ABSTRACT

BALOCH, SAJJAD H. Shape Modeling and Analysis for Object Representation, Reconstruction, and Recognition. (Under the direction of Professor Hamid Krim).

Shape Modeling constitutes a fundamental problem in computer vision. Complexity of the problem arises from the variability of shape realizations, which may be due to their inherent variability or introduced because of adverse situations like noise, pose problem, occlusion, etc. In this thesis, we address this fundamental problem for 2D and 3D shapes in statistical as well as algorithmic settings.

In a probabilistic setting, we present a novel method for 2D shape modeling and template learning, which we call Flexible Skew-symmetric Shape Model (FSSM). It uses an extended class of semiparametric skew-symmetric distributions. The proposed model aims at capturing the inherent variability of shapes so long as the realization contours remain within a certain neighborhood range around a “mean” with high probability. It is flexible enough to capture the non-Gaussianity of underlying data, and allows automatic selection of landmarks. We explore several applications of FSSM, such as, sampling new shapes, learning templates, and classifying shapes.

The algorithmic 2D and 3D shape models are formulated in a Morse theoretic framework, where shapes of arbitrary topology are represented completely by topo-geometric graphs. The idea is to capture topology by localizing critical points of distance function as the Morse function, thereby representing it through skeletal graphs. Geometry, on the other hand, is captured by tracking radii of the corresponding level curves of the distance function (for planar shapes), or by modeling the evolution of these level curves (for 3D shapes). This leads to a weighted skeletal representation, which is then employed for reconstruction, and recognition applications.
Shape Modeling and Analysis for Object Representation, Reconstruction, and Recognition

by

Sajjad H. Baloch

A dissertation submitted to the Graduate Faculty of
North Carolina State University
in partial satisfaction of the
requirements for the Degree of
Doctor of Philosophy

Electrical Engineering

Raleigh

2006

Approved By:

Dr. Alexandra Duel-Hallen
Dr. Irina Kogan

Dr. Hamid Krim
Chair of Advisory Committee
Dr. Brian Hughes
To the humanity.
Biography

Sajjad H. Baloch received his BSc degree in electrical engineering from University of Engineering and Technology, Taxila, Pakistan in 1997 and then MSc degree in systems engineering from Quaid-i-Azam University, Pakistan in 1999, where he secured first position in the department and was awarded gold medal by the President of Pakistan.

From 1997-1999, he was a research fellow at Pakistan Institute of Engineering and Applied Sciences, Pakistan. In 1999, he joined Instrumentation and Computer Control Complex as a systems engineer and also as a consultant to Elixir Technologies, Pakistan. In 2002, he started his PhD studies in Electrical Engineering at North Carolina State University, Raleigh, NC. His research interests include shape modeling and analysis, pattern analysis and recognition, data mining and statistical signal and image processing.
Acknowledgements

I would like to express my gratitude to all those who gave me the possibility to complete this thesis. First, I am deeply indebted to my supervisor Dr. Hamid Krim for his immense guidance and never ending support throughout my doctoral research. His stimulating suggestions and ideas, continued mentoring and encouragement made my research really enjoyable. I am also greatly thankful to Dr. Marc Genton, Drs. Irina Kogan, and Dmitry Zenkov for continuously providing insight, and direction. I also express my gratitude to Drs. Brian Hughes, and Alexandra Duel-Hallen for serving on my advisory committee, for their interest in the topic with further suggestions.

I am grateful to my friends, who benefitted me both academically and personally. Especially, I thank my group members Ben, Alex, Shuo, Djamila, Vaibhav, Yang, and Aysegul for healthy discussions, and unforgettable light moments. I was also lucky enough to have met some wonderful people at NC State, like Mr. Michael Albright, Ms. Tiffany Williams, and Brenda Beaty, who were always willing to help.

I would like to thank my family whom I owe so much – my parents for supporting and believing in me, my siblings for their love and care, my life-long friends for making my life so much enjoyable throughout, my beloved wife, Rukhsana, for being on my side in pain and happiness, and my wonderful children Haider and Farwa for bringing so much joy to me.
Contents

List of Figures viii
List of Tables xiii

1 Introduction 1
  1.1 Shape Descriptors ................................................. 3
  1.2 Proposed Shape Models ........................................... 4
  1.3 Organization of the thesis ....................................... 6

2 Literature Review 7
  2.1 Statistical Methods ............................................. 7
  2.2 Algorithmic Methods ............................................. 9
    2.2.1 3D Shape Models ............................................ 11
  2.3 Contributions of This Thesis .................................... 12
    2.3.1 Probabilistic Shape Modeling ................................. 12
    2.3.2 2D Morse Theoretic Shape Model ............................. 14
    2.3.3 3D Topo-Geometric Shape Model ............................. 14

3 Preliminaries 16
  3.1 Planar Shapes .................................................. 16
  3.2 Graph Theory ................................................... 18
    3.2.1 Decomposition based Subgraph Isomorphism .................. 19
    3.2.2 Combining Decompositions .................................. 20
  3.3 Morse Theory .................................................... 21
    3.3.1 Handle Decomposition ....................................... 24
    3.3.2 Critical Points and Topology ................................ 27

4 Probabilistic Modeling of 2D Shapes 28
  4.1 Motivation ..................................................... 29
  4.2 Admissible Shapes ............................................... 30
  4.3 Problem Statement ............................................... 31
    4.3.1 Posterior Distribution ...................................... 33
    4.3.2 Posterior Learning .......................................... 35
    4.3.3 Angle Distribution .......................................... 35
4.3.4 Overall Shape Distribution ........................................ 36
4.3.5 Template Learning ................................................. 37
4.3.6 Performance Assessment ........................................ 37
4.4 Classification of Shapes ............................................. 39
4.4.1 Sampling from Models ........................................... 39
4.5 Experimental Results ................................................ 41
4.5.1 Template Learning ................................................ 41
4.5.2 Sampling from Models ........................................... 43
4.5.3 Classification ..................................................... 44
4.5.4 Application to Brown Database of Shapes ..................... 45
4.6 Connection with Principal Curves .................................. 47
4.7 Conclusions .......................................................... 51

5 2D Shape Modeling in Morse Theoretic Framework .......... 53
5.1 Motivation and Background ........................................... 54
5.2 Topological Model .................................................... 54
5.2.1 The Distance Function ......................................... 55
5.2.2 Analysis of Planar Shapes ..................................... 56
5.2.3 Algorithm ....................................................... 57
5.3 Geometric Modeling .................................................. 61
5.3.1 Geometric Encoding of Vertices ............................... 61
5.4 Experimental Results ................................................ 63
5.4.1 Examples ........................................................ 63
5.4.2 Application to Planar Shapes .................................. 64
5.5 Conclusions .......................................................... 64

6 Topo-Geometric Modeling of 3D Shapes ....................... 67
6.1 Topological Model .................................................... 68
6.1.1 Scaling and Rigid Motion Invariant Skeletal Graph .......... 68
6.1.2 Equations for the Critical Points of the Distance Function 70
6.1.3 Algorithmic Approach to Topological Analysis ............... 71
6.2 Geometric Model ..................................................... 73
6.2.1 Curve Evolution Model ....................................... 74
6.2.2 Dimension reduction .......................................... 78
6.2.3 Implementation Issues ........................................ 80
6.3 Topo-Geometric Model ............................................. 81
6.3.1 Approximation of a Curve in $\mathbb{R}^n$ .................... 81
6.3.2 Approximation of Elastic Segment in $\mathbb{R}^3$ ............ 82
6.3.3 Piecewise Projection .......................................... 82
6.4 Sampling and level curve pruning ............................... 85
6.5 Experimental Results ............................................... 89
6.5.1 Topological modeling ......................................... 89
6.5.2 Geometric model .............................................. 91
6.6 Conclusions ........................................................ 94
7 Shape Recognition by Graph Edit Operations

7.1 Graph Deformation .............................................. 95
7.2 Cost of Edit Operations ........................................ 98
  7.2.1 Constant Cost ............................................. 98
  7.2.2 Variable Cost ............................................. 98
7.3 Experimental Results .......................................... 100
  7.3.1 Model Learning ............................................ 100
  7.3.2 Shape Recognition ......................................... 100
7.4 Conclusions .................................................... 112

8 Conclusions

8.1 Contributions of The Thesis ................................... 115
  8.1.1 Flexible Skew Symmetric Shape Model ..................... 115
  8.1.2 2D Morse Theoretic Model ................................ 116
  8.1.3 Topo-Geometric Shape Model ............................... 116
  8.1.4 Classification of Shapes from Skeletal Representations 117
8.2 Future Research ............................................... 117
  8.2.1 Extended Flexible Skew-symmetric Shape Model .......... 117
  8.2.2 Shape Recognition through Topo-Geometric Model .......... 118
  8.2.3 Extension to Topo-Geometric Model ......................... 118

Bibliography
# List of Figures

1.1 A typical vision system. .................................................. 2
1.2 Fundamental tasks in machine vision: (a) Denoised images; (b) Edge map; (c) Segmented images; (d) Feature extraction; (e) Shape models in a database. 2

3.1 Graph Decomposition: (a) Model graph $g_1$; (b) Model graph $g_2$; (c) Test graph $g$; (d) Decomposition of $g$ and subsequent combination to find subgraph isomorphism. .................................................. 22
3.2 Critical points of a height function defined on a Manifold $\mathcal{M}$. ................. 24
3.3 Study of a manifold $\mathcal{M}$ with handle decomposition. .............................. 25
3.4 $\epsilon$-Neighborhood of a 0-index critical point. ........................................ 25
3.5 $\epsilon$-Neighborhood of a 1-index critical point. ........................................ 26
3.6 $\epsilon$-Neighborhood of a 2-index critical point. ........................................ 26
3.7 (a) Symbology for different handles; (b) Handle decomposition of the surface given in Fig. 3.3. .............................. 26

4.1 (a) Heart shape; (b) Some realizations superimposed on each other; (c) Sampled superimposed realizations; (d) Constituent realizations. .......... 29
4.2 Circular cloud sliced at angle $\theta = 0$ along with $\epsilon$-neighborhood, $N_\epsilon^0$. ........ 32
4.3 Examples of conditional histograms for distance $r$ given some angle $\Theta = \theta$: $r$-axis is normalized. (a) Skewed normal and skew-$t$ modes for star cloud (Fig. 4.8(b)) given $\theta = 0$; (b) Normal and skewed modes for triangle cloud (Fig. 4.12(b)) given $\theta = \pi/2$. ................. 33
4.4 Singularities and angle distribution: (a) Singularities in shape at angle $\theta = \pi/6, \pi/2, 5\pi/6$; (b) Piecewise uniform distribution; (b) PUT distribution. .... 36
4.5 Illustration of using modes as boundary points: (a) Star shape; (b) Bimodal distributions learned with 100 angles. .............................. 38
4.6 Shape space partitioning via shape distributions. ................. 39
4.7 Square shape: (a) Ideal shape; (b) Simulated realizations; (c) Learned Template using 10 uniform angle samples; (d) Template using 10 PUT angle samples; (e) and (f) Template using 100 angle samples. .......... 41
4.8 Star shape: (a) Ideal shape; (b) Simulated realizations; (c) Template learned with uniform distribution for the angle; (d) Template learned with 30 PUT angle-samples; (e) and (f) Template using 100 angle samples. .......... 41
4.9 Star-II shape: (a) Ideal Star-II shape; (b) Simulated realizations; (c) Template learned with 50 uniform angle samples; (d) Template learned using 50 samples from PUT distribution, given in Fig. 4.10; (e) and (f) Template learned with 100 angle samples.

4.10 PUT distribution $p(\theta)$ for angle $\Theta$: (a) Square shape; (b) Star-II shape.

4.11 Heart shape: Learned templates. (a) Using 20 angle samples drawn from a uniform distribution; (b) Using 20 angle samples drawn from a PUT distribution, given in Fig. 4.10; (c) Template learned with 100 angle samples.

4.12 Triangle shape: (a) Ideal shape; (b) Simulated realizations; (c) Template learned with uniform distribution for the angle; (d) Template learned with 10 PUT angle-samples; (e) and (f) Template using 100 angle samples.

4.13 Brain shape: (a) Human brain; (b) Brain contour; (c) Simulated realizations; (d) Learned template.

4.14 Truck shape: (a) Ideal truck shape; (b) Simulated realizations of truck; (c) Learned template.

4.15 Shape simulation according to the distribution of Eq. 4.14: (a) Ideal shape; (b) Realizations; (c) Realizations simulated using parameters learned from (b).

4.16 Standard deviation $\sigma_s$ of closure pdf $p(r|r_s)$.

4.17 Star and Star-II shapes simulated according to Eq. 4.22: (a) Star without Markov chain; (b) Star with $\sigma_0 = 1$; (c) Star with $\sigma_0 = 0.09$; (d) Star with $\sigma_0 = 0.04$; (e) Star-II simulated with $\sigma_0 = 0.1$; (f) Star-II simulated with $\sigma_0 = 0.04$.

4.18 (a) Test shapes from Brown database (Heart, Crown, F-18, Car, Face, Brick, Misk, Key, Flat fish, Bottle, Text box, Mig-29, Classic, Fgen, and Children); (b) Learned Templates.

4.19 Comparison of normalized likelihoods of templates given in Fig. 4.18 (Rows:shapes, columns:model). Normalized likelihood varies from 0 (black) to 1 (white).

4.20 Test shapes from Kimia database.

4.21 Classification of shapes in Fig. 4.20.

4.22 Similarity scores: (Right) Test shape; (Middle Right) Best match; (Middle Left) Second best match; (Left) Third best match.

4.23 Similarity scores: (Right) Test shape; (Middle Right) Best match; (Middle Left) Second best match; (Left) Third best match.

4.24 Equivalence between principal curves (blue) and FSSM (red).

5.1 Importance of topological information: (a) a nut; (b) Silhouette of a nut with minimal topological information; (c) Segmented nut with all topological information.

5.2 Skeletal graph of an eight shape: (a) shape analyzed with an evolving circle; (b) intersections of the circle and the shape; (c) vertex assignment in the graph.

5.3 Topological analysis of a 2D shape $\mathcal{M}$, where $\mathcal{M}_1, \mathcal{M}_2$ and $\mathcal{M}_3$ define the boundary of $\mathcal{M}$. (a) Critical points of $\mathcal{M}_1, \mathcal{M}_2$ and $\mathcal{M}_3$; (b) handle decomposition of $\mathcal{M}$.

5.4 Skeletonization of a 2D shape $\mathcal{M}$. Note that $\mathcal{A}_c = \mathcal{A}_{c_1} \cup \mathcal{A}_{c_2} \cup \mathcal{A}_{c_3} \cup \mathcal{A}_{c_4}$, and $\mathcal{A}_p = \mathcal{A}_{p_1} \cup \mathcal{A}_{p_2} \cup \mathcal{A}_{p_3} \cup \mathcal{A}_{p_4}$, $\mathcal{M} = \mathcal{M}_1 \cup \mathcal{M}_2 \cup \mathcal{M}_3 \cup \mathcal{M}_4$. 

5.5 Skeletonization of an eight shape with various graph resolutions $K$: (a) $K = 4$; (b) $K = 5$; (c) $K = 7$.  
5.6 Skeletonization of a kettle: (a) $K = 4$; (b) $K = 16$.  
5.7 Skeletonization of an airplane: (a) $K = 4$; (b) $K = 16$; (c) $K = 64$.  
5.8 New definition of graph vertex.  
5.9 Reconstruction of shape.  
5.10 Illustration of difference between two vertex definitions ($N = 0.5(C + D)$).  
5.11 Illustration of difference between two vertex definitions ($N = 0.4C + 0.6D$).  
5.12 Reconstructed boundary points $A$ and $B$.  
5.13 Skeletonization of a kettle: (a) old definition of a vertex; (b) new definition. (Left) $K = 4$; (center) $K = 5$; (right) $K = 16$.  
5.14 Skeletonization: (a) old definition of a vertex; (b) new definition. (Left) horse, $K = 5$; (right) airplane, $K = 8$.  
5.15 Skeletonization: (a) frog, $K = 16$; (b) camel, $K = 16$; (c) airplane, $K = 100$.  
5.16 Reconstruction of shape landmarks: (a) $K = 8$; (b) $K = 100$.  
5.17 Contour fitting: (a) eight shape; (b) horse shape.  
6.1 Weighted skeletal graph: Graph captures the topology while geometry captured via the weights assigned $W_i$.  
6.2 Skeletal graph of double torus: (a) Surface analyzed with an evolving sphere; (b) Intersections of the sphere and the surface; (c) Node assignment in the graph.  
6.3 Skeletonization of a 3D surface $\mathcal{M}$.  
6.4 Simplification of a skeletal graph: (a) A graph learned by the algorithm of Section 6.2. ; (b) Simplified graph. Note how a topologically homogeneous sequence of edge segments marked in (a) maps to an edge in (b).  
6.5 Illustration of geometric modeling: (a) A 3D object; (b) Object sampled at levels $r_1, \ldots, r_m$; (c) Intersections $C_1, \ldots, C_m$ embedded in $\Lambda$ bounding box to compute the distance field; (d) Vectorizing the elements of the distance field yields an $n$ dimensional vector $\rho_i$ for each $C_i$.  
6.6 A level curve of double torus: (a) Curve lies in $\mathbb{R}^3$; (b) Projection to $D \subset \mathbb{R}^2$ via spherical coordinate transformation.  
6.7 Transformation to spherical coordinates: (a) A curve whose projection to $\theta \phi$ plane is not closed; (b) A rotation in $\mathbb{R}^3$ makes the projection closed.  
6.8 Signed distance field: (a) Original curve; (b) Level sets of distance field.  
6.9 Optimal trajectory between each pair of curve vector: (a) Elastic fitted between $\rho_1$ and $\rho_2$ with constraints end point constraints $v_1$ and $v_2$; (b) Problem transformed to $\mathbb{R}^3$, where tangential constraints become $w_1$ and $w_2$ and the displacement between end points is mapped to $w_3$.  
6.10 Computing tangents.  
6.11 Effect of basis rotation: (a) See how $\alpha_4$ and $\alpha_5$ map to $\alpha_4'$ and $\alpha_5'$ after projection from $\mathbb{R}^n$ to $\mathbb{R}^3$, resulting in a high curvature cusp at the node; (b) Rotating the basis at the nodes makes the Frenet frame smoother and the resulting trajectory is first order smooth.
6.12 Piecewise projection: (a) A trajectory in $\mathbb{R}^n$; (b) Its piecewise projection onto $\mathbb{R}^3$. 
6.13 Multiresolution tree for curves $\rho_1, \ldots, \rho_8$. Note that resolution is finer at higher levels. 
6.14 Curves on multiresolution tree after pruning. 
6.15 Illustration of multiresolution modeling of $\alpha_2, \alpha_3, \alpha_4$: (a) $\alpha_{23}(t)$ fitted at fine level for the first modeling interval; (b) $\alpha(p)(t) = \alpha_{24}(t)$ for both intervals. Note that the basis is rotated by angle $\theta$ to align starting tangent $v_2(s)$ of the second arc along the ending tangent $v_{1}(e)$ of the first arc. This results in the rotation of $v_2(e)$ by $\theta$; (c) $\alpha(p)(t) = \alpha_{24}(t)$ fitted at coarse level. $\alpha_3$ is dropped and the resulting trajectory has a curvature less than that at fine level. 
6.16 Extended Reeb graph for double torus along with level curves. 
6.17 Extended Reeb graphs for double torus: (a) $K = 4$; (b) $K = 8$; (c) $K = 16$. 
6.18 Extended Reeb graphs: (a) An airplane; (b) A horse. 
6.19 Extended Reeb graphs for a camel: (a) $K = 8$; (b) $K = 16$; (c) $K = 32$. 
6.20 Rotation invariance of extended Reeb graph: (a) No rotation; (b) Rotation by $\pi/2$; (c) Rotation by $3\pi/4$. 
6.21 Simplified skeletal graph: (a) Original; (b) After simplification; (c) Object color coded according to simplified graph. 
6.22 Surface modeling for double torus: (a) Level curves of edge $e$; (b) Reconstructed subsurface $e$ from the model; (c) Level curve set; (d) Reconstructed double torus. 
6.23 Trajectory for edge $e$ in $\mathbb{R}^3$: (a) Without basis rotation; (b) Smooth Frenet frame along the curve. 
6.24 Polynomial representation of the trajectory in parametric form in $\mathbb{R}^3$: (a) Edge $a$; (b) Edge $e$; (c) Edge $i$. 
6.25 Taylor polynomial coefficients for the three surface trajectories: (a) $x$-coefficients; (b) $y$-coefficients; (c) $z$-coefficients. 
6.26 Reconstruction of the Twins: (a) Original object; (b) Color coded level curves; (c) Reconstructed object. 
6.27 Reconstruction of the Bull: (a) Original object; (b) Color coded level curves; (c) Reconstructed object. 
7.1 Skeletal representation of 3D shapes. 
7.2 Skeletal representation of 3D shapes. 
7.3 Skeletal representation of 3D shapes. 
7.4 Skeletal representation of 3D shapes. 
7.5 Skeletal representation of 3D shapes. 
7.6 Skeletal representation of 3D shapes. 
7.7 Skeletal representation of 3D shapes. 
7.8 Skeletal representation of planar shapes. 
7.9 Skeletal representation of planar shapes. 
7.10 Skeletal representation of planar shapes.
7.11 First level search for test shape (a) resulted in two candidate shapes (b) and (c) with identical genus. ................................................................. 112
7.12 Shape recognition: (a) Test shape; (b) Best match; (c)-(e) Other good matches.113
List of Tables

3.1 Subgraph isomorphisms from $g_1, g_2$ to $g$. . . . . . . . . . . . . . . . . . . . . . . . . . 21
4.1 Performance measures $\tilde{D} \times 10^3$. . . . . . . . . . . . . . . . . . . . . . . . . . 42
4.2 Effect of Shearing on Classification (% classification rates). . 45
4.3 Summary: FSSM learning. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 52
5.1 Skeletonization of planar shapes . . . . . . . . . . . . . . . . . . . . . . . 59
6.1 Algorithm for Constructing Topological Graph . . . . . . . . . . . . . . . . . . . . . . . . . 73
Chapter 1

Introduction

Machine vision ideally amounts to imparting a machine an ability to view things in a way somewhat similar to humans. Inherently, machines are “dumb” and are, hence, unable to meaningfully visualize things with useful decisions. They must be “trained” to, for instance, start recognizing certain shapes as fundamental entities, which may then provide an ability to detect objects of similar shapes. Even the problem of training a machine for a particular class of shapes is quite complex, where the complexity arises from various layers involved in the recognition process. First, images acquired by a machine for scene or image analysis are usually noisy. This typically requires image denoising and their subsequent enhancement, prior to analyzing their contents. Second, a difference in viewing angle and/or transformations applied to objects (e.g., affine), may make different shapes look similar. An efficient machine vision system, should be capable of performing “reasonably” well in the presence of all these adverse scenarios.

A basic machine vision system is shown in Fig. 1.1. As can be seen in the figure, any vision system should be capable of performing three fundamental functions, namely enhancement, segmentation and identification/recognition. The former two form low level part of vision, and help in isolating different objects present in an image. The latter, which amounts to high level vision, identifies each isolated object. Identification of objects, in itself, is a two step process. First a machine should know a sufficient number of particulars about an object. This requires representing all shapes of interest by a shape descriptor that is simple yet sufficient enough for efficient and correct identification. Second, it should be able to extract certain characteristic features from an isolated shape, which are then matched to various descriptors in a database to identify the best candidate and, hence,
classify the shape accordingly. Finding efficient and robust shape descriptors is, therefore, a fundamental problem in machine vision, image understanding and vision perception.

