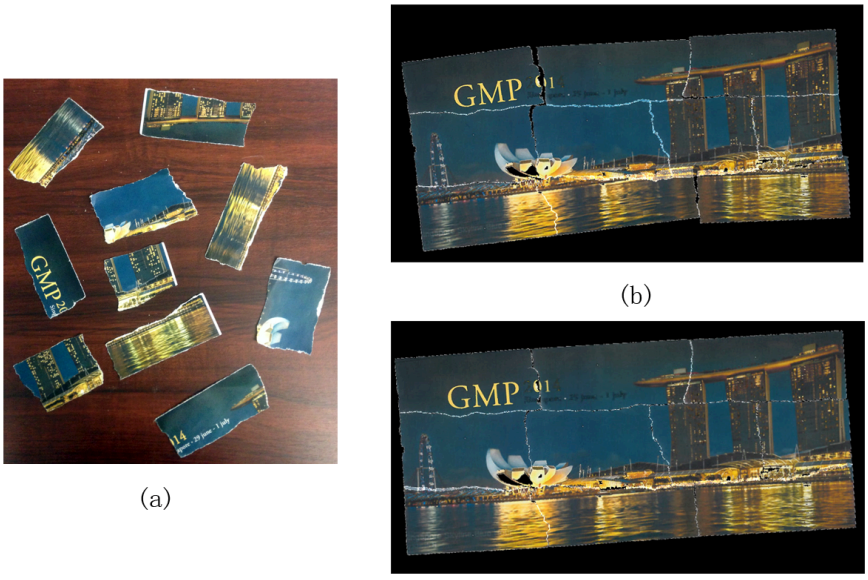
Figure 2.9: (a) House Image Fragments in 10 pieces, (b) Initial reassembly results, (c) Final reassembly result.

2.6.2 Running Time

Table 3.1 shows the running time our algorithm takes for each model. From this table we can see that the total time for our algorithm is from 3 to 15 minutes, depending on the number of fragments. If the number of fragments is the same, then the time for the global reassembly is also roughly the same.

Table 2.1: Runtime table for our reassembly algorithm.

Images	# of fragments	Image size	T (in minutes)
House image	10	686 * 426	2.7
GMP image	10	936 * 399	2.8
Castle image	10	649 * 424	2.5
Flower	26	1024 * 768	14.6



(a) (b) (c)

Figure 2.10: (a) GMP Image Fragments in 10 pieces, (b) Initial reassembly results, (c) Final reassembly results.

2.6.3 Error Evaluation

To evaluate the composition result numerically, we have developed two types of experiments.

1. **Ground Truth Error** δ_G. If the *ground truth* is available, we can measure its deviation from our computed result. We perform the following new "fracturing" simulation: first, digitally slice the images into different pieces; second, transform these fragments using randomly generated rigid transformations; then run our reassembly algorithm to compose the fragments. Finally, we evaluate our composition result by comparing each pixel's displacement from the ground truth. The average error from the ground truth, denoted as the *ground truth error* δ_G, is normalized through dividing the average pixel displacement by diagonal length (unit: pixels). We performed such an experiment on the GMP Logo figure (Fig. 2.14). The image resolution is 936×399. We can see that after pairwise matching, the δ_G is 0.25%. After the groupwise matching and global

(a) (b)

Figure 2.11: (a) Castle Image Fragments in 10 pieces, (b) Final reassembly results.

(a) (b)

Figure 2.12: (a) Flower Image Fragments in 26 pieces, (b) Final reassembly results.

optimization, the error reduces to 0.19%.

2. **Matching Error** δ_M. If the *ground truth* is not available, we evaluate the reassembly by measuring how well (in terms of both the geometry and texture) the adjacent fragments match each other after the composition.

   Specifically, for each boundary pixel p_1 of an image fragment, we compute the distance from p_1 to its corresponding nearest pixel p_2 on neighboring fragments. The valid nearest correspondence is defined in our modified ICP algorithm in Section 2.3.4. We

denote the average distance as the matching error δ_M. The matching error is also normalized by dividing the average distance by the diagonal length of the composed image.

Note that: When no original image is available, we use this normalized matching error to evaluate the composition. In addition, even when the original image is available, (for example, in our experiments, we print an image, physically tear it into pieces, then re-scan the fragments for the digital composition), this matching error δ_M is still a more convenient measurement than the ground truth error δ_G defined above. Because both the *color* and *resolution* of the original image, after printing and re-scan, do not remain the same. To match the composition result with the original image pixel-by-pixel, we will need an *image registration*. Any distortions in these three procedures (print, scan, and registration), aside from the composition algorithm itself, can significantly affect the computation of ground truth error.

We have compute the normalized matching error δ_M in all our composition experiments. The errors are shown in Table 2.2.

Table 2.2: Error table for our reassembly algorithm.

Models	# of fragments	Image size	δ_M
House	10	686 * 426	0.12%
Castle	10	649 * 424	0.10%
GMP	10	936 * 399	0.26%
Flower	26	1024 * 768	0.17%

Figure 2.13: The comparison with existing method. (a) The results computed by the method in [1], (b) The results computed by our method.

2.6.4 Comparison with Previous Method

We implement the algorithm of paper [1]. The side-by-side comparison is given in Figure 2.13. Figure 2.13(a) shows the results generated by the method in the paper [1] and Figure 2.13(b) shows the results computed by our algorithm. The results in our algorithm show visually better alignment between each pair of adjacent fragments than the previous method in [1]. Since there is no ground truth for the results, we adopt the matching error δ_M defined in Section 2.6.3 for the numerical comparison. The comparison of numerical matching error δ_M is shown in Table 2.3. This table demonstrates that our algorithm can generate more accurate reassembly results compared with the previous method.

Table 2.3: Comparison of the Reassembly Generated by [1] and by Our Algorithm. Numerical Matching Error δ_M is compared.

Models	# of Pieces	Image Resolution	δ_M in [Tsamoura et al.]	Our δ_M
House	10	686 * 426	0.53%	0.12%
Castle	10	649 * 424	0.20%	0.10%

(a)

(b)

(c)

Figure 2.14: The error estimation of our algorithm. (a) The ground truth image (b) The initial reassembly result before graph optimization, and $\delta_G = 0.25\%$, (c) Final reassembly result, and $\delta_G = 0.19\%$. We can see that the error is reduced about 20%.

2.7 Conclusion and Future Work

We present a novel computational pipeline for the automatic reassembly of fragmented images. It consists of three main steps: pairwise matching between two image fragments, graph-based global fragment reassembly, and refinement of the reassembly via graph optimization. In Step 1, we present a better curve-matching algorithm: each pair of image fragments is aligned via a pairwise matching integrating both geometry and color. In step 2, we present a reliable graph-based global search algorithm, which reassembles multiple pieces together to reach the overall composition. In step 3, we develop a novel graph optimization strategy on the previous reassembly result, which can effectively refine the position of the fragments globally. We evaluate our algorithm using various real-world images and demonstrate that it is effective and robust.

A limitation of this reassembly algorithm is that, like existing reassembly algorithms, if both the color and geometry of the fragment boundaries are similar, then the algorithm can not easily distinguish the correct match from the incorrect ones, and could fail to extract the correct composition even with global groupwise matching. We will explore better matching strategy for this in the near future. More effective graph searching and backtracking algorithms or some stochastic optimization strategies could improve the reassembly's robustness against local minima. In addition, higher level features/descriptors incorporating not only the boundary color but also interior textures may also help remove some boundary-matching ambiguity and hence facilitate the fragment matching. Interior texture or color information can be incorporated into our framework as a preprocessing step or as an extra term in both pairwise matching and groupwise matching.

3. CURVE NETWORK MATCHING FOR 3D FRAGMENT REASSEMBLY

We extend our 2D reassembly work to the 3D fragment reassembly problem. Geometric restoration that composes 3D fragmented pieces into the original complete object is a geometric problem that has direct applications in many scientific fields such as archaeological reconstruction, digital heritage archiving, and forensic evidence analysis, to name a few. For example, archaeologists reconstruct shards (e.g., ceramic fragments or sherds) into complete pots in order to analyze the information of an ancient society [39]; forensic anthropologists reassemble skull fragments into a complete skull for ancestry assessment and body identification [33]. For these types of applications, the need exists to solve a geometric reassembly of digitized thin-shell fragments of different shapes, sizes, and resolutions. In this project, we explore the application of 3D reassembly for forensic skull modeling. To assist with victim identification in law enforcement investigation cases, forensic anthropologists need to predict the skull's ancestry and create facial reconstruction/superimposition to help identity the body. The skeletal remains, however, are often incomplete or fragmented due to trauma or environmental exposure. Therefore, forensic specialists need to first perform a manual recomposition of skull fragments before subsequent assessment and analysis. With the advances in 3D scanning and 3D printing technologies, it is now possible to accurately digitize the fragments and examine the automation and augmentation of this skull reassembly procedure.

Considering the geometry of a fragment, its boundary surface consists of two types of regions: the *intact regions* and the *fracture regions*. The *intact regions* are those from the original boundary surface of the complete object before fracturing, and the *fracture regions* are those generated due to fracturing. Figure 3.1 (c,d) illustrate the intact regions and fracture regions on a fragment.

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

Figure 3.1: A Fragmented Chicken Model. The complete model (a) is broken into pieces (b). (c,d) On small fragments, intact regions (blue) and fracture regions (red and gray) may not be easily differentiable.

We can generally classify existing 3D fragment reassembly algorithms into two types: (1) reassembly based on fracture-region matching, and (2) reassembly using template guidance. Fracture-region matching approaches exploit similarities in the local fracture geometry of adjacent fragments [6, 29, 30]. The template guidance approaches compose fragmented pieces based on their best match to a complete model [33]. Each approach has advantages and limitations, and reassembly algorithms in both categories report difficulty in effectively processing small fragmented pieces. First, with small fragments, it is particularly challenging to differentiate and segment intact and fracture regions. Second, the number of uncertain potential matches tends to be large and this is non-trivial to process robustly.

We develop a new effective reassembly pipeline integrating both template-guidance and fracture-region matching. The idea is to use the information from both intact and frac-

ture regions to construct many potential matching relationships among the fragments and template; then, through a multi-piece matching optimization, prune and refine these possible matches to obtain globally consistent alignment of the fragments. This approach is formulated in a 3-step pipeline: (1) initial reassembly guided by a template; (2) pairwise fracture matching between fragments; (3) multi-piece matching integrating both intact and fracture information. The **main technical contributions** include (a) a reliable pairwise matching algorithm to align fragments with small overlapping regions, and (b) a multi-piece matching and refinement algorithm effectively integrating both template guidance and pairwise fragment matchings, which iteratively optimizes the positioning of fragments while consistently controlling accumulated error and avoiding penetrations.

3.1 Background and Related Work

3.1.1 3D Fragment Reassembly

Early research on data reassembly focused on solving the 2D jigsaw puzzle problem [40]. 3D objects are projected to 2D [19, 27, 28] and matched by their planar boundaries. In these approaches, planar boundaries are identified manually or detected through the difference of texture/geometry between the intact and fracture regions. These methods work for large thin shell fragments, on which the thickness of the fracture region can be ignored and fracture surfaces can be treated as curves. For 3D algorithms, Cooper et al. [29] assemble 3D pot fragments by matching the break-curves and shard normals. Willis and Cooper [30] improve pottery assembly by applying Bayesian analysis with a semi-automatic matching. Their experiments were shown most effective for axially symmetric and geometrically smooth/simple fragmented pots. Yin et al. [33] use templates to roughly assemble skull fragments, then

perform a break-curve analysis to refine the composition. This method also approximates thin shell fragments as surface patches, which is not suitable for small fragments [33]. For general 3D solid models, Papaioannou et al. [32] segment the fragments and extract fracture regions, then use depth maps for matching. Huang et al. [6] propose a 4-step framework for fragmented solid reassembly. Both of these methods require the segmentation of intact and fracture regions on the fragment, which is often difficult on small fragments. Herein, we develop an algorithm to incorporate intact and fracture geometry without explicit segmentation.

3.1.2 Geometric Partial Matching

3D partial matching is the key enabling tool for fragment reassembly. Partial matching is related to *geometric feature detection*, *feature description*, and *feature correspondence*. **Feature detection** identifies geometrically salient *keypoints*. [41] evaluates the performance of a few recent feature detectors suitable for partial matching [42–45]. Specifically related to our problem, desirable feature detectors should identify keypoints consistently under noise, small resolution variations. **Shape descriptors** are designed to evaluate the similarity between extracted keypoints. Descriptors are typically invariant under either rigid transformation [46, 47] or isometric transformations [48–50]. In this application, we focus on descriptors for rigid transformations. *Curvatures* [47] and *Integral Invariants* [51] are popular local descriptors in object recognition and surface matching. However, these descriptors are sensitive to the local geometric variance in handling template-subject disparity in fragment reassembly. Histogram-based descriptors, such as *spin images* [46] and *shape context* [52] compress geometric structures into bins, hence are more globally discriminate

and less sensitive to local geometric variance. [53] improves them by using a unique local reference frame. This descriptor performs well on partial surface matching. **Feature correspondence refinement** is a procedure to establish a bijective map between features on different shapes. Widely used algorithms for this task include *spectral matching* [54], *graph matching* [55], *voting* [56], *RANSAC* [57], and *forward search* [6].

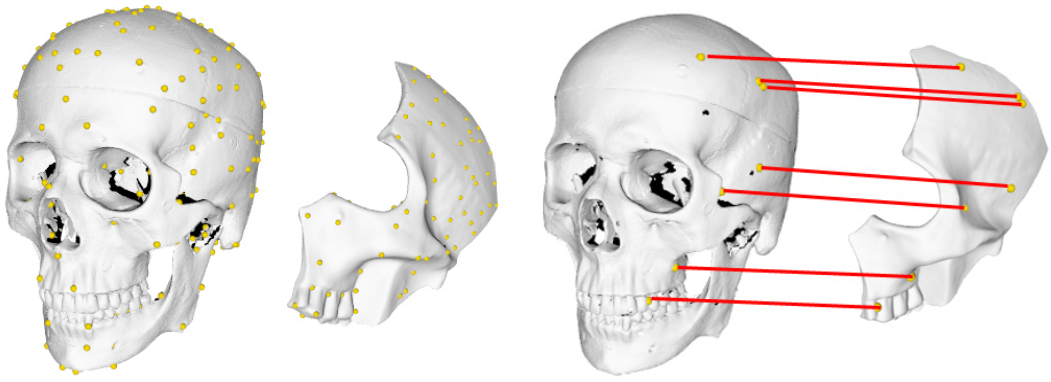
3.2 Template Matching

In some reassembly tasks, complete models with similar geometry to the subject data are available and can be used to guide the reassembly. For example, in forensic skull restoration from fragmented pieces [33], existing skull models can provide useful guidance. In tasks such as building a 3D repository database, models archived in the same category are also useful. Following a template model M, we can reassemble fragmented pieces after solving a partial matching between each fragment F_i with M. A successful partial matching should align the intact region of F_i with M, despite certain geometric disparity between the subject and template. Our template matching algorithm has two steps: (1) feature extraction and initial correlation; (2) correspondence refinement. The output is a list of ranked 3D transformations T_i^j that align F_i with M.

Feature extraction and initial correlation. According to the comparison in the recent survey [41], among popular 3D feature detectors, the *Intrinsic Shape Signatures* (ISS) [42] offers great repeatability and efficiency in rigid partial matching. The ISS constructs the covariance matrix of the support region of each 3D point and whose largest eigenvalues differs most with its second largest eigenvalues are identified as features. We extract features using ISS on F_i and M, respectively. Fig. 3.2(a) illustrates an example of extracted features.

To accommodate geometric disparity between the template and subject, a shape descriptor that is not too sensitive to local geometric variance is preferred. The histogram-based descriptor, such as *spin images* [46] and *shape context* [52], can stably reflect geometric variance. The Signature of Histograms of OrienTations (SHOT) [53] is another effective histogram-based descriptor. SHOT first constructs a unique and unambiguous local reference frame, then calculates the descriptor by compressing the geometric information into bins along these three axes. This descriptor outperforms the spin image in partial matching applications such as object recognition and 3D multi-view reconstruction. Therefore, we use SHOT to describe and compare feature points on M and on F_i.

For each feature p on M, its SHOT descriptor, denoted as $S(p)$, is a 352-dimensional vector, and its top k most similar features on F_i are extracted. Specifically, $p \in M$ and $q \in F_i$ is considered as potential corresponding pair if (i) $D(p,q) = |S(p) - S(q)| < \delta_S$ and (ii) $D(p,q)$ is one of the k smallest $D(p,\cdot)$ values that satisfies (i). Then (p,q) is kept and added into an initial *correspondence set* $\bar{\mathcal{C}}$ ($\delta_S = 0.05$ and $k = 5$ in all our experiments).



(a) (b)

Figure 3.2: Feature Extraction and Matching. (a) ISS keypoints on the template and a fragment, (b) refined feature correspondences.

Correspondence Refinement. To ensure enough number of features are extracted on smaller fragments, M usually contains many keypoints. This results in two challenging issues in getting bijective feature correspondence (1) The size of $\bar{\mathcal{C}}$ is big, because features from other irrelevant regions on M (that does not correspond with F_i) could also contribute k pairs to $\bar{\mathcal{C}}$. (2) The correct correspondence pairs contains only a small portion of $\bar{\mathcal{C}}$. Due to the big size of $\bar{\mathcal{C}}$, powerful correspondence refinement algorithms such as graph matching [55], which needs to construct a huge affinity matrix and then solve an integer quadratic programming, turn out to be too computationally expensive for our pipeline. Instead, we adopt the more efficient RANSAC algorithm [57] to extract the geometrically consistent correspondence pairs in $\bar{\mathcal{C}}$. Geometric consistency is used to define outliers of the RANSAC algorithm. In this problem, given two correspondence pairs $c_1 = (p_1, q_1), c_2 = (p_2, q_2) \in \bar{\mathcal{C}}$, c_1 and c_2 are geometrically consistent if the Euclidean distance between p_1 and p_2 is similar to the distance between q_1 and q_2, namely, $|\|p_1p_2\| - \|q_1q_2\|| < \delta_t$. δ_t can be chosen according to similarity between the template and subject ($\delta_t = 5$mm in our experiments).

In the RANSAC algorithm, for each extracted transformation model, if the ratio of size of inlier set to size of $\bar{\mathcal{C}}$ is bigger than a threshold δ_r, this transformation is accepted. To estimate a suitable threshold δ_r, we observe that it is mainly affected by (1) the size of F_i and (2) k value in the initial correspondence extraction. We set $\delta_r = \lambda \frac{V(F_i)}{k \times V(M)}$, where $V(F_i), V(M)$ are volumes of the fragment and the template approximated by their bounding box volumes, and λ is a weighting factor. $\frac{V(F_i)}{V(M)}$ roughly estimates the ratio of features from M that have corresponding features in F_i, among which only $1/k$ is correct (since top k matches are preserved). We conservatively use $\lambda = 0.5$ to set δ_r in all our experiments. Fig. 3.2 (b) shows an accepted correspondence between a skull fragment and a template.

3.3 Pairwise Fracture Region Matching

Matching with template (that is a different object) usually only roughly positions large fragments. And sometimes, no template is even available. Therefore, effective matching of shared fracture regions of adjacent fragments is a critical component to successful reassembly. In this section we aim to find a set of potential alignments between each pair of fragments. Such an alignment between fragment F_i and F_j is referred to as a *relative transformation* T_{ij}.

The pairwise matching is more challenging than the template matching. Because for thin-shell objects, the actual fracture regions are significantly smaller than intact regions, and these fracture regions often possess little geometric saliency. Very few effective feature points on fracture regions can be extracted to support reliable matching between fragments. Therefore, rather than using feature points, matching based on *boundary curves* [28, 30] or *feature regions* [6] are often used for aligning adjacent fragments. The *region-based matching* extracts salient geometric areas on fracture regions to compute alignments between fragment pieces. But thin-shell objects often have simple and flat fracture regions, lacking salient region to help matching computation. Instead, sharp ridge and valley curves are more salient features for effective partial matching. Therefore, we solve pairwise alignments by matching feature curves extracted on fragments.

3.3.1 Feature Curve Extraction

We extract feature curves on fragments following the algorithm of [58] due to its robustness against noise. First, extract points with high mean curvature. Then build a minimum spanning tree (MST) to connect all these points into a curve network. The weights of edges

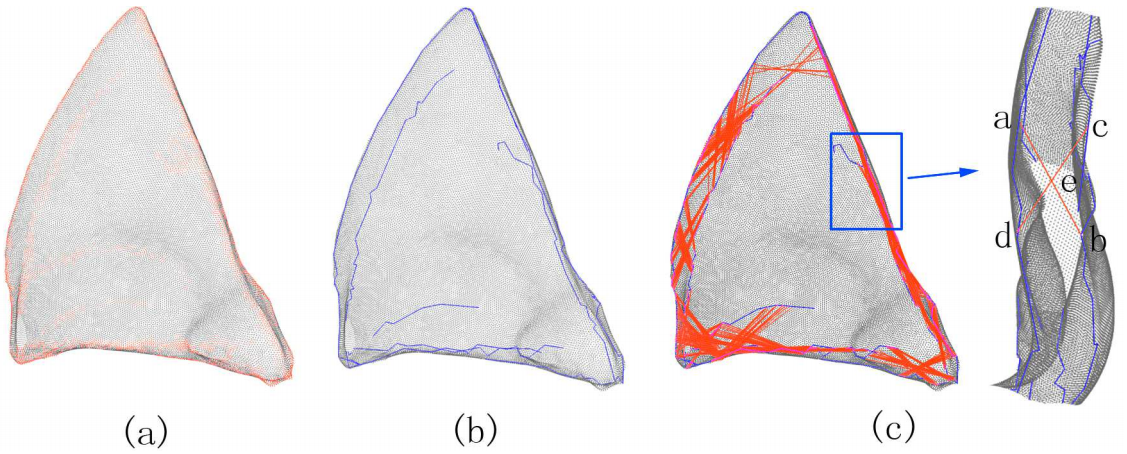


Figure 3.3: Feature Curve Matching. (a) Extracted feature segments (orange) on a fragment, (b) pruned feature network, (c) sampled wide bases (red crosses).

are designed [58] to make MST passing through sharp regions as much as possible. The MST contains many short branches, making the matching of curve network difficult. Then a bottom-up graph pruning is used to eliminate these short branches on the MST tree: (1) sort all leaves of the tree based on their depth; (2) determine the longest path by traversing this tree; (3) remove short branches from the tree. Iterate this pruning until no short branch is left on the tree. Then the final feature curve network, consisting of the remaining salient feature curve paths, is extracted. Note that this algorithm does not rely on the initial root of MST and can remove noisy short feature line segments reliably. Fig. 3.3(a,b) illustrate an example of feature curve extraction.

3.3.2 Feature Curve Matching

In existing fragment reassembly work, a few curve matching algorithms [18,19,27,28,30] have been developed to match adjacent fragments. However, in these curve matching formulations, each fragment's contour is extracted as a simple curve loop, and the matching is performed to find certain "longest common substrings" shared by two curve loops. They assume the

fragment can be scanned in a way that only intact surfaces are obtained. Unfortunately, such a scan and the clear separation between intact and fracture regions are too difficult when fragments are small. Following our feature extraction, the curve networks obtained from fragments are way more complicated than a curve loop, and their alignment needs more sophisticated matching strategy.