Figure 1.2 illustrates the sequence of processing steps, where an image, acquired from a scene, is first processed to remove any noise. Following edge detection (see Fig. 1.2(b)), a segmentation is carried out to isolate various shapes in the image as shown in Fig. 1.2(c). Features are then extracted as shown in Fig. 1.2(d), which in the example are represented by skeletal graphs. These skeletal graphs are finally compared with shape models (see Fig. 1.2(e)) stored in a database to find the best match. The aim of this thesis, is to focus on the latter part of a vision system, i.e., high level vision, which amounts to learning shape models from training shapes, extracting shape features for a test shape, and finally matching the feature set with the models. Our approach is to find efficient shape descriptors, which represent a shape in a robust fashion so as to yield low classification errors, while simultaneously reducing computational complexity.
1.1 Shape Descriptors

In order to match two shapes, one possible approach may be to represent two shapes in the form of long vectors by projecting them to a high dimensional space and then quantifying their similarity by way of the Euclidean distance between them or their mutual correlation. This approach when applied to raw shapes, does not yield acceptable recognition rates. The idea is then to highlight certain shapes features, which capture important and unique characteristics. Classification is then carried out on a shape space defined by the feature set instead of the actual Euclidean space. Several complexities, however, arise in the latter situation, where inherent variability of shapes, noise, occlusion, pose, etc., contribute to complicate correct identification. Note that noise mostly affects low level vision, i.e., segmentation. Segmentation tasks are, however, of secondary importance in this thesis, and we assume that all shapes have already been isolated. The pose problem, nevertheless, appears as a result of the following two cases:

1. Observed view of an object is similar to that in the database, but rotated at some angle. In this case, we need to either align the shape, or look for a model that is invariant under rotation.

2. A slightly different view of an object is observed. In this scenario, we seek a shape descriptor that is robust against affine transformation.

2D shape descriptors are quite vulnerable to pose problem. In this thesis, we will present two 2D shape models and will analyze their robustness to rotation and a subset of affine transformations.

To fully overcome this problem, one approach is to rather consider 3D shapes, which possess complete shape information and are, therefore, view independent. A good shape descriptor will then be the one which is invariant under rigid transformations.

Apart from invariance to a set of transformations, the utility of any shape descriptor lies in its ability to possess the following properties over a wide class of shapes:

**Uniqueness** The feature set for any given shape should be its unique signature. It is very important that two different shapes do not lead to coinciding features.

**Sufficiency** It is desirable to extract a feature set that contains sufficient information about a shape. Uniqueness and sufficiency are very important properties for correct iden-
tification of shapes.

**Compact** A shape descriptor should be compact. This property is desirable for efficient representation and archiving.

**Completeness** Completeness of a descriptor is tantamount to perfect reconstruction capacity. This property forms the basis of shape reconstruction. Note that compactness is meaningless without completeness.

Our contribution herein in this direction consist of two proposed shape descriptors which meet these properties.

### 1.2 Proposed Shape Models

While a lot of shape modeling techniques have been proposed in the past, learning an efficient model that accounts for all shapes in all scenarios is still an open problem. As noted earlier, several factors, namely inherent variability of shapes, noise, occlusion, pose, etc., contribute to the complexity of the problem. These complexities motivate a flexible method that is robust in most practical applications. The problems of classification, reconstruction and sampling of shapes from models are generally of paramount importance. In the first part of the thesis, we address the problem of statistical modeling of simple planar shapes, with the goal of proposing a unified method that:

1. captures inherent shape variability,
2. allows shape classification,
3. permits sampling new shapes from the model, and
4. reflects well skewness of the underlying data without enforcing overly restrictive assumption of Gaussianity.

In short, shape variability is viewed as a random process and is, therefore, captured via a shape distribution. The learned shape distribution is then combined with a maximum likelihood or Bayesian classifier for recognition purposes. In addition, it is formulated that all representative realizations may be sampled from the model. We also demonstrate an equivalence of the proposed method with that of Principal Curves.
The above proposed method models 2D simple shapes and does not extend directly to shapes of arbitrary topology. In order to model 2D shapes of arbitrary topology, we propose shape modeling via skeletal graphs in the second part of the thesis. These skeletal graphs are formulated in a Morse theoretic framework, a rather novel technique for 2D shape modeling. The resulting similarity transformation-invariant shape model, promises completeness and uniqueness of representation – highly desirable properties for shape reconstruction and classification.

Another interesting aspect of the problem coincides with human visual perception. A human visual system combines the outer contour information of objects with that pertaining to their depth and their texture. It is, therefore, believed that a vision system, which exploits the last two features will outperform a simple system operating on planar shapes. This strongly suggests that three dimensional shape modeling in scene analysis, would be more robust. The success of such a system hinges on our ability to construct a 3D scene from images, which in turn is the essence of shape from shading [35, 36, 66, 67, 97, 43, 96] and 3D photography [22, 30, 19, 31] in machine vision. In contrast to 2D shape modeling much work remains in 3D shape modeling and classification. The third part of this thesis focusses on extending the idea of 2D skeletal graphs to 3D shapes and the classification of 3D shapes in the presence of above mentioned limitations. This constitutes an important building block of a robust and reliable machine vision system with an acceptable classification performance.

3D objects are intrinsically defined by two key attributes, namely topology and geometry. In order to capture topology, a Morse function, the distance function, is defined on the surface of an object, and then its critical points are tracked to construct a skeletal graph, whose vertices represent the critical points and whose edges represent topologically homogeneous parts of an object. Geometry, on the other hand, is captured by parametrically modeling the evolution of level curves of the distance function along each graph edge. The skeletal graph is then augmented with the geometric information in the form of weights, derived from the geometric model parameters, to yield a topo-geometric weighted graph. Such an object representation not only achieves data compression, but also forms a unique signature of an object capable of yielding a reconstruction from the topo-geometric graph. Since this model is a unique shape signature, intuitively it will achieve high classification rates. In the last part of the thesis, we classify 3D shapes based on the above approach where error tolerant graph matching is employed for comparing shapes.
1.3 Organization of the thesis

This thesis is organized as follows. We start with a literature review given in Chapter 2. Since the proposed shape models will be developed in a probabilistic and a Morse theoretic framework, we give a brief overview of statistical and algorithmic shape models. In Chapter 3, we provide some mathematical preliminaries, and briefly discuss the machinery that drives the proposed methods. In addition, since our 2D and 3D shape models are graphs-based, we also provide a brief introduction to graph theory.

In Chapter 4, we propose a 2D probabilistic shape model, which we call Flexible Skew-symmetric Shape Model (FSSM), due to its flexibility to adapt to non-Gaussianity of the underlying data. We will start with outlining differences of this approach with existing methods. Later in the chapter, we proceed to represent shapes by shape distributions, which are ultimately useful in classification applications. In addition, we incorporate a Markov process to account for the dependence of shape boundary points on their neighboring points. This allows us to sample smooth shapes from the model.

A 2D Morse theoretic shape model will be presented in Chapter 5, where we represent shapes by skeletal graphs to capture their topology. We will show how geometric information is encoded in graph vertices without additional cost. We will later demonstrate shape reconstruction capabilities of the model. 3D topo-geometric shape model will be discussed in Chapter 6, where we extend the above idea to 3D shapes. Although the extension for topological model follows naturally, geometric encoding requires additional construction. In the chapter, we propose a geometric model, which is formulated as a curve evolution model. We demonstrate the rigid transformation-invariance of the model and utilize the model for reconstruction of 3D shapes.

We present classification results for 2D and 3D shapes in Chapter 7 and finally conclude with a discussion of results, future extensions and applications in Chapter 8.
Chapter 2

Literature Review

Shape modeling is of paramount importance in the fields of machine vision, image understanding and vision perception. It aims at providing a mathematical formalism to represent a class of shapes each with an ideally unique fingerprint. Inherent variability of shapes makes the problem quite challenging, requiring a model that is sufficiently flexible and robust to adapt to the inherent high variability of observations. There have been two major approaches for addressing the problem:

1. Statistical approach

2. Algorithmic/computational geometry based approach

In this chapter, we present a brief overview of the previous work addressing shape modeling problem through these two approaches. In addition, we discuss existing techniques for 3D shape modeling. Later in the chapter, we will introduce our contribution to shape modeling and analysis, along with a comparison with existing methods.

2.1 Statistical Methods

There has been extensive research work on shape modeling in a statistical setting [26, 7, 41, 17, 81, 94, 99] for capturing the inherent variability of shapes.

Deformable templates One of the earliest approaches to statistical shape modeling was based on deformable templates [26, 27], which consider shapes as points on some infinite
dimensional manifold and then model their variations by Lie group actions on the manifold. This approach, however, leads to infinite dimensionality of shape spaces resulting in computationally intensive demands.

**Landmark based methods** To address the above mentioned complexity together with the practicality of data-driven shapes, the so-called Kendall school [10, 20, 41, 80] proposed an $n$-dimensional statistical description of shapes by introducing the notion of landmarks. In this framework, shapes are mapped from preshape spaces ($\mathbb{R}^n$) to finite-dimensional shape spaces by normalizing them with respect to a set of transformation groups, while shape differences are quantified by defining a metric on the space. Cootes et al. later capitalized on this idea and presented so called active shape (ASM) [14] and active appearance models (AAM) [15]. There has been other recent work [95] to alleviate the computational demands of Grenander’s deformable templates.

**Probability distributions** Srivastava, Joshi, Mio and Liu [81], on the other hand, model shapes through probability distributions on tangent spaces to the shape space, which is usually a nonlinear infinite dimensional manifold. Tractability and computational complexity issues are addressed by way of a finite dimensional approximation in the tangent spaces.

**Region-based feature statistics** Zhu [99] uses region-based features such as linelets and rib lengths whose histograms are computed and averaged to yield a Gibbs-like distribution for shape sampling. The use of this method is limited to sampling new shapes, whereas a recognition problem may require additional formulation. The sensitivity of this method to noise arises from its dependence on curvature. The latter problem may, however, be mitigated by using robust curvature estimators [23]. On the other hand, Manay, Hong, Yezzi, and Soatto [55] proposed to represent shapes by their integral invariants, which are inherently much less sensitive to noise. This approach not only allows shape reconstruction but may also be used for shape matching.

**Parametric shape estimation** In a more statistical approach, Ye, Bresler, and Moulin [93, 94] viewed shape modeling as a parameter estimation problem and proposed Cramér-Rao bounds on the accuracy of parametric shape estimation. In addition, they introduced a quantitative notion of confidence region, which is useful in recovering a shape from a given
image. In contrast to the parametric nature of this method, our model is a semi-parametric approach and minimizes a priori assumptions.

Active Contours and Level Set Methods There have also been attempts for low level shape modeling, through active contours [37] and level set methods [52, 53, 54, 75]. Active contours are driven by an optimization of an energy functional of closed curves whose evolution in time approximates shapes. Level set methods, on the other hand, have been popularized by Osher and Sethian’s work [63] on modeling propagation fronts. While these methods have been successfully employed in noise removal, segmentation, and shape extraction, they have, to the best of our knowledge, fallen short of providing a simple and consistent shape model for classification and shape sampling applications.

2.2 Algorithmic Methods

Numerous other techniques have been proposed in an algorithmic/computational geometric setting. An overview of some of these techniques is given below.

Point correspondences Many algorithmic approaches are based on establishing point correspondences between two shapes [72, 6, 7].

In [72], Sclaroff and Pentland proposed a point correspondence method: the correspondences were established in a generalized feature space, which is determined by eigen-modes of a finite element representation of a shape. The resulting correspondence was shown to be invariant under affine transformations and insensitive to noise. This method, however, is highly global and operates on a shape contour as a whole without taking local features into account.

Basri et al. [6], on the other hand, base their shape matching method on a correspondence between the outlining contour points so as to minimize a cost of bending and stretching energy of morphing one shape into the other. This cost then constitutes a basis for comparison. In addition to its limited efficiency, this approach heavily relies on a preordered set of points/landmarks, which may turn out to be difficult in practice.

In [7], Belongie, Malik and Puzicha propose point correspondence for unordered boundary points using shape contexts. Specifically, a shape context evaluated at a point, is a distribution of contour points relative to this point. Shape contexts for two shapes are used
in a $\chi^2$ test statistic-based cost function, whose minimization yields point correspondence. Following the establishment of point correspondences, a transformation is defined to yield a shape distance metric, which is then employed for shape comparisons.

**Medial axis** Another class of computational geometry-based methods is that of medial axis representation of shapes [9, 34]. These models owe much to their simplicity, which is also their limitation of capturing variability across various shapes. Specifically, medial axis may lead to a non-unique representation of a shape. Zhu and Yuille presented a method and referred to it as FORMS, which is based on a variant of medial axis [98]. Their model involves two primitives, which when deformed yield so called *mid-grained shapes* which in turn capture parts of an object. These mid-grained shapes are in the end assembled to represent complete objects by using a custom grammar. The dependence of such a model on primitives and its complexity due to the burden of grammar rules reduces its flexibility.

**Shape axis** Liu et al. propose a method for shape recognition via matching *shape axis trees* which are derived from the shape axis [49]. Shape axis is similar to medial axis and is defined as the locus of midpoints of corresponding boundary points on two given shapes. Shape axis trees are then modified to achieve the best match reflected by an associated cost which is based on the approach in [6]. Although this method addresses articulations and occlusions, it has limitations similar to those in Basri et al. [6].

**Shock graphs** Shock graphs are a variant of medial axis, as they capture its evolution in time. Specifically, it is a locus of singularities (shocks) formed by a propagating wave (grassfire) from the boundaries [42]. A shock graph may be viewed as a medial axis endowed with additional information. Hence, it may result in a unique representation over a wider class of shapes and is, therefore, generally regarded as a better shape descriptor with numerous variants.

Siddiqi et al. compare shock graph-based shapes by viewing them as trees and employing subgraph isomorphism [78] or by finding maximal clique of association graphs [65]. They choose the oldest shock as the root node, which is not always most logical and may contribute to classification errors.

Sebastian et al. [73] simplified shape recognition through shock graphs by partitioning the shape space into equivalence classes of all shapes with the same shock graph
topology. Subsequently, they discretize the deformation space by describing all deformations with same transitions as equivalent. Shape matching is carried out through graph edit distances by finding an optimal sequence of transitions that deforms one graph to the other.

2.2.1 3D Shape Models

A number of techniques have been explored for 3D shape modeling, some of which are presented below:

Mesh representation One of the most common 3D object representation is based on a triangulated mesh parameterizing their surface boundaries. Such a representation, however, suffers from two major drawbacks. First, it does not take into account local correlations or similarities, thus failing to exploit underlying redundancy for a compact representation. Second, it results in intertwined topological and geometrical information, which makes it unsuitable for shape recognition. Various approaches have been proposed in literature for addressing these problems.

Medial axis based models One of the foremost methods for compact representation is a variant of medial axis [1, 50]. Since a medial axis does not contain sufficient information for shape reconstruction, Lee [47] encodes the additional information necessary for complete shape representation by superimposing the radius information of the corresponding circles/spheres on medial axis. The resulting shape model, known as medial axis transform [47, 61, 74, 76], is a complete yet compact shape representation, which allows their reconstruction. It, however, suffers from the drawback that for 2D surfaces, medial axis transform itself happens to be a surface, which limits its utility for shape recognition. Pizer et al. [69], on the other hand, provide the addition information by way of spokes along the medial axis representing the distance from the surface. Although the resulting shape model, known as m-reps has been shown to reconstruct very “simple” shapes, its utility for complex shapes of arbitrary topology is an open question. In addition, it does not decouple topological and geometric information, and hence, is not suitable for shape recognition.

Skeletal models Shinagawa et al. [77] use Reeb graphs based on the height function to obtain skeletal graphs of surfaces representing 3D objects. A similar approach has been
proposed by Carr, Snoeyink, and Axen [13], where they model terrain data using contour trees. Lazarus et al. [46] propose skeletonization based on geodesic distance from a manually chosen source point. The resulting skeletal graphs are called level set diagrams. Hilaga et al. [33] extend this approach by eliminating the need of manual selection of a source point and propose a matching algorithm based on multiresolution Reeb graphs. Tung and Schmitt [84] capitalized on this approach to present augmented multiresolution Reeb graphs thereby capturing additional attribute features which yield better recognition rates.

**Shape distributions** In [28] and [62], 3D objects are represented through shape distributions and then a dissimilarity measure for distributions is employed to classify them.

**Reflective Symmetry Descriptors** In Kazhdan et al. [38], the global properties of 3D objects are captured through a reflective symmetry descriptor, which is defined over a certain parameterization which the authors term as canonical. Kazhdan and Funkhouser [39] use rotation-invariant spherical harmonics as a shape descriptor.

## 2.3 Contributions of This Thesis

As mentioned earlier, we will propose three shape models in this research effort in the following order:

1. Probabilistic 2D model
2. Morse theoretic 2D model
3. Topological-geometric 3D model

The first two methods model 2D shapes, while the latter accounts for 3D shapes.

### 2.3.1 Probabilistic Shape Modeling

In a probabilistic setting, we present a novel method for shape modeling and template learning, which we call Flexible Skew-symmetric Shape Model (FSSM). It uses an extended class of semiparametric skew-symmetric distributions. The proposed model aims at capturing the inherent variability of shapes so long as the realization contours remain within a certain neighborhood range around a “mean” with high probability.
Given several realizations of a shape, we model the latter by a joint distribution of angle and distance from its centroid. The distribution, according to which the points of these realizations are scattered, may be unknown and the template boundary can have any irregular shape. Instead of handling it as a bivariate case, we slice through the image at a specified angle, selected according to some prior distribution and then by maximizing the likelihood, we learn the univariate skewed bimodal density for the cluster of points that fall within some neighborhood of the angle. Using the modes of the learned distribution we can recover the boundary of the “mean shape”.

These conditional distributions are then combined for a global shape representation to capture the essence of all realizations of the “mean shape”.

Of all the techniques mentioned in Section 2.1, our approach is perhaps most related to that in [14]. There are, however, three major differences:

1. While approach in [14] relies on manual selection of landmarks on a shape, which may be tedious at best and time consuming, we proceed to sample shapes by learning an angle distribution, which allows us to automatically capture landmarks.

2. The model in [14] limits variability in the coefficient space of principal components, while we assume that a shape may have any form of variability within some bounds.

3. In addition, the model of [14] assumes Gaussianity of the underlying data which may not be true in general. This may, in fact, be observed in the form of skewness in medical imaging experiments, for instance, X-ray or MRI, where a patient’s motion and imperfect alignment are the rules. This, as a result, calls for a flexible model not only to account for skewness in the data but also to capture the Gaussian case.

Our model, therefore, generalizes that of [14] in some sense. Although the problem of non-Gaussianity of data has also been addressed by Cremers et al. [17, 18], where non-Gaussianity is captured via non-linear statistics, our approach is different since we do not explicitly apply any non-linear transformations and operate directly on the raw data. Specifically, we propose \textit{FSSM} which captures variability with a confidence within a certain neighborhood around a “mean” reflected by the model being sought. The potential excursion of the curves from the mean is captured via conditional probability density functions at given angles, for which we adapt a class of flexible skew-symmetric distributions [51]. In addition to the above features, our model promises to provide an alternative to principal
curve modeling [32] as we highlight their close connection and may indeed be used to extract principal curves in a point cloud.

This shape modeling technique will be presented in Chapter 4, where we will also employ it for shape classification and sampling.

2.3.2 2D Morse Theoretic Shape Model

Although \textit{FSSM} is flexible enough to account for non-Gaussianity of the data, it operates on a rather small class of simple shapes. In this thesis, we present another novel 2D shape model that addresses the problem of modeling shapes of arbitrary topology. This method is developed in a Morse theoretic setting, and results in a unique, compact and yet complete skeletal representation. The resulting skeletal graph may, then, be used for reconstruction as well as classification.

Among all methods discussed in Section 2.2, our method is closest to those based on shock graphs with a larger scope of applicability. To the best of our knowledge no previous attempt has been made to specifically address topologically diverse shapes. Specifically, our method is applicable to shapes which have nonzero genus, while those in [78, 65], for example, will fall short, as they view a shock graph as a tree, which is generally invalid in such cases. We must note that level set methods [64, 48, 83, 18] have also been proposed for shape modeling, which employ distance fields for topology preservation. While these low level modeling methods have remarkable segmentation capabilities, they have not been applied for shape classification. Our proposed method will use a shape (such as the segmented output of level set technique) as an input for modeling to ultimately achieve a compact representation suitable for subsequent storage and classification.

In contrast to our proposed approach, many of the previously described methods, particularly those based on medial axis, minimally invoke the geometric information and hence, do not guarantee unicity of representation. We will discuss this approach in detail in Chapter 5.

2.3.3 3D Topo-Geometric Shape Model

As discussed in Chapter 1, the pose problem in 2D shape modeling motivates the use of 3D shapes for representing real objects. To that end, we propose a method for 3D shape modeling, where the novelty lies in its ability to capture both the topological and
geometric structure of a shape. In the usual mesh representation, these two attributes are intertwined, thus making the representation too complex for classification purposes. We, on the other hand, propose to untangle this information by representing topology in the form of skeletal graphs, and geometry by assigning graph weights.

Previous work on 3D shape modeling does not yield unique shape signatures. Most of the 3D methods either are topology-based or they invoke minimal geometric information, resulting in a non-unique and incomplete shape representation. The proposed shape model, on the other hand, possess uniqueness and completeness properties. In addition, Reeb graph representation [77] suffers from the drawback that it is not rotation invariant, whereas [33] fails to capture the complete geometric structure of a shape. Our approach, however, yields rigid transformation invariance and accounts for complete topological and geometric information. Details of this approach will be presented in Chapter 6.
Chapter 3

Preliminaries

Much of the discussion in this thesis is built on some previous work, and hence, assumes some mathematical background. The goal of this chapter is to familiarize the reader with relevant and necessary tools to help the flow of the thesis.

3.1 Planar Shapes

A 2-D shape, may be represented by its outer contour. In Chapter 4, we will be representing shapes with their outer contours, and in Chapter 6, with their outer surfaces, whereas Chapter 5 considers a shape as a whole. In case, we represent a shape with an outer contour, we need to formally discuss the properties of such contours.

Definition 3.1: (Plane Curve) A parameterized plane curve $C$ is a continuous function that maps a point from a closed interval $I = [a, b] \subseteq \mathbb{R}$ to a point in a plane:

$$C : I \subseteq \mathbb{R} \rightarrow \mathbb{R}^2, \text{or } I \subseteq \mathbb{R} \rightarrow \mathbb{R} \times [0, \pi],$$

where the parameterization $C(t)$ may be represented in the form $(x(t), y(t))$ or $(r(t), \theta(t))$.

If a curve is associated to the outer contour of a shape $S$, we denote it as $C_S$ to emphasize this association. In addition, for simplicity, we may assume that $I \subset \mathbb{N}$, since practically we will be dealing with sampled curves, and subsequently, be representing a shape with a set of $m$ samples $S = \{C_S^i\}_{i=1}^m$, where $C_S^i$ is a point on the curve.

Definition 3.2: (Simple closed curve) A curve is called simple or Jordan, if it is not self-intersecting, i.e., $C(t_1) \neq C(t_2)$ if $t_1 \neq t_2, \forall t_1, t_2 \in I$. It is closed if there exists at least one parameterization of $C : [a, b] \rightarrow \mathbb{R}^2$, such that $C(a) = C(b)$. 
We are now in a position to formally define a shape:

**Definition 3.3: (Planar Shape)** A planar shape $S$ is a collection of points, that are enclosed by an outer contour $C_S$.

In Chapter 4, we will be considering shapes with contours that are strictly simple closed curves, whereas in Chapter 5, we will model a wider class of shapes of arbitrary topology. Throughout, we will be called upon to utilize several characteristics of a shape which we define next:

**Definition 3.4: (Mass of a Shape)** The mass $M$ of a planar shape $S$ with a surface density $\sigma(x, y)$ is given by:

$$M = \int_S \sigma(x, y) \, dx \, dy.$$  

**Definition 3.5: (Shape Centroid)** The centroid $(x_c, y_c)$, or the center of gravity, of a shape $S$ of mass $M$ is defined as the center of its mass. Mathematically,

$$x_c = \frac{\int_S x \sigma(x, y) \, dx \, dy}{M}, \quad (3.2)$$

$$y_c = \frac{\int_S y \sigma(x, y) \, dx \, dy}{M}.$$  

A centroid is the point on which a shape would balance, when placed on a needle.

In order to simplify the above expressions, note that by Definition 3.3 all discretized shapes (as usually is the case in practice) may be represented by a finite set of points enclosed by outer contours. In such a case, a shape consists of a set of $n$ point masses $m_i$ located at positions $(x_i, y_i), i = 1, \ldots, n$. Thus, Eq. 3.2 may be simplified to:

$$x_c = \frac{\sum_{i=1}^{n} m_i x_i}{\sum_{i=1}^{n} m_i} = \frac{\sum_{i=1}^{n} m_i x_i}{M}, \quad (3.3)$$

$$y_c = \frac{\sum_{i=1}^{n} m_i y_i}{\sum_{i=1}^{n} m_i} = \frac{\sum_{i=1}^{n} m_i y_i}{M},$$

which, if all masses are equal, further simplifies to:

$$x_c = \frac{1}{n} \sum_{i=1}^{n} x_i, \quad (3.4)$$

$$y_c = \frac{1}{n} \sum_{i=1}^{n} y_i.$$  

The centroid may, therefore, be computed as an arithmetic mean of the shape points. We will be using this expression frequently in the later chapters.
3.2 Graph Theory

The notion of a graph will be a recurring theme throughout this thesis, and we proceed to briefly describe the fundamental notions necessary for a smooth flow of the remaining development.

**Definition 3.6: (Graph)** A graph $G$ is a 4-tuple $G = (V, E, \mu, \nu)$ where

- $V$ is a set of vertices
- $E$ is a set of edges $e = (v, v')$, with $v, v' \in V$, i.e., $E \subseteq V \times V$
- $\mu: V \rightarrow LV$ is a function assigning labels to vertices
- $\nu: E \rightarrow LE$ is a function assigning labels to edges

One way to compare two graphs is to find the largest subgraph isomorphism from one graph to the other, where subgraph and isomorphism are defined below:

**Definition 3.7: (Subgraph)** Given a graph $G = (V, E, \mu, \nu)$, a subgraph $S$ of $G$ is a 4-tuple $S = (V_s, E_s, \mu_s, \nu_s)$ if $V_s \subseteq V$, $E_s = E \cap (V_s \times V_s)$ and $\mu_s$ and $\nu_s$ are the restrictions of $\mu$ and $\nu$ to $V_s$ and $E_s$ respectively, i.e.,

$$\mu_s(v) = \begin{cases} 
\mu(v) & \text{if } v \in V_s \\
\text{undefined} & \text{o.w.}
\end{cases}$$

$$\nu_s(e) = \begin{cases} 
\nu(e) & \text{if } e \in E_s \\
\text{undefined} & \text{o.w.}
\end{cases}$$

In order to define subgraph isomorphism, we need to introduce two concepts:

**Definition 3.8: (Difference Graph)** Given a graph $G = (V, E, \mu, \nu)$, and a subgraph $S = (V_s, E_s, \mu_s, \nu_s)$ of $G$, i.e., $S \subseteq G$. The difference of $G$ and $S$, i.e., $G - S$, is defined by the set of vertices $V \setminus V_s$.

**Definition 3.9: (Union of graphs)** Given two graphs $G_1 = (V_1, E_1, \mu_1, \nu_1)$ and $G_2 = (V_2, E_2, \mu_2, \nu_2)$, where $V_1 \cap V_2 = \emptyset$ and $E' = (V_1 \times V_2) \cup (V_2 \times V_1)$. Let $\nu: E' \rightarrow LE$ be a labeling function. Then the union of $G_1$ and $G_2$, i.e., $G_1 \cup E', G_2$, w.r.t. $E'$ is the graph $G = (V, E, \mu, \nu)$ such that

- $V = V_1 \cup V_2$
- $E = E' \cup (E_1 \cup E_2)$
\[ \mu(v) = \begin{cases} 
\mu_1(v) & \text{if } v \in V_1 \\
\mu_2(v) & \text{if } v \in V_2 
\end{cases} \]
\[ \nu(e) = \begin{cases} 
\nu_1(e) & \text{if } e \in E_1 \\
\nu_2(e) & \text{if } e \in E_2 \\
\nu(e) & \text{if } e \in E' 
\end{cases} \]

We are now in a position to define graph and subgraph isomorphism:

**Definition 3.10: (Graph Isomorphism)** A bijective function \( f : V \to V' \) is a graph isomorphism from a graph \( \mathcal{G} = (V, E, \mu, \nu) \) to a graph \( \mathcal{G}' = (V', E', \mu', \nu') \) if

1. \( \mu(v) = \mu'(f(v)), \forall v \in V \)
2. For any \( e = (v_1, v_2) \in E \), there exists \( e' = (f(v_1), f(v_2)) \in E' \) such that \( \nu(e) = \nu'(e') \)
3. For any \( e' = (v'_1, v'_2) \in E' \), there exists \( e = (f^{-1}(v'_1), f^{-1}(v'_2)) \in E \) such that \( \nu'(e') = \nu(e) \)

**Definition 3.11: (Subgraph Isomorphism)** An injective function \( f : V \to V' \) is a subgraph isomorphism from \( \mathcal{G} \) to \( \mathcal{G}' \), if there exists a subgraph \( \mathcal{S} \subseteq \mathcal{G}' \) such that \( f \) is a graph isomorphism from \( \mathcal{G} \) to \( \mathcal{S} \).

### 3.2.1 Decomposition based Subgraph Isomorphism

Given a set of model graphs \( \mathcal{G}_1, \ldots, \mathcal{G}_L \) and a test graph \( \mathcal{G}_t \), the problem is to find all subgraph isomorphisms from \( \mathcal{G}_i \) to \( \mathcal{G}_t \).

A naive inefficient approach is to sequentially match \( \mathcal{G}_i \) to \( \mathcal{G}_t \). A better approach is to decompose model graphs recursively and then match a test graph with the decompositions [57, 58]. Using this strategy, if a subgraph occurs multiple times in a graph or in several graphs, it is matched only once, thus reducing the computational overhead.

**Graph Decomposition**

Let \( B = \{ \mathcal{G}_1, \ldots, \mathcal{G}_L \} \) be a set of model graphs. A decomposition of \( B \), i.e., \( D(B) \), is a finite set of 4-tuples \( (\mathcal{G}, \mathcal{G}', \mathcal{G}'', E) \), where

1. \( \mathcal{G}, \mathcal{G}' \) and \( \mathcal{G}'' \) are graphs with \( \mathcal{G}' \subseteq \mathcal{G} \)
2. \( E \) is a set of edges such that \( \mathcal{G} = \mathcal{G}' \cup_E \mathcal{G}'' \)
3. For each \( \mathcal{G}_i \), there exists a 4-tuple \( (\mathcal{G}_i, \mathcal{G}', \mathcal{G}'', E) \in D(B) \) with \( \mathcal{G} = \mathcal{G}_i; \ i = 1, \ldots, L \)
4. For each 4-tuple \((G, G', G'', E) \in D(B)\) there exists no other 4-tuple \((G_*, G'_*, G''_*, E_*) \in D(B)\) with \(G = G_*\).

5. For each 4-tuple \((G, G', G'', E) \in D(B)\)

   (a) if \(G'\) consists of more than one vertex then there exists a 4-tuple \((\hat{G}, \hat{G}', \hat{G}'', \hat{E}) \in D(B)\) such that \(\hat{G} = G'\)

   (b) if \(G''\) consists of more than one vertex then there exists a 4-tuple \((\hat{G}, \hat{G}', \hat{G}'', \hat{E}) \in D(B)\) such that \(\hat{G} = G''\)

   (c) if \(G'\) consists of one vertex then there exists no 4-tuple \((G_*, G'_*, G''_*, E_*) \in D(B)\) such that \(G_* = G'\)

   (d) if \(G''\) consists of one vertex then there exists no 4-tuple \((G_*, G'_*, G''_*, E_*) \in D(B)\) such that \(G_* = G''\)

In above description of graph decomposition, \(G', G''\) are subgraphs of \(G\), whereas \(E\) denotes the set of edges between them. Statements (3) and (4) respectively ensure that every model is decomposable and the decomposition is unique. Statements 5(a) and 5(b) mean that every decomposition itself has a decomposition, thereby yielding a recursion. In other words, a decomposition is a recursive partitioning of graphs into smaller subgraphs, starting with complete models and terminating at the level of single vertices. If several models have a common subgraph \(S\), or if \(S\) occurs multiple times in a model, it is sufficient to represent all such occurrences by just one 4-tuple \((S, S', S'', E_s) \in D(B)\), and to match these occurrences only once.

The choice of decompositions is clearly not unique for a given set \(B\), and we seek the optimal decomposition, the one that gives the largest subgraph common to each \(G_i \in B\). Finding the optimal decomposition is a exponentially complex problem. One way to reduce its complexity is to sequentially decompose each graph \(G_i\) into two, such that one of the subgraphs is the largest subgraph decomposition of \(G_i\) that has already been added to the set of subgraphs resulting from decompositions. If no such subgraph exists, then the subgraphs themselves are recursively decomposed until we get single vertices.

3.2.2 Combining Decompositions

For each 4-tuple \((G, G', G'', E) \in D(B)\), we find isomorphisms from \(G'\) and \(G''\) to \(G_i\) and then combine the subgraph isomorphisms as described above to get subgraph
isomorphisms from $\mathcal{G}$ to $\mathcal{G}_t$. This process is started with single vertex graphs $\mathcal{G}'$ and $\mathcal{G}''$ and recursively continued until a level is reached where $\mathcal{G}$ represents a complete model. We illustrate this idea with the following example:

Example

Consider two model graphs $g_1$ and $g_2$ as shown in Fig. 3.1. Their decomposition is shown at the top of Fig. 3.1(d), where each subgraph consists of a single vertex. Given the test graph of Fig. 3.1(c), the decompositions are iteratively combined to find the largest subgraph isomorphism of $g$. As shown in Fig. 3.1(d), graph 1 can be matched to vertex 4 of $g$, graph 2 can be matched to any of the vertices 1 and 3 of $g$, and so on. Graphs 2 and 3 are then combined to get graph 6, which can be matched to subgraphs $\{12\}$ and $\{32\}$ of $g$. At the next stage, we get graph 7, which is isomorphic to subgraphs $\{124\}$ and $\{324\}$ of $g$. The process is continued until we get a graph that is not isomorphic to any subgraph of $g$. All subgraph isomorphisms from $g_1$ and $g_2$ to $g$ are given in Table 3.1. It can be seen that the largest subgraph isomorphism from $g_1$ to $g$ corresponds to graph 7 ($\{124\}, \{324\}$), whereas that from $g_2$ to $g$ corresponds to graph 10 ($\{12456\}$). This clearly shows that relative to $g_1$, $g_2$ is more “similar” to $g$.

<table>
<thead>
<tr>
<th>Decomposition</th>
<th>Subgraph of $(g_1 \text{ or } g_2)$</th>
<th>Isomorphic subgraphs of $g$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$g_1 \cdot g_2$</td>
<td>${4}$</td>
</tr>
<tr>
<td>2</td>
<td>$g_1, g_2$</td>
<td>${1}, {3}$</td>
</tr>
<tr>
<td>3</td>
<td>$g_1, g_2$</td>
<td>${2}, {5}$</td>
</tr>
<tr>
<td>4</td>
<td>$g_1, g_2$</td>
<td>${6}$</td>
</tr>
<tr>
<td>5</td>
<td>$g_2$</td>
<td>-</td>
</tr>
<tr>
<td>6</td>
<td>$g_1, g_2$</td>
<td>${12}, {32}$</td>
</tr>
<tr>
<td>7</td>
<td>$g_1, g_2$</td>
<td>${124}, {324}$</td>
</tr>
<tr>
<td>8</td>
<td>$g_2$</td>
<td>${1245}$</td>
</tr>
<tr>
<td>9</td>
<td>$g_1$</td>
<td>-</td>
</tr>
<tr>
<td>10</td>
<td>$g_2$</td>
<td>${12456}$</td>
</tr>
<tr>
<td>11</td>
<td>$g_2$</td>
<td>-</td>
</tr>
</tbody>
</table>

3.3 Morse Theory

In this section, we review some basic concepts of Morse theory [56, 59] as a prelude to the methods proposed in Chapters 5 and 6. While Morse theory provides basic framework
Figure 3.1: Graph Decomposition: (a) Model graph $g_1$; (b) Model graph $g_2$; (c) Test graph $g$; (d) Decomposition of $g$ and subsequent combination to find subgraph isomorphism.
for topological analysis of smooth manifolds, we will be briefly discussing its role for the
analysis of smooth compact surfaces embedded in $\mathbb{R}^n$.

Morse theory relates the topology of a smooth manifold with the number of the
critical points of a Morse function (see Definition 3.15) defined on this manifold.

A $k$-dimensional manifold $\mathcal{M}$ may be locally parameterized as

$$
\phi : \Omega \rightarrow \mathcal{M}
$$

that is, $\Omega \ni u \mapsto \phi(u) \in \mathcal{M}$, where an open connected set $\Omega \subset \mathbb{R}^k$ represents the parameter
space. Let $f : \mathcal{M} \rightarrow \mathbb{R} \subseteq \mathbb{R}$ be a real-valued function defined on $\mathcal{M}$. By definition,
the function $f$ is smooth if the composition $f \circ \phi : \Omega \rightarrow \mathbb{R}$ is smooth for each local
parameterization of $\mathcal{M}$.

**Definition 3.12:** (Critical Point) A point $x = \phi(u) \in \mathcal{M}$, where $u \in \Omega$ is called
a critical point of $f$ if the gradient of $f \circ \phi$ vanishes at $u$, i.e., $\nabla f \circ \phi(u) = 0$, and $f \circ \phi(u)$
is referred to as the corresponding critical value.

There are two types of critical points, which are of interest, namely degenerate and non-
degenerate critical points:

**Definition 3.13:** (Non-degeneracy) A critical point $x \in \mathcal{M}$ is called non-degenerate
if the Hessian $\nabla^2 f \circ \phi(u)$ is non-singular at $\phi(u)$.

Note that this definition is independent of the choice of the local parametrization in the
neighborhood of the critical point.

**Lemma 3.14:** (Morse Lemma) There exists a parametrization of a neighborhood
of a non-degenerate critical point of $f$ in which $f \circ \phi$ attains a quadratic form.

For instance, the function $f(x) = x^2$ has a non-degenerate critical point at $x = 0$, which is
in accordance with the local quadraticity of the function.

If $f$ is a smooth function on a two-dimensional manifold $\mathcal{M}$, three possible types of
non-degenerate critical points exist, namely the local minimum (index 0), the saddle point
(index 1), and the local maximum (index 2).

**Definition 3.15:** (Morse function) A smooth function $f : \mathcal{M} \rightarrow \mathbb{R} \subseteq \mathbb{R}$ on a
smooth manifold $\mathcal{M}$ is called a Morse function if all of its critical points are non-degenerate.

A Morse function satisfies the following basic properties:

- Critical points of a Morse function are isolated
Figure 3.2: Critical points of a height function defined on a Manifold $\mathcal{M}$.

- The number of critical points of a Morse Function is *stable*, that is, a small perturbation of the function neither creates nor destroys critical points
- The number of critical points of a Morse function on a compact manifold is finite

The level set $\mathcal{L}_t = f^{-1}(t) \subset \mathcal{M}$ of a Morse function $f : \mathcal{M} \to \mathbb{R} \subseteq \mathbb{R}$, for some $t \in \mathbb{R}$, is called *critical*, if it contains a critical point of $f$. For instance, in Fig. 3.3, $\mathcal{L}_c$ is a critical level of the surface $\mathcal{M}$ corresponding to the critical value $c$. According to the Morse Deformation Lemma, if any two levels $\mathcal{L}_{c_1}$ and $\mathcal{L}_{c_2}$ have different topological types, there is a number $c \in (c_1, c_2)$ such that $\mathcal{L}_c$ is a critical level. In other words, a change of topology occurs only at a critical point.

**Example 3.16: (The Height Function on a Sphere)** The height function defined on a unit sphere $\mathcal{M} = S^2$ is a real valued function $h : \mathcal{M} \to \mathbb{R}$ such that $h(x, y, z) = z, \forall (x, y, z) \in \mathcal{M}$. This function has two critical points, minimum at the south pole and maximum at the north pole. It is straightforward to show that both are non-degenerate indicating that $h$ is a Morse function.

Fig. 3.2 illustrates the critical points of the height function on a double torus. There are six critical points: a minimum, a maximum, and four saddle points.

### 3.3.1 Handle Decomposition

Topological analysis can also be explained in terms of *handle decompositions*. Consider a height function $h$ defined on a surface $\mathcal{M}$ shown in Fig. 3.3, mapping $\mathcal{M}$ onto the interval $[a, b]$ where $a$ and $b$ correspond to the two extrema. Studying the topology of $\mathcal{M}$ is tantamount to looking at its intersections with the level sets $h = t \in [a, b]$ of $h$, where $t$ is gradually increased.
Starting at $t = a$, we figure out that the portion of $\mathcal{M}$ that lies underneath is an empty set. Clearly, there is a critical point $p_0$ corresponding to the value $t = a$, and when viewed in an $\epsilon$-neighborhood of $a$, we get a subsurface $\mathcal{M}_{p_0 + \epsilon}$, illustrated in Fig. 3.4. This cup shaped subsurface is diffeomorphic to a $D^2$-disc, referred to as a 0-handle in accordance with the index of the critical point. Hence, whenever a minimum is encountered, we attach a 0-handle to the portion of the surface that lies underneath, which, in turn, is an empty set in this example.

Consider the case when we are just about to encounter the saddle point $c_0$, as shown in Fig. 3.5(a), which corresponds to the level set $L_{c_0 - \epsilon}$. The subsurface $\mathcal{M}_{c_0 - \epsilon}$ that lies under this level set contains two branches. As $t$ is gradually increased from $L_{c_0 - \epsilon}$, we gradually move up the hill eventually connecting the two limbs at the critical point $c_0$. The bridge formed between the two limbs, as shown in Fig. 3.5(b), is diffeomorphic to a $D^1 \times D^1$ disc, and is referred to as a 1-handle. Hence, whenever a 1-index critical point is encountered, we attach a 1-handle to the portion of the surface that lies underneath.

Similarly, the cup corresponding to the maximum gives rise to a 2-handle (Fig. 3.6). Recall that a compact surface has finite critical points, hence it can be decomposed in a finite set of handles. Another point that becomes clear in this discussion is the fact that a change in topology occurs only at a saddle point, where we actually observe a bifurcation or a merger of branches in a surface. The symbology for different handles of a surface is given in Fig. 3.7(a), whereas the handle decomposition of Fig. 3.3 is given in Fig. 3.7(b).
Figure 3.5: $\varepsilon$-Neighborhood of a 1-index critical point.

Figure 3.6: $\varepsilon$-Neighborhood of a 2-index critical point.

Figure 3.7: (a) Symbology for different handles; (b) Handle decomposition of the surface given in Fig. 3.3.
3.3.2 Critical Points and Topology

We now briefly explain how the topology of a compact orientable surface may be linked to the number of critical points of a Morse function defined on this surface. This, however, requires defining two new interrelated concepts. First, the genus of a surface is defined as the number of “handles” one needs to add to a sphere to synthesize the surface. Second, the Euler characteristics, \( \chi \), is defined as the sum of the number of vertices and the number of faces minus the number of edges of an arbitrary triangulation of \( M \). The topological type of a compact orientable surface is in one-to-one correspondence with either of these two numerical invariants. It follows from the previous section that the genus is directly related to the topological type of a compact surface. On the other hand, it can be shown that for any Morse function defined on a compact orientable surface \( M \), the Euler characteristic equals the number of maxima plus the number of minim minus the number of saddle points. Thus, if a Morse function on \( M \) is found, one may compute the corresponding Euler characteristic and hence, the genus, since both are interrelated by the formula \( \chi = 2 - 2g \), which in turn will determine the topological type of \( M \).
Chapter 4

Probabilistic Modeling of 2D Shapes

Statistical skewness of shape data often arises in applications (e.g., medical image analysis), and is often overlooked in statistical shape models. In most practical applications, a Gaussian assumption is unrealistic and a formulation of a general shape model, which accounts for skewness is in order. In this chapter, we present a novel statistical method for shape modeling, which we refer to as Flexible Skew-symmetric Shape Model (FSSM). The model is sufficiently flexible to accommodate a departure of the data from Gaussianity, and is fairly general to learn a “mean shape” (template), with a potential for classification and random generation (i.e., sampling) of new realizations of a given shape. Accounting for skewness results from deriving the FSSM from an extended class of flexible skew-symmetric distributions. In addition, we demonstrate that the model allows us to extract principal curves in a point cloud.

The idea is to view a shape as a realization of a spatial random process, and to subsequently learn a shape distribution which captures the inherent variability of realizations, provided they remain, with high probability, within a certain neighborhood range around a mean. Specifically, given shape realizations, FSSM is formulated as a joint bimodal distribution of angle and distance from the centroid of an aggregate of random points. A mean shape is recovered from the modes of the distribution, while the maximum likelihood criterion is employed for classification.
Figure 4.1: (a) Heart shape; (b) Some realizations superimposed on each other; (c) Sampled superimposed realizations; (d) Constituent realizations.

4.1 Motivation

A highly flexible shape model is motivated by its ubiquitous importance in practical applications, and its adaptability to high variability as may be observed, for instance, in anatomical shapes. Several anatomic organs indeed have temporal shape variations. Occlusion coupled with a pose problem may also cause different realizations of a shape to look different. In the presence of all these difficulties with observed shapes, we seek to provide a unified modeling and analysis method with the following properties:

- A capacity to capture inherent shape variability,
- A flexibility in shape classification,
- A means to sample new shapes from a model, and
- A potential to reflect well skewness of the underlying data.

By constraining the aforementioned shape variability to within some statistical bounds, we are able to address the modeling problem, and to contemplate solving classification and recognition problems. Specifically, we restrict the realizations to lie within a tubular region (interpreted as permissible shape region) around a mean with probability close to 1 as illustrated in Fig. 4.1(c). When all such realizations are superimposed and subsequently sampled at each specified angle, they may be viewed as a cloud of points in the neighborhood of a template boundary as shown in Fig. 4.1(c). We begin with a basic assumption that these sampled points on individual realizations are independently distributed. This overly restrictive assumption, which albeit gives reasonably good results,
will be relaxed later on, where we incorporate a Markovian property to account for statistical dependence of the points at neighboring angles.

4.2 Admissible Shapes

In this chapter, we will be investigating a class of simple closed shapes \( S \), whose centroid satisfy a certain admissibility condition \([70]\). In case they do not, we relax the condition to allow a wider class of shapes, which have at least one point, not necessarily the centroid, that satisfies the admissibility condition. This condition is based on the following definition:

**Definition 4.1:** (Regular Admissible Center) Given a shape \( S \) with a contour denoted by \( C_S \). A point \( \gamma_0 \in S \) is called a regular admissible center of \( S \), if there exists no half-line \( \ell \) going from \( \gamma_0 \) that is tangent to \( C_S \). A collection of all regular admissible centers is called the admissible set of \( S \) and is denoted by \( A_S \).

In simple words, an admissible shape is the one that contains at least one regular admissible center. The following theorem \([70]\) allows an easy computation of a regular admissible center:

**Theorem 4.2:** Let \( C_S : [a, b] \to \mathcal{C} \subset \mathbb{R}^2 \) be the outer contour of a shape \( S \). For any \( t \in [a, b] \), let \( \tau(t) \) be the unit tangent at point \( C_S(t) \) and \( \gamma_0^0(t) \) be a vector from any shape point \( \gamma_0 \in S \) to \( C_S(t) \), i.e.,

\[
\gamma_0^0(t) = C_S(t) - \gamma_0, \quad t \in [a, b].
\]

Then \( \gamma_0 \in A \) if and only if the orientation of the pair of vectors \( \{\tau(t), \gamma_0^0(t)\} \) is a non-zero constant for all \( t \in [a, b] \). The latter condition means:

\[
\text{sgn}\left( \begin{vmatrix}
\gamma_1(t) & \gamma_2(t) \\
\tau_1(t) & \tau_2(t)
\end{vmatrix}\right) = \alpha, \forall t \in [a, b],
\]

where \( \gamma_0^0(t) = (\gamma_1(t), \gamma_2(t))^T, \tau(t) = (\tau_1(t), \tau_2(t))^T \) and \( \alpha \neq 0 \) is some constant.