To match two feature curve networks C_1 and C_2 reduces to finding a rigid transformation T that $T(C_1)$ and C_2 have the largest overlap. Inspired by [59], we formulate this problem as a Largest Common Point-set (LCP) problem: given two point sets X and Y, the LCP under δ_D-congruence finds a transformation T and a subset $P = \{p_i\} \subset X$ with the largest cardinality, such that $Dist(T(p_i), Y) < \delta_D$, where $Dist(T(p_i), Y)$ measures the distance from transformed point p_i to point set Y, given a distance threshold δ_D. We also call the cardinality of P the *LCP score*.

We propose a voting algorithm to solve the LCP problem. (1) Extract *4-point congruent sets* (Section 3.3.2) on C_1 and C_2, compute their matchings. (2) Conduct a voting and get the transformations that are top k favorably voted (Section 3.3.2). After the top-k LCP transformations are obtained, we refine each transformation using the iterative closed points (ICP) [60] algorithm and discard alignments that lead to fragment penetration. The penetration is checked by the collision detection package named SWIFT++ [61]. Note that, unlike [59] which conducts a RANSAC for randomly generated points on surfaces, we conduct a thorough extraction of 4-point congruent sets on pruned feature curves. Such a thorough extraction on well pruned samples makes our curve matching much more robust than [59] when there is a high percentage of outliers, which always happens in pairwise fragment matching.

4-Points Wide Base Extraction and Matching

Inspired by [59], we match curve networks using a modified *4 points congruent set* (4PCS) algorithm. A *4-points wide base* is a set of 4 points that are coplanar. The coplanar 4 points set is wide if the length between each two points is larger than a threshold. A *wide base* created by selecting points that are far from each other results in more stable alignments [62]. However, if the points are too far away, the selected points will not all lie in the overlap regions. Therefore, with two controlling parameters d_{min}, d_{max}, we first randomly pick 3 points such that the distance between each two points is in the range of $[d_{min}, d_{max}]$, then select the 4th point that is in the same distance range and is coplanar with these three points. d_{min} and d_{max} are chosen according to the estimated thickness d_t of the fragments: $d_{min} = 0.5 * d_t$ and $d_{max} = 2 * d_t$. Fig. 3.3(c) shows an example of the 4-points wide bases on the curve network.

Given a 4-points wide base $X = \{a, b, c, d\}$, let ab and cd be the two lines intersecting at an intermediate point e. We can build a 5-dimensional descriptor vector $v = \{l_1, l_2, \theta, r_1, r_2\}^T$, where $l_1 = \|ab\|$ and $l_2 = \|cd\|$ are the lengths of segments ab and cd, θ is the angle between them, and r_1 and r_2 are computed as $r_1 = \|ae\|/\|ab\|$, $r_2 = \|ce\|/\|cd\|$. Two 4-points wide bases are congruent if their descriptors are similar, and matching them introduces a rigid transformation by a least square fit.

Voting and Evaluation of Alignments

We first compute the sets of 4-point wide bases S_1 and S_2 for feature curve networks C_1 and C_2. Then for any wide base $B_i \in S_1$, find its congruent wide base $B_j \in S_2$ and compute the transformation between B_i and B_j. The number of transformations introduced by congruent

wide base is large, and directly evaluating all their effects is computationally expensive. Therefore, based on the observation that valid transformations are usually supported by many mutually congruent corresponding base pairs, we do a voting to pre-select potential transformations. First, a transformation inferred by two 4-point wide bases is transformed into a 6D vector (ax, ay, az, x, y, z), where ax, ay, az are rotation (Euler) angles and x, y, z are translations. Then the 6D parameter spaces are divided into buckets. Finally, the k buckets with most vectors are selected as the transformation. The total time complexity of this algorithm is $O(nlogn)$. Each pre-selected transformation receives a certain number of votes, but it does not necessarily have high *LCP scores*. Therefore, to re-rank the transformations following the *LCP scores*, for each pre-selected transformation T_i, we compute $T_i(C_1)$ and count points in $T_i(C_1)$ within δ_D-distance to C_2. We use Approximate Nearest Neighbor (ANN) [63] for efficient neighborhood query. Then, the transformations are re-ranked, and the top K_p (in our experiments, $K_p = 200$) transformations are kept.

3.4 Multi-piece Matching

The template matching (Section 3.2) suggests a set of possible alignments between the fragments and template; and the pairwise matching (Section 3.3) suggests a set of possible alignments among the fragment pairs. Now we perform a graph-based search algorithm to extract globally coherent pairwise matchings from the above possible alignments.

3.4.1 Reassembly Graph Representation

All the fragments, the template (if used), together with all the extracted potential correspondences are encoded in a *reassembly graph* $G = (V, E)$. Each node $n_i \in V$, denoting a

fragment F_i, is associated with a transformation matrix X_i that needs to be solved, to get its final position, $X_i(F_i)$, in the reassembly. X_i is a 3×4 matrix $[R|t]$ where R is a 3×3 rotation matrix and t indicates a 3D translation. The template model M is denoted as a base node $n_0 \in V$ in G. To get a unique solution, we set template's transformation matrix X_0 to be $[I_3|0_3]$, where I_3 is the 3×3 identity matrix and 0_3 is the 3D zero vector.

A directed edge $e^k_{i,j} \in E$ from node n_i to n_j denotes an alignment from fragment F_i to F_j. This alignment is represented by a relative transformation $T^k_{i,j}$ on each edge $e^k_{i,j}$. Note that between a pair of nodes n_i and n_j, multiple edges ($k = 1, 2, \ldots$) may exist, so G is not a simple graph. These relative transformations $T^k_{i,j}$ are pairwise alignments computed in the previous steps. The matches from each fragment F_i to the template M are also represented as edges $e^k_{i,0}$ and relative transformations $T^k_{i,0}$.

To simplify the formulation in the following, given a directed edge $e^k_{i,j}$, we can also consider it in its opposite direction, as $e^k_{j,i}$, from n_j to n_i. Accordingly, the associated alignment is naturally defined as $T^k_{j,i} = (T^k_{i,j})^{-1}$. With this we can study G as an undirected graph. For example, we can say edges $e^1_{1,2}$ and $e^2_{1,2}$ form a loop, without specifically modifying the order of the subscripts.

3.4.2 Matching Objective Function and Constraints

We define a few matching scores to measure how well fracture or intact regions overlap. Two regions are considered *overlapped* if their Hausdorff distance is smaller than a threshold ξ_h (based on the precision of our 3D scanner, in all our experiments $\xi_h = 3$mm). Given two nodes n_i and n_j and their associated transformation X_i and X_j, we define a **fracture matching** score $S_f(X_i, X_j)$ as the area of the overlapping regions between these two fragments after

applying X_i and X_j. Similarly, we define an **intact matching** score $S_i(X_i, X_0)$ as the area of overlapping regions between $X_i(F_i)$ and the template M. Naturally, $S_i(X_i, X_j) = 0$ between fragment nodes for $i, j \neq 0$, and $S_f(X_i, X_0) = 0$ between each fragment node and the template node. Then, combining them together, between each pair of nodes n_i and n_j, we define a total **region matching** score $S_r(X_i, X_j) = S_f(X_i, X_j) + \alpha S_i(X_i, X_j)$, where α is a weighting factor that can be adjusted according to the availability of a good template. By adding template node n_0 and the intact matching term, our algorithm can make good use of available template to guide multi-piece matching. α and the available template affect the reassembly results. Ideally, when template is similar to the ground truth, α should be set big, while when no good template is available, α should decrease. When a wrong template is used, the reassembly will be bad if α is big. Setting $\alpha = 0$ reduces the reassembly to only relying on fracture region matching of adjacent pieces. By default, we set $\alpha = 0.1$, allowing the pairwise fragment matching to play a more important role.

Finally, the multi-piece matching problem reduces to solving an optimal subgraph $G' = (V', E'), V' \subseteq V, E' \subset E$, and a set of transformations $\{X_i | n_i \in V'\}$ maximizing the following accumulated region matching score $\Phi(G')$,

$$\Phi(G') = \sum_{n_i, n_j \in V'} S_r(X_i, X_j), \tag{3.1}$$

subject to the following *validity constraints*.

Validity Constraints. A subgraph $G' = (V', E')$ and associated transformations infer a valid reassembly if

1) G' is a *simple graph*. At most one edge can exist between two nodes, hence a unique alignment is selected between these two fragments.

2) *Loop closure* constraint: for any loop in G', composing the relative transformations along this loop should yield an identity transformation.

3) *Non-intersection* constraint: reassembled fragments should not spatially penetrate each other.

It is not hard to verify that a graph that satisfies the *loop closure* constraint must be a *simple graph*. Because two different transformations between a fragment pair automatically infer a non-identity loop between these two nodes.

3.4.3 Optimizing Multi-piece Matching

We develop an iterative algorithm to optimize the multi-piece matching. First, initialize an empty graph G'. Then, iteratively grow it by adding a new node with its incident edges from G, maximizing $\Phi(G')$ subject to the validity constraints. During the growing of G', we simultaneously refine all the alignments added in G'. This algorithm is formulated as follows.

In: $G = (V, E)$ and $\{T_{ij}^k\}$;

Out: $G' = (V', E'), V' \subseteq V, E' \subset E$, and X_i for $n_i \in V'$.

1. $G' = \emptyset$; add n_0 to G' and let X_0=$[I_3|0_3]$;
2. Find an edge $e_{i,j}^k \in E$ where $n_i \in V', n_j \in V \backslash V'$, such that adding $e_{i,j}^k$ to G' does not violate *validity constraints* and leads to maximal increase of $\Phi(G')$. Add this $e_{i,j}^k$ to E' and add n_j to V'; set $X_j = (T_{i,j}^k)^{-1} * X_i$.
3. Perform a graph optimization refinement on G';
4. If $V' = V$ or no further edge $e \in E$ is added into G', STOP; otherwise, GOTO Step 2.

Evolving G' through Beam Search. In the above Step 2, we add a new edge whose insertion results in a valid new G' with largest increase of F. G' is called the evolution of G. This greedy approach may easily get stuck in local minima. In this problem, between fragment pairs, multiple $T^k_{i,j}$ exist and the actually correct alignment may not be highest ranked. To more robustly circumvent such local minima, we implement a beam search strategy [64]. In each iteration, instead of choosing the single best alignment, we explore m highest-scored $e^k_{i,j}$. First, G' evolves into m different graphs. On each of these graphs, expand its n best subsequent evolutions, which will result in totally $n \cdot m$ possible graphs. Among these graphs, the m highest-scored graphs are again selected and preserved. This process repeats until no new edge can be added into G'. Through beam search, G' has a better chance to circumvent small local minima; meanwhile, this avoids the exponential time complexity and memory requirements in searching for a global optimum through complete enumeration.

3.4.4 Multi-piece Reassembly Refinement

Transformations X_i are computed through the composition of relative transformations during the growth of G', small errors can accumulate and affect the subsequent X_i computation and penetration test. To minimize this error accumulation, we apply a graph optimization on G' to globally refine existing transformations. We minimize the following least square function defined on G',

$$\phi(\mathbf{X}) = \sum_{e_{i,j} \in E'} ||T^k_{i,j} - X_i \times X_j^{-1}||^2 \times S_r(X_i, X_j), \tag{3.2}$$

where $\mathbf{X} = \{X_1, \ldots, X_{|V'|}\}$. Node transformation X_i should be coherent with the selected relative transformations $T_{i,j}$ on its incident edges. X_i is first computed by one $T_{i,j}$ by $X_i =$

$T_{i,j} * X_j$ when n_i was added to G'. But a later added node n_r may introduce another edge $e_{i,r}$ and associated $T_{i,r}$. This consistency $X_i = T_{i,r}^h * X_r$ may not exactly be satisfied due to accumulated noise or numerical errors. Therefore, we globally refine $\mathbf{x}$ following Eq. (3.2) on G'. We implement this optimization following the algorithm of [38]. With analytic derivative information available, this optimization converges very efficiently.

3.5 Experimental Results

Experiment Setting. We evaluate our reassembly algorithm on various ceramic models and real forensic skull data. For ceramic models, we first scan the complete models as ground truth for result evaluation. Then break each object into fragments and scan them individually. Our algorithm reassembles the digital fragments and we compare our result with the ground truth. We also use our algorithm to compose real fragmented skull data provided by forensic specialists. In practical law investigation cases, forensic specialists restore the skull geometry from fragments manually, then perform subsequent ancestry analysis, facial reconstruction for body identification. Our digital restoration results on these skulls are visualized. All the 3D objects and fragments are digitized using a NextEngine scanner (with reported accuracy $\pm 0.3mm$). We also estimate and document our scanning accuracy in these experiments by performing multiple scans on each model and computing Hausdorff distances between different scans.

Evaluation of Reassembly Error. To numerically evaluate reassembly accuracy, we define two error metrics: the *Reference Error* ε_R and *Matching Error* ε_M. The completed model $S = \bigcup X_i(F_i)$ (accordingly, also fragments) is scaled to a unit box for consistent error evaluation. If the ground truth model $\hat{S}$ is available, we can measure the **Reference Error**

ε_R by comparing S and $\hat{S}$: compute a displacement field $\eta_r(x)$ on $\hat{S}$ to record the distance from each point $x \in \hat{S}$ to its nearest point on S, then use the average, $\varepsilon_R = \sqrt{\frac{\sum_{v\in\hat{S}} f^2(v)}{|\hat{S}|}}$, where $|\hat{S}|$ represents the number of points of $\hat{S}$. When ground truth is not available, we can use a **Matching Error** ε_M to estimate how well the adjacent fragments align with each other after the reassembly. For each point p on the $X_i(F_i)$, if its nearest corresponding point q on an adjacent fragment $X_j(F_j)$ has an opposite normal direction from p's normal, then p and q are paired, and $d(p) = \|pq\|$. Then the average distance of paired points is calculated $\varepsilon_M = \sqrt{\frac{\sum_{p\in X_i(F_i)} d(p)}{|P|}}$, where $|P|$ represents the number of such pairs. Note that, the normal constraint in pairing is used to avoid measuring points from intact regions.

3.5.1 Reassembly Results

Our reassembly program was run on a laptop with 2.0GHz Core i7 CPU and 6GB RAM. Table 3.1 shows the running time on our examples. The total computation needs from 4 to 14 minutes, depending on the number of fragments and their geometry. The pairwise matching to collect potential alignment between fragments is the most time-consuming step in this reassembly pipeline.

Table 3.1: Runtime Table: #V is the total vertex number of each fragmented model, #F is the number of fragments, T_t, T_p, and T_m are computational times (in seconds) for template-guided reassembly, pairwise matching, and multi-piece matching, respectively.

Models	# V	# F	T_t	T_p	T_m
Child	1066 k	8	123	89	31
Chicken	882 k	11	183	244	37
Buddha Head	1152 k	12	–	142	43
Skull 1	1294 k	19	146	614	64
Skull 2	427 k	10	93	141	20

We first test a simpler scenario on ceramic objects, using the ground truth model as the template. Fig. 3.4 illustrates reassembly of the fragmented Child Buddha model and chicken

Figure 3.4: Reassembling fragmented Child Buddha model and Chicken model.

model. The physical dimensions of the smallest pieces here are about $3cm \times 2cm \times 0.4cm$. Note that the scan resolution is about $0.3mm$, and the final matching error ε_M about $0.8mm$. These experiments show that small fragments are reassembled effectively and accurately.

We also examine our algorithm when only *template with salient geometric difference* is available. Fig. 3.5 shows a reassembly example of a gnome model. The broken gnome model (a) is quite different from the available template gnome (b). When a big template-guidance weight $\alpha = 1$ is used, the template (b) more strongly affects the composition and leads to the result (c). To better align the shoes (red box in (c)), there is a gap in the final reassembly (see region blue box). When a small $\alpha = 0.01$ is used, the reassembly, after initial positioning, more relies on inter-fragment alignment, and actually leads to a more accurate

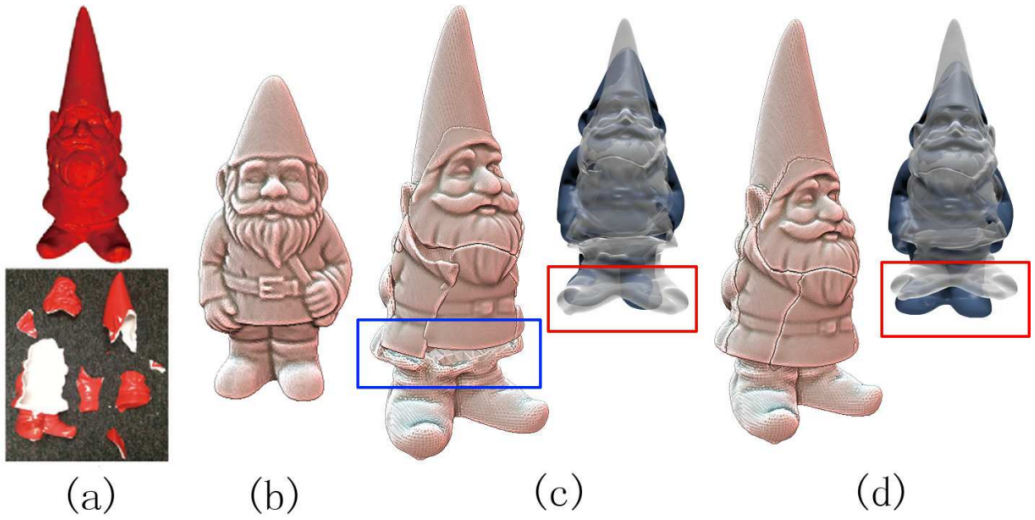


Figure 3.5: (a) The ground truth model and fragments, (b) the gnome template, (c) a reassembly result ($\alpha = 1$), (d) another reassembly result ($\alpha = 0.01$).

result (d). Human skulls possess geometric similarity. Hence, complete human skulls are suitable templates in fragmented skull reassembly. We collected a group of standard skull models from all the major ancestries: {male, female} $\times$ {white, black, Asian} skulls, using them in guiding skull reassembly. Fig. 3.6 shows our reassembly on a ceramic skull model. The model was broken into 19 pieces. We randomly used a white male skull model as a template to perform the reassemblies. Despite the geometric disparity between the template and the ceramic models (see the accompanying video for details), the fragmented skull was successfully reassembled, showing that our reassembly is robust against template-fragment disparity. Fig. 3.7 shows our reassembly on a real fragmented skull from forensic anthropologists in which the victim's skull is severely damaged due to a gun shot. The two sides of the skull are highly fragmented and some are missing. Its manual restoration becomes challenging for two reasons. First, on some small pieces the fracture and intact regions are hard to distinguish. Second, the fracture regions of the fragments are very thin and hard

Figure 3.6: Reassembling a 19-piece fragmented skull model.

to align. By effectively integrating the guidance from a template skull model and fracture region matching, our algorithm reassembled this damaged skull satisfactorily.

The numerical reassembly error on different models is documented in Table 3.2. The error distribution is also illustrated in our accompanying video. From the table we can see the algorithm is effective, and in particular, can handle small fragments that are not processed well by existing algorithms such as [6, 33].

3.5.2 Comparison With Existing Methods

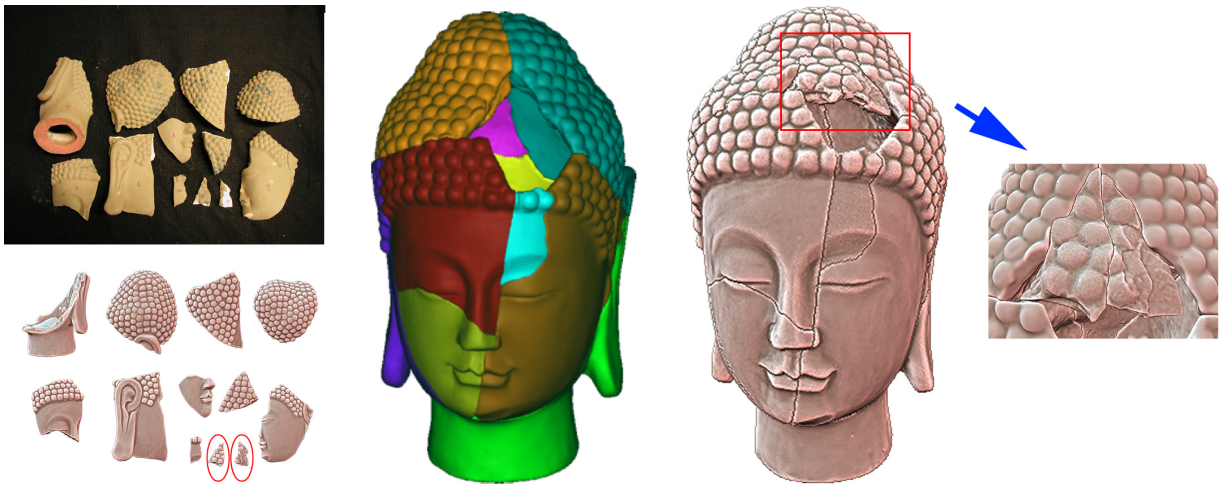
Currently there is no public benchmark dataset online to evaluate the reassembly algorithms. We will publish our data for future comparative study. Here we analyze and compare our algorithm with the related existing methods. Algorithms that solve only pairwise matching, such as [32], cannot not effectively handle ambiguity in pairwise alignment. Algorithms that

Figure 3.7: Reassembling a fragmented skull obtained in a real law investigation case.

treat fragments as surface patches, such as [28, 30, 33, 65], require that only intact surfaces are scanned and fragment boundaries are simple curve loops. This is practically difficult when fragments are relatively small. Algorithms such as [6] require explicit segmentation between intact and fracture regions, which is often difficult for small fragments; also, this method requires fracture regions to be big and possess salient geometric variance to support reliable matching. Hence it is more suitable to handle solid rather than thin-shell objects. Our algorithm effectively tackles all the above three unsolved challenging issues in thin-shell object reassembly. In Fig. 3.8, the Buddha head model from [6] was reported to be a challenging example due to the existence of small fragments. As shown in (b), The algorithm of [6] fails to reassemble the two small pieces (marked in red circles in (a)). In contrast, our algorithm successfully reassembles all the fragments including these small pieces, as shown in (c). Since no template is available, $\alpha = 0$ and no template guidance is used.

Table 3.2: The Reference Error and Matching Error on different models. $\#F$ is the number of fragments. ε_R and ε_M are the absolute reference error and the absolute matching error (in millimeter). δ_R and δ_M are the reference and the matching error ratios. The error ratio is measured as a percentage of the bounding box diagonal length (i.e., average error / diagonal length $\times 100\%$). Intuitively, for a model with the bounding box diagonal is 1 decimeter, 1% error is 1 millimeter. Some reference errors are not available due to the lack of ground truth model.