The following theorem \([70]\) explains that the regular admissible centers cannot be scattered all over a shape but instead form a path-connected set \( A_S \).

**Theorem 4.3:** (Convexity) Let \( S \) be a shape with an admissible set \( A_S \). Then \( A_S \) is convex, i.e., for any \( \gamma_1, \gamma_2 \in A_S \), \( (\gamma_1, \gamma_2) \subset A_S \), where

\[
(\gamma_1, \gamma_2) = \{\alpha \gamma_1 + (1 - \alpha) \gamma_2 : \alpha \in [0, 1]\}.
\]

(4.3)
We are now in a position to define the class of shapes we will be investigating in this chapter. We will seek probabilistic models for simple closed 2-D shapes, which have at least one regular admissible center.

As mentioned in Chapter 3, we will consider the outer contour $C_S$ of a shape $S$, which we parameterize in the form of modified polar coordinates $(r, \theta)$ instead of Cartesian coordinates. The modified polar coordinates of a point $(x, y) \in \mathbb{R}^2$ are defined as:

$$
|r| = \sqrt{x^2 + y^2}, \quad \theta = \arctan(y/x).
$$

Although angle based shape models have been historically popular [45, 81], we opt for the modified polar coordinates. The choice of this alternative formulation, where $\theta \in [0, \pi]$ instead of $[0, 2\pi]$, and $r \in \mathbb{R}$ will become clear in the next section.

Because of variability of shapes, the contour $C_S(t)$ may be viewed as a random process, which assumes values over a set of all possible realizations. Given several realizations of a shape $S$, the problem is to seek a probabilistic model in terms of curve parameterization.

In the preprocessing stage, we standardize location, scale and orientation information, since the proposed method itself does not adapt to rigid transformations among shapes. There are various methods for shape normalization [25, 8, 79, 24, 44]. In this chapter, we employ one such approach, known as Procrustes analysis [25, 24, 80], because of its simplicity. Note that Procrustes analysis yields a reference origin at a shape centroid. If the centroid is, however, not a regular admissible center, we then take the origin at the centroid of the region of regular admissible centers.

**4.3 Problem Statement**

Following the preprocessing stage, the aggregate shape is randomly sampled through the centroid at angles $\Theta \in [0, \pi]$ according to a prior distribution $p(\theta)$. In other words, all realizations are sampled at angles $\theta_1, \ldots, \theta_n$, which will result in dense sets of samples at each angle on account of:

1. The continuity of the contour

2. Realizations form a dense set of shapes within the permissible tubular region.
Practically, however, due to the discretization of the contours, sampling at specific angles may not yield any samples. We can circumvent this problem by identifying the samples lying within some $\epsilon$-neighborhood around an angle $\theta_i$, as shown in Fig. 4.2, and then projecting them to the $\theta = \theta_i$ axis. For a sufficiently small $\epsilon$, the projected radii closely reflect actual radii to the sample points. On the other hand, the purpose of considering such neighborhoods is to get a statistically significant number of samples for a given angle. The size $\epsilon$ must be carefully chosen to ensure an ample set of samples to learn the model without compromising much on accuracy. It should be emphasized that the notion of $\epsilon$-neighborhood is introduced for practical considerations, which in essence divides a shape in a number of sectors. Later in Section 4.3.4, we will consider the special case, where we use non-overlapping sectors, which will allow us to represent shapes with shape distributions.

It may be readily observed that we do not need to keep track of the entire $\theta$ range of $[0, 2\pi]$ since the closure of a shape allows us to associate two clusters of samples on either side of the centroid at $\theta_i$, with a relative phase difference of $\pi$ as illustrated in Fig. 4.2. This justifies our curve parameterization of Eq. (4.4). The distances $R$ from the centroid to the samples in two different clusters are, however, weighted by $\pm 1$ to distinguish the two corresponding clusters, and are subsequently modeled by a bimodal distribution. This approach is rationalized on three counts:

1. It reduces the learning space to half the actual space and, hence, alleviates the complexity,

2. When the centroid does not lie in the interior of a shape, we still get a bimodal density for simple shapes, and

3. It provides a straightforward extension to multimodal distributions to eventually accommodate multi-loop shapes.
4.3.1 Posterior Distribution

In Fig. 4.3, some conditional histograms of \( R \) given \( \Theta \) are presented, illustrating the bimodality and skewness of conditional distributions, \( p(r|\theta) \). These conditional distributions cannot be modeled by a symmetric distribution (e.g., Gaussian, or Laplacian). Although any distribution may be expanded into Gram-Charlier series, and therefore, modeled as a sum of Gaussians, the number of terms required to accomplish this may not be straightforward in many cases, and may limit the utility of a sum of Gaussians. A natural choice to adopt is a flexible functional form for a pdf which may handle such cases. Wang et al. [85] proposed a skew-symmetric (SS) representation for such a class of conditional distributions:

\[
p(r|\theta) = 2f \left( \frac{r - \xi}{\sigma} \right) h \left( \frac{r - \xi}{\sigma} \right),
\]

where \( \xi \) is the location parameter, \( f \) is any symmetric pdf with standard deviation \( \sigma \) and \( h \) is a skewing function that satisfies the following two conditions:

\[
0 \leq h(r) \leq 1,
\]
\[
h(-r) = 1 - h(r).
\]

A semiparametric form of SS representation, called flexible skew-symmetric (FSS) distribution [51], is obtained by choosing \( h(r) = H(P_K(r)) \), where \( H \) is any cumulative distribution function of a continuous random variable, and symmetric around zero, while \( P_K \) is an odd polynomial of order \( K \). On a compact support, FSS distributions

\[
p(r|\theta) = 2f \left( \frac{r - \xi}{\sigma} \right) H \left( P_K \left( \frac{r - \xi}{\sigma} \right) \right)
\]

form a dense subclass of skew-symmetric (SS) distributions [51].

Unfortunately, an odd polynomial, \( P_K \), cannot attain a symmetry of a bimodal density. To that end, we have the following:
Proposition 4.4: An FSS distribution, as given by Eq. (4.7), and using an odd polynomial \( P_K \), is not symmetric for any odd order \( K \).

Proof: Without loss of generality, assume \( \sigma = 1 \) and \( \xi = 0 \). Further assume that \( p(r|\theta) \) is symmetric. Then, \( 2f(r)H(P_K(r)) = 2f(-r)H(P_K(-r)) = 2f(r)[1 - H(P_K(r))] \).

Hence, \( H(P_K(r)) = 1/2 \), which is a contradiction, since \( H \) is an increasing functional of an odd function and hence cannot be even. Hence, \( p(r|\theta) \) is not symmetric.

As an alternative, we propose to use an arbitrary polynomial \( P_K^* \) of order \( K \) instead of odd polynomial \( P_K \). To preserve the validity of the pdf, we introduce an appropriate scaling factor \( \omega \) to obtain a general FSS structure which satisfies Eq. (4.6):

\[
p(r|\theta) = 2\omega f\left(\frac{r - \xi}{\sigma}\right)H\left(P_K^*\left(\frac{r - \xi}{\sigma}\right)\right),
\]

(4.8)

where \( P_K^* \) is any polynomial of order \( K \) with coefficients \( \alpha \). This may, then, be interpreted as defining a new skewing function \( h^*(r) \) written as:

\[
h^*(r) = \omega H\left(P_K^*\left(\frac{r - \xi}{\sigma}\right)\right),
\]

(4.9)

where \( \omega \) is a function of \( \xi, \sigma, \) and \( \alpha \) and is given by:

\[
\omega(\xi, \sigma, \alpha) = \frac{1}{2 \int_{-\infty}^{+\infty} f\left(\frac{r - \xi}{\sigma}\right)H\left(P_K^*\left(\frac{r - \xi}{\sigma}\right)\right) dr}.
\]

(4.10)

Note that the density parameters \( \xi, \sigma \) and \( \alpha \) are functions of \( \theta \) in Eqs. (4.5), (4.7) and (4.8). Depending on a shape, the parameters \( \xi, \sigma \) and \( \alpha \) and hence, the polynomials \( P_K \) and \( P_K^* \) may be different for different values of \( \theta \), and may result in different posterior densities.

Model Size

As mentioned earlier, even order terms are required for capturing symmetry, while odd order terms have been shown to yield skewness. It has also been shown [51] that a third order odd polynomial, for instance, enjoys skewness and bimodality properties while it cannot accommodate symmetric models. It is, henceforth, required that \( K \geq 3 \) in Eq. (4.8) to ensure all the nice properties our models may require. Since in this chapter, we are interested in shapes which strictly lead to bimodal distributions, the obvious choice for \( K \) is 3. In Section 4.5, we experimentally demonstrate that good results for shapes of interest are achieved and increasing \( K \) beyond 3, does not improve much on performance.
4.3.2 Posterior Learning

Given a data set of sufficient size, the next step is to learn conditional densities. We may, for instance, maximize the log-likelihood function for each given angle $\Theta = \theta_0$:

$$L(\xi, \sigma, \alpha|\theta_0) = m \log (2\omega(\xi, \sigma, \alpha)) + \sum_{i=1}^{m} \log f\left(\frac{r_i - \xi}{\sigma}\right)$$

$$+ \sum_{i=1}^{m} \log H\left(\sum_{k=1}^{K} \alpha_k \left(\frac{r_i - \xi}{\sigma}\right)^k\right),$$

where $m$ is the number of radius samples, $r_1, \ldots, r_m$. The negative log-likelihood function for a Gaussian pdf, $f$, may be written as:

$$J(\xi, \sigma, \alpha|\theta_0) = -L(\xi, \sigma, \alpha|\theta_0)$$

$$= m \log \frac{\sigma}{\omega(\xi, \sigma, \alpha)} + \sum_{i=1}^{m} \frac{(r_i - \xi)^2}{2\sigma^2} - \sum_{i=1}^{m} \log H\left(\sum_{k=1}^{K} \alpha_k \left(\frac{r_i - \xi}{\sigma}\right)^k\right).$$

Hence,

$$\beta(\theta_0) = \arg \min_{\beta} J(\xi, \sigma, \alpha|\theta_0),$$

where $\beta(\theta_0) = (\xi(\theta_0), \sigma(\theta_0), \alpha(\theta_0)^T)^T$.

It must be noted that for a Gaussian distribution, $p(r|\theta)$ given by Eq. 4.8 forms a dense set of skew-normal distributions, which is sufficient for capturing skewness and bimodality.

4.3.3 Angle Distribution

Since contour points $(X, Y)$ of a shape are randomly distributed around a mean shape and determine both $R$ and $\Theta$, the latter, in turn, are random, thus justifying the specification of a distribution on $\Theta$ to complete a shape representation. The angle distribution $p(\theta)$ should be such that it facilitates the capture of features like corners and other characteristic directions without (or at most with a minimal) manual assistance.

A hypothesized uniform distribution for $\Theta$ seems a reasonable choice for completely representing a shape. Singularities in a shape contour like cusps, for instance Fig. 4.4(a), complicate this choice. The nonuniform manifestation of such events allows us to consider
Figure 4.4: Singularities and angle distribution: (a) Singularities in shape at angle $\theta = \pi/6, \pi/2, 5\pi/6$; (b) Piecewise uniform distribution; (b) PUT distribution.

distributions that deviate from uniform. Higher sample density in the region in question suggests a piecewise uniform distribution for the angle $\Theta$, as illustrated in Fig. 4.4(b). Another choice is to employ a piecewise uniform tapered (PUT) distribution, as shown in Fig. 4.4(c), which follows from the fact that the density of points varies gradually around a singularity. Other nonlinear extensions of PUT distribution are also possible; specifically, automatic estimation of $p(\theta)$, based on the fraction of samples lying within an angular neighborhood, yields a PUT distribution that exhibits slightly nonlinear behavior in the otherwise uniform region. Later in the chapter, we will quantitatively justify an appropriate choice of angle distribution.

Such an approach is advantageous as an alternative to manual selection of landmarks [14] along a shape, as $p(\theta)$ allows us to automatically account for landmarks.

### 4.3.4 Overall Shape Distribution

We are now in a position to propose a shape distribution, which we construct using an angle distribution and conditional densities of radii $R$. To proceed, we first introduce some notations: we denote a parametric form of a shape $S$ by $\{Z_j\}_{j=1}^m$, where $Z_j = (r_j, \tilde{\theta}_j)$ is an observed point with radius $r_j$ and angle $\tilde{\theta}_j$. Note the difference in notation between the two angles. The angle that an observation point assumes is denoted by $\tilde{\theta}_j$, while an angle sampled from an angle distribution is represented as $\theta_i$. Thus, $\{Z_j\}_{j=1}^m$ is a set of all points on an observation boundary. When we sample $n$ angles, $\theta = (\theta_1, \ldots, \theta_n)^T$, according to a distribution $p(\theta)$ and consider their $\epsilon$-neighborhoods, we essentially divide the shape into $n$-sectors, where each sector is independent of the other, provided non-overlapping neighborhoods are assumed. As a result, for each $\theta_i, i = 1, \ldots, n$, the likelihood that a
point \((r_j, \tilde{\theta}_j)\) falls in an \(\epsilon\)-neighborhood of \(\theta = \theta_i\), \(N^\theta_i\), is \(p(r_j|\theta_i)p(\tilde{\theta}_j)\), and the likelihood of the boundary segment in the \(i\)-th sector is:

\[
p(Z_j \in N^\theta_i | \theta, \epsilon) = \prod_{Z_j \in N^\theta_i} p(r_j|\theta_i)p(\tilde{\theta}_j).
\]

(4.13)

The independence of points lying in \(N^\theta_i\) for different \(\theta_i\), due to non-overlapping neighborhood assumption, yields the following conditional likelihood for the overall shape:

\[
p(Z_1, \ldots, Z_m | \theta) = \prod_{i=1}^{\infty} \prod_{Z_j \in N^\theta_i} p(r_j|\theta_i)p(\tilde{\theta}_j).
\]

(4.14)

As \(n \to \infty\) and \(\epsilon \to 0\), we get \(p(S) \to p(Z(t)|\Theta)\), where \(\Theta \sim p(\theta)\), which represents the overall distribution of a shape with continuous boundary.

4.3.5 Template Learning

We define a template as the best realization of a shape \(S\) in a maximum likelihood sense. In other words, the template, \(R_S\) \(\text{template}\), maximizes the conditional likelihood function Eq. (4.14) over the set of all possible realizations:

\[
R_S\text{template} = \arg \max_{R_S} p(Z_1, \ldots, Z_m | \theta) |_{R_S}.
\]

(4.15)

This is equivalent to using modes of the bimodal distribution as the most likely boundary points. We, therefore, estimate the modes of the posterior for each angle, a set of which will then be taken as a template boundary. In the limiting case, as \(n \to \infty\), we will get a closed contour for the template. This is illustrated in Fig. 4.5, where conditional distributions \(p(r|\theta_i), i = 1, \ldots, n\), given \(n\) randomly drawn \(\Theta\)-samples, have been plotted for a star shape. An examination of the crests in the figure reveals the presence of a desired shape pattern which is recovered by the modes of the bimodal distributions.

4.3.6 Performance Assessment

In order to compare feature capturing capabilities of various angle distributions, we present a performance measure that compares templates learned using various angle distributions with the actual shape. Such a measure will aid an appropriate choice for \(p(\theta)\).

Suppose \(r^{(1)}(\theta)\) and \(r^{(2)}(\theta)\) represent two boundary points of a given shape at an angle \(\theta\), while \(\tilde{r}^{(1)}(\theta)\) and \(\tilde{r}^{(2)}(\theta)\) are the boundary points of a realization at the same angle.
Then the dissimilarity between the two is defined as:

$$dr_i(\theta) = r^{(i)}(\theta) - \hat{r}^{(i)}(\theta); \quad i = 1, 2.$$ \hspace{1cm} (4.16)

The cumulative dissimilarity or deviation of the realization from the shape is, therefore, given by the $L_2$-norm of the above dissimilarities:

$$\mathcal{D} = \left[ \int_0^{\pi} (dr_1^2(\theta) + dr_2^2(\theta)) \, d\theta \right]^{\frac{1}{2}}. \hspace{1cm} (4.17)$$

Although Eq. (4.17) is a good dissimilarity measure between a shape and its realizations/observations, it may not be practically applicable since it requires the boundary to be continuous. In order to accommodate discrete shapes, we consider only a discrete set of angles $A_D = \{\theta_1, \ldots, \theta_n\} \subset [0, \pi]$ and approximate $\mathcal{D}$ by:

$$\tilde{\mathcal{D}} = \left[ \sum_{j=1}^{n} (\tilde{dr}_1^2(\theta_j) + \tilde{dr}_2^2(\theta_j)) \right]^{\frac{1}{2}}, \hspace{1cm} (4.18)$$

where $\tilde{dr}_i(\theta_j)$ is an approximation of $dr_i(\theta_j)$ and is computed by replacing the actual boundary points in Eq. (4.16) with their respective approximates $\tilde{r}^{(i)}(\theta_j)$, where $\tilde{r}^{(i)}(\theta_j)$ is an average of all shape samples lying within some $\epsilon$-neighborhood $N^{(\theta_j)}$ of $\theta_j \in A_D$:

$$\tilde{r}^{(i)}(\theta_j) = \frac{1}{l} \sum_{l_k \in N^{(\theta_j)}} r^{(i)}(\hat{\theta}_k), \hspace{1cm} (4.19)$$

where we have used the notation defined in Section 4.3.4, whereas $l$ is the number of samples lying in $N^{(\theta_j)}$. In other words, for discrete curves, we may not have well defined $r^i(\theta_j)$ due to discontinuities, so we approximate it with the average of points that lie very close to it.

Eq. (4.18) represents the deviation of a realization from a shape, and since a template is another realization, it quantifies the quality of a learned template and allows us to select angle distributions which eventually yield a small $\tilde{\mathcal{D}}$. 

Figure 4.5: Illustration of using modes as boundary points: (a) Star shape; (b) Bimodal distributions learned with 100 angles.
4.4 Classification of Shapes

With the probabilistic model of Eq. (4.14) in hand, we may now employ a maximum likelihood (or Bayesian) classifier. Suppose that we have learned models $p_S$ for $\ell$ shapes $\{S_i : i = 1, \ldots, \ell\}$. Given a test shape, $S_0$, we compute $\ell$ likelihoods $\{p_{S_i}(S_i | S_0) : i = 1, \ldots, \ell\}$ and assign $S_0$ to the class that achieves the largest likelihood. This is illustrated in Fig. 4.6, where shape distributions effectively partition the shape space for a 6 class classifier.

4.4.1 Sampling from Models

Eq. (4.14) not only affords shape classification but also allows sampling new realizations of a shape. In order to sample a cloud of shapes, we need to learn the conditional distributions, $p(r|\theta)$, where $\theta \in \{\theta_1, \ldots, \theta_n\}$ are drawn according to a PUT distribution. Once the learning phase is complete, we proceed to generate $l \gg n$ additional angle samples from the same PUT distribution. For each angle sample $\theta$, the best approximation $\tilde{\theta}$ is selected:

$$\tilde{\theta} = \arg\min_{\theta_i \in \{\theta_1, \ldots, \theta_n\}} |\theta - \theta_i|,$$

and finally $r$ is generated according to $p(r|\tilde{\theta})$ to get one point $(r, \theta)$ in the cloud.

Sampling closed shapes with a smooth boundary from our proposed model requires a regularization via a Markov property. The Markovian model is motivated on two counts:

1. It captures the dependency of immediately adjacent landmarks resulting in a recursion,
2. It is sufficiently simple to be tractable.
Each value $r$ assumed by $R$, clearly, depends on its previous value $r_0$ along with the angle $\Theta = \theta$. Assuming $r_0$ to be independent of $\theta$, as a simplification, we get:

$$
p(r|\theta, r_0) = p(r|\theta)p(r_0|r)/p(r_0) = p(r|\theta)p(r_0)/p(r),
$$

(4.21)

where $p(r|\theta)$ is given by Eq. (4.8). Hence, the modified shape model is given by:

$$
p(Z_1, \ldots, Z_m|\theta) = \prod_{i=1}^{n} \prod_{j=1}^{m} p(r_j|\theta_i, r_0) p(\tilde{\theta}_j).
$$

(4.22)

$p(r|r_0)$ may be assumed to be Gaussian for each mode, with mean $r_0$. Its variance $\sigma_0$ may be treated as a smoothness parameter. Laplace distribution is, however, more appropriate because of its heavier tails. To further improve upon the tail behavior, a product of a Gaussian distribution and a bimodal FSS distribution $f(r)$ may be used for $p(r|r_0)$; this not only limits the variation between $r$ and $r_0$ but also ensures that $r$ is selected from a permissible shape region:

$$
p(r|r_0) = \frac{f(r)}{\lambda \sigma_0} \left( \exp \left[ -\frac{(r^{(1)} - r^{(1)}_0)^2}{2\sigma_0^2} \right] + \exp \left[ -\frac{(r^{(2)} - r^{(2)}_0)^2}{2\sigma_0^2} \right] \right),
$$

(4.23)

where $\lambda$ is a normalizing factor for $p(r|r_0)$ and $r_0 = [r_0^{(1)}, r_0^{(2)}]^T$. In essence, the introduction of a Markov chain probabilistically narrows the tubular permissible shape region.

In order to ensure closure of a shape, a current boundary point $r$ must be dependent on the starting point $r_s$ such that as $\theta \to \pi$, $r \to r_s$. Again a Gaussian distribution is found to be a good choice. Hence,

$$
p(r|r_s) = \frac{1}{\sqrt{2\pi\sigma_s}} \exp \left( -\frac{(r - r_s)^2}{2\sigma_s^2} \right).
$$

(4.24)

This dependence is, however, weak, initially ($\theta = 0$), but as a curve is traversed, the dependence gradually becomes stronger. Such a behavior is desirable, as the trajectory is minimally restricted at the beginning, while it is forced to close on itself towards the end. This requires $\sigma_s$ to be a decreasing function of $\theta$ as in the following functional form:

$$
\sigma_s(\theta) = \gamma - \exp(\theta - \eta\pi),
$$

(4.25)

where $\gamma$ has a sufficiently large value and $\eta$ is such that as $\theta \to \pi$, $\sigma_s \to \delta_\sigma$, where $\delta_\sigma$ is sufficiently small. Other functional forms are certainly possible, so long as they have a negligible decay initially, coupled with fast decay in the closing stages of the curve.
4.5 Experimental Results

Some practical applications of the proposed technique are presented in this section, with all shapes normalized in the preprocessing stage using Procrustes analysis. A third order polynomial ($K = 3$) is chosen for the extended flexible skew-normal distributions of Eq. (4.8) and $\epsilon$ is selected to be $\pi/300$.

4.5.1 Template Learning

Upon using both uniform and PUT distributions for template learning, we find that PUT is better suited for capturing singularities. An example is illustrated in Fig. 4.7(a) where simulated shape variability is given in Fig. 4.7(b). Two sets of 10 angle samples were drawn from an automatically estimated PUT distribution, shown in Fig. 4.10, and from a uniform distribution, and were subsequently used to learn templates given in Fig. 4.7(c) and (d). A template learned with dense (100) angle samples is shown in Fig. 4.7(e). Additional results are given in Figs. 4.8, 4.9, 4.11 and 4.12. Comparison of results for the two distributions with the fine sampling results, demonstrates that the PUT templates closely resemble the ideal shapes and outperform the uniform distribution templates. This is also evident by the respective performance measures given in Table 4.1. With very fine sampling of the angle space, e.g., 100 samples in above case studies, $\tilde{D}$ is indistinguishable for both distributions. Some additional results are given in Figs. 4.13 through 4.14.
Figure 4.9: Star-II shape: (a) Ideal Star-II shape; (b) Simulated realizations; (c) Template learned with 50 uniform angle samples; (d) Template learned using 50 samples from $PUT$ distribution, given in Fig. 4.10; (e) and (f) Template learned with 100 angle samples.