Model	$\#F$	ε_R	δ_R	ε_M	δ_M
Child Buddha	8	0.853	0.33%	0.118	0.42%
Chicken	11	0.820	0.27%	0.633	0.21%
Gnome	8	0.575	0.15%	0.697	0.19%
Head	12	n/a	n/a	0.760	0.13%
Skull 1	19	0.435	0.16%	0.595	0.23%
Skull 2	10	n/a	n/a	2.688	0.83%



(a) (b) (c)

Figure 3.8: Reassembling the Buddha Head. (b) The algorithm of [6] fails to reassemble small pieces (in red circles in (a)). (c) Our algorithm correctly reassemble all the small pieces.

3.6 Conclusion and Future Work

We present a 3D reassembly algorithm for thin-shell fragmented objects. We developed new partial matching algorithms for effective template matching, cross fragment matching, and multi-piece alignment. Our algorithms are specifically suitable for these matching problems where the overlap regions on fragments are small and lacking geometric saliency. Our strategy extracts plenty of potential alignments, then utilizes both template guidance (if available) and inter multi-fragment matchings to prune and refine these alignments and obtain final reassembly. Our method is demonstrated effective and robust in restoring various fragmented ceramic objects and forensic skull models. It works especially well when small pieces exist and tedious intact-fracture region segmentation preprocessing is unavailable.

Limitations and Future Work. If fragments undergo erosion or deformation, their reassembly becomes more difficult. Utilizing other information such as texture or color may help. Also, our algorithm currently uses only one template. Incorporating information from multiple template sources may further improve the reassembly performance.

4. SURFACE MATCHING AND SSM FOR DATA COMPLETION

The digitization of physical objects often results in missing regions on the geometric models, due to scanning occlusions, calibration errors, noise of hardware, or damage on the physical objects themselves. Repairing these missing regions (holes) to get a complete digital model is desirable for various modeling and analysis purposes. This data completion problem is a long-standing and fundamentally challenging geometric modeling problem. For holes that are small or located in simple/smooth geometric regions, filling them using a smooth patch would result in satisfactory results. However, things become more challenging when the holes are big or located on regions with complex geometry or with rich details. Simple smoothing won't produce accurate enough repair in practical modeling tasks.

Using certain prior knowledge on geometry or semantics of the object to repair, one can perform more complex completions. Such kind of algorithms are often referred to as *context-based completion*. Two representative strategies are symmetry-guided repair and template-guided repair. Based on its symmetry, an object can be repaired using other regions on itself. This requires the object to be (at least local) symmetric and the missing region has a repeated region that remains complete. The template-guided repair can be used more generally, by transplanting corresponding regions from a template model to patch the missing holes. However, the result depends on the template model, and finding a desirable (e.g., with good geometric resemblance) template is non-trivial and sometimes not practically possible.

Statistical shape analysis is a tool to do geometrical analysis on a set of shapes. The statistical properties of all these shapes are modeled to encode the similarity and difference among all the available data. It was first developed by Cootes et al. [66], and named as *Statistical Shape Models (SSM)*. It uses a point distribution model to represent the geometric shapes. This point distribution models the mean geometry of a set of shapes and geometric variation among these shapes. SSM has been successfully used in many applications including shape segmentation, object recognition, shape compression, and geometric repair [67].

Different from other context-based methods that reconstruct the missing regions based on one shape (either a template or the model itself), SSM can utilize all the existing data to compose a statistically reasonable guess to the damaged object based on this object's other existing regions. For example, in our projects, we have a database containing many human bodies with different heights and obesities. Then, given a new incomplete digital human body scan, the missing geometry can be repaired based on its "body profile" extracted from other regions, such a profile suggests this object's height and obesity percentile among all available human data, which then is used to compose the repair. Therefore, SSM can often produce more desirable completion result when no exact template model is available.

However, a key assumption for statistical shape modeling is that *all the data (including the object to repair) should follow the same statistical distribution*, so that the fitted statistical shape model can be used to predict the missing regions. This, as we will elaborate in the following, may not always be true. There are two main issues in many current algorithms that directly apply SSM for geometric data analysis. First, data may not globally follow a single distribution. For example, different poses/motions in the digitization of articulated bodies such as humans or animals (Fig. 4.1), and different expressions in the digitization of human

faces, all significantly affect each sampling point's coordinates. Second, the most commonly used statistical model is the Gaussian distribution. But in many practical scenarios, the distribution follows more complicated distributions than a simple normal distribution.

Based on the above observations, our basic idea is that, first, before applying any SSM analysis, the data should be examined and appropriately pre-processed. Specifically here in our human body completion task, we suggest to partition the data into subparts, so that each of which will follow a uniform statistical distribution. Second, the local distribution should be estimated so that suitable distribution models will be used to describe the data. We demonstrate that in the practical human body completion task that we are working on, a T-distribution can better describe the data variance, and thus produce better reconstruction. The *main contributions* and novelty of this work is that we suggest a new SSM-based completion framework. The geometric objects should be adaptively partitioned into subregions that can be locally modeled by a statistical distribution. A local statistical shape model based on T-distribution is then derived for data completion. We therefore call our new completion algorithm as a *Local T-distribution based Statistical Shape Model* (LTSSM). Compared with the completion using the conventional global Gaussian-based SSM model, our algorithm demonstrates clear advantages and improvements in repairing articulated geometric models.

4.1 Related Work

4.1.1 Geometric Data Completion

Geometric completion algorithms can be generally classified into two types: (1) methods repairing missing regions without using context knowledge (*context-insensitive* approaches),

and (2) methods that are data context driven (*context-sensitive* approaches).

Context-insensitive geometric completions are either *surface-based* or *volume-based*. *Surface-based* methods construct the hole-filling surface patches by maximizing smoothness or minimizing curvature variance near the missing regions. Some algorithms first topologically delimit the holes, then refine the patching geometry through smoothing or remeshing operations [68] [69] [70] [71]. Some others fill the missing region by directly applying geometric data fitting [7, 72, 73] using moving least squares, radial basis functions, or other techniques. The surface-based methods are often simple, efficient, and can produce nice repair when the holes are small or locate at regions without complicated geometric details. However, they are often insufficient in recovering the geometric details that are often important in data analysis. *Volume-based* methods consider the object as a solid model [74–78]. Such approaches often use a signed distance function (defined on volumetric grids) to describe the embedding of the object in $\mathbb{R}^3$. The final complete mesh is extracted from the zero-level set of the signed distance function. Volume-based methods are often robust in handling holes that are big or have complex (e.g. non-convex) boundary geometry. However, they are often relatively slow and still cannot reproduce subtle geometric details.

The *context-sensitive* methods repair the model using available prior knowledge of the data. For example, shape symmetry is a useful context information in repairing an object using another region of the object itself [9, 79, 80]. These methods can successfully recover the geometric details if the missing regions of a repeated pattern are available and the symmetry can be detected correctly. However, if the object is not symmetric or both the corresponding regions are damaged, then this approach fails. Another group of approaches repairs the holes by transferring the corresponding regions from a template model that shares similar

geometry. These methods [8] [81] [82] [83] need to compute a map between the incomplete object and template model, then use this correspondence to identify and deform a patch from the template to fill the hole. These template-based methods are generally robust in dealing with various holes and are capable to recover details using the template. A key limitation of these methods is that the selection of template model usually affects the completion result and suitable template model is not always available or could be automatically found. In fact, template is often from a set of data where the object to repair does not belong. Therefore, no predetermined template can perfectly reproduce the right geometry of the target object. Another recent type of context-sensitive completion algorithm is based on the statistical shape model (SSM) [84]. Utilizing multiple templates and their distribution, the SSM has been shown desirable in robustly producing adaptive results on repairing categorized datasets such as human faces [67]. More about SSM is discussed in the following section. An extensive discussion of data completion techniques can be found in the surveys [85] and [86].

4.1.2 Statistical Shape Models

The best known methods of Statistical Shape Models are the Active Shape models [66] and Active Appearance models [87] proposed by Cootes et al. All the points' coordinates on the models are concatenated into one vector P and the dimensionality of the variation of the vectors is reduced by Principal Component Analysis (PCA). Then any shape can be approximately described by a linear combination of the first c eigenvectors computed by PCA: $P = Wx + \mu$. The authors [88] propose probabilistic Principal Component Analysis (PPCA) based on the Gaussian distribution. They build a latent variable model for PCA as $P = Wx + \mu + \varepsilon$. With this additional residual variance ϵ, PPCA can more accurately model

the dataset and also can be used to estimate the missing data. However, PPCA requires a relatively big training set to make the SSM accurate. Therefore, recent research in SSM is mainly along two directions (see a survey paper [84]). (1) One is to build training set using novel synthesis algorithms [89] [90]. (2) The other is to divide the SSM into smaller subparts, each of which can be fitted using smaller training set. The authors [91] employ the wavelet transform to organize their model into several parts and each band is modeled independently. Another scheme, proposed by [92], describes the MR brain images based on mesh segmentation. Besides Gaussian distributions, some other models have been studied in building statistical data models. The authors [93] use Gaussian mixture models to describe the variation of the brain stem in MR image. The authors [94] use T-distribution for image compression and recognition.

4.2 Basic Ideas and Algorithm Overview

Directly constructing a global Gaussian-based SSM is often not a good idea. More naturally, data need to be analyzed and pre-processed before suitable SSM can be constructed. Specifically, in this project, our goal is to repair the missing regions of the digitized human bodies. For such kind of articulated geometric models, we propose a new completion algorithm based on a local T-distribution based SSM. Our data completion algorithm mainly has two components:

1. **Data Preparation**: The set of (training) template data should be evaluated and, if necessary, pre-processed, namely, partitioned then transformed, so that each subpart follows a uniform statistical distribution.

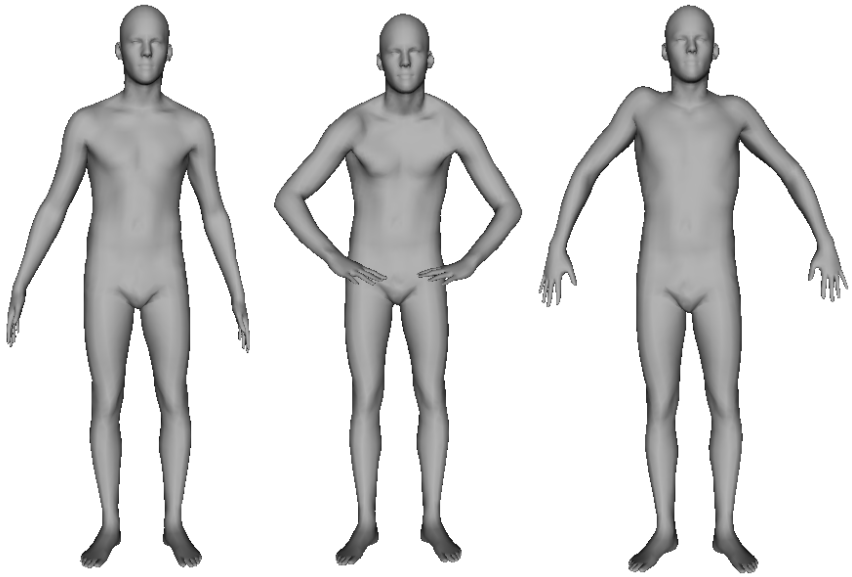


Figure 4.1: Human Bodies with Different Poses.

2. **SSM Construction and Completion**: Construct a T-distribution based SSM (TSS-M) for each local subpart individually. The incomplete object can then be repaired by the set of local T-Statistical Shape Models.

4.3 Data Preparation

To ensure a meaningful SSM can be constructed on a set of shapes, we propose to preprocess the shapes using the following paradigm.

1. Verify whether the data follow a statistical distribution (Sec. 4.3.2). IF yes, THEN the data are ready for SSM construction, STOP.

2. IF the verification in STEP 1 fails, THEN partition the data (Sec. 4.3.3).

3. Align each subpart (Sec. 4.3.4); GOTO STEP 1 to verify this subpart.

4.3.1 Shape Representation

Given a mesh M with n vertices, suppose the 3D coordinates of the vertex i are $p_i = (p_i^x, p_i^y, p_i^z)$, we can use a $3n$ vector P to represent this mesh M by concatenating all the coordinates: $P = (p_1^x, p_1^y, p_1^z, p_2^x, p_2^y, p_2^z, \ldots, p_n^x, p_n^y, p_n^z)$. For all the available datasets M_i in the dataset, construct a consistent parameterization among them (see below). Therefore, every object M_i is represented using a $3n$-dimensional vector P_i.

Consistent Shape Parameterization. To build a statistical shape model on a set of geometric objects requires that all these data are consistently parameterized, with corresponding regions among different objects being sampled coherently. Effective computation of inter-shape correspondence has been studied in geometric modeling literature in recent years, and many algorithms have been developed to solve low-distorted and feature-aligned bijective inter-surface maps [95–99]. Since this mapping computation is not a focus of this paper, we refer the interested readers to recent surveys on this topic [2,100] for more details. In this work, we adopt the algorithm from [99] to establish the mapping between human body models from various dataset sources. The mapping is optimized through minimizing the distance distortion induced by the correspondence, and solved using a novel integer linear programming solver. With consistent correspondence computed across all the digital models, each model can then be represented using a $3n$-dimensional vector.

4.3.2 Distribution Verification

To determine whether a statistical dataset D follows a certain distribution, the *statistical hypothesis test* is commonly used. From a distribution model, this test first generates a syn-

thetic dataset $\hat{D}$, then determines whether D and $\hat{D}$ are related. We define a *null hypothesis* that D and $\hat{D}$ have no relationship (not from the same distribution), and the *alternative hypothesis* is that D and $\hat{D}$ are related. The null hypothesis is rejected if the probability of the alternative hypothesis occurring (p-value) is higher than a threshold T_ϵ. We use Anderson-Darling test (adtest) [101] to determine whether the hypothesis is accepted or not. In practice, the significance level threshold T_ϵ is often set to 0.05. If the p-value of the test running on a set of data is larger than the T_ϵ, then we can say this dataset follows this specific distribution.

4.3.3 Data Partitioning

We propose a clustering-based algorithm to partition the geometric models. This differs from the several partitioning strategies used in existing SSM research [92], which mainly adopt mesh segmentation algorithms straightforwardly. Our partitioning adopts *fuzzy clustering method*. Every element unit (here, a triangle) may belong to not just one subpart. Every triangle i is associated with a set of weights w_{ib}, measuring how likely it belongs to the b-th cluster. The design of suitable weights depends on applications. For example, in many clustering algorithms, the weights are calculated as the inverse of the distances from the point to the centroid of the cluster. However, in our application of human body modeling, we can naturally use the weights given by the *skinning* of the human body.

Skinning

In mesh skinning, a vertex weight is used to describe how much this vertex is influenced by a joint. These weights can naturally be used for the subpart clustering. We utilize the

skinning technique introduced in [102] to associate joints with vertices. The basic approach is briefly recapped here. Suppose we have a mesh sequence of $T+1$ meshes $P^t, t = 0, \ldots, T$. The mesh skinning algorithm consists of two steps

1. **Computing Bone Transformations**: The rotation of the triangle j at mesh P^t relative to a reference mesh, say P^0, is computed. The rotation is denoted as R_j^t. A mean shift clustering is used to cluster triangles with similar R_j^t. Within each cluster, the rigid transformation for a corresponding bone T_b^t is then estimated by averaging the rotation of all the triangles in this cluster.

2. **Estimating Vertex Weight**: Based on a linear blending model, we can establish the following over-constrained least square problem:

$$\begin{aligned} w_{ib} = argmin_{w_{ib}} \| \sum (T_b^t p_i^0) w_{ib} - p_i^t \|^2, \\ s.t. \sum w_{ib} = 1, \text{and } w_{ib} >= 0. \end{aligned} \tag{4.1}$$

   After solving the vertex weights w_{ib}, each triangle's weight is computed as the average of the weights of its three vertices.

Adaptive Data Clustering

For each triangle i, all the weight w_{ib} is sorted and the largest k weights are extracted. This indicates that which k parts this triangle is most likely to belong to. Meanwhile, w_{ib} is removed if it is smaller than a threshold η (in our experiments, $\eta = 0.01$). This double thresholding ensures each triangle is associated with closely related up-to-k clusters. All the triangles that belong to the same cluster are grouped together and form a subpart. The subparts generated through such a clustering is fuzzy, that is, clusters could overlap with

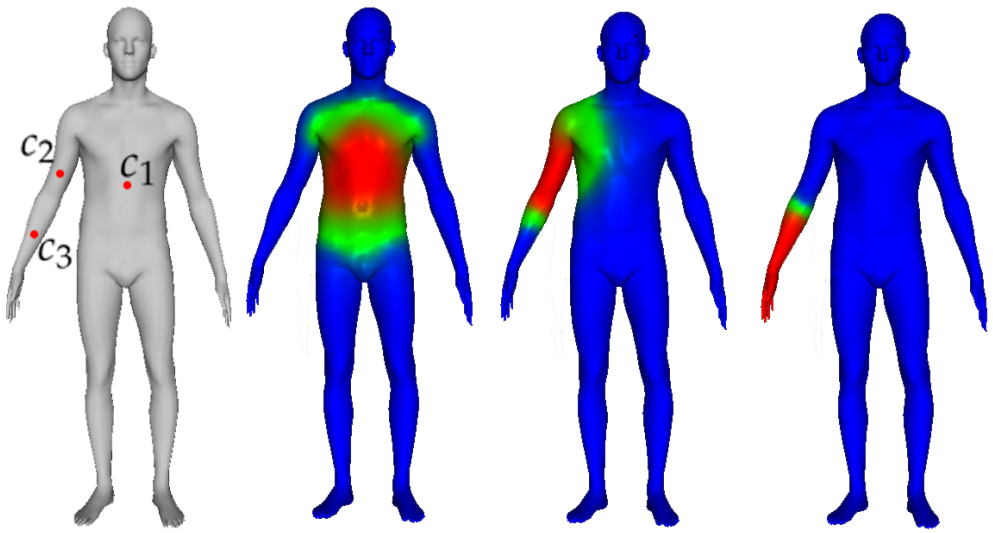


Figure 4.2: The clustering on the human body data. (a) shows the center of three clusters (bones); (b) shows the color-coding of the distribution of the vertex weights with respect to each bone. Note that different subparts or clusters can overlap with each other.

others. Fig 4.2 shows some local parts of the mesh and the coloring shows the vertex weight (from blue representing 0 to red representing 1).

Note that in Sec. 4.3.3, the mean shift clustering is used to cluster the triangles with similar rotations. A single intuitive *rotation tolerance parameter*, ξ, controls the number of clusters generated in mean shift clustering method. Smaller ξ will lead to more clusters and finer partitioning. To avoid the unnecessary over-segmentation, we first choose $\xi = 0.1$. Then we test whether partitioned data follows the specific distribution (Sec. 4.3.2). If not, we will decrease the value of ξ to obtain more clusters. Finally, the data will be partitioned such that each subpart follows a statistical distribution.

4.3.4 Data Transformation

The geometric variation among a group of similar objects often comes from two factors: (1) the intrinsic difference between different individual objects, which is what we aim to model

using SSM, and (2) some global or local transformations on the data (e.g., pose/expression change, etc.), whose effect is what we want to eliminate. After partitioning, each subpart of the data needs to be transformed to eliminate the transformation effect. This transformation and normalization are done as follows. First pick a reference mesh, then transform all the other subparts to align with the reference by calculating the transformation $T_b = (F, t)$, where F is the rotation and scaling matrix and t is the translation. T_b is calculated by minimizing the triangle area-weighted sum of the squared position errors between transformed triangle centroids $F\tilde{c}_i$ and the reference centroids c_i over all triangles on the mesh:

$$\sum a_i ||c_i - F\tilde{c}_i - t||^2 \tag{4.2}$$

where a_i is the area of the triangle i. Then by applying T_b on each part of the data P_b, this subpart of data is normalized.

4.3.5 Global versus Local SSM after Data Preparation

When data preparation discussed in the above sections has correctly partitioned and transformed each subpart of the shape, it is possible to construct an SSM globally. For example, in the human body modeling case, suppose one can apply some technique to normalize the poses of all the human body models, then after that, one may construct a global SSM to describe all these human bodies under the same pose. However, using individually constructed local SSM (that we will propose in the following section) is usually better than using a global SSM on the pose-normalized models. This is mainly due to two reasons.

(1) Fundamentally, our verification-partitioning framework is data-driven and it prepares the data for correct SSM modeling. While in this human body modeling task, skin-

ning is used to generate attributes on each vertex for fuzzy clustering, in other data modeling tasks, different attributes may be used. In contrast, global SSM based on pose-normalization may not always work.

(2) Locally constructed SSMs are often more accurate, more reliable, and more efficient

   a. Different subparts may have different variations, where one global SSM may not be sufficient.

   b. Smaller parts exhibit less variation, which require fewer training samples. Therefore, datasets with moderate sizes are sufficient for reliable training and model construction [92].

   c. The data in each subpart has smaller dimension. So the local SSM construction, including PCA computation and parameter estimation, is much faster than the construction of a global SSM.

To illustrate the above observations, we use a set of 2D points sampled from three characters (Fig. 4.3). Suppose different people are writing these 3 characters. Due to their habits in writing different characters, each character may follow a different distribution. This is simulated by using different Gaussian perturbations on each character ($\mu_i = 0$, $\sigma_1 = 0.1$ for 'S', $\sigma_2 = 1$ for 'P', and $\sigma_3 = 2$ for 'M'). 3000 sampling data are generated accordingly (three data shown in the top three rows of Fig. 4.3).

To compare the reconstruction results using global SSM vs local SSM, a 10-fold cross-validation is used: first, divide the data into 10 groups, and first 9 groups are used as the training set for SSMs, while the last group is the testing set. Some points in the char-

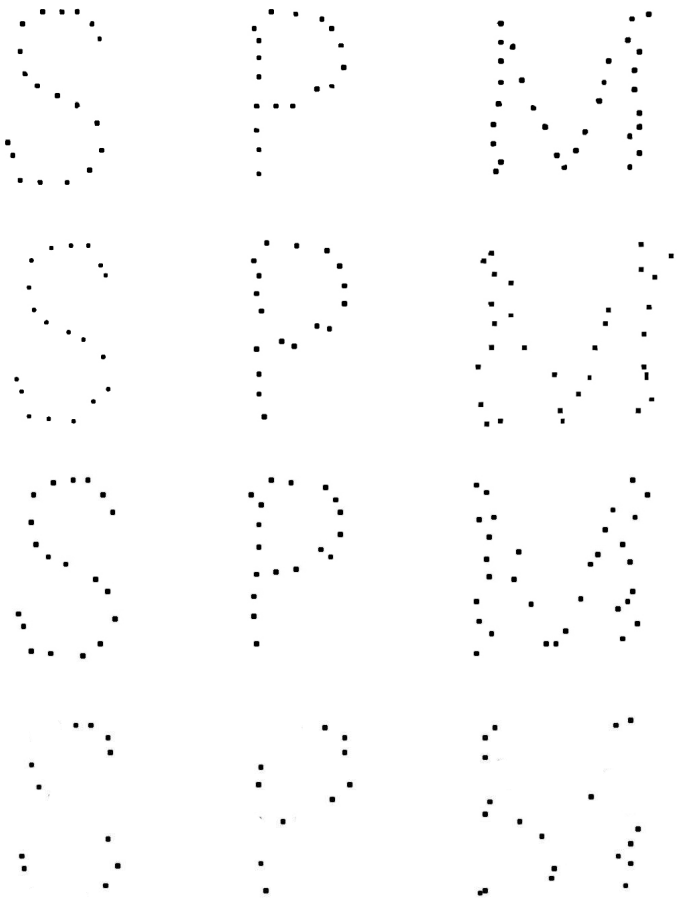


Figure 4.3: Synthesized characters with different Gaussian distribution (the top three rows) and the incomplete characters (the last row).

acters from the testing set are randomly removed (an example is shown in the bottom row of Fig. 4.3) This reconstruction test is repeated 10 times. The average completion error (equ. 4.17) using local SSM is 0.228 while the error from a global SSM is 0.764. Therefore, we believe if the data intrinsically exhibit different local variances in different local parts, then local SSMs are naturally offer more accurate description of the data.

4.4 SSM Construction and Data Completion

After the data preparation step, each (partitioned) dataset follows a certain distribution. This section discusses how we build the SSM on each dataset.

4.4.1 Gaussian-distribution Based SSM

Given data P which exhibits multiple unknown variables in it, we want to calculate those unknown variables using the known variables based on the correlation between them. *Factor analysis* is a powerful tool to explore the correlation between the variables in high dimensional data. In factor analysis, any d-dimensional vector P can be related to a corresponding q-dimensional vector of latent variables x:

$$P = Wx + \mu + \varepsilon. \tag{4.3}$$

The $d \times q$ matrix W relates the two sets of variables and μ is the mean of P. The motivation is that the high dimensional observed variables can be described by lower dimensional unobserved variables called *factors*. Conventionally, the latent variable follows the Gaussian distribution $x \sim N(0, I)$ and the error is likewise Gaussian $\varepsilon \sim N(0, \Psi)$. So the observations P is also Gaussian $P \sim N(\mu, WW^T + \Psi)$. The key assumption behind factor analysis is that the error covariance Ψ is a diagonal matrix. Therefore, the observed variables P_i are conditionally independent given the latent variables x. Thus, x is intended to explain the correlations between P_i while ε_i represents variability unique to a particular P_i.

The matrix W is solved using maximum-likelihood estimates and will generally not correspond to the principal subspaces of the observed data. However, when the error covariance Ψ is constrained to be *isotropic*, i.e., $\Psi = \sigma^2 I$, then there is a certain link between the factor

analysis and PCA. Suppose the covariance matrix for N observed data P is

$$S = \frac{1}{N}\sum_{n=1}^{N}(P_n - \mu)(P_n - \mu)^T. \tag{4.4}$$

By using maximum likelihood algorithm introduced in [88], the solution for $d \times q$ matrix W:

$$W = U_q(\Lambda_q - \sigma^2 I)^{\frac{1}{2}} \tag{4.5}$$

where the q column vectors in the $d \times q$ matrix U_q are the principal eigenvectors of S, with the corresponding eigenvalues λ_i in the $q \times q$ diagonal matrix Λ_q,and the solution for σ is given by

$$\sigma^2 = \frac{1}{d-q}\sum_{j=q+1}^{d}\lambda_j \tag{4.6}$$

which is the average of the lost eigenvalues. Thus in the following work, we will make the assumption that the error model is isotropic, and the above statistical models are called *Gaussian-distribution based SSM (GSSM).*

GSSM offers a natural approach to estimate the missing variables in data P [103]. We can estimate the latent variable x based on the known variables in P using maximum likelihood method which will be captured in Section 4.4.3. Then the missing variables can be computed by $P = Wx + \mu$.

4.4.2 T-distribution Based SSM

The multivariate T-distribution $t(\mu, \Sigma, v)$ has the following density function

$$\begin{aligned} P(x) = &\frac{\Gamma[(v+d)/2]|\Sigma|^{-1/2}}{\Gamma(v/2)v^{d/2}\pi^{d/2}}\times \\ &[1 + \frac{1}{v}(x-\mu)^T\Sigma^{-1}(x-\mu)]^{-(v+d)/2}, \end{aligned} \tag{4.7}$$

where d is the dimension of the observations. Compared with the Gaussian distribution, a T-distribution has heavier tails. This means that T-distribution is more likely to produce values that fall far from the mean, which is called rare case. Thus, when outliers do occasionally appear in a dataset (e.g., in human body modeling, there are people who are very fat or thin, very tall or short), using Gaussian distribution to model these data will underestimate the probability of the rare case occurring.

The T-distribution is in theory a generalization of Gaussian distribution and allows more free parameters to be tuned according to the data. The degree of freedom v controls how heavy its tails are. When $v \to \infty$, T-distribution becomes Gaussian distribution. The T-distribution could more reliably describe data in many applications. In fact, we have estimated the partitioned body data, and found out that they generally follow the T-distributions rather than Gaussian distributions. Fig. 4.4 shows the probability-probability plot (P-P plot) of the data (vertex coordinates from the human body models) versus normal distribution and T-distribution respectively. From the figure we can see that the data are more T-distributed than normal distributed. Therefore, in our work, we propose a T-distribution Based SSM (TSSM) that generalizes the Gaussian-based SSM for shape completion.

The equation of TSSM is the same as the equ. (4.6), but $x \sim t(0, I, v)$ and $\varepsilon \sim t(0, \sigma^2 I, v)$. Then the observations t is also T-distributed $P \sim t(\mu, WW^T + \sigma^2 I, v)$. We use the EM method [94] to estimate the parameters (W, μ, σ, v) in equ. (4.6).

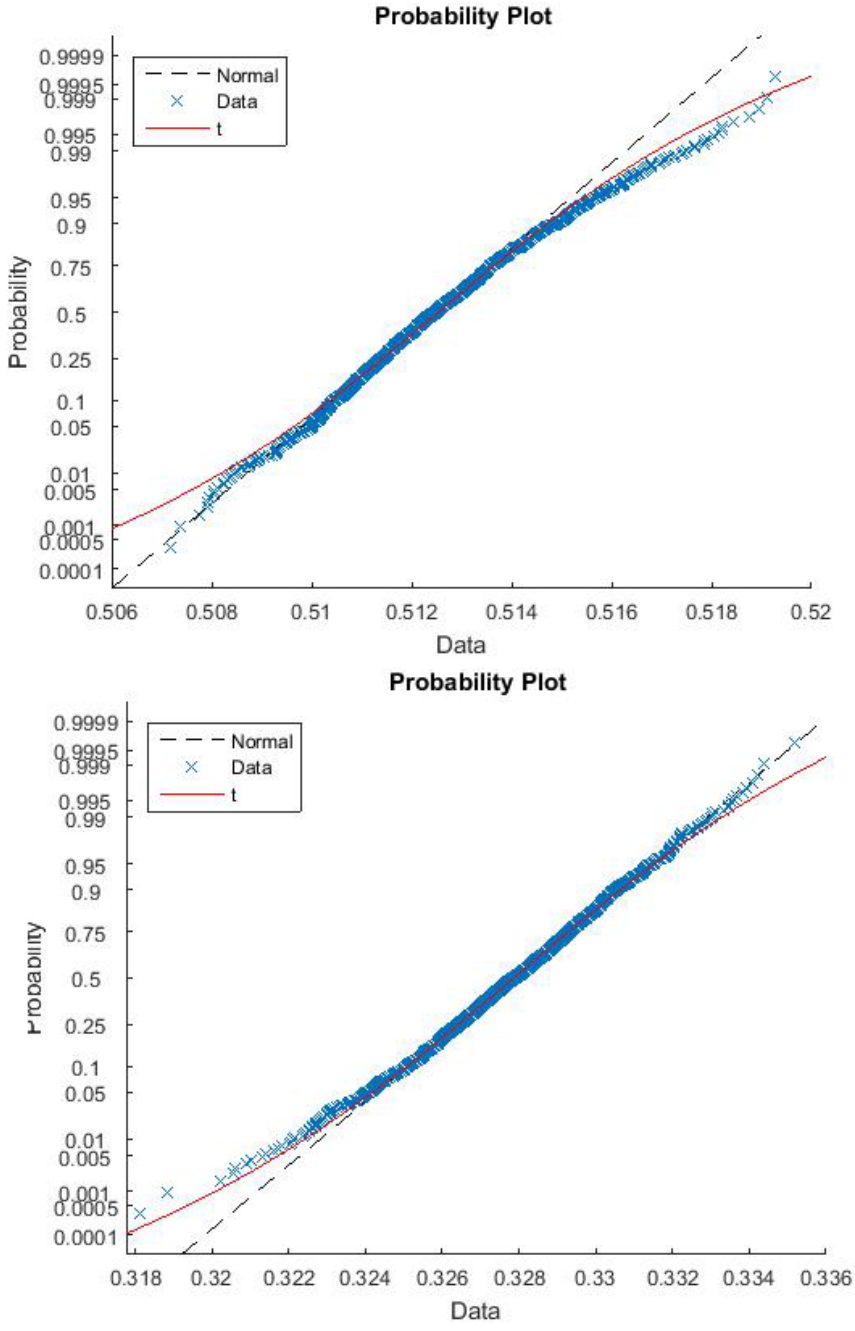


Figure 4.4: The P-P plot of the dataset, and its Gaussian distribution fitting (dashed curves) and T-distribution fitting (solid curves). We can see that the data follow T-distribution better.

4.4.3 Missing Data Computation

Determining Unknown Components

Given an incomplete mesh P^*, first, we register it with a complete reference model P and get a map $\Phi : P \rightarrow P^*$ (Sec. 4.3.1). Φ is used to resample and tessellate P^* following P. Since P^* is incomplete, some vertices $p_u \in P$ will not find corresponding points on P^*. In addition, we also identify the near-boundary points p_u, whose images, $\Phi(p_u)$, have a small distance to the boundary of missing region in P^* (specifically, distance $< 2\lambda$, where λ is the mesh resolution of P). The coordinates of these vertices p_u are considered unknown or unreliable, and need to be estimated in this section during completion.

Since P is decomposed into B different subparts $P_b, (b = 1, \ldots, B)$, we can obtain the decomposition of the incomplete mesh $P_b^* = \Phi(P_b)$. Each P_b^* containing unknown coordinates will be repaired using TSSM.

Computing Missing Data Using TSSM

In this section we aim to derive the formula to solve coefficient x for the incomplete data P_b^* based on a T-distribution based Statistical Shape Model. We use the maximum-likelihood estimation to estimate the coefficient x.

The likelihood function is defined as:

$$P(x|P_b^*) = \frac{P(x) * P(P_b^*|x)}{P(P_b^*)} \propto P(x) * P(P_b^*|x) \tag{4.8}$$

Since $x \sim t(0, I, v)$, then:

$$P(x) = \frac{\Gamma[(v+q)/2]}{\Gamma(v/2)v^{q/2}\pi^{q/2}[1 + \frac{1}{v}||x||^2]}, \tag{4.9}$$

where q is the dimension of x. Since $\varepsilon \sim t(0, \sigma^2 I, v)$, then $P_b^*|x \sim t(Wx + \mu, \sigma^2 I, v)$:

$$P(P_b^*|x) = \frac{\Gamma[(v+q')/2]}{\Gamma(v/2)v^{q'/2}\pi^{q'/2}[1 + \frac{1}{v}||L(P_b^* - Wx - \mu)||^2]} \tag{4.10}$$

where q' is the number of known vertices in P_b^*, and L is the $q' \times q$ matrix such that $L(i', i) = 1$ where i' is the index of known vertex which is rearranged from 0 to $q-1$, and i is the index of vertex in P_b^*. Thus, LP_b^* is the q'-dimensional vector that consists of all the known vertices. To maximize the function (4.8), we take the logarithm-inverse of the likelihood function. The log-inverse-likelihood function is then:

$$\begin{aligned} -\log P(x|P_b^*) &= -\log P(x) - \log P(P_b^*|x) \\ &= log(1 + \frac{1}{v}||x||^2) + \\ &\quad log(1 + \frac{1}{v}||L(P_b^* - Wx - \mu)||^2) + c, \end{aligned} \tag{4.11}$$

where c is a constant. To maximize the likelihood function is to minimize its log-inverse. Let $E1 = \frac{1}{v}||x||^2$ and $E2 = \frac{1}{v}||L(P_b^* - Wx - \mu)||^2$. That is, we aim to minimize:

$$\begin{aligned} E &= log(1 + E1) + log(1 + E2) \\ &= log(1 + E1 + E2 + E1 * E2) \end{aligned} \tag{4.12}$$

Since $E1$ and $E2$ is small, $E1 * E2$ can be ignored. Finally the objective function to minimize is

$$E = ||x||^2 + ||L(P_b^* - Wx - \mu)||^2, \tag{4.13}$$

Surprisingly, this lead to the same objective function in [67]. Thus, the optimal x is:

$$x = V(W^2 + \sigma^2 I)^{-1} W U^T L(P_b^* - \mu), \tag{4.14}$$

where U and V are the Singular Value Decomposition (SVD) of $Q = LW$: $Q = U\Sigma V^T$. The

completion result is therefore:

$$P_b = WV(W^2 + \sigma^2 I)^{-1} W U^T L(P_b^* - \mu) + \mu, \tag{4.15}$$

We run the optimal completion equation (4.16) for all P_b that contain unknown elements.

Shape Blending: we use the linear blending to calculate the final complete data, for each vertex i:

$$p_i = \sum_{b \in \mathscr{B}_i} (T_b^{-1} p_{ib}) w_{ib} \tag{4.16}$$

where T_b is the bone transformation calculated in Sec. 4.3.4, $\mathscr{B}_i$ is the set of parts that vertex i belongs to, p_{ib} is the position of vertex i in part b and w_{ib} is the weight calculated in Sec. 4.3.3.

4.5 Experimental Results

In this project, our target application is to repair digital human body scans for medical analysis on the effectiveness of diet and exercises on obesity treatment. During a period of time, volunteers (including people with normal weights and those affected by obesity) are being scanned regularly using a set of Kinect sensors (see details in Sec. 4.5.2). Completing the scanned digital models reliably and accurately is critical for subsequent body circumferences measurement and local volume computation.

Therefore, we perform multiple experiments to evaluate our LTSSM model and data completion algorithm on human body datasets. Our experiments are classified into two types: (1) data completion on benchmark human body datasets; and (2) data completion on our scan data. We used two benchmark human body datasets, including the MPI FAUST dataset [104] and the processed CAESAR dataset [105]. On the one hand, we use these models as training

data to build the SSM model. On the other hand, we use them as ground truth to evaluate the effectiveness of our algorithm and to compare with other completion algorithms. These experiments on repairing benchmark data with ground truth are shown in Sec. 4.5.1. Finally, we also demonstrate the completions of our Kinect scans in Sec. 4.5.2.

4.5.1 Repairing Benchmark Datasets

Training Data Preparation. In order to evaluate our LTSSM algorithm on the benchmark datasets, we first randomly partition the available data into 10 sets that roughly have the equal sizes. Then pick one among them to be the *testing data* while the other nine sets are used as *training data*. Within the testing data, we again pick a few models and synthesize damages on the model. These damaged regions are then repaired using various completion algorithms. Since all the human body models are firstly parameterized consistently to a mesh with n vertices, the removed regions (vertices), after repairs, can be directly compared with the ground truth model (before regions are removed).

Error Evaluation. The completion error on each vertex v_i is measured as:

$$e_i = \|\mathbf{p}_i - \mathbf{p}_i^*\| \tag{4.17}$$

where $\mathbf{p}_i$ is the 3D position of v_i on the repaired region and $\mathbf{p}_i^*$ is the 3D position of this vertex in the ground truth model.

Comparison with Context-insensitive Methods: First, we compare our completion result with the result using context-insensitive method [7], which fits the holes with a parametric patch. We make the holes on the arm of 10 body data randomly chosen from the benchmark set and repair them using the two methods respectively. Fig 4.5(b) shows one

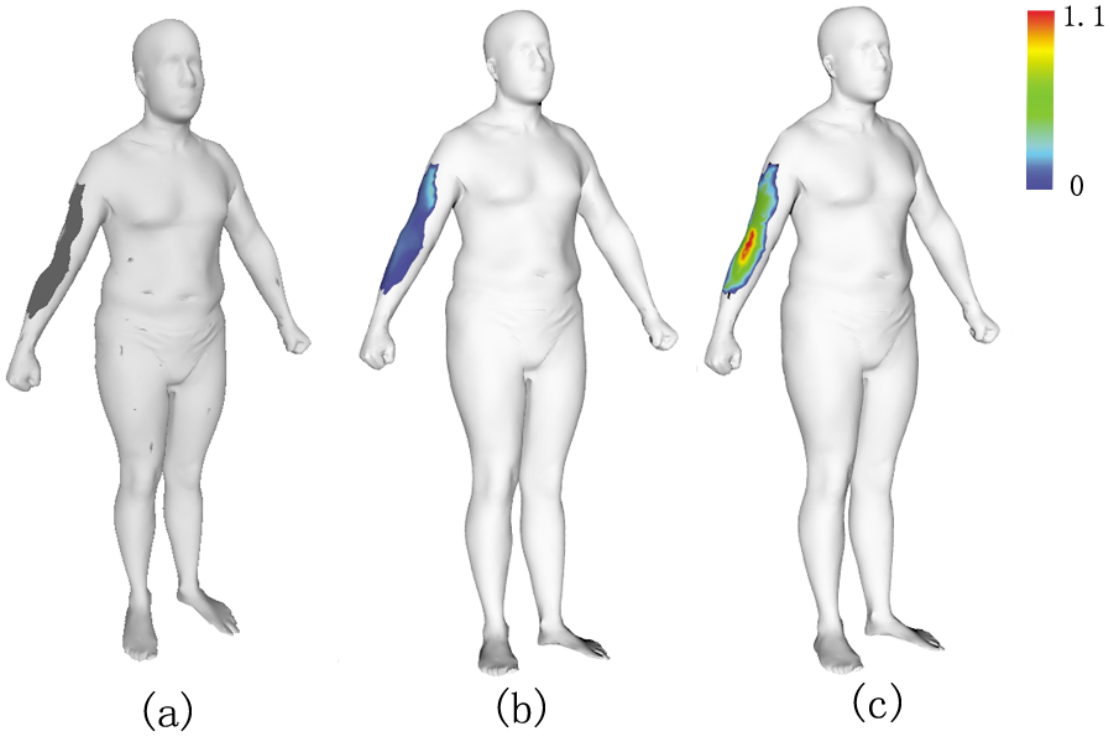


Figure 4.5: Color-encoded completion error. (a) Incomplete mesh; (b) completion result using LTSSM. (c) completion result using context-insensitive method [7].

example of the completion result using LTSSM and fig 4.5(c) is the result using [7]. Our method can produce more accurate completion results: the average error (equ. 4.17) by [7] is $0.6cm$ while the error by LTSSM is about $0.1cm$. Our method is 83% better than [7].

Comparison with Context-sensitive Methods: For more complex problem, context-insensitive method will fail while template-based method can handle it. However, choosing an appropriate template is difficult. We compare our result with the result by template-based method [8], which deforms the template to fill the missing region. We make the holes on the hand of 10 body data randomly chosen from the benchmark set and repair them using the two methods respectively. Fig. 4.6(b) shows the template model. Fig. 4.6(c) is one example of the completion result produced by LTSSM. Fig. 4.6(d) shows the result using [8].

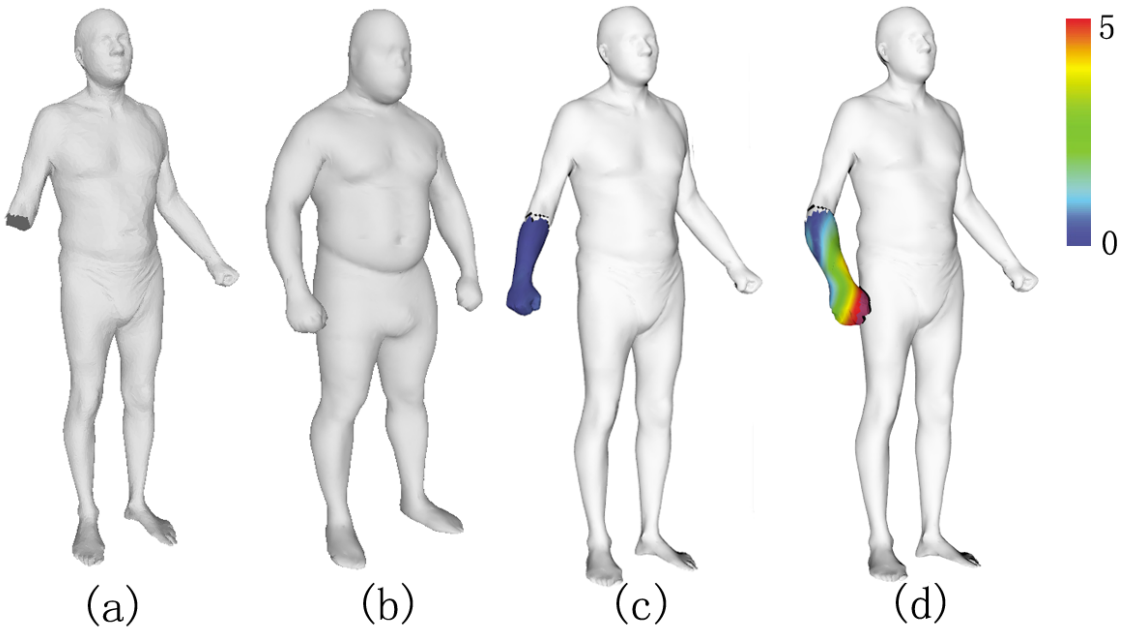


Figure 4.6: Color-encoded completion error. (a) Incomplete mesh; (b) Template; (c) completion result using LTSSM. (c) completion result using template-based method [8].

We can observe that the completion result by [8] is similar to the template. Therefore, the completion error by [8] is larger than ours. Specifically, the average error (equ. 4.17) by [8] is 3.3cm while the error by LTSSM is about 0.25cm. Our method is 77% better than [8].

Comparison between local SSM and global SSM: *Data with different poses*: In order to demonstrate that our method is capable of handling the incomplete data that has non-rigid transformations, we perform our method on several body data with different poses and successfully repair them. Fig. 4.7 shows the completion results with different poses. Direct global SSM fails on these data. Note that, here the repaired geometry is a closed hand while the original model is an opened hand. This is because the majority of example models in the database has closed hands. In fact, if more gestures of hands are captured in the datasets and are used for training finer subparts, the different gestures could also be recovered.