Figure 4.10: $PUT$ distribution $p(\theta)$ for angle $\Theta$: (a) Square shape; (b) Star-II shape.

Figure 4.11: Heart shape: Learned templates. (a) Using 20 angle samples drawn from a uniform distribution; (b) Using 20 angle samples drawn from a $PUT$ distribution; (c) Template learned with 100 angle samples.

Figure 4.12: Triangle shape: (a) Ideal shape; (b) Simulated realizations; (c) Template learned with uniform distribution for the angle; (d) Template learned with 10 $PUT$ angle-samples; (e) and (f) Template using 100 angle samples.

Table 4.1: Performance measures $\tilde{D}$ ($\times 10^3$).

<table>
<thead>
<tr>
<th>$p(\theta)$</th>
<th>Square</th>
<th>Star</th>
<th>Star II</th>
<th>Heart</th>
<th>Triangle</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uniform</td>
<td>0.764</td>
<td>0.483</td>
<td>0.287</td>
<td>0.163</td>
<td>0.487</td>
</tr>
<tr>
<td>$PUT$</td>
<td>0.395</td>
<td>0.184</td>
<td>0.210</td>
<td>0.155</td>
<td>0.239</td>
</tr>
<tr>
<td>100 Samples</td>
<td>0.079</td>
<td>0.092</td>
<td>0.130</td>
<td>0.114</td>
<td>0.059</td>
</tr>
</tbody>
</table>
4.5.2 Sampling from Models

In order to experimentally demonstrate that the model given by Eq. (4.14) represents an entire class of shapes, we sample shapes from the model. Fig. 4.15(c) illustrates a cloud of points simulated for Mickey face from FSSM, using 100 \( \Theta \)-samples drawn according to a \( PUT \) distribution. 10000 \((\tilde{\Theta}, R)\)-samples were drawn from the model as discussed in Section 4.4.1. A comparison of Fig. 4.15(b) and Fig. 4.15(c) reveals that both are nearly identical, which shows that any shape in the permissible shape region may be sampled from the model. Note that the choice of an angle distribution is crucial if one is to preserve singularities, such as cusps.

Before concluding this section, we present some results for star shapes sampled from an FSSM model. We used \( \sigma_s \) given by Eq. (4.25) with \( \gamma = 3 \) and \( \eta = 0.653 \) and shown in Fig. 4.16. Results for different values of \( \sigma_0 \) are given in Fig. 4.17, and clearly demonstrate its smoothing capacity.

![Figure 4.13: Brain shape: (a) Human brain; (b) Brain contour; (c) Simulated realizations; (d) Learned template.](image)

![Figure 4.14: Truck shape: (a) Ideal truck shape; (b) Simulated realizations of truck; (c) Learned template.](image)

![Figure 4.15: Shape simulation according to the distribution of Eq. 4.14: (a) Ideal shape; (b) Realizations; (c) Realizations simulated using parameters learned from (b).](image)
4.5.3 Classification

In this section, we demonstrate the potential of our proposed model for classification applications. We exploit a database of 739 shape realizations of car, triangle, star, square, heart, brain, truck, and star-II shapes. Variability of realization contours has a standard deviation of 2% of image size. Given any observation $R_S^j \in \{R_S^\ell; \ell = 1, \ldots, 739\}$, we tested the following $M$-ary hypotheses ($M = 7$):

$$H_i : R_S^j \text{ is the } i\text{th shape. : } i = 1, \ldots, 7,$$  \hspace{1cm} (4.26)

where the set of possible shapes is \{car, triangle, star, square, heart, brain, truck, star-II\} and $i$ indexes a shape.

Likelihoods $\{p_{S_i}^j\}_{i=1}^7$ for $R_S^j$ were computed and were subsequently used to classify the shapes with 100% success rate.

We, next, evaluate the robustness of an FSSM model against shear and occlusion.

**Shear:**

Shearing in 2D may be defined by a transformation $T_s = \begin{pmatrix} 1 & s_x \\ s_y & 1 \end{pmatrix}$ where $s_x$ and $s_y$ quantify the amount of shear in $x$ and $y$ directions. To evaluate the robustness to shearing, we applied the transformation $T_s$ to the learned templates, where $s_x$ and $s_y$ were varied from 0 to 0.5. Note that shapes sheared with factors of 0 to 0.2, are still correctly classified. This demonstrates the flexibility of the model which tries to capture any
variability within the permissible shape region. Increasing the amount of shear decreases the recognitions rates, which is justified, because shearing may map a shape out of the permissible region.

Table 4.2: Effect of Shearing on Classification (% classification rates).

<table>
<thead>
<tr>
<th>$s_y$ \ $s_x$</th>
<th>0.0</th>
<th>0.1</th>
<th>0.2</th>
<th>0.3</th>
<th>0.4</th>
<th>0.5</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>86</td>
<td>57</td>
<td>14</td>
</tr>
<tr>
<td>0.1</td>
<td>100</td>
<td>100</td>
<td>86</td>
<td>29</td>
<td>14</td>
<td>0</td>
</tr>
<tr>
<td>0.2</td>
<td>86</td>
<td>71</td>
<td>43</td>
<td>14</td>
<td>14</td>
<td>0</td>
</tr>
<tr>
<td>0.3</td>
<td>57</td>
<td>57</td>
<td>29</td>
<td>14</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0.4</td>
<td>57</td>
<td>43</td>
<td>14</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Occlusion:

We introduced random occlusions in shape realizations, ranging from 0 to 40% of the original shape in steps of 5%. We have recorded no misclassification for the above shape set, albeit for one occlusion per realization.

4.5.4 Application to Brown Database of Shapes

For evaluation purposes, we used shapes from Kimia database [86] as our test cases, some of which are shown in Fig. 4.18(a). Note that the training set that is presented here, consists of shapes which are very similar. This will allow us to demonstrate how FSSM performs when there is a significant overlap in the permissible regions of two different shapes.

The templates sampled from an FSSM are given in Fig. 4.18(b). A comparison of template likelihoods in terms of a likelihood matrix is given in Fig. 4.19, where each row corresponds to a template $R_i$ and each column represents the shape $S_j$ whose probabilistic model $p_{S_j}$ is used to compute the likelihood. For instance, Column 1 shows the likelihoods $p_{S_1}(R_i)$ of all templates using the Crown shape model ($j = 1$). Note that the indicated values are likelihoods normalized by $p_{S_i}(R_i)$, which means that diagonal elements are unity while off diagonal elements are less than 1.

The likelihood matrix clearly indicates that the training set selected in this experiment, consists of several shapes which are fairly similar, e.g., {F-18 and Mig-29}, {Bottle, Children, Key and Flat fish}, {Face and Textbox}, and {Car, Classic, and Brick} form sets
Figure 4.18: (a) Test shapes from Brown database (Heart, Crown, F-18, Car, Face, Brick, Misk, Key, Flat fish, Bottle, Text box, Mig-29, Classic, Fgen, and Children); (b) Learned Templates.

of similar shapes. This overlap in the permissible regions of various shapes will help in evaluating model performance in such adverse scenarios.

Figure 4.19: Comparison of normalized likelihoods of templates given in Fig. 4.18 (Rows:shapes, columns:model). Normalized likelihood varies from 0 (black) to 1 (white).

Finally, 192 test shapes from the Kimia database, shown in Fig. 4.20, were classified using FSSM models stored in the database. On average, it took 0.12s to compute normalized likelihood, and 1.82s to classify a shape using MATLAB on a 3.4GHz machine. Corresponding classification results are provided in Fig. 4.21. Although some of the shapes are highly similar as depicted in Fig. 4.19, which makes the classification task difficult, FSSM achieved a classification rate of 95.3%. Even in the presence of partial occlusion,
the method successfully classified very similar realizations of different shapes, for instance, F-18 and Mig-29. Similarity scores for various test shapes were computed as a ratio of model log-likelihood to shape log-likelihood, and are given in Figs. 4.22 and 4.23 for three best matches. Note that a high value of similarity score corresponds to a good match. It is evident from the figure that best matches are in accordance with shape similarity given by the likelihood matrix.

Figure 4.20: Test shapes from Kimia database.

4.6 Connection with Principal Curves

If FSSM template and the principal curve [32, 21] are learned for a given data set, we found that both led to the same result modulo the explicit smoothness constraint of the principal curve [32], which is probabilistically implied in our case. Roughly speaking a
Figure 4.21: Classification of shapes in Fig 4.20.
Figure 4.22: Similarity scores: (Right) Test shape; (Middle Right) Best match; (Middle Left) Second best match; (Left) Third best match.
Figure 4.23: Similarity scores: (Right) Test shape; (Middle Right) Best match; (Middle Left) Second best match; (Left) Third best match.
principal curve is defined as a curve that passes through the middle of a dataset [21], and the equivalence arises in using conditional modes for the template boundary, which minimize the least square error for the Gaussian case. This makes the criterion coincide with that of principal curves, forcing the boundary to pass through the middle. One such example is given in Fig. 4.24. Smoothness in the case of principal curves follows from the functional form of the curve, which may, however, in some cases fail to capture the closedness of the curve. FSSM template can, therefore, be regarded as yet another method for finding principal curves, provided that we have sufficient samples in our data set. In such a case, FSSM is more flexible and demonstrates several additional features, which may make it preferable to classical methods of finding principal curves. For instance, its mathematical formulation allows its use for classification and sampling purposes. In addition, since FSSM is a density based method, a learned template is less sensitive to outliers than its traditional counterpart.

![Figure 4.24: Equivalence between principal curves (blue) and FSSM (red).](image)

### 4.7 Conclusions

In this chapter, we presented a novel approach to statistical shape modeling exploiting an extended class of flexible skew-symmetric distributions, where each shape is represented by a distribution. Computer simulations show that the probabilistic nature of the FSSM model makes it quite effective and robust in capturing inherent shape variability. Shape singularities may be captured which to a large extent depends on an appropriate angle distribution that is automatically learned, thus eliminating the need for manual selection of landmarks. Applications of the models include classification and sampling. Although we employed maximum likelihood classifier, a MAP classifier may also be used. Classification
results demonstrate robustness of the model to occlusion and shear provided the realization contours do not violate the assumed permissible shape region. *FSSM* template may also be thought as an equivalent to principal curves, however, designed for a different goal.

**Table 4.3: Summary: FSSM learning.**

**Preprocessing:**

1. Normalize shape realizations to remove location, scale and orientation information. The realizations may be the output of edge detection and/or denoising algorithm(s).
2. Superimpose realizations on top of each other such that their centroids are aligned. Sample the superimposed image.
3. Choose the centroid as origin.
4. If not known *a priori*, find angles at which singularities are present. Densities of sample points may be used to identify their presence. Select (learn) an appropriate *PUT* angle distribution $p(\theta)$ accordingly.

**Shape Learning:**

5. Select size $K \geq 3$ for the model.
6. Choose $m$ samples for $\Theta \in [0, \pi]$ according to the distribution $p(\theta)$. For each $\Theta = \theta$, repeat steps 7 through 10.
7. Slice through the image at angle $\theta$. Identify two clusters of points which lie within some $\epsilon$-neighborhood of $\theta$, $N^\theta_\epsilon$.
8. Learn the bimodal conditional distribution $p(r|\theta)$, given by Eq. 4.8, for distances $\{r_j : (\theta, r_j) \in N^\theta_\epsilon, \forall j\}$ from the origin. For instance, the objective function $\mathcal{J}$, given by Eq. 4.12, may be minimized to find density parameters $(\xi, \sigma, \alpha)$.
9. Compute the modes of the bimodal density.
10. The set of modes for all the angles will constitute the boundary.
Chapter 5

2D Shape Modeling in Morse Theoretic Framework

As mentioned in Chapter 1, topology and geometry are the attributes that uniquely define a shape. Existing shape descriptors, however, pay little attention to the topology of shapes and instead operate on a smaller subset, where all shapes are assumed to have a genus of one. In this chapter, we will describe a novel 2D shape modeling method that keeps track of the topology of a shape in combination with its geometry for a robust shape representation. Using a Morse theoretic approach, we focus on representing planar shapes of arbitrary topology. The proposed approach extends existing modeling techniques in the sense that it encompasses a larger class of shapes.

In short, we represent a shape in the form of a topo-geometric graph, which encodes both of its attributes. The topological structure is captured via skeletal graphs, where the graph vertices correspond to the critical points. Geometry, on the other hand, is encoded by carefully annotating graph vertices. We show that this leads to a representation that is rigid transformation-invariant, and demonstrate that the original shape may be recovered from the model.

The balance of this chapter is organized as follows. We start the next section with motivation for our work. In Section 5.2, we present our proposed skeletal model and its properties. Section 5.3 details the geometric encoding of skeletal graphs. We conclude the chapter with substantiating examples in Section 5.4.
Figure 5.1: Importance of topological information: (a) a nut; (b) silhouette of a nut with minimal topological information; (c) segmented nut with all topological information.

5.1 Motivation and Background

As noted above, little work to our knowledge has attempted to account for topological features of shapes. In many practical instances, however, the topological information is of paramount importance. Stripping a shape of its topological information as illustrated in Fig. 5.1, makes recognition difficult even for a human observer. Our goal of accounting for such information (i.e., arbitrary topology) in shapes is to propose a shape model, which provides a unique rigid transformation-invariant signature and with sufficient information to reconstruct the shape.

In the proposed model, the topology of a shape is captured by a Morse theoretic skeletal graph constructed through the critical points of a distance function defined on it. The resulting skeletal graph also includes all geometric information of a shape contour by encoding the order in which it evolves. In other words, it preserves the properties of shock graphs, with additional characteristics to mitigate their limitations. In addition, we will demonstrate that our graph is a unique signature of a shape in the sense that one may reconstruct the original shape.

5.2 Topological Model

For topological analysis of 2D shapes, we use the distance function as a Morse function, which may be shown to be invariant under rotation, translation and scaling.
5.2.1 The Distance Function

Consider a distance function \( d : p \mapsto \|p\| \) in \( \mathbb{R}^2 \). Given a generic shape \( M \subset \mathbb{R}^2 \), the restriction of the distance function on \( M \),

\[
d : M \to \mathbb{R}_+,
\]

(5.1)
is a Morse function. The distance function may, therefore, be used in constructing skeletal graphs.

To study a compact shape by way of a Morse distance function, we start at \( d(p) = 0 \) and gradually increase the value of the distance function in \( K \) steps to a sufficiently large value which we denote \( b \). The integer \( K \) is called the resolution of the skeletal graph. The larger the resolution, the greater the precision of capturing the structural changes in the level sets of the distance function. Recall that such changes only occur at critical level sets.

The level sets of \( d \) are concentric circles. We find the intersection of the shape with circles of radius \( d \), for all \( d \in [0, b] \) and assign a vertex to each connected component in an intersection. This is illustrated in Fig. 5.2. Hence, skeletal graphs associated with the distance function may be described as a quotient space \( M/\sim \) where the equivalence relation \( \sim \) is defined as follows:

**Definition 5.1: (Equivalence)** Two points \( p \) and \( q \in M \) are equivalent, i.e., \( p \sim q \), if they belong to the same connected component of the level set of the function \( d \). Mathematically, \( p \sim q \) if \( d(p) = d(q) \) and \( p \in \text{ConnComp}_d(q) \).

The skeletal representation is, hence, a set of all such equivalence classes with each equivalence class mapped to the same value through a distance function. Mathematically, this quotient space is defined as \( M/\sim := \{[p] : p \in M\} \), where \( [p] = \{q \in M : q \sim p\} \) and
the equivalence relation $\sim$ is as defined above.

Note that $d$, given by Eq. (5.1), is translation dependent. However, if the origin is taken at the centroid $\mu$ of a shape, we achieve translation invariance:

$$d_\mu(p) := \|p - \mu\|.$$  \hfill (5.2)

We can introduce scale invariance through the following transformation:

$$\tilde{d}_\mu(p) = \frac{d_\mu(p) - d_{\min}}{d_{\max} - d_{\min}}.$$  \hfill (5.3)

**Proposition 5.2: (Invariance)** The distance function given by Eq. (5.3) is rotation, translation and scale invariant.

**Proof:** Proof follows directly as a special case of Proposition 6.1.

The above proposition demonstrates the invariance of the distance function to rigid body transformation under the condition that the centroid of the manifold must be translated to the origin.

### 5.2.2 Analysis of Planar Shapes

In order to capture the topology of a shape $M$, as shown in Fig. 5.3, we have to identify the special landmarks marked on $M$. To exploit the Morse function formalism, we concentrate on the boundary of $M$, which is composed of 3 disjoint sets $M_1, M_2$ and $M_3$.

Of course, if we were to use the distance function as a Morse function we could have easily identified the maxima and the minima of the independent curves. In practice, we have to identify the critical points, represent them as graph vertices and subsequently establish their mutual relationships, i.e., their connectivity to other critical points in a graph.

This requires a slightly different strategy, where a shape is considered as a whole. We focus on the neighborhood of $M_2^{\max}$ in Fig. 5.3(a), which is a 1-index critical point of a height function $h$. If we are moving in the decreasing $h$ direction, as we encounter $M_2^{\max}$, we detect a bifurcation of the shape in two branches. Although $M_2^{\max}$ is not a saddle point, it behaves in a similar way here and we, therefore, call it a *pseudo saddle point*. Its similarity with a saddle point is evident from the fact that it defines a change in the topology of a shape much like its actual counterpart on a surface. Clearly, if we look at the neighborhood around this point, we notice that as we move towards it, we indeed attach a 1-handle to the portion of the shape that lies above it. The idea of handle decomposition is, therefore,
Figure 5.3: Topological analysis of a 2D shape $\mathcal{M}$, where $\mathcal{M}_1, \mathcal{M}_2$ and $\mathcal{M}_3$ define the boundary of $\mathcal{M}$. (a) Critical points of $\mathcal{M}_1, \mathcal{M}_2$ and $\mathcal{M}_3$; (b) handle decomposition of $\mathcal{M}$.

still applicable and provides the basic framework for capturing topology. In other words, it allows us to establish the connectivity of vertices and, hence, to represent the topology of a planar object in the form of a skeletal graph.

A handle decomposition of an eight shape is shown in Fig. 5.3(b). Note that here we are using a height function to explain the idea, but the formalism is valid for any Morse function, and, hence, for our choice of distance function.

5.2.3 Algorithm

The algorithm for computing a Morse theoretic skeletal graph is illustrated in Fig. 5.4. Note that the level sets of the distance function are concentric circles and their intersections with any shape will always be circular arcs. In order to generate a topological graph, we start with a circle of smallest radius which is gradually increased. In the process, we monitor its intersections with the shape. Each intersection arc is subsequently specified with a graph vertex at its centroid defined as an arithmetic mean of the points lying on the arc. Connectivity between the vertices is established by looking at the connectivity of the circular arcs at two different levels. For instance, at a particular instant, we have a current circle $\mathcal{C}_c$ and a previous circle $\mathcal{C}_p$ as illustrated in Fig. 5.4. Analyzing the shape with $\mathcal{C}_p$ and $\mathcal{C}_c$ gives two sets of intersection arcs $\{A_{p1}, A_{p2}, A_{p3}, A_{p4}\}$ and $\{A_{c1}, A_{c2}, A_{c3}, A_{c4}\}$, which respectively yield two sets of vertices $\{N_{A_{pi}}\}_{i=1}^4$ and $\{N_{A_{ci}}\}_{i=1}^4$. To establish relationships between the two sets of vertices, we look at the regions enclosed between the two circles, which in this example are $\{\mathcal{M}_1, \mathcal{M}_2, \mathcal{M}_3, \mathcal{M}_4\}$. Note that there is only one arc at the current level that is connected to an arc at the previous level via a shape region. For instance, it is only $\mathcal{A}_{c1}$ that is connected to $\mathcal{A}_{p4}$ through $\mathcal{M}_1$. This allows us to add an edge.
Figure 5.4: Skeletonization of a 2D shape $\mathcal{M}$. Note that $\tilde{A}_c = A_{c_1} \cup A_{c_2} \cup A_{c_3} \cup A_{c_4}$, and $\tilde{A}_p = A_{p_1} \cup A_{p_2} \cup A_{p_3} \cup A_{p_4}$, $\tilde{M} = M_1 \cup M_2 \cup M_3 \cup M_4$.

$(N_{A_{p_1}}, N_{A_{c_1}})$ to the skeletal graph. Other edges in this example are determined similarly, i.e., $\{(N_{A_{p_i}}, N_{A_{c_i}})\}_{i=2}^4$.

We may now summarize the algorithm as given in Table 5.1.

**A Sampling View:**

Aside from the Morse theoretic framework, there is an alternative interpretation of
the previously described methodology. The intersections of a shape with concentric circles
may be viewed as an isotropic sampling process. This means that we need to define a point
spread function (PSF) that allows us to identify circular arcs. In polar coordinates, we
represent the PSF by $K(r, \theta)$. Analyzing a shape at a given point $(r, \theta)$ means convolving
it with the PSF:

$$\Lambda = S \ast K(r, \theta), \quad (5.4)$$

where $S$ is a shape in polar coordinates. This convolution is computed at all angles and
all radii to get $\{\{\Lambda(r, \theta) : \theta \in [-\pi, \pi] \} : r \in [d_{\text{min}}, d_{\text{max}}]\}$, which represents all inter-
section arcs and, therefore, defines all vertices in a skeletal graph. Note that the inner
set represents an intersection of a shape $S$ with a circle of radius $r$ and, therefore,
$\text{Mean} (\text{ConnComp} (\{\Lambda(r, \theta) : \theta \in [-\pi, \pi]\}))$ defines a graph vertex. The vertices whose degree
is not two identify critical points of the distance function.
Table 5.1: Skeletonization of planar shapes

1. Find the centroid of the shape \( \mathcal{M} \) as the arithmetic mean of the shape points.
2. Find \( d_{\text{max}} \) as the maximum distance from the centroid.
3. Given \( K \), define:
   \[
   r_k := k \frac{d_{\text{max}}}{K}, \quad k = 1, \ldots, K
   \]  \hspace{1cm} (5.3)
4. Generate the previous circle \( \mathcal{C}_P \) with radius \( R_p = r_1 \).
5. Find \( \tilde{A}_P = \mathcal{M} \cap \mathcal{C}_P \).
6. Find the connected components in \( \tilde{A}_P \). Each connected component \( A_P \) in \( \tilde{A}_P \) will be a circular arc. Assign a vertex \( N_{A_P} \) to each \( A_P \) at its centroid.
7. For \( k = 2 \) to \( K \):
   - Generate the current circle \( \mathcal{C}_C \) with radius \( R_c = r_k \).
   - Find \( \tilde{A}_C = \mathcal{M} \cap \mathcal{C}_C \).
   - If \( \tilde{A}_C = \mathcal{C}_C \), goto Step 7.
   - Find \( \tilde{M}_C = \mathcal{M} \cap ([\mathcal{C}_C] \cap [\mathcal{C}_P]) \), where \([\cdot]\) and \(\lceil\cdot\rceil\) identify interior and exterior of a closed contour. Hence, \( \tilde{M}_C \) will be the portion of \( \mathcal{M} \) that lies in between \( \mathcal{C}_P \) and \( \mathcal{C}_C \).
   - Find the connected components in \( \tilde{A}_C \).
   - For each connected component \( A_C \) in \( \tilde{A}_C \) do:
     - Assign a vertex \( N_{A_C} \) at the centroid of \( A_C \).
     - For each connected region \( \mathcal{M}_C \) in \( \tilde{M}_C \) do:
       * If the number of connected regions in \( \mathcal{M}_C \cup A_C \) is one, find the arc \( A_P \) in \( \tilde{A}_P \) such that \( (\mathcal{M}_C \cup A_C) \cup A_P \) has only one connected region. Connect \( N_{A_C} \) to \( N_{A_P} \).
     - end for
   - end for
   - \( \tilde{A}_C = \tilde{A}_C \).
7. end for
8. end for
Figure 5.5: Skeletonization of an eight shape with various graph resolutions $K$: (a) $K = 4$; (b) $K = 5$; (c) $K = 7$.