Data with same pose: Even if the data is in the same pose, local SSM is still better than

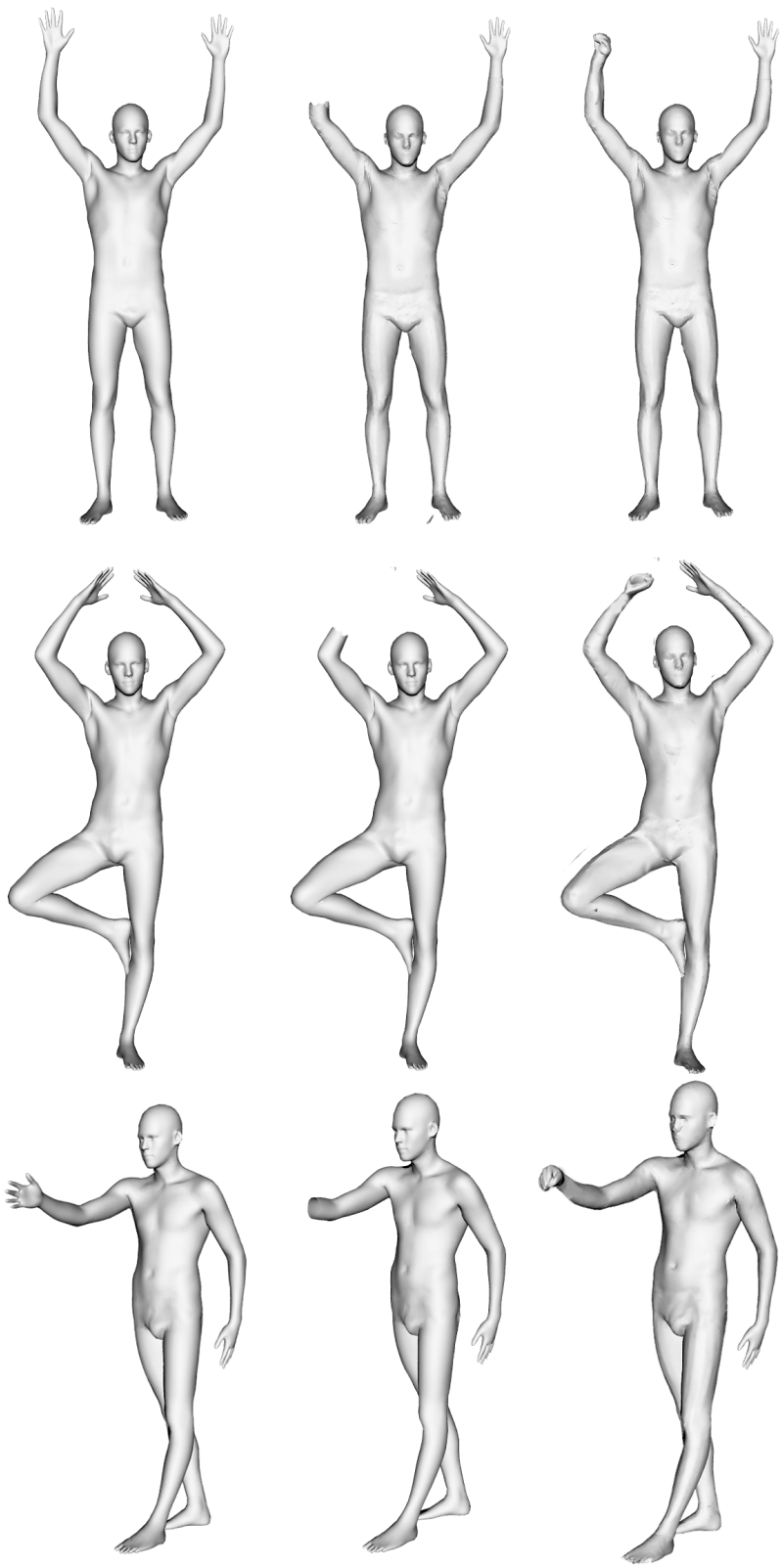


Figure 4.7: Completion using a model with a different pose. Left column: The original model; Middle: The incomplete model; Right: The completion result.

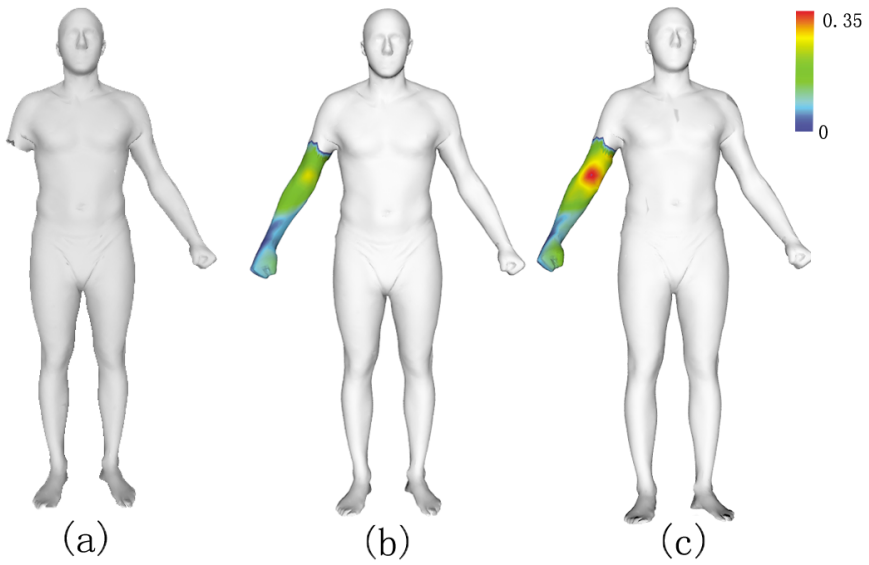


Figure 4.8: Color-encoding of the completion error. The original model (a) is repaired using local SSMs (b) and a global SSM (c).

global SSM. In Sec. 4.3.5 we discussed the comparison between the local SSM and global SSM on the synthetic data. Here we run the local SSM and global SSM to fill the missing region on 10 body data with the same pose. Fig. 4.8 shows the error distributions of one example result by the two methods. The average error of local SSM is $0.34cm$ while the error of global SSM is $0.49cm$. Thus local SSM is 30% better than global SSM.

Comparison between TSSM and GSSM We also run the TSSM versus GSSM on 10 body data. Fig. 4.9 shows the error distributions of one example from the results of the two methods respectively. The average error of GSSM is $0.19cm$ while the error of TSSM is $0.11cm$. Thus TSSM is 40% better than GSSM.

4.5.2 Completion of the Our Scanned Data

We built a body scanning system (Fig. 4.10) consisting of 16 Kinects mounted from different directions. The scanned models inevitably contain holes due to

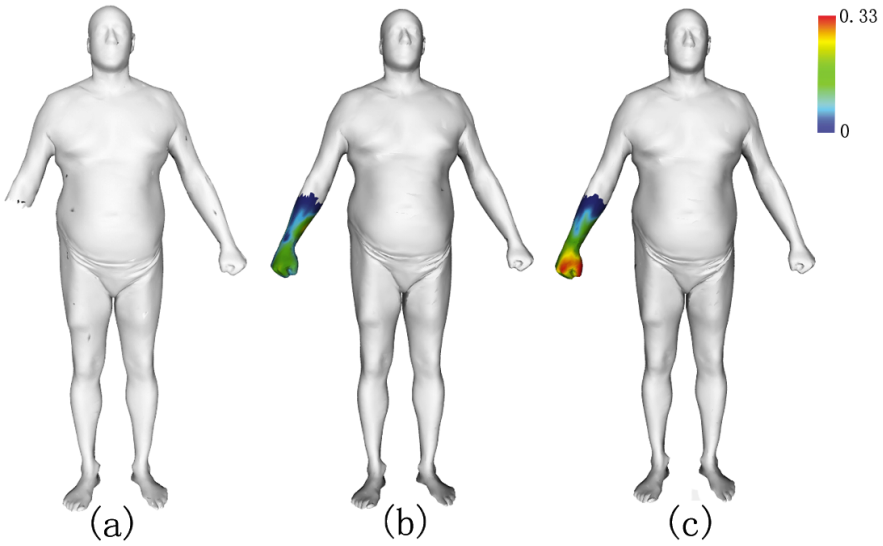


Figure 4.9: Color-encoding of the completion error. The original model (a) is repaired using T-distribution based SSMs (b) and Gaussian-based SSMs (c).

1. The data is occluded by the body itself. It often occurs between the arms and body and between the two legs.

2. Some part of the body is outside the range of the Kinect, therefore cannot be scanned.

We have scanned about 200 bodies from volunteers with various heights and obesity. We applied our completion algorithm to repair all these data. Fig. 4.11 shows a few completion examples on our scans. We have provided these data to the medical scientists for subsequent anthropometry analysis in their clinical research.

4.6 Conclusion and Future Work

We propose a new completion algorithm to repair the damaged or missing regions of 3D geometric data using a novel Local T-distribution based Statistical Shape Model. First, we suggest that general geometric models should be evaluated and appropriately partitioned

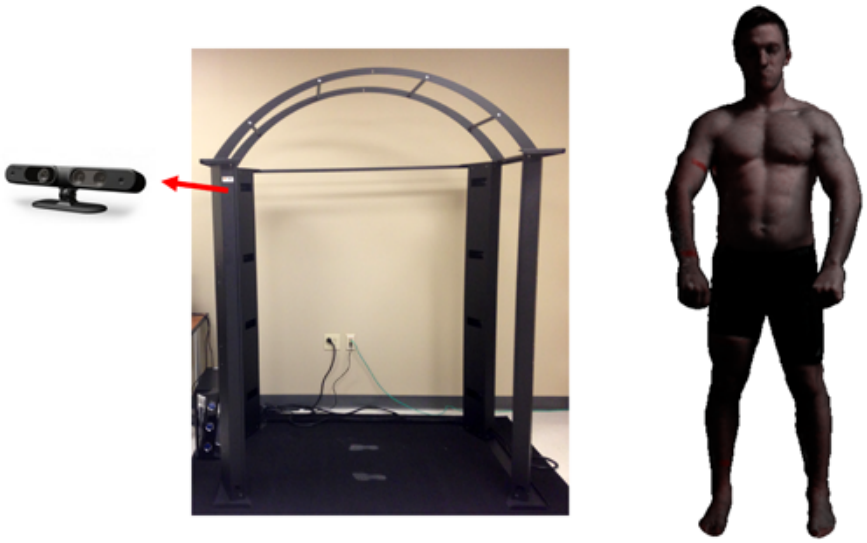


Figure 4.10: Our body scanning system. Left: the scanning device where 16 Kinects are mounted on a frame; Right: a completed scan with texture.

into subparts that follow a statistical distributions. Second, the subparts are modeled locally using a T-distribution based SSM. We examine the effectiveness of this framework in our specific application of the completion of human body scans with big missing regions and with arbitrary poses. Tested on the human body benchmark with ground truth, our LTSSM-based completion demonstrates to be more effective in restoring the geometry than other representative completion algorithms.

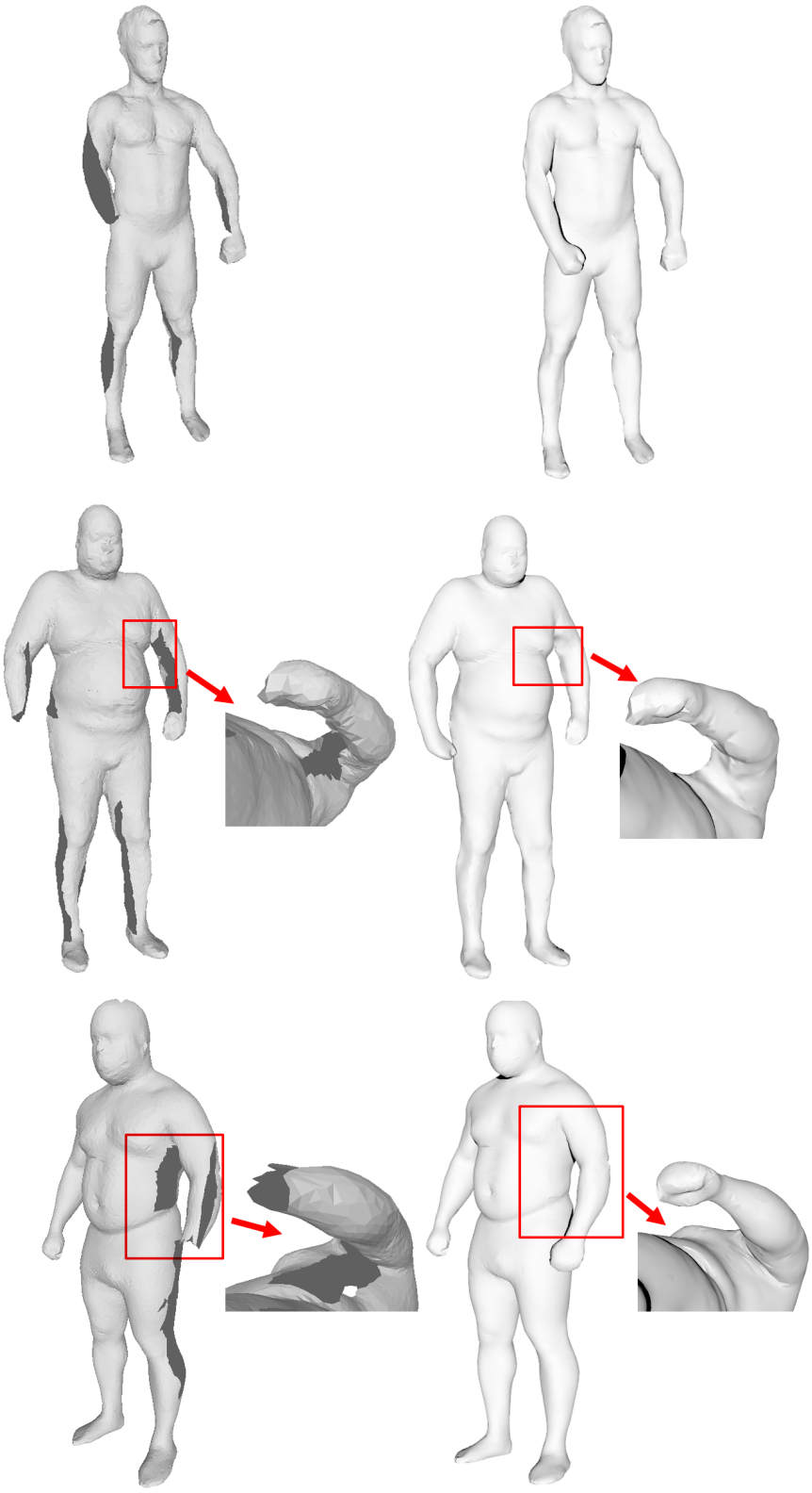


Figure 4.11: Some completion results on our digital scans. Left: Incomplete models; Right: completed models.

5. CURVE MATCHING AND CLASSIFICATION FOR ANCESTRY ANALYSIS

Assessing ancestry from the skeleton is an important problem of the practice of forensic anthropology because it aids with the identification of unknown individuals in law investigations. For example, in computer-aided skull processing fragmented skull restoration, correctly identified ancestry can lead to more natural/accurate selection of skull template [9]. In forensic facial reconstruction [10,11], applying appropriate tissue depths also relies on the understanding of the subject skull's ancestry.

We study the palate shape of the skull, or as it is sometimes called, the dental arcade, as the indicator. For the three main ancestry groups: *white*, *black*, and *Asian/native American*, the dental arcades fall into three categories [12,13]: the *parabolic* (or triangular), the *hyperbolic* (or rectangular), and the *elliptical* (or rounded), respectively. As three geometrically discrete shapes, it stands to reason that a parabola, hyperbola, and ellipse would lend themselves easily to metric assessment, and, thus, a more objective method for assessing ancestry from the palate could be formulated [14, 15]. However, practical computer-aided identification systems based on this idea do not exist. In this dissertation, we model this problem as a curve matching and classification problem and design a computer-aided system to determine the ancestry of the skull based on the palate shapes. Then we evaluate the effectiveness of palate shape analysis in ancestry determination.

Computer-aided Curve Matching and Classification. Curve classification and matching are important techniques used in many computer vision and computer graphics tasks such as signature verification, handwriting recognition, object recognition and reconstruction and shape matching. **Curve Matching** is often used in object reconstruction for finding corresponding regions/parts. In [106], the authors build a string representation for the curve based on the curvature and torsion of the curve, and then used a string matching algorithm to do the curve matching. Sunder et al. [107] approximate 3-d shapes using their skeletons (media axis) and perform shape comparison/retrieval based on the matching of the abstracted skeleton graphs. However, curve matching cannot handle the intra-class variability well.

Many effective **curve classification** algorithms have been developed recently. A general approach is to extract some features of the curve, both globally and locally, then use machine learning techniques to do classification. Bunke et al. [108] use temporal stroke sequence as the feature of the handwriting and use Hidden Markov Models to do the recognition. Meenakshi et al. [109] propose GSC feature to describe the signature and use a Bayes classifier in the verification process. Fang et al. [110] combine a smoothness feature with global features in signature classification. In [111], the authors use training data to build a codebook for the classical curves and computed the histogram of curves based on the codebook, then use a linear SVM to do the classification.

We first construct a parametric spline representation for the sampled palate points, then extract some global and local features of these curves based on forensic domain knowledge. Finally, we perform curve classification and verify our results. The experiments are performed on about 370 palate curves collected from the William M. Bass Skeletal Collection, a modern sample, housed at the University of Tennessee, the Hispanic Collection housed at

the Pima County Office of the Medical Examiner (PCOME) in Tucson, Arizona which was also a modern collection comprised primarily of undocumented Mexican immigrants, and the Robert J. Terry Skeletal Collection, a more historic sample, housed at the Smithsonian Institution's Museum Support Center. The correctness ratio on ancestry prediction produced by our system is compared with identification done by forensic specialists.

5.1 Related Work

Many techniques exist for skull ancestry assessment. They can be classified into two categories as *metric* and *non-metric* methods. The metric techniques measure a set of lengths between pairs of landmarks on the skull, and perform the prediction based on statistical analysis of existing skull collections. These methods are usually objective; but they rely heavily on a large set of data points. Furthermore, metric techniques require mostly intact skeletal material, so that the data points necessary to run the analysis are present. These analyses are also dependent on the availability of the technology needed to analyze the data [12].

In contrast, non-metric, or morphological, methods of ancestry determination are easier and faster to use because they do not require specialized equipment or analysis. However, currently these methods rely on the observer's experience [112] and can be subjective.

We develop a computer-aided system to digitize this entire non-metric determination process and conduct a thorough and subjective numerical evaluation of this method. We also aim to explore whether with advanced computer vision and pattern recognition algorithms the system can do a better job than human beings with forensic experience.

5.2 Algorithms

Our developed ancestry estimation algorithm has 4 steps: (1) data acquisition, (2) curve preprocessing (conversion to parametric spline representation, data smoothing, and normalization), (3) global and local geometric feature extraction, and (4) curve classification and prediction. We elaborate these sub-steps as follows.

5.2.1 Data Acquisition

We use a MicroScribe 3D digitizer [113] to acquire the palate curve. This digitizer uses a stylus to record the position of points in three-dimensional space. We record the center of each tooth where it meets with the alveolar bone (see Fig. 5.1 for illustration).

Figure 5.1: Photograph of how the points are recorded using the digitizer.

5.2.2 Preprocessing

Spline Construction. As the palate curve shapes are collected using digitizer and are relatively sparse and the sample points are non-uniform (especially when parts of the skeleton

are missing), we first convert these discrete sampling points into a continuous parametric representation. A *cubic b-spline* curve [114] is a piecewise polynomial function that has C^2 continuity. B-Spline has many nice properties in representing palate curves.

1. It is smooth and continuous, therefore is more reliable against the acquisition noise.

2. It has a parametric closed-form representation, so we can easily calculate the curve's various intrinsic properties such as curvatures on arbitrary point of the curve.

3. The spline is invariant to the affine transformation.

Given a set of data points $D = \{\mathbf{d}_0, \mathbf{d}_1, \ldots, \mathbf{d}_n\}$ and a curve degree k, fitting a B-spline curve of degree k passing all data points reduces to solving $n+1$ control points $\{\mathbf{p}_i\}$ through a linear system. Specifically, a B-spline curve can be formulated as:

$$\mathbf{r}(t) = \sum_{i=0}^{n} N_{i,k}(t)\mathbf{p}_i. \tag{5.1}$$

where $P = \{\mathbf{p}_0, \ldots, \mathbf{p}_n\}$ are $n+1$ unknown 3-dimensional control points and $N_{i,k}$ are kth-order basis functions defined on the parametric knot spans $t_i, \ldots, t_{i+k+1}$. Each parameter t_i here corresponds to a data point $\mathbf{d}_i$ and it can be determined using e.g. arc-length parameterization. The unknown control points can be solved by the linear system $D = N \cdot P$.

Curve Smoothing. Due to the irregularity of the teeth and the acquisition noise (often inevitable during the usage of digitizer to record the points), the collected data may be noisy. Therefore, instead of constructing a B-spline curve that precisely interpolates all the sample data which can cause unnatural wiggles, we compute a smoother B-spline curve to approximate these data. The approximating curve pass the first and last data points but is not required to pass other points. The approximation is measured by error distance between

data points and the curve. When the number of control points is equal to the data points, the equation system is full ranked and the B-spline curve passes all the data points; when fewer number of the control points is used, the approximating B-spline curve is computed in a least square manner. The sum of all squared error distances to minimize is

$$\epsilon(\mathbf{p}_1, \ldots, \mathbf{p}_{h-1}) = \sum_{i=1}^{h-1} |\mathbf{d}_i - \mathbf{r}(t_i)|^2. \tag{5.2}$$

This approximate b-spline curve is resilient to the data noise. The number of control points should not be too small because we will lose the details of the data. In our experiments, we use 11 control points to construct the spline curve.

Curve Normalization. In forensic anthropology, shape of the palate projected to 2D plane is matched with three special cases.

We also fit such a plane using the sampled data points and project the palate curve onto this plane. The curve is normalized so that its length is 1. Finally, rotate the curve so that its line of reflection is on the y-axis and the front curve is towards $y+$ direction (see Fig. 5.2)

Figure 5.2: Rotation Normalization. (a) Original curve (b) Curve after rotation.

5.2.3 Extracting Geometric Features

In this section we will introduce how we extract the features of the dental curve for their classification. We build our model based on the domain knowledge that the shape of Asian curve is close to ellipse, white curve close to parabola, and black close to rectangular hyperbola. An intuitive approach, therefore, is to fit the curve with those three different types of quadratic curve and see how much it deviates from these curves respectively. We also incorporate several other local geometric features to describe the shape of the two sides and the front part of the curve, which turns out to improve the ancestry estimation.

Fitting Ellipse. First, we fit the curve with the ellipse. The equation for ellipse is $Ax^2 + Bxy + Cy^2 + Dx + Ey + F = 0$ satisfying the constraint $B^2 - 4AC < 0$. By sampling a set of points uniformly from the spline curve, we can obtain a big linear equation system. By solving this equation system in least square sense we can compute the best fitting ellipse for this curve. However, this simple fitting method would cause a problem. In Fig. 5.3, although the shape of this curve is close to an ellipse, it matches well with a small portion of the ellipse. The intuition is to evaluate the matching between the curve and a big part of the ellipse. Therefore, we design the following 3-step fitting algorithm.

1. Translate the curve so that its centroid is on y-axis and its two endpoints are on the x-axis (Fig. 5.4(a));

2. Iteratively move the ellipse towards the $y-$ direction with a small step δ, and fit it with a standard ellipse $x^2/a^2 + y^2/b^2 = 1$ (Fig. 5.4(b));

3. Repeat step 2 until the curve is entirely below the y-axis; choose the ellipse with the best fitting error (Fig. 5.4(c)).