Figure 5.6: Skeletonization of a kettle: (a) $K = 4$; (b) $K = 16$.

**Physical Interpretation:**

In addition to its location, each graph vertex attribute includes the radius of the corresponding concentric circle. This allows us to keep track of the order in which vertices are birthed. This process may intuitively be viewed as a wavefront emanating at a point source located at a shape centroid and propagating outwards. The shape itself may be viewed as a dense material with some reflective index. For simplicity, we assume that the material has directional reflectivity, i.e., it can only reflect an outward propagating wave. Thus, as the wave propagates through the material at each instance a portion of it is reflected back to the source, which acts as the focal point of the reflecting medium. The time to record a reflected wave is proportional to the distance that it travels before reflection, which in turn equals the radius of the concentric circles. Keeping track of the order of vertex evolution largely aligns this method with the shock graph technique, which in turn differs from the medial axis based methods in terms of this additional information encoded in corresponding graphs.

Some illustrative skeletal graphs of shapes are given in Figs. 5.5 through 5.7 to demonstrate the potential of the proposed technique to capture topology.
5.3 Geometric Modeling

While the skeletal graphs presented in Section 5.2 demonstrate their capacity to capture topological structure, they fall short of a complete shape representation, i.e., a shape cannot be reconstructed given its skeletal representation due to lack of geometric information. In this section, we investigate encoding the geometric information in a graph with no additional cost. To proceed, first recall that graph vertices coincide with the centroid of the circular arcs, which is in turn defined as the arithmetic mean of the points in a given arc. We explore an alternative definition of a centroid resulting from a geometric construction to yield a complete shape representation.

5.3.1 Geometric Encoding of Vertices

Suppose we are given an arc $\alpha$ as shown in Fig. 5.8, with end points $A(x_A, y_A)$ and $B(x_B, y_B)$. This corresponds to an intersection arc of a level set of the distance function. Clearly, $A$ and $B$ lie on a shape boundary. Without loss of generality we assume that the shape centroid lies at the origin, then:

$$\alpha(x, y) : x^2 + y^2 = r^2.$$ (5.5)

1. Find the perpendicular bisector $CD$ of the line segment $AB$, where $D$ is the point where it intersects the circular arc $\alpha$.

2. Take the midpoint $N(x_N, y_N)$ of $CD$ as a new definition of the centroid. In other words, a vertex coincides with $N$.

With the above construction, given a vertex $N(x_N, y_N)$ at radius $r$, we can always recover boundary/landmark points $A$ and $B$ using a simple geometric manipulation as given below:
Reconstruction of Planar Shapes  The construction given above allows the recovery of boundary points from a given vertex. To that end, we first find the midpoint $C$ of the desired line segment $\overline{AB}$ (whose length is yet known). The following steps are involved (see Fig. 5.9):

1. Identify point $D$ as an intersection between $\overline{ON}$ and the circle $\beta$ of radius $r$
2. Find point $C(x_C, y_C)$ on $\overline{OD}$ such that $\overline{CN} = \overline{DN}$
3. Find the intersection of $\beta$ with the perpendicular to the segment $\overline{CD}$ passing through $C$
4. The two intersections, i.e., points $A(x_A, y_A)$ and $B(x_B, y_B)$ give the boundary points

Recall that each shape is represented by a graph $\mathcal{G}$ which consists of a set of vertices $\{N(r_i) : i = 1, \ldots, n\}$ corresponding to radii $\{r_i : i = 1, \ldots, n\}$. This means that we can recover the corresponding set of boundary points $\{A(r_i), B(r_i) : i = 1, \ldots, n\}$, and as the graph resolution goes to infinity we get a continuous boundary.
5.4 Experimental Results

To substantiate the preceding geometric encoding construction, we randomly simulate arcs and subsequently evaluate them for specific shapes.

5.4.1 Examples

Randomly generated arcs $\alpha$, parameterized as $\alpha(t) : t \in [0, 1]$, with end points $A = \alpha(0)$ and $B = \alpha(1)$ are illustrated in Fig. 5.10. Let $N_0$ denote a vertex computed using the old definition of the centroid. If we use $N = 0.5(C + D)$ as indicated in Step 2 of Section 5.3.1, the resulting vertex $N \neq N_0$, indicating an error. This will ultimately yield a skeletal graph that does not coincide with the skeletal graph constructed with the old definition.

![Figure 5.10: Illustration of difference between two vertex definitions ($N = 0.5(C + D)$).](image)

As illustrated in Fig. 5.10, the point $N$ needs to be moved closer to $N_0$ by adjusting the weights for $C$ and $D$ in $N(\gamma) = \gamma C + (1 - \gamma)D$. $\gamma$ should be such that it minimizes the mean square error between $N$ and $N_0$ for all cases. Such a minimization yields $\gamma = 0.4$. Note that decreasing $\gamma$ may move point $N$ closer to $N_0$ in some cases (see Fig. 5.11(a)), but may also overshoot $N_0$ in other cases (see Fig. 5.11(c)). There should, therefore, be an equilibrium between two forces that are pushing $N$ in opposite directions, as illustrated in the state given in Fig. 5.11(d) for $\gamma = 0.4$.

![Figure 5.11: Illustration of difference between two vertex definitions ($N = 0.4C + 0.6D$).](image)
We now reconstruct boundary points from a given vertex \(N\), the results for which are shown in Fig. 5.12.

![Figure 5.12: Reconstructed boundary points A and B.](image)

5.4.2 Application to Planar Shapes

Applying the methodology to shapes is illustrated in Figs. 5.13 and 5.14 where a comparison of results for old and new vertex definitions is carried out. We note that there is no visual difference between the two sets of graphs. The advantage of the new definition is that it allows shape reconstruction. Fig. 5.15 presents some additional results while Fig. 5.16 shows reconstructed shapes. Although the results shown in Fig. 5.16 actually reconstruct landmarks on the boundary, instead of the boundary itself, fitting a contour to these landmarks may be trivially carried out in most cases. Since landmark points lie on the boundary, active contours [92] or principal curves [40] may give reasonably good results due to the fact that the images are noiseless. Some results are given in Fig. 5.17.

Since we can reconstruct a shape from a graphical representation, we have experimentally confirmed that our skeletal graph forms a unique signature of the corresponding shape.

5.5 Conclusions

In this chapter, we addressed the problem of modeling a topologically diverse class of shapes, a previously unaddressed problem. Our approach, based on Morse theoretic skeletal graphs, not only models topology but fully captures the geometry of a shape as well. The model, which is inherently rotation, translation and scale invariant, is therefore, a simpler, unique and compact shape representation. Applications include storage and shape classification.
Figure 5.13: Skeletonization of a kettle: (a) old definition of a vertex; (b) new definition. (Left) \( K = 4 \); (center) \( K = 5 \); (right) \( K = 16 \).

Figure 5.14: Skeletonization: (a) old definition of a vertex; (b) new definition. (Left) horse, \( K = 5 \); (right) airplane, \( K = 8 \).

Figure 5.15: Skeletonization: (a) frog, \( K = 16 \); (b) camel, \( K = 16 \); (c) airplane, \( K = 100 \).
Figure 5.16: Reconstruction of shape landmarks: (a) $K = 8$; (b) $K = 100$.

Figure 5.17: Contour fitting: (a) eight shape; (b) horse shape.
Chapter 6

Topo-Geometric Modeling of 3D Shapes

We propose a new method for representing the topology and geometry of 2D compact surfaces embedded in a three-dimensional space. Topology is modeled through a skeletal graph, where a distance function is employed as a Morse function. Invariance to similarity transformations follows directly from the choice of the Morse function. Geometry, on the other hand, is captured by modeling the evolution of level curves of the distance function along topologically homogenous parts of the surface, which in turn correspond to various graph edges. Combining topological and geometric information leads to a weighted skeletal graph representation of a surface (Fig. 6.1 for instance), where edge weights are constructed from corresponding curve evolution models. Uniqueness and completeness of representation make the method particularly suitable for storage, surface reconstruction, and shape recognition problems.

The chapter is organized as follows: We start with topological modeling of a 3D surface and describe the construction of a similarity transformation invariant skeletal graph in Section 6.1. Geometric modeling and its implementation are presented in Section 6.2 and Section 6.3, respectively. We conclude the chapter with substantiating examples in Section 6.5.
6.1 Topological Model

In this section, we employ Morse theory for studying the topology of smooth compact surfaces (two-dimensional manifolds) embedded in \( \mathbb{R}^3 \), which may be locally parameterized as

\[
\phi : \Omega \rightarrow \mathcal{M}
\]

that is, \( \Omega \ni u \mapsto \phi(u) \in \mathcal{M} \), where an open connected set \( \Omega \subset \mathbb{R}^2 \) represents the parameter space. As outlined in Chapter 3, Morse theory allows one to relate the topology of a smooth manifold with the number of critical points of a Morse function \( f : \mathcal{M} \rightarrow \mathbb{R} \subseteq \mathbb{R} \) on the manifold \( \mathcal{M} \).

In this chapter, we identify the points in \( \mathbb{R}^3 \) with their position vectors, which are typed in bold. The parameter space \( \Omega \) is two-dimensional and the points in \( \Omega \) are written as \( (u, v) \). Thus, if \( x \) belongs to the surface, we write \( x = x(u, v) \).

6.1.1 Scaling and Rigid Motion Invariant Skeletal Graph

As discussed in Chapter 3, height function has been employed for topological analysis in Reeb graphs, which suffer from a drawback that the resulting Reeb representation is not rotation invariant. To address this problem, we propose to use distance function as our choice of Morse function for graph construction, which automatically yields invariance to similarity transformations.
The distance function

A distance function defined on a surface $\mathcal{M}$ maps each point $p$ on a surface $\mathcal{M}$ to its distance from the origin, i.e., $d : p \mapsto \|p\|$, $\forall p \in \mathcal{M}$. One can show that for generic surfaces $\mathcal{M} \subset \mathbb{R}^3$, the restriction of the distance function $d : \mathcal{M} \rightarrow \mathbb{R} \subseteq \mathbb{R}_+$ on $\mathcal{M}$ is a Morse function. We, therefore, use it for constructing skeletal graphs.

Note that the function $d$ given above is not invariant with respect to translation and scaling. In order to achieve this invariance, we take the origin at the centroid $\mu$ of the surface of interest and scale the surface accordingly to get:

$$d_\mu(p) := \|p - \mu\|,$$
$$\tilde{d}_\mu(p) = \frac{d_\mu(p) - d_{\min}}{d_{\max} - d_{\min}}. \quad (6.2)$$

**Proposition 6.1: (Invariance)** The distance function given by Eq. (6.2) is rotation, translation and scale invariant.

*Proof:* We will prove this statement for two-dimensional manifolds. Let $\mu = (\mu_x, \mu_y, \mu_z)$ be the centroid of the manifold $\mathcal{M}$, and let $p$ be an arbitrary point on $\mathcal{M}$.

We first confirm that the function $d_\mu$ is translational and rotational invariant. An arbitrary Euclidean transformation of $\mathbb{R}^3$ is given by $p \mapsto Ap + b$, where the vector $b$ represents the translation and the orthogonal matrix $A$ represents the rotation. Since $\mu$ undergoes the same transformation, we have

$$d_{A\mu + b}(Ap + b) = \|(Ap + b) - (A\mu + b)\|$$
$$= \|A(p - \mu)\|$$
$$= \|p - \mu\|$$
$$= d_\mu(p).$$

Therefore, $\tilde{d}_\mu$ is invariant with respect to rotations and translations in $\mathbb{R}^3$.

Now if we scale the manifold by a factor of $a$, then

$$d_\mu(a p) = \sqrt{(ax)^2 + (ay)^2 + (az)^2}$$
$$= a\sqrt{x^2 + y^2 + z^2}$$
$$= a d_\mu(p),$$
and therefore,

\[ \tilde{d}_\mu(a p) = \frac{d_\mu(a p) - a d_{\min}}{a d_{\max} - a d_{\min}} = \frac{d_\mu(p) - d_{\min}}{d_{\max} - d_{\min}} = \tilde{d}_\mu(p). \]

Proposition 6.1 demonstrates the invariance of the distance function to translation, rotation and scaling under the condition that the centroid of the manifold/surface must be translated to the origin.

### 6.1.2 Equations for the Critical Points of the Distance Function

In order to identify the topological type of a surface, we need to locate the critical points of the distance function. In this section, we present an analytic solution to the problem.

Assume that the surface \( M \) is locally parameterized as a patch, \( p = (u, v, g(u, v)) \), and that the centroid is located at the origin. The distance function \( d(x, y, z) = \sqrt{x^2 + y^2 + z^2} \) restricted to \( M \) reads

\[ \tilde{d}(u, v) = d|_M = \sqrt{u^2 + v^2 + g^2(u, v)}. \]

The partial derivatives of \( \tilde{d} \) are

\[ \frac{\partial \tilde{d}}{\partial u} = \frac{2u + 2gg_u}{2\sqrt{u^2 + v^2 + g^2(u, v)}}, \]

\[ \frac{\partial \tilde{d}}{\partial v} = \frac{2v + 2gg_v}{2\sqrt{u^2 + v^2 + g^2(u, v)}}. \]

At a critical point the partial derivatives of \( \tilde{d} \) vanish. Therefore, the critical points of the distance function on the surface \( M \) are the solutions of the system

\[ g_u(u, v) + \frac{u}{g(u, v)} = 0, \]

\[ g_v(u, v) + \frac{v}{g(u, v)} = 0. \]

The solution of these coupled partial differential equations leads to the critical points, which may then be used to find the genus of a shape via Euler’s characteristic.
Although this approach leads to a numerical solution, complexity lies in one’s ability to find a patch to represent a surface, which may not be straightforward for complex shapes. In addition, this approach is highly sensitive to noise, due to its dependence on higher order derivatives. We, therefore, adopt an algorithmic approach for topological analysis, by finding a topological structure that gives us more than simple topological information, eventually leading to a machinery for capturing complete shape information.

6.1.3 Algorithmic Approach to Topological Analysis

The basic idea for topological analysis of a compact surface by a distance function as a Morse function, follows from the Morse deformation lemma, which states that a change of topology occurs only at the critical level of the Morse function. We, therefore, need to study level sets of the distance function, and to identify the levels where a change of their topology occurs.

This requires scanning the surface with level sets of the distance function, which happen to be concentric spheres. We, therefore, gradually increase the value of the distance function in $K$ steps from 0 to a sufficiently large value, say $b$, and find the intersections of the surface with corresponding spheres of radii $r \in [0, b]$. The parameter $K$, in other words, acts as the resolution of the skeletal graph. The larger the $K$, the better the precision of capturing the structural changes in the level sets of the distance function.

We, then, assign a node to each connected component in an intersection as illustrated in Fig. 6.2. The skeletal graph associated with the distance function may, hence, be described as a quotient space $\mathcal{M}/\sim$ where the equivalence relation $\sim$ is defined as follows:

**Definition 6.2:** (Equivalence) The points $p$ and $q \in \mathcal{M}$ are equivalent if they belong to the same connected component of the level set of the function $d$. We write this equivalence as $p \sim q$.

Distance function based topological graph is, therefore, a quotient space $\mathcal{M}/\sim:= \{[p] \mid p \in \mathcal{M}\}$, where the equivalence class $[p]$ of the point $p \in \mathcal{M}$ is the set of all points $q \in \mathcal{M}$ such that $q \sim p$. 
Algorithm

The definition of distance function based topological graph given above, yields a construction algorithm, which is illustrated in Fig. 6.3 and given in Table 6.1.

The algorithm yields a graph similar to the one shown in Fig. 6.4(a), which is composed of edge segments between various nodes. Note that not all of these nodes are related to the critical points of the distance function, which demonstrates redundancy present in the learned graph. We eventually remove this redundancy via graph simplification by merging the nodes (and the edge segments) which lie on a topologically homogeneous path along the graph. A simplified graph is shown in Fig. 6.4(b), where the merging of redundant nodes gives rise to graph vertices and the collapsing of redundant edge segments forms graph edges. Note that the vertices of the graph in Fig. 6.4(b) correspond to critical levels of the distance function, and mark a change in the topology of level curves, while the number of
Table 6.1: Algorithm for Constructing Topological Graph

- Find the centroid of the surface \( \mathcal{M} \) as the arithmetic mean of the vertices of the triangulated mesh and place the origin at the centroid.
- Find \( d_{\text{max}} \), the maximum distance from the centroid to \( \mathcal{M} \).
- Given \( K \), define:
  \[ r_k := k \frac{d_{\text{max}}}{K}, \quad k = 1, \ldots, K \]
- Generate the spheres \( S_1 \) and \( S_2 \) with radii \( R = r_1 \) and \( R = r_2 \), respectively.
- Find \( \tilde{\mathcal{M}}_p = \mathcal{M} \cap ([S_1] \cap [S_2]) \), where \([\cdot]\) and \(\lceil\cdot\rceil\) identify the interior and exterior of a closed surface; \( \tilde{\mathcal{M}}_p \) is, therefore, the part of \( \mathcal{M} \) that lies between \( S_1 \) and \( S_2 \).
- Assign a node \( N_{\tilde{\mathcal{M}}_p} \) to each connected component \( \mathcal{M}_p \) of \( \tilde{\mathcal{M}}_p \) at the centroid of \( \mathcal{M}_p \).
- For \( k = 3 \) to \( K \)
  - Generate the “current” sphere \( S_k \) with radius \( R = r_k \).
  - Find \( \tilde{\mathcal{M}}_c = \mathcal{M} \cap ([S_{k-1}] \cap [S_k]) \). Hence, \( \tilde{\mathcal{M}}_c \) is the portion of \( \mathcal{M} \) that lies in between \( S_{k-1} \) and \( S_k \).
  - Find the connected components \( \mathcal{M}_c \) of \( \tilde{\mathcal{M}}_c \).
  - For each \( \mathcal{M}_c \in \tilde{\mathcal{M}}_c \) do
    * Assign a node \( N_{\mathcal{M}_c} \) at the centroid of \( \mathcal{M}_c \).
    * Find the connected region \( \mathcal{M}_p \in \tilde{\mathcal{M}}_p \) such that \( \mathcal{M}_c \cup \mathcal{M}_p \) is a single connected region. Add an edge segment between \( N_{\mathcal{M}_c} \) and \( N_{\mathcal{M}_p} \).
  - end for
- \( \tilde{\mathcal{M}}_p = \tilde{\mathcal{M}}_c \).
- end for.

edges connecting two nodes equals the number of connected components of the level curves of the distance function between two critical points. On the other hand, the number of cycles reflect the genus of a given surface. In subsequent discussion, we only consider the simplified graph.

6.2 Geometric Model

The skeletal representation given in the preceding section completely describes the topological structure of a surface. It, however, suffers from two limitations, both arising from the lack of complete geometric information. First, it does not allow surface reconstruction.
Second, any shape recognition based only on topological information will be unable to differentiate geometric changes, resulting in low recognition rates. We, therefore, proceed to capture geometric information, which will then be combined with a topological model for complete shape representation.

In order to capture geometry, we note that the algorithm given in Section 6.2, not only constructs topological graph but also yields level curves of the distance function. We, therefore, model the evolution of these curves along various graph edges corresponding to topological homogeneous parts. Since we are working with compact surfaces, a non-singular connected component of a level set is always a simple closed curve. The example given in Fig. 6.5 illustrates the idea, where a height function is employed for illustration purposes, and the above algorithm results in a single edge graph. Clearly, any surface may be considered as a dense set of level curves. Given a small subset of these curves, the problem is to model their evolution in a way so as to reconstruct the whole surface. The smaller subset in turn is generated via a proper choice of graph resolution parameter. The constructed curve model is then used to learn some weights, which are finally assigned to respective edges. We will later demonstrate that a surface reconstructed from a weighted graph represents the original surface arbitrarily closely.

### 6.2.1 Curve Evolution Model

An acceptable curve modeling framework is the one that not only preserves the topology of individual level curves, but also the smoothness of the corresponding surface. The idea is to map each curve to a higher dimensional space and to fit a smooth trajectory
Figure 6.5: Illustration of geometric modeling: (a) A 3D object; (b) Object sampled at levels $r_1, \ldots, r_m$; (c) Intersections $C_1, \ldots, C_m$ embedded in $\Lambda$ bounding box to compute the distance field; (d) Vectorizing the elements of the distance field yields an $n$ dimensional vector $\rho_i$ for each $C_i$.

Figure 6.6: A level curve of double torus: (a) Curve lies in $\mathbb{R}^3$; (b) Projection to $D \subset \mathbb{R}^2$ via spherical coordinate transformation.

that passes through these points. Topological preservation is achieved through distance fields, which have been successfully used for capturing topological variations in level set methods [75], whereas smoothness of the trajectory is enforced by using elasticæ. We combine these two properties by formulating a curve evolution model in the space of distance fields instead of the space of curves.

In order to explain the idea, note that level curves of the distance function are spatial curves, each curve being a subset of a sphere. Spherical coordinates, therefore, map these level curves onto the curves in $\Lambda = [-\pi, \pi] \times [-\frac{\pi}{2}, \frac{\pi}{2}]$, as illustrated in Fig. 6.6. Since the sphere cannot be covered by a single chart, the closed curves on the sphere may produce non-closed curves in $\Lambda$. An example of such a case is shown in Fig. 6.7(a) which arises when the original level curve cycles around the $z$-axis. However, the curve modeling scheme [60, 3] that we adopt here is strictly valid for closed curves. There are various ways to address this
situation. In some cases, it is adequate to rotate the coordinate system in $\mathbb{R}^3$ in order to move the $z$-axis out of the level curve. In other situations, when this rotation may either be impossible or undesirable to perform, we use appropriate coordinates on the sphere to carry out our calculations. For instance, when the level curve is close to a big circle on a sphere, the stereographic projection is a good choice of coordinates. We, henceforth, assume that all transformed curves are closed.

In order to model these simple closed curves, we view each curve as a point in a high dimensional space, and fit a trajectory that passes through these points while preserving the curve topology and surface smoothness. As mentioned earlier, for topology preservation, we employ a signed distance field, which is always bounded, since surfaces of interest are compact.

Let $C_r$ be a closed curve that corresponds to the $r$-level curve of the distance function. We represent $C_r$ in terms of $\theta - \phi$ curve corresponding to the spherical parameterization. This allows us to map $C_r$ to the bounded box $\Lambda$. For such $C_r$, we define the signed distance field $\rho_r : \Lambda \rightarrow \mathbb{R}$ by

$$
\rho_r(x, y) = \begin{cases} 
+D((x, y), C_r) & \text{if } (x, y) \in [C_r] \\
-D((x, y), C_r) & \text{if } (x, y) \in [C_r], 
\end{cases} 
$$

where $D((x, y), C_r)$ denotes the distance from the point $(x, y)$ to the set $C_r$, i.e.,

$$
D((x, y), C_r) = \min_{(u, v) \in \tilde{C}_r} \tilde{D}((x, y), (u, v)),
$$

where $\tilde{D}$ is the Euclidean distance, and $[C_r]$ and $[C_r]$ represent the interior and exterior of $C_r$, respectively. This is illustrated in Fig. 6.8. The contour $C_r$ itself corresponds to the
Figure 6.8: Signed distance field: (a) Original curve; (b) Level sets of distance field.

isoset $\rho^{-1}_r(0)$ and since it corresponds to a curve at level $r$, we may think of it as a subset of $\Lambda \times \mathbb{R}_+$. In practice, $\Lambda$ is replaced with an $n_1 \times n_2$ grid, and vectorizing the $n = n_1 n_2$ elements of the distance field defined on $\Lambda$ yields a function $\rho : \Lambda \to \mathbb{R}^n$ whose components are $(\rho_1, \ldots, \rho_n)$. This is illustrated in Fig. 6.5, where a vase is sampled by $m$ horizontal planes at levels $r_1, \ldots, r_m$ to get intersection curves $C_1, \ldots, C_m$ shown in Fig. 6.5(b) which are then embedded in the bounding box $\Lambda$ as in Fig. 6.5(c) to compute the distance field. The vectorization of the corresponding distance fields yields a collection of $m$ points, $\rho_1, \ldots, \rho_m$ in $\mathbb{R}^n$ as depicted in Fig. 6.5(d). Each point is a vectorized distance field and our goal is to model a trajectory that best fits these points in $\mathbb{R}^n$ according to some criterion such that the original vase may be reconstructed from the level curves.