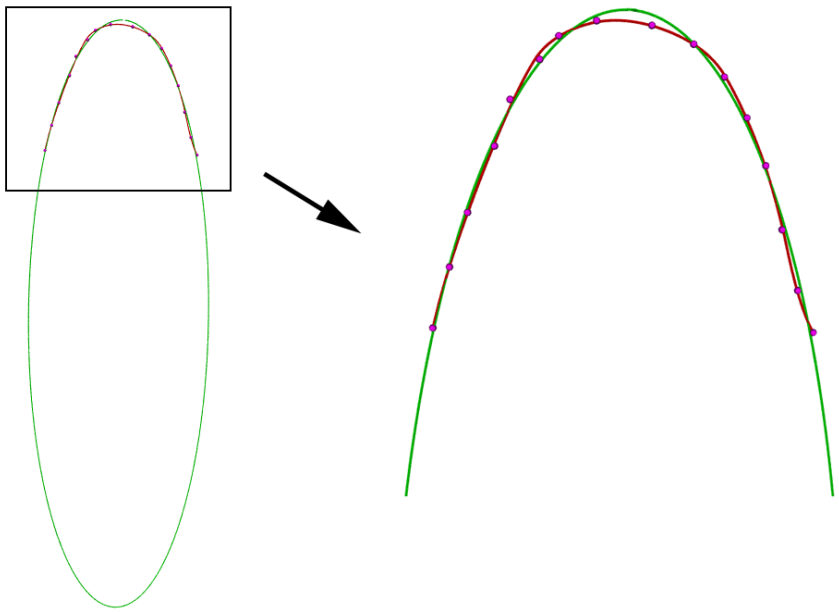


Figure 5.3: The issue when fitting ellipse.

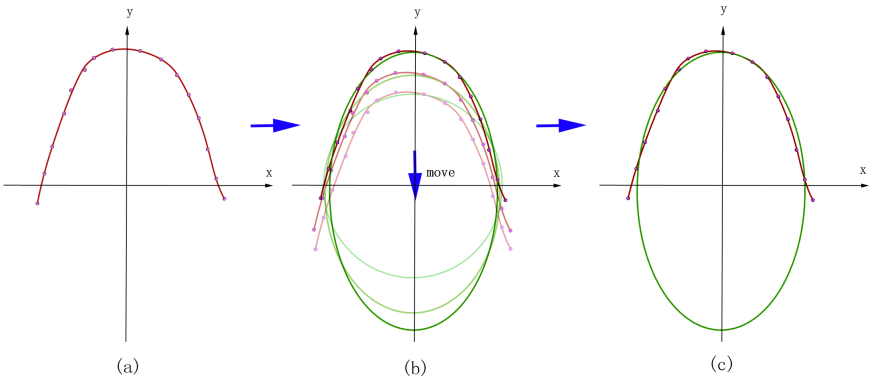


Figure 5.4: The method to fit ellipse.

Fitting Parabola. Fitting with parabola is easier. The equation for the parabola is $Ax^2 + Bxy + Cy^2 + Dx + Ey + F = 0$ satisfying the constraint $B^2 - 4AC = 0$. Here we can set $B = C = 0$. Then the equation can be written as $(y - c_y)^2 = 4p(x - c_x)$. Still by solving the linear equation system in the least square sense we can find the best fitting parabola for this curve.

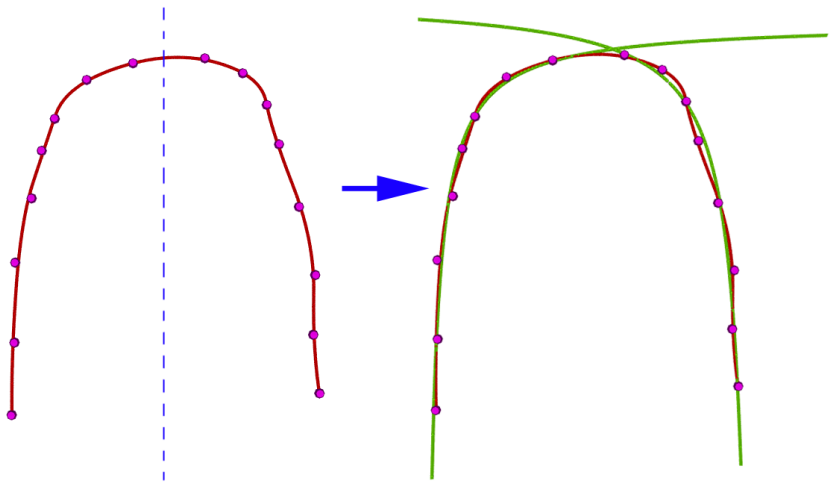


Figure 5.5: Fitting Rectangular Hyperbola.

Fitting Rectangular Hyperbola. The equation for the rectangular hyperbola is $Ax^2 + Bxy + Cy^2 + Dx + Ey + F = 0$ while satisfying the constraint $B^2 - 4AC > 0$ and $A + C = 0$. Here we simply set $A = C = 0$. Then the equation can be written as $(x - c_x) * (y - c_y) = m$. We split the curve into 2 parts by its symmetric line and fit each part with a rectangular hyperbola (Fig. 5.5).

Describing the Two Sides of the Curve

The shape of the two sides of the curve is helpful in ancestry estimation. Generally, the shape of the two sides for black is parallel to each other, for white it is angled, for Asian data it is rounded. So we can approximate the two sides of the curve with 2 straight lines and use them as features. Fig. 5.6 shows line fitting for 3 different ancestry dental curves. In this figure we can see that lines of Asian and black curves will have large slope while the line of white curve will have small slope. Meanwhile the line fitting error for Asian curve could be large while it is small for black and white curve.

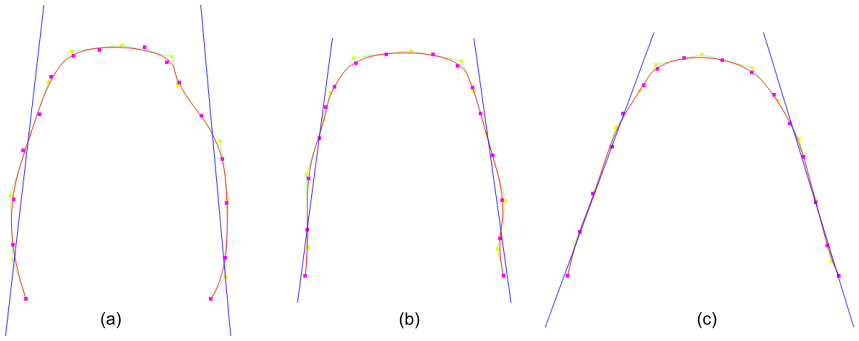


Figure 5.6: Fitting Rectangular Hyperbola.

Curvature and Turning Angles of the Front Teeth

The shape of the front teeth seems to be an important indicator for the palate shape. Generally, the shape of front teeth for black is like flat in the middle and rectangular at the two sides, and the shape for white is pinched/pointed and rounded for Asian. Here we use the angle formed by the control points of the B-Spline to indicate those shapes. Suppose we have 5 control points with the highest y-coordinate values, regarded as the control points controlling the front teeth part of the curve. Connecting them sequentially we can get 4 straight lines. The 3 angles α_1, α_2 and α_3 formed by adjacent lines would be our another 3 dimensions of the feature. From Fig 5.7 we can see that for black the angle α_1 and α_3 are closer to 90 degrees and α_2 is closer to 180 degree. Furthermore, we calculate the integral of the curvature of the front curve $\int_\ell \kappa(t)dt$, which describes the flatness of the front region.

The Accumulated Feature of the Palate

Fig. 5.8 shows the fitting results for 3 different ancestry dental curves. From the figure we can see that for different types of the dental curve the fitting error for the quadratic curve is different. So we use these 3 fitting errors as first 3 dimensions of the feature of the dental

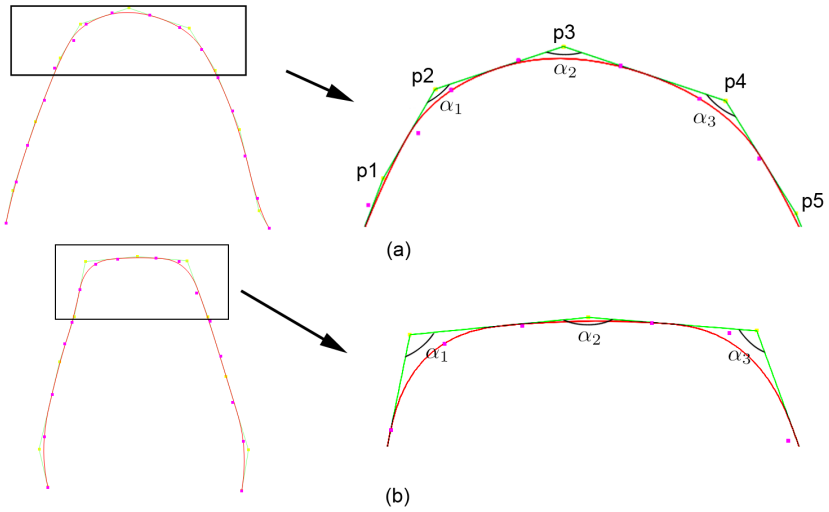


Figure 5.7: Front Shape for two ancestry. (a) Shape for white (b) Shape for black (c) Fitting for White.

curve. Meanwhile, the shape of the quadratic curve is also important. For ellipse with equation $x^2/a^2 + y^2/b^2 = 1$, parabola with equation $(y - c_y)^2 = 4p(x - c_x)$, and rectangular hyperbola with equation $(x - c_x) * (y - c_y) = m$, we use b/a, p and m as another 3 dimensions of the feature, which indicates the shape of the quadratic curve. Furthermore, by fitting two straight lines, we can get two slopes $k1, k2$ and the total fitting error, which can be set as another 3 dimensions of features. Additional 4 dimensions of feature are also needed for describing the shape of the front teeth. Therefore, we can totally build a 13-dimensional feature vector f for each dental curve, such that f_0 is the fitting error for ellipse, f_1 is the fitting error for parabola and f_2 is the fitting error for rectangular hyperbola; $f_3 = b/a$, $f_4 = p$ and $f_5 = m$; f_6 is the fitting error for line, $f_7 = k1$ and $f_8 = k2$; $f_9 =$ angle α_1, $f_{10} =$ angle α_2, $f_{11} =$ angle α_3 and $f_{12} = \int_\ell \kappa(t)dt$ (ℓ is the front region of the curve).

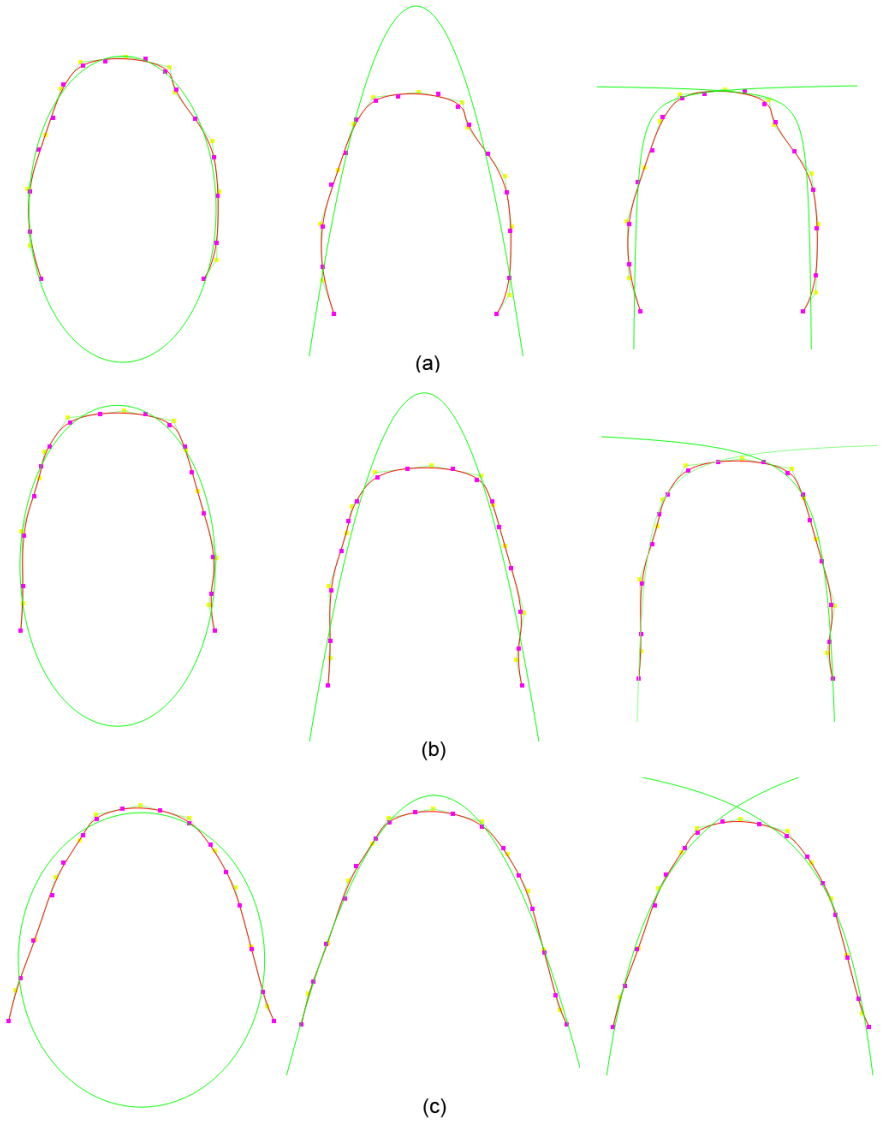


Figure 5.8: Fitting Results. (a) Fitting for Asian (b) Fitting for Black (c) Fitting for White.

5.2.4 Classification

Each dental curve is associated with a 13-dimensional vector. Then we use the multi-class support vector machine (SVM) to classify those vectors into 3 categories. SVMs [115], which are widely used in machine learning, are supervised learning models with associated learning algorithms that analyze data and recognize patterns, used for classification and regression analysis. Given a set of training examples, an SVM algorithm will build a model to mark each of them into one of two categories. When a new example comes in, the SVM algorithm will calculate and classify which category it belongs to. The multi-class SVM [116] constructs M binary classifiers. The ith classifier output function ρ_i is trained taking the examples from one class as positive and the examples from all other classes as negative. For a new example x, it assigns it to the class with the largest value of ρ_i. Here we use the WEKA [117] software available online to do our classification.

5.3 Experimental Results

5.3.1 Sample

The sample for this study is composed of 310 individuals, including 83 white male, 85 white female and 65 black male, 51 black female, 3 Asian male, 1 Asian female, 20 Hispanic and 2 Hispanic female. This sample came from the William M. Bass Skeletal Collection, a modern sample, housed at the University of Tennessee, the Hispanic Collection housed at the Pima County Office of the Medical Examiner (PCOME) in Tucson, Arizona which was also a modern collection comprised primarily of undocumented Mexican immigrants, and the Robert J. Terry Skeletal Collection, a more historic sample, housed at the Smithsonian

Figure 5.9: Photograph of how the points are recorded using the digitizer.

Institution's Museum Support Center. Meanwhile, since the palate shape of Hispanic people is unknown, we exclude the 22 Hispanic palate curves. Therefore, the total number of our data sample is 288.

5.3.2 Data Collection

A *Microscribe*^{TM} digitizer, which uses a stylus to record points in three-dimensional space, is used to record data for each palate. This is accomplished by recording the point on the bone that represented the approximate center of each tooth (see Fig. 5.9).

5.3.3 Classification Accuracy

Two types of the tests are performed to assess the accuracy: manual assessment and assessment by our algorithm.

Manual Assessment

A test of 20 randomly selected curves from the sample is used to assess the accuracy of the human practitioner using palate shape to estimate ancestry. Five graduate students, all first or second year master's degree students with backgrounds in forensic anthropology, are asked to assign ancestry based on the digital representation of a palate curve. Each observer is asked to assign white, black, or Asian category. This is done three times to obtain the average accuracy for each observer. The mean of those averages is used as the manual accuracy for human practitioners. Table 5.1 shows the accuracy for a human observer. We can see that the results vary from different observer and different trials, which means that the manual assessment is highly subjective and dependent on the background of the human practitioner. The average accuracy is about 40%.

Table 5.1: Accuracy for a human observer

Observer	Trial 1	Trial 2	Trial 3	Average
1	0.5(10/20)	0.55(11/20)	0.55(11/20)	0.533
2	0.15(3/20)	0.55(11/20)	0.6(12/20)	0.433
3	0.35(7/20)	0.35(7/20)	0.3(6/20)	0.333
4	0.2(4/20)	0.2(4/20)	0.4(8/20)	0.267
5	0.45(9/20)	0.55(11/20)	0.3(6/20)	0.433
Average	0.33	0.44	0.43	0.399

Assessment By Our Algorithm

The algorithm is tested by ten-fold testing procedure. At first, the dataset is randomly divided into ten groups. For every iteration of testing, we use nine of the ten groups as the training set to build up the classifier and the tenth group is used as the testing set to test the accuracy. Table 5.2 shows the accuracy of our algorithm for each ancestry. The average accuracy is about 58%, which is higher than the human observer. However, it is still not

accurate. The possible reason is that the admixture of different ancestry, plus a variety of additional environmental factors, will cause the change in the human skeleton. This is called the secular changes.

Table 5.2: Accuracy for our algorithm

Actual Ancestry		Classified Ancestry		
		White	Black	Asian
White	168	111 (60%)	56 (33%)	1 (1%)
Black	116	60 (52%)	56 (48%)	0 (0%)
Asian	4	2 (50%)	2 (50%)	0 (0%)
Total	288	173 (60%)	114 (40%)	1 (0%)

5.4 Conclusion and Future Work

In conclusions, palate shape appears to be a useful indicator of ancestry from the human skull, and potentially more useful when assessed by our algorithm compared with a human observer. However, the admixture and environmental factor are causing secular change in the shape of the palate, which prevent our algorithm from being accurate. Our future work includes developing more accurate indicators. We can add more useful metrics, for example, the depth of the palate, to help increase the accuracy of our algorithm.

6. NON-RIGID SURFACE MATCHING FOR FACIAL RECONSTRUCTION

The reconstruction of unidentified body's face from its skull has been performed by forensic anthropologists manually to reveal the facial appearance of the body. This reconstructed facial geometry is often the only available information allowing investigators to identify the body when no other ID information is available. The current skull and face modeling techniques adopted in law investigation performed manually by forensic specialists using physical sculpting clay takes an experienced expert about two weeks to finish on average. Our ultimate goal is to automate this procedure in an digital environment. In this project, we perform a first-step study on this problem and develop an interactive modeling system for this digital facial synthesis task.

Following the current practice, given a subject skull with its ancestry information, forensic specialists follow the statistical tissue depth data measured on a set of landmarks on the skull (face). Each measurement describes the total distance from skin surface to the skull, including the thickness of both fat and muscle layers. Markers with proper lengths are placed on the skull or a cast made from it, oriented orthogonally to the skull surface, corresponding to the direction of the tissue thickness measurements. Then a facial geometry reconstruction can be performed, ensuring the thicknesses of the clays equal the tissue depths on these landmarks (Fig. 6.1). The tissue depths for different ethnic groups are examined in the paper [118] and the relationship between the skull and facial features are studied

in the work [119]. Others found correlations between muscle activity and skull shape are documented in [120] [121]. A complete survey for facial reconstruction can be found in [122].

Figure 6.1: (Left) The skull and its landmarks, (Middle) The reconstructed face using clay, (Right) The reconstructed face with texture.

To mimic this reconstruction in a digital environment, we develop a 3D modeling framework to synthesize the face from the skull. The idea is to model the manual tissue depth method as an optimization problem. Given a template face, we aim to deform the face, preserving its symmetry and physical characteristics as a human face as much as possible, onto the top of the subject skull and fit the landmark constraints defined above. Our main **technical contributions** include (a) a mathematical model for facial reconstruction and (b) the method for as-rigid-as-possible deformation while maintaining the symmetry.

6.1 Related Work

6.1.1 Facial Reconstruction.

Recently, several manual methods for facial reconstruction has been used in practice by physically modeling the face from the skull. The manual facial reconstruction can be classified into general types. The Russian method [123] reconstructs the face by modeling the

complete anatomy of muscles and soft-tissues covered by a thin layer onto the skull. The American method [124] approximates the average tissue depths at a sparse set of landmarks on the skull. More recently, the method combining Russian and American methods, proposed by [125], is often called the Manchester method [126]. The above manual facial reconstruction methods require a high degree of forensic specialists' expertise, and is time-consuming and labor intensive.

Recently, the emerging 3D scanning techniques have been facilitating the computer-aided facial reconstruction. Using 3D scanning techniques, we can easily digitize the skull the reconstruct the face using computer by mimicking the manual reconstruction procedures. In the paper [127] a volume deformation method is proposed to deform a source head model such that the skull of that head model approximately matches the target skull. Archer [128] reconstructs the face by fitting a hierarchical B-spline head model to a skull. A deformation technique is proposed by Vanezis et al. [129]. In their work, a facial template is chosen from face database and deformed to match the position of target face landmarks, which is calculated by adding tissue depths to the corresponding skull landmarks. Kähler et al. [130] develop reconstructions from one head template with relatively few markers and use additional mechanisms to improve the reconstruction results. They can not only reconstruct the face, but also produce the expressive facial animation. A complete survey on recent computer aided facial reconstruction techniques can be found in [131].

6.1.2 Surface Registration

Given a source and a target surface, the goal of registration is to find a deformation that optimally positions points on the source surface to the target surface. Recently, many sur-

face registration algorithms have focused on rigid registrations. Famous rigid registration algorithms are Iterative Closest Point (ICP) [132] and its variant [133]. ICP requires a good intial positioning of the source and target surface and easily gets trapped in local minima. To tackle this problem, a set of approaches [134] [135] [136] [137] [138] uses feature matches to generate initial guess of the optimal transformation of the source surface, which can then be optimized by ICP.

Despite the rigid registration methods, the non-rigid registration has been proposed. Allen et al. [139] smoothly deform the template human meshes to fit with the incomplete human shape by optimizing the energy integrating the smoothness and data fitting error. Other non-rigid registration techniques have been developed [97, 140–145]. Olga et al. [146] propose a framework to deform the surface as rigid as possible and use it for many surface modeling and animation. Recent surveys for surface registration can be found in [147] [2].

6.2 Skull and Template Face Preparation

Before facial reconstruction, we need to carefully prepare the skull and template face data. In this section, we introduce the method to acquire the complete skull data, the landmarks on the skull and template face and the tissue depths on the corresponding landmarks.

6.2.1 Acquisition of the Complete Skull

The skull data is often highly damaged and fragmented, thus is hard to processed for facial reconstruction. First, the fragments of the skull are digitized using 3D scanner. Then, we use the method in [137] to reassemble the fragmented skull. The figure 6.2 shows the reassembly result. After the reassembly of the fragmented skull, there may still be missing regions that

need to be filled. Then we use the method introduced in [148] to complete the fragmented skull (Fig. 6.3).