We address the problem by adopting a piecewise interpolation approach, where for all $i$, we fit an arc $\rho^{(i)}$ between the points $\rho_i$ and $\rho_{i+1}$ in $\mathbb{R}^n$ subject to the constraints:

$$
\begin{align*}
\rho^{(i)}(r_i) &= \rho_i, \\
\rho^{(i)'}(r_i) &= v_i, \\
\rho^{(i)}(r_{i+1}) &= \rho_{i+1}, \\
\rho^{(i)'}(r_{i+1}) &= v_{i+1}.
\end{align*}
$$

(6.4)

where the end point tangent vectors $v_i$ and $v_{i+1}$ are computed from the data, and where the arc $\rho^{(i)}$ is parameterized by $r$. The purpose of introducing tangent constraints is to ensure smoothness at joints (end points), when later we combine arcs for all such consecutive pair of points. The optimal arc, $\rho^{(i)}(r)$, is the one that minimizes a certain energy functional subject to the stated constraints. We, thus, construct a sequence of arcs between successive points, which when put together, yields a $C^1$ trajectory from $\rho_1$ to $\rho_m$, passing through $\rho_2, \rho_3, \ldots, \rho_{m-1}$. Note that each arc $\rho^{(i)}(r)$ belongs to an affine subspace of $\mathbb{R}^n$ through
the point $\rho_i$ and spanned by the vectors $v_i$, $v_{i+1}$, and $\rho_{i+1} - \rho_i$.

### 6.2.2 Dimension reduction

Finding the optimal arc in $\mathbb{R}^n$ with $n \gg 3$ is computationally very intensive. This complexity may, however, be greatly reduced by projecting the problem to $\mathbb{R}^3$ by a set of transformations, which are derived by noting that the arc belongs to an affine subspace of $\mathbb{R}^n$ through the point $\rho_i$ and spanned by vectors $v_i$, $v_{i+1}$, and $d_i := \rho_{i+1} - \rho_i$ [3] (See Fig. 6.9(a)). These vectors are generically independent but not necessarily orthogonal. We, therefore, employ Gram–Schmidt orthogonalization to get an orthonormal set of basis vectors $\{b_k, k = 1, 2, 3\}$ spanning the above mentioned subspace.

\[
\begin{align*}
    b_1 &= v_1, \\
    b_2 &= \frac{v_2 - (v_2, b_1)b_1}{\|v_2 - (v_2, b_1)b_1\|}, \\
    b_3 &= \frac{d - (d, b_1)b_1 - (d, b_2)b_2}{\|d - (d, b_1)b_1 - (d, b_2)b_2\|}.
\end{align*}
\]

This allows us to map the vectors $v_i, v_{i+1}, d_i \in \mathbb{R}^n$ onto vectors:

\[
\begin{align*}
    w_1 &= e_1, \\
    w_2 &= (v_{i+1}, b_1)e_1 + (v_{i+1}, b_2)e_2 + (v_{i+1}, b_3)e_3, \\
    w_3 &= (d_i, b_1)e_1 + (d_i, b_2)e_2 + (d_i, b_3)e_3,
\end{align*}
\]

where $e_1, e_2, e_3$ form the canonical basis for $\mathbb{R}^3$ and $w_k \in \mathbb{R}^3, k = 1, 2, 3$. The problem is now reduced to finding elastic $\alpha^{(i)} : I = [r_i, r_{i+1}] \rightarrow \mathbb{R}^3$ satisfying $\alpha^{(i)}(r_i) = 0, \alpha^{(i)}(r_{i+1}) = w_3$.
with starting and ending tangents \( \frac{\alpha^{(i)}(r_i)'}{\|\alpha^{(i)}(r_i)\|} = w_1 \) and \( \frac{\alpha^{(i)}(r_{i+1})'}{\|\alpha^{(i)}(r_{i+1})\|} = w_2 \), respectively as shown in Fig. 6.9(b). We choose the optimal solution to the problem as the one that minimizes the bending energy:

\[
E(\alpha^{(i)}) = (s_{i+1} - s_i) \int_{s_i}^{s_{i+1}} \kappa_{\alpha^{(i)}}^2(s) \, ds, \tag{6.7}
\]

where \( s \) is the arc length and \( \kappa_{\alpha^{(i)}} \) is the curvature. Since \( ds = \|\alpha^{(i)}'(r)\| dr \), and the curvature is:

\[
\kappa_{\alpha}(r) = \frac{\|\alpha'(r) \times \alpha''(r)\|}{\|\alpha'(r)\|^3},
\]

the bending energy (6.7) becomes:

\[
E(\alpha^{(i)}) = (r_{i+1} - r_i) \int_{r_i}^{r_{i+1}} \frac{\|\alpha^{(i)}'(r) \times \alpha^{(i)}''(r)\|}{\|\alpha^{(i)}'(r)\|^2} \, dr, \tag{6.8}
\]

whose minimization subject to above mentioned constrains [60] yields the optimal trajectory \( \alpha^{(i)}(r) \).

To reconstruct the part of the surface corresponding to the interval \([r_i, r_{i+1}]\), we need to traverse this trajectory. Hence, for any \( r \in [r_i, r_{i+1}] \), the corresponding \( \alpha^{(i)}(r) \in \mathbb{R}^3 \) is mapped to a unique \( \rho^{(i)}(r) \in \mathbb{R}^n \):

\[
\rho^{(i)}(r) = \rho_i + \langle \alpha^{(i)}(r), e_1 \rangle b_1 + \langle \alpha^{(i)}(r), e_2 \rangle b_2 + \langle \alpha^{(i)}(r), e_3 \rangle b_3, \tag{6.9}
\]

Each \( \rho^{(i)}(r) \), therefore, models the vectorized distance fields of the level curves for \( r \in [r_i, r_{i+1}] \). To recover a level curve from \( \rho^{(i)}(r) \), we first need to unvectorize it to get the corresponding distance field which is defined on \( \Lambda \), and then to find the zero level set of this distance field. For a complete representation of an entire topologically homogenous part of a surface, we glue together the corresponding \( \rho^{(i)}(r) \) to get a trajectory \( \rho(r) \subset \mathbb{R}^n, r \in [r_1, r_m] \), which is \( C^1 \) smooth and is in essence a piecewise curve modeling approach:

\[
\rho(r) = \bigcup_{i=1}^{m-1} \rho^{(i)}(r) 1 (r \in [r_i, r_{i+1}]), \tag{6.10}
\]

where \( 1(\cdot) \) is an indicator function, which assumes a value of unity when the argument is true. The distance field trajectory \( \rho \), given by Eq. 6.10, is, therefore, the geometric model, capturing the evolution of level curves. Each graph edge is finally assigned the corresponding model of \( \rho(r) \subset \mathbb{R}^n \).
6.2.3 Implementation Issues

We now discuss some implementational issues.

Tangent computation

Given a set of $m$ curves, $\rho_1, \ldots, \rho_m$, the starting and ending tangents, $v_i$ and $v_{i+1}, i = 1, \ldots, m$, may be approximately computed as in [3]:

$$v_k = \frac{\rho_k - \rho_{k-1}}{\|\rho_k - \rho_{k-1}\|}, \quad k = i, i + 1.$$  \hfill (6.11)

To avoid a longer vector (of larger norm) overbiasing the direction of the approximate tangent vector, we use a weighted difference of the two curve vectors,

$$v_k = \frac{\eta_{k-1}\rho_k - \eta_k\rho_{k-1}}{\|\eta_{k-1}\rho_k - \eta_k\rho_{k-1}\|},$$  \hfill (6.12)

where $\eta_k = \|\rho_k\|; k = i, i + 1$.

Tangents at the Terminal Points

At the initial and terminal points, i.e., $i = 1$ and $i = m$, we find starting and ending tangents respectively as

$$v_1 = \frac{\eta_0\rho_1 - \eta_1\rho_0}{\|\eta_0\rho_1 - \eta_1\rho_0\|},$$  \hfill (6.13)

$$v_{m+1} = \frac{\eta_m\rho_{m+1} - \eta_{m+1}\rho_m}{\|\eta_m\rho_{m+1} - \eta_{m+1}\rho_m\|},$$  \hfill (6.14)

where we assume that $\rho_0$ and $\rho_{m+1}$ are well defined. This is shown in Fig. 6.10.

Graph resolution

The topo-geometric model affords reconstruction of parts of a surface using the geometric model and their connection using topological information. This in turn suggests that for reconstruction purposes, we need not to keep track of entire subparts but actually
only a smaller subsets of level curves. The number of level curves is determined by the graph resolution parameter $K$ defined earlier. Due to the absence of a 2D sampling theorem, we need to adopt a practical approach for selecting the “optimal” subset of level curves. Since each curve is mapped to a point in a higher dimensional space of distance fields, we exploit correlations among various curves to rule out the curves which lie very close to other curves in the optimal set of curves. We, therefore, sample a surface very finely (arbitrarily large $K$), to get a large number of redundant curves. To find the optimal subset, we start with just two curves, for which we choose the outermost curves. This corresponds to curves at the maximum and minimum distance along the corresponding subpart. We, then, enlarge this subset by selecting two curves at a time, which are dissimilar to the curves in the existing subset and adding them to the optimal set. This is repeated by moving inward from the two outermost curves, until all curves are handled. As a measure of dissimilarity, we use the correlation of a curve with its immediate neighbor, and regard them dissimilar if this correlation is less than some bound $\epsilon_B$. With an appropriate choice of $\epsilon_B$, we can represent a surface arbitrarily closely.

6.3 Topo-Geometric Model

We now encode the trajectory $\rho(r)$ by a finite dimensional weight vector. Note that segments of $\rho(r) \in \mathbb{R}^n$ have one-to-one mapping with corresponding $\alpha^{(i)}(r) \in \mathbb{R}^3$. A smooth trajectory $\alpha$ in $\mathbb{R}^3$ corresponding to $\rho$ is obtained by gluing these $\alpha^{(i)}$, while simultaneously translating and rotating them to have the end points aligned, as described in the following section. Graph edges are finally encoded by corresponding $\rho$ or $\alpha$, thus resulting in a complete shape representation, which allows surface reconstruction.

6.3.1 Approximation of a Curve in $\mathbb{R}^n$

One way to encode geometric information along graph edges is to assign a weight vector comprised of the coefficients of a polynomial or spline approximation of the curve $\rho(r) \subset \mathbb{R}^n$. Performing such approximation in a high-dimensional space is, however, computationally inefficient. It is preferable to perform approximations in $\mathbb{R}^3$. 
6.3.2 Approximation of Elasticæ Segment in \( \mathbb{R}^3 \)

The curve \( \rho \) is obtained by lifting the segments \( \alpha^{(i)} \in \mathbb{R}^3 \) according to Eq. (6.9) and then gluing them together to get a curve. Each \( \alpha^{(i)} \) represents a curve segment between two consecutive nodes on the same edge. It can be approximated by splines, and the corresponding coefficients can be assigned to each segment. This method assigns a weight vector to each edge segment between two consecutive nodes. Let \( r_0 \) and \( r_K \) be radii that correspond to two vertices of a skeletal graph, connected by an edge. The points \( r_1, \ldots, r_{K-1} \), divide the segment \([r_0, r_K]\) into \( K \) intervals. For each interval \([r_i, r_{i+1}]\), we obtain a curve \( \alpha^{(i)} \) in \( \mathbb{R}^3 \) according to Section 6.2. Thus, for each edge, we construct \( K \) sets of spline coefficients. Each set corresponds to an edge segment along that particular edge. These coefficients encode the geometric information to any desired precision, and are assigned to the edge segments as weights. A possible drawback of this method is that the size of weight vectors increases with the number of edge segments even if the level curves themselves do not have a significant variation along them. In addition, such a representation is highly local and falls short of a global representation of an edge, i.e., a topologically homogeneous part of a surface.

6.3.3 Piecewise Projection

Another method of constructing the weight vectors is based on the following approach. While the smooth curve \( \rho \subset \mathbb{R}^n \) is obtained by lifting and gluing elasticæ segments \( \alpha^{(i)} \), the segments themselves start at the origin of \( \mathbb{R}^3 \). These segments are, thus, translated in \( \mathbb{R}^3 \) to have the end points matched. This produces cusps at the end points because each curve segment \( \alpha^{(i)} \) was computed in its own basis. Recall that the initial tangent vector \( w_1 \) is always equal to \( e_1 \) as shown in Fig. 6.11(a). In order to eliminate these cusps, we rotate the curve segments so that the tangents become aligned. In this way, we obtain a \( C^1 \) curve in \( \mathbb{R}^3 \) as shown in Fig. 6.11(b).

An alternative approach consists of performing rotation of the arcs of the curve \( \rho(r) \subset \mathbb{R}^n \), so that the resulting smooth curve lies in a three-dimensional affine subspace of \( \mathbb{R}^n \). We call this procedure the piecewise projection. Recall that the arc of \( \rho^{(i)}(r) \) connecting the \( i \)th and \( (i+1) \)st node lies in the three-dimensional affine subspace \( E_i \) of \( \mathbb{R}^n \) spanned by the vectors \( \mathbf{d}_i, \mathbf{v}_i, \mathbf{v}_{i+1} \), and passing through the point \( \rho_i \). The intersection of subspaces \( E_i \) and \( E_{i+1} \) contains the vector \( \mathbf{v}_{i+1} \) as illustrated in Fig. 6.12(a). Each consecutive pair
Figure 6.11: Effect of basis rotation: (a) See how $\mathbf{\alpha}_4$ and $\mathbf{\alpha}_5$ map to $\mathbf{\alpha}'_4$ and $\mathbf{\alpha}'_5$ after projection from $\mathbb{R}^n$ to $\mathbb{R}^3$, resulting in a high curvature cusp at the node; (b) Rotating the basis at the nodes makes the Frenet frame smoother and the resulting trajectory is first order smooth.

of arcs $\rho^{(i)}(r), \rho^{(i+1)}(r)$, therefore, lies in a five-dimensional affine subspace spanned by the vectors $\mathbf{v}_i, \mathbf{v}_{i+1}, \mathbf{v}_{i+2}, \mathbf{p}_{i+1} - \mathbf{p}_i, \mathbf{p}_{i+2} - \mathbf{p}_{i+1}$, and passing through $\mathbf{p}_{i+1}$. By applying an orthogonal transformation $T$ that preserves the vector $\mathbf{v}_{i+1}$, we can “align” the subspaces $E_i$ and $TE_{i+1}$, the image of $E_{i+1}$ under this transformation. In other words, the three-dimensional affine subspaces $E_i$ and $TE_{i+1}$ are the same. We subsequently perform this “alignment” at each node, eventually obtaining a smooth curve in a three-dimensional affine subspace of $\mathbb{R}^n$. The procedure is illustrated in Fig. 6.12, where the three-dimensional affine subspaces $E_i$ are drawn as quadrilaterals. Two consecutive affine subspaces have a common line, and a sequence of such subspaces appears like a folded sheet of paper with folds along common lines. The curve $\rho$ is $C^1$-smooth, since it is tangent to the folds. The rotation that preserves the vectors $\mathbf{v}_i$ corresponds to the process of unfolding the sheet of paper (Fig. 6.12(b)). The result is a smooth curve $\beta \subset \mathbb{R}^3$. On the computational level, this is equivalent to performing rotations of the basis in $\mathbb{R}^3$, as described in the first paragraph.

To assign a weight vector to the entire edge, we use a spline or polynomial approximation of the curve $\beta : [a, b] \to \mathbb{R}^3, \beta : t \mapsto (\beta_1(t), \beta_2(t), \beta_3(t))$, where $\beta_i(t), i = 1, 2, 3$, are the coordinate functions of the curve obtained by the piecewise projection of $\rho$ onto $\mathbb{R}^3$. 
Figure 6.12: Piecewise projection: (a) A trajectory in $\mathbb{R}^n$; (b) Its piecewise projection onto $\mathbb{R}^3$. 
The set of coefficients of the polynomial/spline in tandem with the set of parameters of all required orthogonal transformations forms a single weight that we assign to each homogeneous part of the graph. A skeletal graph equipped with such weights contains sufficient amount of information needed to reconstruct the original surface with desired precision. A weighted skeletal graph may, therefore, be used for the storage and classification of objects. In addition, the proposed construction is invariant to rigid motions and scaling.

6.4 Sampling and level curve pruning

Recall that while constructing a skeletal graph, the level curves of the distance function are modeled to provide the weights of corresponding graph edges. This raises a question about the evolution speed of intersecting spheres which define these samples. A fine sampling (large $K$) achieves a skeletonization that captures most critical points well. On the other hand, it yields too many level curves resulting in a Reeb graph that is too “weighty” to be of any practical significance. Accounting for redundancy of level curves and preserving a minimal significant number is, hence, of paramount importance. In this section, we present two criteria for optimal subsampling/pruning of finely sampled curves of a 3D object; one is correlation-based while the other is based on curvature.

Multiresolution tree

In order to implement such a subsampling algorithm, we find it easier to work with a multiresolution tree, where each node points to a pair of level curves. We are given a set of finely sampled curve vectors $\rho_1, \ldots, \rho_m \in \mathbb{R}^n$ corresponding to the level sets $r_1, \ldots, r_m$. The root node points to just the first and the last curves, i.e., $\rho_1$ and $\rho_m$, which correspond to the level sets $r_1$ and $r_m$ respectively. At level 2 of the tree, we divide the region between $r_1$ and $r_m$ into two to get another curve vector corresponding to the level set $r_{m/2}$, assuming $m$ is even, for convenience. Thus, the two nodes at level 2 correspond to $(\rho_1, \rho_{m/2})$ and $(\rho_{m/2}, \rho_m)$ respectively. The process is repeated down the tree until no further division may be carried out. Hence, as shown in Fig. 6.13, each tree level represents nodes corresponding to some level of resolution, and moving down the tree amounts to moving from coarser to finer resolution, with level 1 being coarsest and terminal level being finest.

As the tree is traversed upwards proceeding from leaves towards the root, two costs is computed at each node whose children are leaves in order to make a pruning decision.
In order to explain the idea, consider the subtree $T_s$ enclosed in a rectangle in Fig. 6.13. Clearly, both children of the root of $T_s$ are terminal nodes. We, therefore, compute the cost of the parent as $C(\rho_2, \rho_4)$, which is actually the cost of going from $\rho_2$ to $\rho_4$. On the other hand, the combined cost of the children is the cost of going from $\rho_2$ to $\rho_4$ via $\rho_3$, i.e., $\tilde{C}(\rho_2, \rho_4 | \rho_3) = C(\rho_2, \rho_3) + C(\rho_3, \rho_4)$. We then proceed to make a decision for redundancy of $\rho_3$. If $\tilde{C}(\rho_2, \rho_4 | \rho_3) > C(\rho_2, \rho_4)$, we merge the two children to the parent node. The process is repeated to prune the tree of any redundant leaves.

The cost should be such that there is a smooth variation among the samples which suggests a functional that depends on the curvature of elasticae and would in fact be the same as the bending energy of the curve evolution trajectory:

$$C_{ij} = C(\rho_i, \rho_j)$$
$$= \int K_{\alpha_{ij}}^2 ds,$$  
(6.15)

where $K_{\alpha_{ij}}$ is the curvature of the elastica $\alpha_{ij}$ fitted between $\rho_i$ and $\rho_j$. For example, in subtree $T_s$, there are three level curves $\rho_2, \rho_3, \rho_4$ on the leaves, with $\alpha_2, \alpha_3, \alpha_4$ as their respective projections on $\mathbb{R}^3$ and corresponding elastica $\alpha^{(c)}(t) = \tilde{\alpha}_{24|3}(t)$ fitted to the children. The notation indicates the learning between $\alpha_2$ and $\alpha_4$ knowing that $\alpha_3$ lies in between, as shown in Fig. 6.15(b). Recall, two additional points are needed at the two ends for tangent estimates. A terminal point is taken from the next node at the same level, which happens to be $\rho_5$. 

Figure 6.13: Multiresolution tree for curves $\rho_1, \ldots, \rho_8$. Note that resolution is finer at higher levels.
Figure 6.14: Curves on multiresolution tree after pruning.

The same is carried out for a parent of two leaves, i.e., between $\alpha_2$ and $\alpha_4$, to get $\alpha^{(p)}(t) = \alpha_{24}(t)$, as illustrated in Fig. 6.15(c). Clearly, if the points $\alpha_2$ and $\alpha_4$ are very dissimilar, the tangents at the two points will point in directions with a large angle disparity. This means that the cost $C_{24}$ will be larger than $\tilde{C}_{24|3}$. If $C_{24} \leq \tilde{C}_{24|3}$, we may conclude that the point $\alpha_3$ between $\alpha_2$ and $\alpha_4$ is redundant and does not add any new information. This results in merging the two child nodes into the parent. The procedure is recursively repeated for all leaves, until we finally converge to a stable tree, similar to that shown in Fig. 6.14. An illustration of the process is given in Fig. 6.15.

There is, however, a constraint that should be introduced. As depicted in Fig. 6.15(c), minimizing the cost may result in some error between the actual $\alpha_3$ and the estimated point, $\hat{\alpha}_3$, corresponding to $\rho_3$, which we would like to be upper bounded by $\varepsilon_B$. Defining the error between $\alpha^{(p)}(t)$ and $\alpha_k$ as:

$$
\varepsilon_j(t) = \| \alpha^{(p)}(t) - \alpha_j \|,
$$

and the optimal estimate, $\hat{\alpha}_j$, is that which minimizes the cost subject to the constraint, $\varepsilon_j(t) < \varepsilon_B$.

Since all considered curves lie along the same graph edge, they may be very highly correlated especially when finely sampled. This suggests that the degree of correlation be-
Figure 6.15: Illustration of multiresolution modeling of $\alpha_2, \alpha_3, \alpha_4$: (a) $\alpha_{23}(t)$ fitted at fine level for the first modeling interval; (b) $\alpha^{(c)}(t) = \alpha_{243}(t)$ for both intervals. Note that the basis is rotated by angle $\theta$ to align starting tangent $v_{2}^{(s)}$ of the second arc along the ending tangent $v_{1}^{(e)}$ of the first arc. This results in the rotation of $v_{2}^{(e)}$ by $\theta$; (c) $\alpha^{(p)}(t) = \alpha_{24}(t)$ fitted at coarse level. $\alpha_3$ is dropped and the resulting trajectory has a curvature less than that at fine level.
Figure 6.16: Extended Reeb graph for double torus along with level curves.

Figure 6.17: Extended Reeb graphs for double torus: (a) $K = 4$; (b) $K = 8$; (c) $K = 16$.

tween any consecutive pair of curves may be used to drop a sample, as this indicates that sufficient information about their evolutionary behavior is already available. A multiresolution tree ensures the retention of the initial and terminal curves, which are very important for interpolation.

Note that the first approach we described in this section is applied in the lower dimensional space after projecting curves from $\mathbb{R}^n$ to $\mathbb{R}^3$, while the latter is applied directly to the curve points in high dimensional space.

6.5 Experimental Results

In this section, we present some skeletonization and geometric modeling results.

6.5.1 Topological modeling

Skeletal graphs of 3D objects of various complexity are given in Figs. 6.16 through 6.20. Fig. 6.16(b) illustrates the skeletal representation of a double torus along with the level curves, and demonstrates that a graph represents topology, while curves capture geometry to provide a complete description of an object. Fig. 6.20 illustrates rotational-invariance of the proposed technique.