Figure 6.2: (Left) The fragments of the skull, (Right) The reassembled skull.

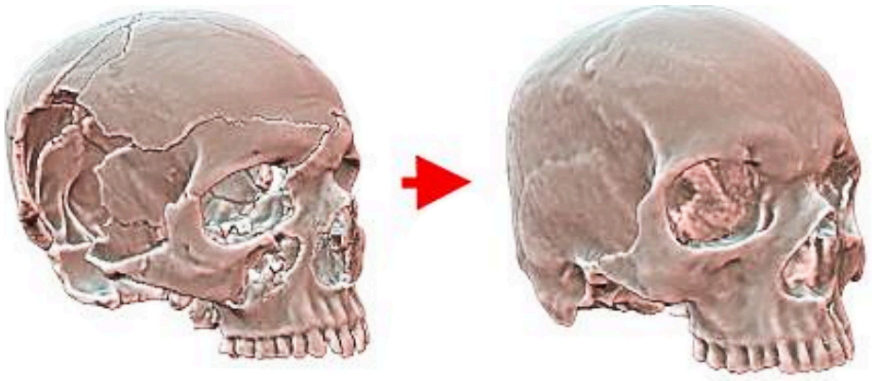


Figure 6.3: The Completion of the skull.

6.2.2 Landmarks on Skull, Template Face and Corresponding Tissue Depth

After acquiring the complete skull, we need to pick the landmarks on the skull data. Our landmarks on the skull are picked according to the paper [149]. Meanwhile, we pick the corresponding landmarks on the template face which is chosen from the database, which

has the same sex and origin with the skull. Figure 6.4 shows the landmarks on the skull and the face respectively. Then, the facial landmarks on the skull are calculated by adding the corresponding tissue depths from the paper [149] to the landmarks on the skull. The vectors from the landmarks on the skull to the facial landmarks should be orthogonal to the skull (see Figure. 6.5). Those facial landmarks on the skull are served as the target marker positions in the next section.

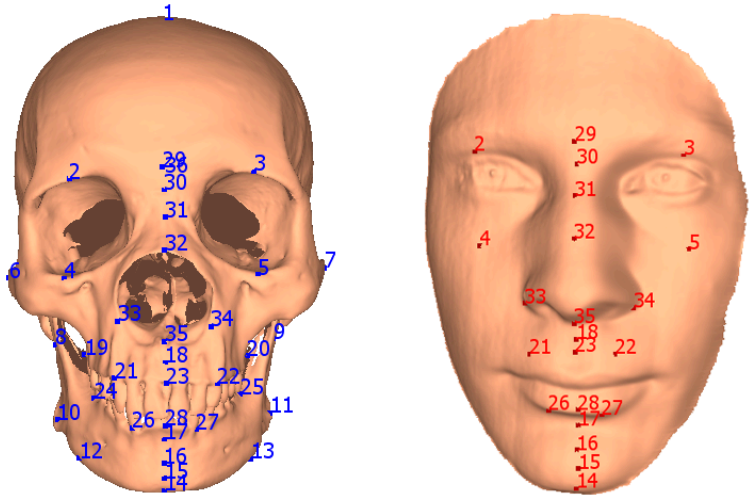


Figure 6.4: (Left) The landmarks on the skull, (Right) The corresponding landmarks on the face.

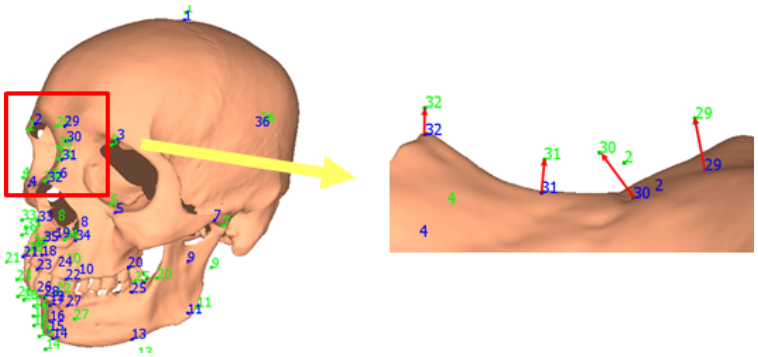


Figure 6.5: The facial Marker on the skull. The vectors from the landmarks on the skull to the facial landmarks should be orthogonal to the skull.

6.3 Symmetry-guided Non-rigid Deformation

In this section, we aim to deform the template face F such that the landmarks on the template face can approximately match the facial landmarks that are derived from the landmarks on the skull. Let us denote the set of landmarks on the template face as S_f and the set of the facial landmarks that are derived from the landmarks on the skull as S_s. Deforming F is reduced to calculate a non-rigid registration $f(S_f) \mapsto S_s$, so that $S_f \in F$ can deform elastically to fit S_s. Second, the registration should be as rigid as possible. Third, after deformation, $f(F)$ should remain symmetric. Therefore, we develop a energy function that consists of three terms. By optimizing the combined energy function, we can calculate the registration.

6.3.1 Non-rigid Surface Registration

Following the registration framework of [139], an affine transformation T_i is defined on each vertex v_i of F. We first conduct a global rigid alignment and scaling [132] to align S_f to approximately fit S_s. Our registration energy consists of three terms: (1) Landmarks fitting error, (2) Smoothness error and (3) Symmetry error.

6.3.2 Landmarks Fitting Error

The deformed landmarks $f(S_f)$ need to be as close to S_s as possible. An error term E_l is defined as the sum of the squared distance between vertices of S_f and the corresponding vertices of S_s.

$$E_d = \sum_{v_i \in S_f} dist^2(T_i v_i, D_i), \tag{6.1}$$

where v_i is the landmarks in S_f, and D_i is the corresponding landmarks in S_s. The $dist(T_i v_i, D_i)$ function computes the Euclidean distance from the deformed points to the target points. This term measures how close $f(S_f)$ matches S_s.

6.3.3 Smoothness Error

Second, we require the warping function f to be smooth. The smoothness error which measures the smoothness of f is formulated as:

$$E_s = \sum_{\{i,j|\{v_i,v_j\} \in edges(F)\}} ||T_i - T_j||_F^2, \tag{6.2}$$

That is, the transformations on any two neighboring vertices should be as close as possible.

6.3.4 Symmetry Error

Since the face is symmetric, we need to make sure that the deformed face should be as symmetric as possible. The symmetry error which measures the symmetry of $T(M)$ is formulated as:

$$E_{symm} = \sum ||T_s v_i - v_{s_i}||^2, \tag{6.3}$$

where T_s is the symmetric transformation, and v_{s_i} is the symmetric correspondence of v_i which is the closest point of the reflection of v_i.

6.3.5 Combining Error

We combine all the above error:

$$E = w_1 E_d + w_2 E_s + w_3 E_{symm} \tag{6.4}$$

Here w_1, w_2 and w_3 are the weight to control the importance of each term. The above energy is quadratic, thus it is easy to optimize by taking the derivative of E and solving the resultant equation system. By minimizing E, we can solve the optimal non-rigid registration f. By applying f on the template face F, the resultant deformed face $f(F)$ is our facial reconstruction result.

6.3.6 Implementation

We only consider translation: $T_i v_i = v_i + t_i$. According to the work [82], using affine transformations has the advantage of better handling global rotations and scaling. If we factor out global rotations and scaling during the initial alignment, it becomes less critical. Moreover, by only using translations, the linear system is reduced from $12n$ to $3n$ unknowns, which leads to the significant performance improvement.

For symmetric transformation, we compute the symmetry plane for the target skull [148]. Then the target skull and the template face F is transformed such that the symmetry plane is $x = 0$. Therefore, the symmetric transformation should be $T_s =$

$$\begin{pmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

.

6.4 Experimental Results

Our algorithm is implemented in C++ and performed on a desktop with 2.27GHz Xeon(R) CPU and 12 GB RAM.

Fig. 6.6 shows the facial reconstruction result from a digitized complete skull. The skull is from a white male person. Fig. 6.6(a) shows the landmarks on the skull and face we choose for facial reconstruction and Fig. 6.6(b) shows the facial reconstruction using our system. Two more results, which are from a asian male person and a white female person respectively, are illustrated in Fig 6.7.

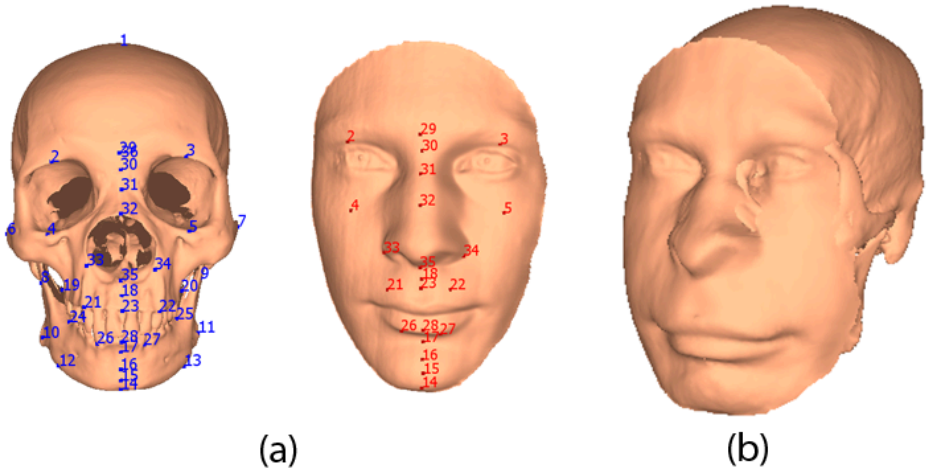


Figure 6.6: (a) The template face and skull with landmarks (b) The facial reconstruction result.

Finally, we perform our algorithm on a real fragmented skull from forensic anthropologists in which the victim's skull is severely damaged due to a gun shot. The fig. 6.8 illustrates the final facial reconstruction result.

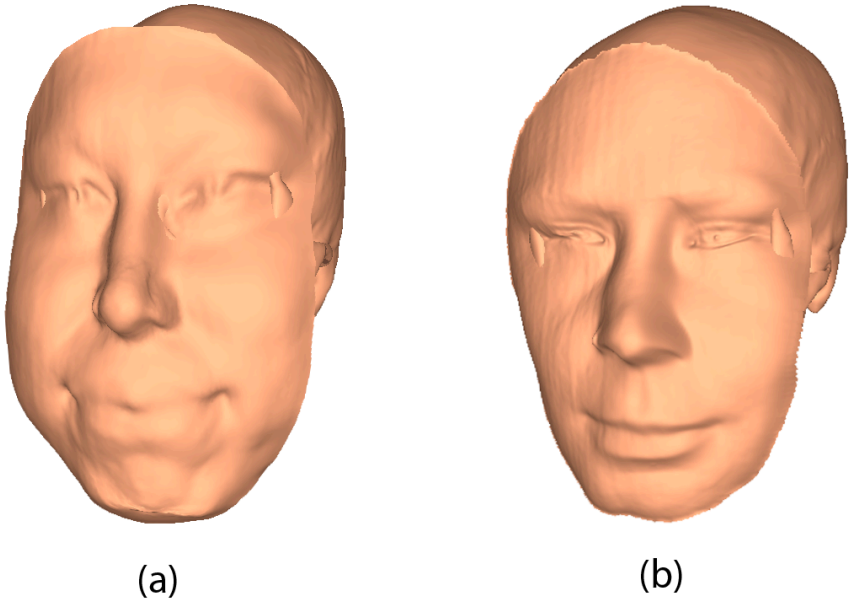


Figure 6.7: (a) The facial reconstruction result of asian male (b) The facial reconstruction result of white female.

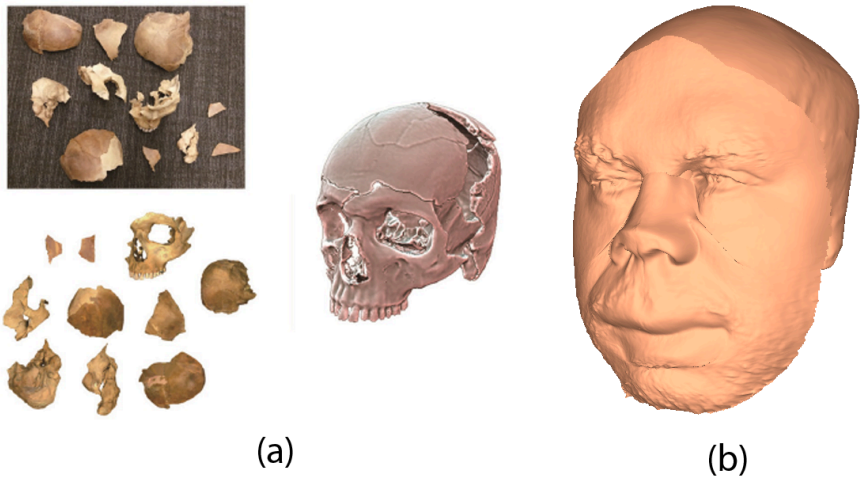


Figure 6.8: (a) The reassembled fragmented skull (b) The facial reconstruction result.

6.5 Conclusion and Future Work

We develop an automatic system to reconstruct the face from the skull. Given a template face, we deform the template face as rigid as possible with respect to the distance constraint that define the relation between the face and skull at selected sample positions. Meanwhile, the symmetry of the template face is maintained. Our main **technical contributions** include (a) a mathematical model for facial reconstruction and (b) the method for as-rigid-as-possible deformation while maintaining the symmetry. We demonstrate the effectiveness of our method on various skulls and template faces, including the real skull data provided by the forensic anthropologists.

Limitations and Future Work. In our framework, there is limited control on the details of the face, like the nose, eyes and hairs. A more sophisticated modeling on these feature regions could result in more realistic facial reconstruction. Second, adopting volumetric registration and deformation [144] [145] that models both face surface and tissue volumes may better model tissue volume preservation and avoid respective artifacts. Third, using additional information like age and ancestry could improve the accuracy of tissue depth selection. This requests a finer facial tissue classification and data collection. Finally, using multiple template faces and adaptively select suitable features from suitable individuals could also improve the robustness of the algorithm, making it less template-dependent.

7. CONCLUSIONS

In this dissertation we have studied the 3D geometric matching for data restoration and its forensic application. In chapter 2, We propose a graph-based optimization framework for automatic 2D image fragment reassembly using **Curve Matching**, **Pairwise Matching** and **Multi-piece Matching**. First, we compute the potential matching between each pair of the image fragments based on their geometry and color. After that, a novel multi-piece matching algorithm is proposed to reassemble the overall image fragments. Finally, the reassembly result is refined by applying the graph optimization algorithm. We perform experiments to evaluate our algorithm on multiple torn real-world images, and demonstrate the robustness of this new assembly framework outperforms the existing algorithms in both reassembly accuracy (in handling accumulated pairwise matching error) and robustness (in handling small image fragments). Then, in chapter 3, we extend our 2D image fragment reassembly work to 3D. We develop a reassembly algorithm to effectively integrate both guidance from a template and from matching of adjacent pieces' fracture-regions. First, we compute partial matchings between fragments and a template using **Curve Matching** and **Pairwise Matching** among fragments. Many potential matches are obtained and then selected/refined in a **Multi-piece Matching** stage to maximize global groupwise matching consistency. This pipeline is effective in composing fragmented objects that are thin-shells and contain small pieces, whose pairwise matching is usually unreliable and ambiguous. Af-

ter 3D reassembly, there may still be large missing regions that need to be filled in. Hence in chapter 4, we propose a new geometric completion algorithm to repair the damaged or missing regions of 3D geometric data based on a novel Local T-distribution based Statistical Shape Model (LTSSM) which relies on the **Surface Matching** and **Template Matching** from the missing data to multiple templates. In this framework, geometric models will first be partitioned into subregions each of which satisfies a T-distribution. A statistical model is constructed to describe each subregion of these models. Then, a given incomplete model will be repaired by the geometry of its undamaged regions with respect to the example datasets. We conduct several experiments to demonstrate the effectiveness of this method. After we obtain the complete skull data, we need to identify that which ancestry of the skull is. In chapter 5, we study the palate shape of the skull, or as it is sometimes called, the dental arcade, as the indicator. For the three main ancestry groups: *white*, *black*, and *Asian/native American*, the dental arcades fall into three categories [12, 13]: the *parabolic* (or triangular), the *hyperbolic* (or rectangular), and the *elliptical* (or rounded), respectively. As three geometrically discrete shapes, it stands to reason that a parabola, hyperbola, and ellipse would lend themselves easily to metric assessment, and, thus, a more objective method for assessing ancestry from the palate could be formulated [14, 15]. However, practical computer-aided identification systems based on this idea do not exist. We model this as a **Curve Matching** and classification problem and use **Template Matching** to evaluate the effectiveness of palate shape analysis in ancestry determination. Finally, since we have the complete skull data and its ancestry information, we can reconstruct the face from the skull. Facial reconstruction for postmortem identifications of humans from their skeletal remains is a long-standing and challenging problem. Recently it is solely carried out by forensic artists

using physical sculpting with clay, which is time-consuming and labor-intensive. In chapter 6, we develop a automatic system based on **Surface Matching** and **Template Matching** to reconstruct the face from the skull. We pick the landmarks on the skull. Given the tissue depths on the skull landmarks, we deform the template face as rigid as possible while the distances between the corresponding skull landmarks and facial landmarks should be equal to the tissue depths. Meanwhile, we maintain the symmetry of the face. We demonstrate the effectiveness of our method on various skulls and template faces.

Our future work includes:

1. Utilize more information to help the pairwise matching, including texture or color, for both 2D and 3D fragment reassembly problem.

2. Use multiple templates to help the template matching for reassembly and facial reconstruction problem.

3. Use a better graph representation, like hyper-graph, in multi-piece matching to model the matching among objects. We believe that using better graph representation can lead to more optimal solution in multi-piece matching part.

BIBLIOGRAPHY

[1] E. Tsamoura and I. Pitas, "Automatic color based reassembly of fragmented images and paintings," *IEEE Trans. on Image Processing*, vol. 19, no. 3, pp. 680–690, 2010.

[2] X. Li and S. Iyengar, "On computing mapping of 3d objects: A survey," *ACM Computing Surveys (CSUR)*, vol. 47, no. 2, p. 34, 2015.

[3] K. Zhang, W. Yu, M. Manhein, W. Waggenspack, and X. Li, "3d fragment reassembly using integrated template guidance and fracture-region matching," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2138–2146, 2015.

[4] S. S. Iyengar, X. Li, H. Xu, S. Mukhopadhyay, N. Balakrishnan, A. Sawant, and P. Iyengar, "Toward more precise radiotherapy treatment of lung tumors," *Computer*, no. 1, pp. 59–65, 2011.

[5] H. Xu and X. Li, "A symmetric 4d registration algorithm for respiratory motion modeling," in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2013*, pp. 149–156, Springer, 2013.

[6] Q.-X. Huang, S. Flöry, N. Gelfand, M. Hofer, and H. Pottmann, "Reassembling fractured objects by geometric matching," in *ACM SIGGRAPH 2006 Papers*, SIGGRAPH '06, (New York, NY, USA), pp. 569–578, ACM, 2006.

[7] J. Wang and M. M. Oliveira, "Filling holes on locally smooth surfaces reconstructed from point clouds," *Image and Vision Computing*, vol. 25, no. 1, pp. 103 – 113, 2007.

[8] B. Allen, B. Curless, and Z. Popović, "The space of human body shapes: Reconstruction and parameterization from range scans," *ACM Trans. Graph.*, vol. 22, pp. 587–594, July 2003.

[9] Z. Yin, L. Wei, M. Manhein, and X. Li, "An automatic assembly and completion framework for fragmented skulls," in *International Conference on Computer Vision (ICCV)*, pp. 2532–2539, 2011.

[10] M. Manhein, G. Listi, R. Barsley, R. Musselman, N. Barrow, and D. Ubelaker, "In vivo facial tissue depth measurements for children and adults," *J Forensic Sci.*, vol. 45, no. 1, pp. 48–60, 2000.

[11] C. Wilkinson, *Forensic Facial Reconstruction*. Cambridge University Press, 2004.

[12] G. Gill, "Challenge on the frontier: discerning american indians from whites osteologically.," *Journal of forensic sciences*, vol. 40, no. 5, p. 783, 1995.

[13] S. Rhine, "Non-metric skull racing," *Skeletal attribution of race: Methods for forensic anthropology*, no. 4, pp. 9–20, 1990.

[14] B. Burris, E. Harris, *et al.*, "Identification of race and sex from palate dimensions.," *Journal of forensic sciences*, vol. 43, no. 5, p. 959, 1998.

[15] S. Byers, S. Churchill, B. Curran, *et al.*, "Identification of euro-americans, afro-americans, and amerindians from palatal dimensions," *Journal of forensic sciences*, vol. 42, pp. 3–9, 1997.

[16] F. Amigoni, S. Gazzani, and S. Podico, "A method for reassembling fragments in image reconstruction," in *Proceedings of International Conference on Image Processing*, vol. 3, pp. III–581–4 vol.2, 2003.

[17] M. Sagiroglu and A. Ercil, "A texture based matching approach for automated assembly of puzzles," in *18th International Conf. on ICPR*, vol. 3, pp. 1036–1041, 2006.

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

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

[20] E. Justino, L. S. Oliveira, and C. Freitas, "Reconstructing shredded documents through feature matching," *Forensic Science International*, vol. 160, no. 2-3, pp. 140 – 147, 2006.

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

[22] H. Wolfson, "On curve matching," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 12, no. 5, pp. 483–489, 1990.

[23] Y. Chen and G. Medioni, "Object modeling by registration of multiple range images," in *IEEE International Conference on Robotics and Automation*, pp. 2724–2729 vol.3, 1991.

[24] S. Rusinkiewicz and M. Levoy, "Efficient variants of the ICP algorithm," in *Third International Conference on 3D Digital Imaging and Modeling (3DIM)*, June 2001.

[25] H. Wolfson, E. Schonberg, A. Kalvin, and Y. Lamdan, "Solving jigsaw puzzles by computer," *Ann. Oper. Res.*, vol. 12, pp. 51–64, Feb. 1988.

[26] G. Ucoluk and I. H. Toroslu, "Automatic reconstruction of broken 3-d surface objects," *Computers and Graphics*, vol. 23, no. 4, pp. 573–582, 1999.

[27] J. C. McBride and B. Kimia, "Archaeological fragment reconstruction using curve-matching," in *Conference on Computer Vision and Pattern Recognition Workshop*, vol. 1, pp. 3–3, 2003.

[28] A. Willis and D. Cooper, "Estimating a-priori unknown 3d axially symmetric surfaces from noisy measurements of their fragments," in *Third International Symposium on 3D Data Processing, Visualization, and Transmission*, pp. 334–341, 2006.

[29] D. B. Cooper, A. Willis, S. Andrews, J. Baker, Y. Cao, D. Han, K. Kang, W. Kong, F. F. Leymarie, X. Orriols, S. Velipasalar, E. L. Vote, M. S. Joukowsky, B. B. Kimia, D. H. Laidlaw, and D. Mumford, "Assembling virtual pots from 3d measurements of their fragments," in *Proceedings of the 2001 conference on Virtual reality, archeology, and cultural heritage*, pp. 241–254, ACM, 2001.

[30] A. R. Willis and D. B. Cooper, "Bayesian assembly of 3d axially symmetric shapes from fragments," *CVPR*, vol. 1, pp. 82–89, 2004.

[31] G. Papaioannou, E.-A. Karabassi, and T. Theoharis, "Virtual archaeologist: assembling the past," *Computer Graphics and Applications, IEEE*, vol. 21, no. 2, pp. 53–59, 2001.

[32] G. Papaioannou and E. aggeliki Karabassi, "A.: On the automatic assemblage of arbitrary broken solid artefacts," *Image and Vision Computing*, vol. 21, pp. 401–412, 2003.

[33] Z. Yin, L. Wei, M. Manhein, and X. Li, "An automatic assembly and completion framework for fragmented skulls," in *International Conference on Computer Vision (ICCV)*, pp. 2532–2539, 2011.

[34] W. Yu, M. Li, and X. Li, "Fragmented skull modeling using heat kernels," *Graphical Models*, vol. 74, no. 4, pp. 140 – 151, 2012.

[35] X. Li, Z. Yin, L. Wei, S. Wan, W. Yu, and M. Li, "Symmetry and template guided completion of damaged skulls," *Computers and Graphics*, vol. 35, no. 4, pp. 885–893, 2011.

[36] L. Wei, W. Yu, M. Li, and X. Li, "Skull assembly and completion using template-based surface matching," in *International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission*, pp. 413–420, 2011.

[37] D. H. Douglas and T. K. Peucker, "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*, vol. 10, pp. 112–122, Oct. 1973.

[38] R. Kummerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "G2o: A general framework for graph optimization," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3607–3613, 2011.

[39] A. Willis and D. Cooper, "Computational reconstruction of ancient artifacts," *Signal Processing Magazine, IEEE*, vol. 25, no. 4, pp. 65–83, 2008.

[40] D. Goldberg, C. Malon, and M. Bern, "A global approach to automatic solution of jigsaw puzzles," in *In Proc. Conf. Computational Geometry*, pp. 82–87, 2002.

[41] F. Tombari, S. Salti, and L. DiStefano, "Performance evaluation of 3d keypoint detectors," *International Journal of Computer Vision*, vol. 102, no. 1-3, pp. 198–220, 2013.

[42] Y. Zhong, "Intrinsic shape signatures: A shape descriptor for 3d object recognition," in *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th Int'l Conf. on*, pp. 689–696, 2009.

[43] A. Mian, M. Bennamoun, and R. Owens, "On the repeatability and quality of keypoints for local feature-based 3d object retrieval from cluttered scenes," *Int. J. Comput. Vision*, vol. 89, no. 2-3, pp. 348–361, 2010.

[44] A. Zaharescu, E. Boyer, K. Varanasi, and R. Horaud, "Surface feature detection and description with applications to mesh matching," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pp. 373 –380, june 2009.

[45] U. Castellani, M. Cristani, S. Fantoni, and V. Murino, "Sparse points matching by combining 3d mesh saliency with statistical descriptors," *Comput. Graph. Forum*, vol. 27, no. 2, pp. 643–652, 2008.

[46] A. Johnson and M. Hebert, "Using spin images for efficient object recognition in cluttered 3d scenes," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 21, no. 5, pp. 433–449, 1999.

[47] S. Ruiz-Correa, L. Shapiro, and M. Melia, "A new signature-based method for efficient 3-d object recognition," in *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, vol. 1, pp. 769 – 776, 2001.

[48] R. M. Rustamov, "Laplace-beltrami eigenfunctions for deformation invariant shape representation," in *Proc. of the fifth Eurographics symp. on Geometry processing*, pp. 225–233, 2007.

[49] J. Sun, M. Ovsjanikov, and L. Guibas, "A concise and provably informative multi-scale signature based on heat diffusion," in *Proceedings of the Symposium on Geometry Processing*, SGP '09, pp. 1383–1392, 2009.

[50] M. Aubry, U. Schlickewei, and D. Cremers, "The wave kernel signature: A quantum mechanical approach to shape analysis," in *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pp. 1626–1633, 2011.

[51] N. Gelfand, N. J. Mitra, L. J. Guibas, and H. Pottmann, "Robust global registration," in *Proceedings of the third Eurographics symposium on Geometry processing*, SGP '05, 2005.

[52] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 24, no. 4, pp. 509–522, 2002.

[53] F. Tombari, S. Salti, and L. Di Stefano, "Unique signatures of histograms for local surface description," in *Proceedings of the 11th European conference on computer vision conference on Computer vision: Part III*, ECCV'10, pp. 356–369, 2010.

[54] A. Berg, T. Berg, and J. Malik, "Shape matching and object recognition using low distortion correspondences," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1, pp. 26–33, 2005.

[55] D. Conte, P. Foggia, C. Sansone, and M. Vento, "Thirty years of graph matching in pattern recognition," *International journal of pattern recognition and artificial intelligence*, vol. 18, no. 03, pp. 265–298, 2004.

[56] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[57] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[58] M. Pauly, R. Keiser, and M. Gross, "Multi-scale feature extraction on point-sampled surfaces," in *CGF*, vol. 22, pp. 281–289, Wiley Online Library, 2003.

[59] D. Aiger, N. J. Mitra, and D. Cohen-Or, "4-points congruent sets for robust pairwise surface registration," in *ACM SIGGRAPH 2008 papers*, SIGGRAPH '08, (New York, NY, USA), pp. 85:1–85:10, ACM, 2008.

[60] P. Besl and N. D. McKay, "A method for registration of 3-d shapes," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 14, no. 2, pp. 239–256, 1992.

[61] S. A. Ehmann and M. C. Lin, "Accurate and fast proximity queries between polyhedra using convex surface decomposition," *Computer Graphics Forum*, vol. 20, no. 3, pp. 500–511, 2001.

[62] M. T. Goodrich, J. S. Mitchell, and M. W. Orletsky, "Practical methods for approximate geometric pattern matching under rigid motions:(preliminary version)," in *Proc. of 10th symp. on Computational geometry*, pp. 103–112, ACM, 1994.

[63] S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Y. Wu, "An optimal algorithm for approximate nearest neighbor searching fixed dimensions," *JACM*, 1998.

[64] S. J. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*. Pearson Education, 2 ed., 2003.

[65] L. Wei, W. Yu, M. Li, and X. Li, "Skull assembly and completion using template-based surface matching," in *3DIMPVT, Intl Conf. on*, pp. 413–420, May 2011.

[66] T. Cootes, C. Taylor, D. Cooper, and J. Graham, "Active shape models-their training and application," *Computer Vision and Image Understanding*, vol. 61, no. 1, pp. 38 – 59, 1995.

[67] C. Basso and T. Vetter, "Statistically motivated 3d faces reconstruction," in *Proceedings of the 2nd international conference on reconstruction of soft facial parts*, vol. 31, Citeseer, 2005.

[68] P. Liepa, "Filling holes in meshes," in *Proceedings of the 2003 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, SGP '03, pp. 200–205, 2003.

[69] U. Clarenz, U. Diewald, G. Dziuk, M. Rumpf, and R. Rusu, "A finite element method for surface restoration with smooth boundary conditions," *Computer Aided Geometric Design*, vol. 21, no. 5, pp. 427 – 445, 2004.

[70] J.-P. Pernot, G. Moraru, and P. VšŠron, "Filling holes in meshes using a mechanical model to simulate the curvature variation minimization," *Computers and Graphics*, vol. 30, no. 6, pp. 892 – 902, 2006.

[71] C. Xiao, W. Zheng, Y. Miao, Y. Zhao, and Q. Peng, "A unified method for appearance and geometry completion of point set surfaces," *The Visual Computer*, vol. 23, no. 6, pp. 433–443, 2007.

[72] J. Branch, F. Prieto, and P. Boulanger, "A hole-filling algorithm for triangular meshes using local radial basis function," in *Proceedings of the 15th International Meshing Roundtable*, pp. 411–431, 2006.

[73] C.-Y. Chen and K.-Y. Cheng, "A sharpness-dependent filter for recovering sharp features in repaired 3d mesh models," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 14, pp. 200–212, Jan 2008.

[74] T. Ju, "Robust repair of polygonal models," in *ACM SIGGRAPH 2004 Papers*, SIGGRAPH '04, pp. 888–895, ACM, 2004.

[75] J. Davis, S. Marschner, M. Garr, and M. Levoy, "Filling holes in complex surfaces using volumetric diffusion," in *3D Data Processing Visualization and Transmission, 2002. Proceedings. First International Symposium on*, pp. 428–441, June 2002.

[76] J. Verdera, V. Caselles, M. Bertalmio, and G. Sapiro, "Inpainting surface holes," in *In Int. Conference on Image Processing*, pp. 903–906, 2003.

[77] T.-Q. Guo, J.-J. Li, J.-G. Weng, and Y. ting Zhuang, "Filling holes in complex surfaces using oriented voxel diffusion," in *Machine Learning and Cybernetics, 2006 International Conference on*, pp. 4370–4375, Aug 2006.

[78] S. Bischoff, D. Pavic, and L. Kobbelt, "Automatic restoration of polygon models," *ACM Trans. Graph.*, vol. 24, pp. 1332–1352, Oct. 2005.

[79] A. Sharf, M. Alexa, and D. Cohen-Or, "Context-based surface completion," *ACM Trans. Graph.*, vol. 23, pp. 878–887, Aug. 2004.

[80] M. Pauly, N. J. Mitra, J. Wallner, H. Pottmann, and L. J. Guibas, "Discovering structural regularity in 3d geometry," *ACM Trans. Graph.*, vol. 27, pp. 43:1–43:11, Aug. 2008.

[81] V. Kraevoy and A. Sheffer, "Template-based mesh completion.," in *Symposium on Geometry Processing*, pp. 13–22, 2005.

[82] M. Pauly, N. J. Mitra, J. Giesen, M. H. Gross, and L. J. Guibas, "Example-based 3d scan completion," in *Symposium on Geometry Processing*, no. EPFL-CONF-149337, pp. 23–32, 2005.

[83] R. Gal, A. Shamir, T. Hassner, M. Pauly, and D. Cohen-Or, "Surface reconstruction using local shape priors," in *Symposium on Geometry Processing*, no. EPFL-CONF-149318, pp. 253–262, 2007.

[84] T. Heimann and H.-P. Meinzer, "Statistical shape models for 3d medical image segmentation: a review," *Medical image analysis*, vol. 13, no. 4, pp. 543–563, 2009.

[85] T. Ju, "Fixing geometric errors on polygonal models: a survey," *Journal of Computer Science and Technology*, vol. 24, no. 1, pp. 19–29, 2009.

[86] M. Attene, M. Campen, and L. Kobbelt, "Polygon mesh repairing: An application perspective," *ACM Computing Surveys (CSUR)*, vol. 45, no. 2, p. 15, 2013.

[87] T. F. Cootes, G. J. Edwards, and C. J. Taylor, "Active appearance models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, no. 6, pp. 681–685, 2001.

[88] M. E. Tipping and C. M. Bishop, "Probabilistic principal component analysis," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 61, no. 3, pp. 611–622, 1999.

[89] Y. Wang and L. H. Staib, "Boundary finding with prior shape and smoothness models," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 22, no. 7, pp. 738–743, 2000.

[90] J. Lötjönen, K. Antila, E. Lamminmäki, J. Koikkalainen, M. Lilja, and T. Cootes, "Artificial enlargement of a training set for statistical shape models: Application to cardiac images," in *Functional Imaging and Modeling of the Heart*, pp. 92–101, Springer, 2005.

[91] C. Davatzikos, X. Tao, and D. Shen, "Hierarchical active shape models, using the wavelet transform," *Medical Imaging, IEEE Transactions on*, vol. 22, no. 3, pp. 414–423, 2003.

[92] Z. Zhao, S. R. Aylward, and E. K. Teoh, "A novel 3d partitioned active shape model for segmentation of brain mr images," in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2005*, pp. 221–228, Springer, 2005.

[93] T. F. Cootes and C. J. Taylor, "A mixture model for representing shape variation," *Image and Vision Computing*, vol. 17, no. 8, pp. 567–573, 1999.

[94] J. Zhao and Q. Jiang, "Probabilistic pca for t distributions," *Neurocomputing*, vol. 69, no. 16, pp. 2217–2226, 2006.

[95] V. Kraevoy and A. Sheffer, "Cross-parameterization and compatible remeshing of 3D models," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 861–869, 2004.

[96] E. Praun, W. Sweldens, and P. Schröder, "Consistent mesh parameterizations," in *SIGGRAPH*, pp. 179–184, 2001.

[97] X. Li, Y. Bao, X. Guo, M. Jin, X. Gu, and H. Qin, "Globally optimal surface mapping for surfaces with arbitrary topology," *IEEE Trans. on Visualization and Computer Graphics*, vol. 14, no. 4, pp. 805–819, 2008.

[98] E. Rodola, S. Bulo, T. Windheuser, M. Vestner, and D. Cremers, "Dense non-rigid shape correspondence using random forests," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4177–4184, 2014.

[99] Q. Chen and V. Koltun, "Robust nonrigid registration by convex optimization," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2039–2047, 2015.

[100] O. Kaick, H. Zhang, G. Hamarneh, and D. Cohen-Or, "A survey on shape correspondence," *Comput. Graph. Forum*, vol. 30, no. 6, pp. 1681–1707, 2011.

[101] T. W. Anderson and D. A. Darling, "Asymptotic theory of certain" goodness of fit" criteria based on stochastic processes," *The annals of mathematical statistics*, pp. 193–212, 1952.

[102] D. L. James and C. D. Twigg, "Skinning mesh animations," in *ACM Transactions on Graphics (TOG)*, vol. 24, pp. 399–407, ACM, 2005.

[103] V. Blanz, A. Mehl, T. Vetter, and H.-P. Seidel, "A statistical method for robust 3d surface reconstruction from sparse data," in *3D Data Processing, Visualization and Transmission, 2004. 3DPVT 2004. Proceedings. 2nd International Symposium on*, pp. 293–300, IEEE, 2004.

[104] F. Bogo, J. Romero, M. Loper, and M. J. Black, "FAUST: Dataset and evaluation for 3D mesh registration," in *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, (Piscataway, NJ, USA), IEEE, June 2014.

[105] Y. Yang, Y. Yu, Y. Zhou, S. Du, J. Davis, and R. Yang, "Semantic parametric reshaping of human body models," in *3D Vision (3DV), 2014 2nd International Conference on*, vol. 2, pp. 41–48, IEEE, 2014.

[106] G. Üçoluk and I. HakkI Toroslu, "Automatic reconstruction of broken 3-d surface objects," *Computers and Graphics*, vol. 23, no. 4, pp. 573–582, 1999.

[107] H. Sundar, D. Silver, N. Gagvani, and S. Dickinson, "Skeleton based shape matching and retrieval," in *Shape Modeling International, 2003*, pp. 130–139, IEEE, 2003.

[108] H. Bunke, M. Roth, and E. Schukat-Talamazzini, "Off-line cursive handwriting recognition using hidden markov models," *Pattern recognition*, vol. 28, no. 9, pp. 1399–1413, 1995.

[109] K. Meenakshi, S. Srihari, and A. Xu, "Offline signature verification and identification using distance statistics," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 18, no. 07, pp. 1339–1360, 2004.

[110] B. Fang, Y. Wang, C. Leung, K. Tse, Y. Tang, P. Kwok, and Y. Wong, "Offline signature verification by the analysis of cursive strokes," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 15, no. 04, pp. 659–673, 2001.

[111] V. Ferrari, L. Fevrier, F. Jurie, and C. Schmid, "Groups of adjacent contour segments for object detection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 30, no. 1, pp. 36–51, 2008.

[112] J. Hefner, "Cranial nonmetric variation and estimating ancestry*," *Journal of forensic sciences*, vol. 54, no. 5, pp. 985–995, 2009.

[113] "*microscribe*^{TM} 3d digitizer @ONLINE."

[114] G. E. Farin, *Curves and surfaces for computer-aided geometric design: a practical code*. Academic Press, Inc., 1996.

[115] N. Cristianini and J. Shawe-Taylor, *An introduction to support vector machines and other kernel-based learning methods*. Cambridge university press, 2000.

[116] K.-B. Duan and S. Keerthi, "Which is the best multiclass svm method? an empirical study," in *Multiple Classifier Systems* (N. Oza, R. Polikar, J. Kittler, and F. Roli, eds.), vol. 3541 of *Lecture Notes in Computer Science*, pp. 278–285, Springer Berlin Heidelberg, 2005.

[117] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software: An update," *SIGKDD Explor. Newsl.*, vol. 11, pp. 10–18, Nov. 2009.

[118] G. Lebedinskaya, T. Balueva, and E. Veselovskaya, "Principles of facial reconstruction," *Forensic analysis of the skull*, pp. 183–198, 1993.

[119] B. A. Fedosyutkin and J. V. Nainys, "The relationship of skull morphology to facial features," *Forensic Analysis of the Skull: Craniofacial Analysis, Reconstruction, and Identification*, pp. 119–213, 1993.

[120] W. J. Moore and C. L. Lavelle, *Growth of the Facial Skeleton in the Hominoidea*. Academic Press, 1974.

[121] W. Weijs and B. Hillen, "Correlations between the cross-sectional area of the jaw muscles and craniofacial size and shape," *American Journal of Physical Anthropology*, vol. 70, no. 4, pp. 423–431, 1986.

[122] K. T. Taylor, *Forensic art and illustration*. CRC Press, 2000.

[123] M. M. Gerasimov, "face finder,"

[124] C. C. Snow, B. P. Gatliff, and K. R. McWilliams, "Reconstruction of facial features from the skull: an evaluation of its usefulness in forensic anthropology," *American Journal of Physical Anthropology*, vol. 33, no. 2, pp. 221–227, 1970.

[125] R. NEAVE, *Making faces: using forensic and archaeological evidence*. 1997.

[126] C. Wilkinson, *Forensic facial reconstruction*. Cambridge University Press, 2004.

[127] S. Michael and M. Chen, "The 3d reconstruction of facial features using volume distortion," in *Proc. 14th Eurographics UK Conference*, vol. 305, 1996.

[128] K. M. Archer, *Craniofacial reconstruction using hierarchical B-spline interpolation*. PhD thesis, University of British Columbia, 1997.

[129] P. Vanezis, M. Vanezis, G. McCombe, and T. Niblett, "Facial reconstruction using 3-d computer graphics," *Forensic science international*, vol. 108, no. 2, pp. 81–95, 2000.

[130] K. Kähler, J. Haber, and H.-P. Seidel, "Reanimating the dead: reconstruction of expressive faces from skull data," in *ACM Transactions on Graphics (TOG)*, vol. 22, pp. 554–561, ACM, 2003.

[131] P. Claes, D. Vandermeulen, S. De Greef, G. Willems, J. G. Clement, and P. Suetens, "Computerized craniofacial reconstruction: conceptual framework and review," *Forensic science international*, vol. 201, no. 1, pp. 138–145, 2010.

[132] P. J. Besl and N. D. McKay, "Method for registration of 3-d shapes," in *Robotics-DL tentative*, pp. 586–606, International Society for Optics and Photonics, 1992.

[133] S. Rusinkiewicz and M. Levoy, "Efficient variants of the icp algorithm," in *3-D Digital Imaging and Modeling, 2001. Proceedings. Third International Conference on*, pp. 145–152, IEEE, 2001.

[134] N. Gelfand, N. J. Mitra, L. J. Guibas, and H. Pottmann, "Robust global registration," in *Symposium on geometry processing*, vol. 2, p. 5, 2005.

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

[136] K. Zhang and X. Li, "A graph-based optimization algorithm for fragmented image reassembly," *Graph. Models*, vol. 76, no. 5, pp. 484–495, 2014.

[137] K. Zhang, W. Yu, M. Manhein, W. Waggenspack, and X. Li, "3d fragment reassembly using integrated template guidance and fracture-region matching," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2138–2146, 2015.

[138] S. Zheng, J. Hong, K. Zhang, B. Li, and X. Li, "A multi-frame graph matching algorithm for low-bandwidth rgb-d slam," *Computer-Aided Design*, 2016.

[139] B. Allen, B. Curless, and Z. Popović, "The space of human body shapes: reconstruction and parameterization from range scans," in *ACM transactions on graphics (TOG)*, vol. 22, pp. 587–594, ACM, 2003.

[140] B. J. Brown and S. Rusinkiewicz, "Global non-rigid alignment of 3-d scans," *ACM Transactions on Graphics (TOG)*, vol. 26, no. 3, p. 21, 2007.

[141] N. J. Mitra, S. Flöry, M. Ovsjanikov, N. Gelfand, L. J. Guibas, and H. Pottmann, "Dynamic geometry registration," in *Symp. on geometry processing*, pp. 173–182, 2007.

[142] Q.-X. Huang, B. Adams, M. Wicke, and L. J. Guibas, "Non-rigid registration under isometric deformations," in *Computer Graphics Forum*, vol. 27, pp. 1449–1457, 2008.

[143] X. Li, X. Gu, and H. Qin, "Surface mapping using consistent pants decomposition," *IEEE Trans. on Visualization and Computer Graphics*, vol. 15, no. 4, pp. 558–571, 2009.

[144] X. Li, X. Guo, H. Wang, Y. He, X. Gu, and H. Qin, "Harmonic volumetric mapping for solid modeling applications," in *Proc. ACM symp. on Solid and physical modeling*, pp. 109–120, 2007.

[145] X. Li, X. Guo, H. Wang, Y. He, X. Gu, and H. Qin, "Meshless harmonic volumetric mapping using fundamental solution methods," *IEEE Trans. on Automation Science and Engineering*, vol. 6, no. 3, pp. 409–422, 2009.

[146] O. Sorkine and M. Alexa, "As-rigid-as-possible surface modeling," in *Symposium on Geometry processing*, vol. 4, 2007.

[147] G. K. Tam, Z.-Q. Cheng, Y.-K. Lai, F. C. Langbein, Y. Liu, D. Marshall, R. R. Martin, X.-F. Sun, and P. L. Rosin, "Registration of 3d point clouds and meshes: a survey from rigid to nonrigid," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 19, no. 7, pp. 1199–1217, 2013.

[148] X. Li, Z. Yin, L. Wei, S. Wan, W. Yu, and M. Li, "Symmetry and template guided completion of damaged skulls," *Computers and Graphics*, vol. 35, no. 4, pp. 885–893, 2011.

[149] C. N. Stephan and E. K. Simpson, "Facial soft tissue depths in craniofacial identification (part i): an analytical review of the published adult data," *Journal of Forensic Sciences*, vol. 53, no. 6, pp. 1257–1272, 2008.

VITA

Kang Zhang was born in the city of Chengdu, Sichuan Province in China in 1988. He graduate from University of Science and Technology of China, Hefei, China in July 2010 with the bachelor's degree in Computer Science. In 2010, he began the doctoral program in the Department of Electrical and Computer Engineering from Louisiana State University, Baton Rouge, Louisiana. Kang Zhang will graduate in June 2016 with the degree of Doctor of Philosophy in Electrical and Computer Engineering.