A graph with high resolution will result in a dense set of nodes along the edges as
Figure 6.18: Extended Reeb graphs: (a) An airplane; (b) A horse.

Figure 6.19: Extended Reeb graphs for a camel: (a) $K = 8$; (b) $K = 16$; (c) $K = 32$.

Figure 6.20: Rotation invariance of extended Reeb graph: (a) No rotation; (b) Rotation by $\pi/2$; (c) Rotation by $3\pi/4$. 
shown in Fig. 6.21(a), which also contains redundant geometric information. A simplified graph is illustrated in Fig. 6.21(b), where each color coded edge represents a topologically homogeneous portion of the corresponding object as in Fig. 6.21(c).

6.5.2 Geometric model

Suppose we want to model edge $e$ given in Figs. 6.21(b) and 6.22(a), where the surface is sampled with $K = 200$, resulting in 98 level curves along the edge, which are then processed for pruning by a multi-resolution tree. The optimal tree results in a minimal set of 13 curves, removing redundant curves, a reduction by a factor of 7.5. Elasticae model is then learned for each pair of curves resulting in a trajectory, shown in Fig. 6.23(b), that is smooth at joints. From the model, we sampled several curves which are then rendered in Fig. 6.22(b). The reconstructed surface closely resembles the original subsurface marked as edge $e$ in Fig. 6.22(a). Likewise, we reconstruct all other edges of the object, as illustrated in Fig. 6.22(c).

The evolution trajectories of curves along respective edges may be represented by Taylor series, the coefficients of which are eventually assigned as weights to the skeletal graph. Curve evolution trajectories for edges $a$, $e$ and $i$ and their respective Taylor polynomial representations are given in Figs. 6.24 and 6.25, which show that the coefficients uniquely represent an edge. These coefficients are assigned to the edges to get a weighted skeletal graph, which represents a unique topo-geometric model of the object.

Note that the percentage error bound and the correlation angle threshold are taken to be 1% and $\pi/400$ respectively. Further examples of reconstructed objects are given in
Figure 6.22: Surface modeling for double torus: (a) Level curves of edge \( e \); (b) Reconstructed subsurface \( e \) from the model; (c) Level curve set; (d) Reconstructed double torus.

Figure 6.23: Trajectory for edge \( e \) in \( \mathbb{R}^3 \): (a) Without basis rotation; (b) Smooth Frenet frame along the curve.

Figure 6.24: Polynomial representation of the trajectory in parametric form in \( \mathbb{R}^3 \): (a) Edge \( a \); (b) Edge \( e \); (c) Edge \( i \).
Figure 6.25: Taylor polynomial coefficients for the three surface trajectories: (a) $x$-coefficients; (b) $y$-coefficients; (c) $z$-coefficients.
6.6 Conclusions

In this chapter, we presented a 3D object modeling scheme that captures the topological and geometric information of an object. Topology and rough geometry are captured through an extension of the Reeb graph by employing a distance function for object analysis. The formulation for this extended Reeb graphs, preferably referred to as topo-geometric graphs, has also been shown to be rotation, translation and scale invariant.

To preserve and encode the geometric information, we proposed elasticæ to model the evolution of level curves along various edges along the graph resulting in a curve evolution trajectory in $\mathbb{R}^3$. This trajectory is subsequently represented by its Taylor polynomial, whose coefficients yield edge weights. The weighted skeletal graph may potentially be used as templates with important applications in archival and classification of 3D objects.
Chapter 7

Shape Recognition by Graph Edit Operations

In this chapter, we use the graphical models constructed in Chapters 5 and 6 for shape recognition. The method that we employ here is based on graph matching [57, 58], where two shapes are compared on the basis of establishing an isomorphism between their respective graphs. As mentioned in Chapter 1, measurement error, noise, occlusion and pose result in variations in shape realizations. An exact graph isomorphism is, therefore, not achievable, and one may proceed to deform a graph so that it becomes isomorphic to the other graph. The amount of deformation, as a result, quantifies the dissimilarity between two shapes. This dissimilarity is subsequently employed to find the best match of a shape among several models.

7.1 Graph Deformation

A recognition system should be able to recognize real world objects, given prior knowledge about objects of interest (e.g., in the form of models, which are graphs in our case), typically stored in a database. Although representative models may be “perfect” representation of corresponding objects, distortions due to noise, occlusion, etc., contribute to imperfect representation of a test object and, hence, to the complexity resulting from an inexact match between the test object and the models.

A solution may be to quantify the dissimilarity between a test shape and the models. This dissimilarity may be defined as a subgraph edit distance, which in essence
compensates for these distortions in a test graph. It entails deforming the model graphs through a sequence of edit operations, until a subgraph isomorphism is established between deformed graphs and a test graph. The edit operations are based on the physical changes that may occur due to aforementioned distortions. This includes insertion or deletion of missing or extraneous vertices or edges, and distortion of vertex or edge labels. Each edit operation, however, has an associated cost, contributing to the total cost of a deformation. The subgraph distance between a model graph and a test graph, therefore, corresponds to the sequence of edit operations that results in minimum total cost.

Definition 7.1: (Edit Operation) Given a graph $\mathcal{G} = (V, E, \mu, \nu)$, a graph edit operation $\delta$ on $\mathcal{G}$ is any of the following:

- **Correction of vertex label distortions**: Substituting the label $\mu(v)$ of a vertex $v$ by $l$:
  \[ \mu(v) \rightarrow l, v \in V, l \in L_V \]

- **Correction of edge label distortions**: Substituting the label $\nu(e)$ of an edge $e$ by $l'$:
  \[ \nu(e) \rightarrow l', e \in E, l' \in L_E \]

- **Correction of missing vertices**: Deleting the vertex $v$ from $\mathcal{G}$:
  \[ v \rightarrow \emptyset, v \in V \]

- **Correction of missing edges**: Deleting the edge $e$ from $\mathcal{G}$:
  \[ e \rightarrow \emptyset, e \in E \]

- **Correction of extraneous edges**: Inserting an edge between vertices $v_1$ and $v_2$ of $\mathcal{G}$:
  \[ \emptyset \rightarrow e = (v_1, v_2), v_1, v_2 \in V \]

Each of the edit operations $\delta$ described above, therefore, deforms a graph $\mathcal{G}$ to $\delta(\mathcal{G})$:

Definition 7.2: (Edited Graph) Given a graph $\mathcal{G} = (V, E, \mu, \nu)$ and an edit operation $\delta$, the edited graph, $\delta(\mathcal{G})$, is a graph $\delta(\mathcal{G}) = (V_\delta, E_\delta, \mu_\delta, \nu_\delta)$, where:

1. $V_\delta = \begin{cases} V - \{v\} & \text{if } \delta = (v \rightarrow \emptyset) \\ V & \text{otherwise} \end{cases}$
2. \( E_\delta = \begin{cases} 
E \cup \{e\} & \text{if } \delta = (\otimes \rightarrow e) \\
E \setminus \{e\} & \text{if } \delta = (e \rightarrow \otimes) \\
E \cap (V_\delta \times V_\delta) & \text{otherwise}
\end{cases} \)

3. \( \mu_\delta(v) = \begin{cases} 
l & \text{if } \delta = (\mu(v) \rightarrow l) \\
\mu(v) & \text{otherwise}
\end{cases} \)

4. \( \nu_\delta(e) = \begin{cases} 
l' & \text{if } \delta = (\nu(e) \rightarrow l') \\
\nu(e) & \text{otherwise}
\end{cases} \)

A graph must undergo a sequence of edit operations to account for all distortions that may occur.

**Definition 7.3:** (Deformation) Given a graph \( G = (V, E, \mu, \nu) \) and a sequence of edit operations \( \Delta = (\delta_1, \ldots, \delta_k) \), the edited graph, \( \Delta(G) \), is a graph \( \Delta(G) = \delta_k(\ldots \delta_2(\delta_1(G)) \ldots) \).

The total cost \( C(\Delta) \) of deforming a graph \( G \) into \( \Delta(G) \) equals the sum of individual costs of each edit operation:

\[
C(\Delta) = \sum_{i=1}^{k} C(\delta_i). \tag{7.1}
\]

**Definition 7.4:** (Error Correcting Subgraph Isomorphism) Given two graphs \( \mathcal{G} \) and \( \mathcal{G}' \), an error-correcting (EC) subgraph isomorphism \( f \) from \( \mathcal{G} \) to \( \mathcal{G}' \) is a 2-tuple \( f = (\Delta, f_\Delta) \), where \( \Delta \) is a sequence of edit operations such that there exists a subgraph isomorphism \( f_\Delta \) from \( \Delta(G) \) to \( \mathcal{G}' \).

The sequence of edit operations leading to error-correcting subgraph isomorphism is not unique. We, therefore, seek the one that results in a minimal cost.

**Definition 7.5:** (Subgraph Distance) Given two graphs \( \mathcal{G} \) and \( \mathcal{G}' \), the subgraph distance from \( \mathcal{G} \) to \( \mathcal{G}' \), \( d(\mathcal{G}, \mathcal{G}') \), is the minimum cost taken over all error-correcting subgraph isomorphisms \( f \) from \( \mathcal{G} \) to \( \mathcal{G}' \):

\[
d(\mathcal{G}, \mathcal{G}') = \min_{\Delta} \{C(\Delta) : \exists f = (\Delta, f_\Delta)\}, \tag{7.2}
\]

where \( f \) is an EC isomorphism from \( \mathcal{G} \) to \( \mathcal{G}' \). \( f \) that corresponds to the minimum distance is known as optimal error-correcting (OEC) subgraph isomorphism.

In other words, the 2D and 3D shape models given in Chapters 5 and 6 respectively, reduce the problem of shape comparison to that of graph matching.
7.2 Cost of Edit Operations

In this section, we discuss how cost may be assigned to various edit operations. Edit operations that we are particularly interested in relate to insertion and deletion of graph edges. Although this involves inserting or deleting graph vertices, our emphasis on edges is based on the fact that geometric information is encoded along various graph edges. We consider two approaches for assigning cost to an edit operation.

7.2.1 Constant Cost

One approach is to assign a constant cost $C = C(\delta_i)$ to each edit operation $\delta_i$. The problem is then to find the subgraph isomorphism from a model graph $G_k$ to a test graph $G_t$, that minimizes the total cost defined by Eq. (7.1). Bunke et. al. [11] define the distance between $G_k$ and $G_t$ in terms of the maximal subgraph $\mathcal{S}(G_k; G_t)$ of $G_k$ that is isomorphic to $G_t$:

$$d(G_k, G_t) = 1 - \frac{|\mathcal{S}(G_k; G_t)|}{\max(|G_k|, |G_t|)},$$

(7.3)

where $|.|$ is the number of nodes in the corresponding graphs.

7.2.2 Variable Cost

The problem with the above approach is that it depends on the size of the subgraph and does not account for other graph attributes. Instead of assigning a uniform cost to all edit operations, we derive it from physical characteristics of a surface, which include topological as well as geometric features of the surface.

Component Cost

As described in Section 6.2, the graph construction algorithm results in graph edges, each actually consisting of several edge segments. The number of edge segments corresponding to an edge, therefore, reflects its relative importance. For instance, an edge consisting of just one edge segment may not have significant information about a shape and may be a result of surface distortion due to noise. On the other hand, an edge composed of several edge segments is more interesting, since it represents a large part of a surface.
that is topologically homogeneous. We, therefore, define component cost $C_c$ in terms of the number of edge segments $N$ in an edge:

$$C_c(\delta) = \frac{N}{M},$$

(7.4)

where $M$ is the total number of edge segments in the graph.

**Proximity Cost**

Heuristically, an edit operation involving vertices which lie in close physical proximity should have a small associated cost than that of vertices which are farther. The cost of an edit operation $\delta$ for matching a vertex $v_i$ of a model graph $G_k$ to a vertex $v_j$ of the test graph $G_t$ is, therefore, defined as:

$$C_p(\delta) = d(v_i, v_j),$$

(7.5)

where $d(v_i, v_j)$ is the Euclidean distance between the physical location of $v_i$ and $v_j$.

The proximity cost for insertion or deletion of an edge, hence, equals the cost of adding or removing the corresponding vertices $v_i$ to or from the edge. In other words, deletion or insertion of an edge between two distant vertices is penalized more, which intuitively corresponds to a potentially large surface variation for that edge.

Another factor that should be incorporated in proximity cost is the order of connectivity of various vertices to account for their relative importance. Obviously, a high degree vertex is more important than a small degree vertex, suggesting that an edit operation involving the former has a higher associated cost. This, therefore, requires the introduction of a weighting factor in the proximity cost as given below:

$$C_p(\delta) = \max\{D_i, D_j\} \cdot d(v_i, v_j),$$

(7.6)

where $D_i$, and $D_j$ are the degrees of vertices $v_i$ and $v_j$ respectively.

**Geometric Cost**

A third component to the cost arises from geometric attributes of edges. Since geometric properties are encoded as edge attribute vectors, we formulate our cost on the basis of dissimilarity of attribute vectors. Given a model graph $G_k$ and a test graph $G_t$, the geometric cost of an edit operation leading to the insertion or deletion of $e_i$ to or from $G_k$, 


to achieve an error-correcting isomorphism from $G_k$ to $G_t$ matching $e_i$ to $e_j$ of $G_t$ is defined as:

$$C_g(\delta) = d(w_i, w_j),$$

(7.7)

where $w_i$ and $w_j$ are geometric attribute vectors corresponding to $e_i$ and $e_j$.

**Total Cost**

Total cost $C$ of an edit operation $\delta$ is defined as the weighted sum of the above three costs, $C(\delta) = \alpha_c C_c(\delta) + \alpha_p C_p(\delta) + \alpha_g C_g(\delta)$, where $\alpha_c$, $\alpha_p$ and $\alpha_g$ are the corresponding weights. Note that the component cost takes into account topological information, while geometric cost is related to geometric information. The proximity cost, on the other hand, is a hybrid of both topological and geometric information. In case, all vertices and edges are considered to be equally important, and have uniform attribute vectors, then the variable cost reduces to the constant cost discussed above.

### 7.3 Experimental Results

In this section, we present extensive experimental results not only for model construction but also for shape recognition, using edit operations on skeletal models.

#### 7.3.1 Model Learning

In our experiments, we consider shapes from various classes such as human, animal, pottery, airplane, plant, fish, etc., some of which are shown in Figs. 7.1 through 7.10. Mesh representations of 3D shapes were downloaded from [87, 88, 89, 90, 91]. First, their skeletal representations are constructed as shown in Figs. 7.1 through 7.10, which demonstrates topology capturing capabilities of the method for a wide variety of shapes. Another observation follows from Figs. 7.8(d) and 7.10(b), where the algorithm succeeded in constructing individual graphs when multiple objects were embedded in single images. This property is highly desirable for shape and part segmentation.

#### 7.3.2 Shape Recognition

Instead of using the variable cost, we employ a simplified matching criterion, which leads to a hierarchical model search.
Figure 7.1: Skeletal representation of 3D shapes.
Figure 7.2: Skeletal representation of 3D shapes.
Figure 7.3: Skeletal representation of 3D shapes.
Figure 7.4: Skeletal representation of 3D shapes.
Figure 7.5: Skeletal representation of 3D shapes.
Figure 7.6: Skeletal representation of 3D shapes.
Figure 7.7: Skeletal representation of 3D shapes.
Figure 7.8: Skeletal representation of planar shapes.
Figure 7.9: Skeletal representation of planar shapes.
Figure 7.10: Skeletal representation of planar shapes.
Hierarchical Model Search

In the following experiments, we propose a strategy for graph matching that is computationally efficient. The idea is to exploit the simplicity of the constant cost together with important features of the variable cost, without enhancing complexity of the problem. This leads to the following hierarchical approach:

1. The first step is to simplify a graph by eliminating two types of edges, (a) those are composed of one edge segment even if the length of the edge is “too” large and (b) those with degree one vertices, and whose removal does not produce additional degree one vertices. This idea effectively accounts for low cost associated with the two types. The former compensates for component cost, while the latter compensates for proximity cost, by removing spurious edges possibly arising from shape distortions.

2. The second step involves carrying out a first level search in the database for the best match on the basis of genus of a test shape. This immediately prunes the set of shapes to those with topological type similar to that of the test shape.

3. The third step is related to the second level search, where the optimal isomorphisms from the model graphs to a test graph is found, assuming constant cost of an edit operation without considering edge attributes. This consequently gives a model which is the closest match to the test graph in the graph distance sense.

This strategy assumes zero geometric cost, corresponding to $\gamma = 0$ in Eq. (7.1), and considering all vertices of degree two and above to be equally important. In addition, Step 1 is based on infinite cost for edit operations that yield topological changes in a graph. In some cases, however, one may want to allow topological changes at a relatively high but finite penalty. In such a case, another term should be included in the total cost that takes topological changes into account.

Skeletal graphs shown in Figs. 7.1 through 7.10 are then simplified according to Step 1, which are finally stored in a database. This also eliminates artifacts due to manifold boundaries and noise.

For recognition, input (test) shapes are first represented by their simplified skeletal representation, before carrying out the first level hierarchical search, where all shapes in the database with same genus are found. In order to determine the genus, we adopt a
Figure 7.11: First level search for test shape (a) resulted in two candidate shapes (b) and (c) with identical genus.

depth-first-search based approach [16]. This in some cases greatly reduces the number of models that would eventually be considered as candidates for graph matching. For instance, when searching a match for double torus, this strategy resulted in only two candidates as shown in Fig. 7.11. Since graph matching procedure is quite expensive, the above scheme, therefore, considerably speeded up search process.

Eventually, recognition is carried out according to Step 3, with results presented in Fig. 7.12. Some of the best matches are provided in the figure, where given shapes are correctly recognized. Results also demonstrate that if a “nice” 2D view of a 3D object is given, it can be matched to the corresponding 3D object. This property makes the method very powerful, especially in applications where one may want to find canonical views of a 3D shape.

7.4 Conclusions

In this section, we presented an application of models developed in Chapters 5 and 6, to shape recognition. The ideas for shape recognition via skeletal representations were borrowed from the rich field of graph theory, where we employed error-correcting graph matching. We, however, fell short of utilizing comprehensive skeletal models for recognition application, in spite of deriving a strategy that employs a variable cost function. This allowed us to achieve high recognition rates for the first match. It should be noted that for some shapes, it resulted in more than one match with identical cost, where at least one of them was an incorrect recognition. This is due to the fact that in those cases there is nothing discriminating in the corresponding shapes descriptors, therefore suggesting that
Figure 7.12: Shape recognition: (a) Test shape; (b) Best match; (c)-(e) Other good matches.
geometric properties are indeed necessary for correct and unique recognition. Since the geometric cost was derived from unique surface characteristics, it strongly suggests that it will potentially yield high recognition rates for a wide class of shapes.
Chapter 8

Conclusions

In this thesis, we addressed a fundamental problem in machine vision, i.e., the modeling and analysis of 2D and 3D shapes, in statistical as well as deterministic settings. The models formulated in this thesis have a great potential for a wide variety of applications in computer vision and image/shape analysis. With the help of models derived in this thesis, we specifically addressed the following problems:

- Classification of shapes
- Sampling new shapes
- Template learning
- Shape representation and reconstruction

In this chapter, we will summarize the contributions of the thesis and will outline future extensions of this work to further this fundamental research area.

8.1 Contributions of The Thesis

8.1.1 Flexible Skew Symmetric Shape Model

Based on a class of distributions, we presented a data mining approach to shape modeling in Chapter 4, known as FSSM. This probabilistic approach captures the variability of shapes in the presence of skewness using shape distributions. Currently, very few approaches for shape modeling address the problem of skewness. We employed this
method for template learning, shape classification, and sampling new shapes. The model has been experimentally shown to be robust to shear and occlusion. Our method differs from landmark based approaches, which employ support vector machines (SVM) or some other classifier to classify shapes on the basis of landmark data. First, we allow automatic landmark selection via angle distributions, hence making our approach more probabilistic than existing methods. Second, it is pointed out that an SVM classifier is suboptimal albeit it yields high classification results. Our approach is motivated by the fact that a shape distribution combined with a Bayesian classifier results in optimal classification rates for a particular class of shapes.

8.1.2 2D Morse Theoretic Model

The majority of 2D shape modeling techniques operate on a smaller subset of simple closed shapes. In Chapter 5, we presented a novel method for modeling 2D shapes of arbitrary topology. To the best of our knowledge, it is the first time 2D shape modeling has been addressed in a Morse theoretic framework. The power of this method is that it encodes both the topological and the geometric properties in a single skeletal representation, which was shown to be a complete and compact shape representation. Completeness and uniqueness properties of the model were demonstrated by reconstructing shapes from models. Although this method bears some similarities with shock graph based methods, it operates on a wider class of shapes with much less computational complexity, since it does not employ any grammar rules.

8.1.3 Topo-Geometric Shape Model

In Chapter 6, we addressed the problem of modeling 3D shapes by capturing their topological and geometric features. The resulting skeletal representation captures critical points of the distance function on surfaces representing 3D solids, to account for topological information. The choice of distance function stems from its invariance to rigid transformations. The advantage of this approach over Medial Axis Transform (MAT) is three fold. First, MAT representation is more complex, since it consists not only of lines but also surface elements. Second, it is computationally more intensive ($O(n^2 \log n)$ [76]), compared to our method ($O((n+k) \log(n+k))$), where $n$ is the number of vertices in the corresponding mesh representation and $k$ is the number of vertices inserted in the mesh while intersecting
a surface with a sphere. Third, small perturbations on a surface leads to spurious edges in MAT, whereas our method filters out some of those small effects with a proper choice of graph resolution parameter. This adverse feature of MAT limits their utility for recognition applications. Similarly, compared with our approach, geodesic distance based multiresolution graph (MRG) [33] also suffers from high computational complexity (\(O(n^2 \log n)\)), in addition to its inability to glean geometric information as topological information in unfolded.

In addition to topology, geometric encoding in the form of weights on the graph results in a complete shape representation, and, therefore, may be used for shape reconstruction. To gain further insight into topo-geometric model and its applications in visualization, we presented simulation results for shape reconstruction.

### 8.1.4 Classification of Shapes from Skeletal Representations

In Chapter 7, we employed models developed in Chapters 5 and 6 for classification via edit operations on graphs. We discussed two types of cost functions for graph matching, and presented a strategy for hierarchical shape classification, that fuses these two costs. Several experiments were provided demonstrating much improved performance of the proposed methodology in 2D and 3D shape matching.

### 8.2 Future Research

Several interesting research directions have emerged in this thesis in addition to possible future extensions.

#### 8.2.1 Extended Flexible Skew-symmetric Shape Model

As mentioned in Section 4.2, \(FSSM\) models a particular class of simple closed shapes. The idea may be extended to a larger class of shapes, by considering a different shape parametrization and subsequently establishing a correspondence among shape realizations. A parameterization, for instance, based on turning angle will also achieve rotation invariance, which in turn will be an important feature of the method. In addition, this method has strong potential for applications such as image segmentation, which need to be investigated.
8.2.2 Shape Recognition through Topo-Geometric Model

In Section 7.2.2, we presented a variable cost that accounts for both topological as well as geometric features of a surface. The classification strategy discussed in Section 7.3, however, does not fully employ the topo-geometric shape model. We, therefore, submit that using the variable cost for shape recognition applications may be a good alternative.

8.2.3 Extension to Topo-Geometric Model

Although topo-geometric shape model is rigid transformation invariant, it does not account for other non-affine transformations arising from the movement of object parts. To achieve invariance to such a group of transformations, a geodesic distance based graph similar to [33] may be considered along with iso-geodesic curves to account for geometric encoding. Further work for the refinement of the method is to model texture information.

In addition, other important related problems are extending topo-geometric model to capture time varying objects and volumetric data. Other possible research avenues of this framework are face recognition, image and shape registration, and tracking.
Bibliography


[86] www.lems.brown.edu/vision/software/index.html

[87] www.3dcafe.com

[88] www.fantasticarts.com

[89] www.graphics.stanford.edu/data/3Dscanrep

[90] www.sculptureweb.org

[91] www.shape.cs.princeton.edu/benchmark


