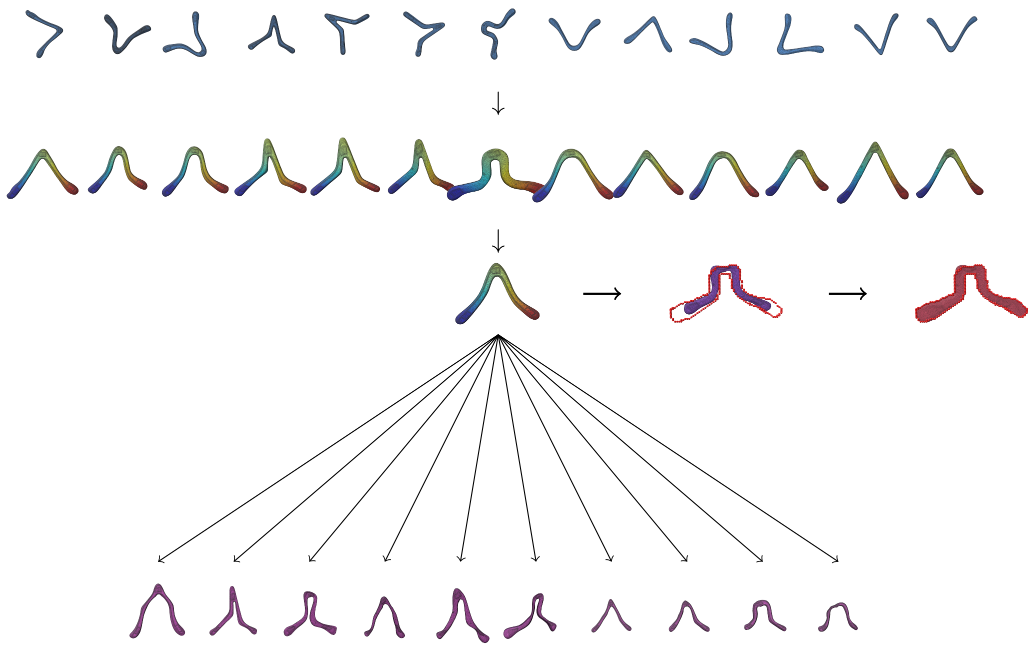
Shape Correspondences and Example-Based Modelling for Boomerang Design

A Framework for Alignment, Parameterization, Modelling and Analysis of Aerodynamic Boomerang Shapes

Computer Science, Master Thesis

Nils van Veen



Shape Correspondences and Example-Based Modelling for Boomerang Design

A Framework for Alignment, Parameterization, Modelling and Analysis of Aerodynamic Boomerang Shapes

by

Nils van Veen

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Friday May 16, 2025 at 10:00 AM.

Student Number:	4917863
Main Supervisor:	K. Hildebrandt
Second Supervisor:	R. Marroquim
Third committee member:	X. Zhang
Project Duration:	September, 2024 - May, 2025
Faculty:	Faculty of Electrical Engineering, Mathematics & Computer Science, Delft

Preface

Boomerangs have previously been studied from a physics perspective, yet a fundamental understanding of their shape variations remains elusive. What makes one shape superior to another? How do subtle geometric differences influence performance?

This thesis addresses the correspondence problem for boomerang-like shapes, aiming to develop a tool that enables users to analyze and compare different designs based on shape correspondences. This work allows users to explore and refine boomerang shapes by providing a structured approach to understanding these relationships.

The journey of completing this thesis has been both challenging and rewarding. I am deeply grateful for the support and guidance I received along the way.

First and foremost, I would like to express my sincere gratitude to Klaus Hildebrandt and Ricardo Marroquim, who supervised me throughout this project. Klaus's expertise in geometry processing and Ricardo's deep knowledge of computer graphics and 3D scanning have been invaluable. Their support began even before the official start of the thesis, as this project was self-initiated, and I greatly appreciate their openness to the idea from the start.

I would also like to thank the NewMedia Centre at the TU Delft, particularly Sharif Bayoumy, for his assistance with the initial setup for 3D scanning. His guidance helped lay the groundwork for the practical aspects of this research. I am also grateful to Roland van Roijen for his flexibility in allowing me to use the equipment.

I extend my gratitude to a few individuals from the boomerang community: Manuel Schütz, for our insightful discussions on the parameterization of boomerangs. His expertise has been instrumental in refining my approach. Bill Hirst, for his scientific knowledge and for challenging me with thought-provoking questions that shaped the direction of this thesis.

Finally, I would also like to thank my girlfriend, Brannie Haigh, for her unwavering support, patience, and encouragement throughout this journey. Sharing intermediate results with her, discussing ideas, and working alongside each other—me on my thesis and her on her dissertation—helped me stay grounded and focused. Her presence made the challenges more manageable and the progress more rewarding.

This work would not have been possible without the encouragement and expertise of those around me, and I am incredibly appreciative of their contributions.

Nils van Veen
Delft, May 2025

Abstract

This thesis presents a computational framework aimed at enabling the analysis and modeling of boomerangs from example shapes. The goal is to provide a systematic and data-driven tool for boomerang design based on real-world geometries. A key challenge in this context is establishing accurate shape correspondences between handcrafted, asymmetric, and aerodynamically functional boomerangs.

To address this, the proposed pipeline integrates multiple components: landmark-based pre-alignment, boundary extraction using alpha shapes, curve parameterization, and Least Squares Conformal Mapping (LSCM) to compute surface correspondences. Building on these correspondences, the framework further incorporates principal component analysis (PCA) and Free-Form Deformation (FFD) to enable the generation of new shapes.

Experimental results show that the method achieves low Hausdorff and Chamfer distances and has been evaluated using area- and shear-based distortion metrics. Nonetheless, some localized inaccuracies - particularly near high-curvature regions - highlight areas for improvement in boundary handling and local control. Additional limitations include the reliance on manual landmark selection, the linearity of PCA, and the sensitivity of the pipeline to alpha shape parameters.

By combining elements of computational geometry with principles of aerodynamic design, this research bridges the gap between empirical craftsmanship and formal shape analysis. The resulting methodology not only enables the comparison and modeling of boomerangs but also lays the groundwork for future tools in boomerang shape exploration and design.

Contents

Nomenclature

Abbreviations

Abbreviation	Definition
BFF	Bounded Free-Flattening
CAD	Computer-Aided Design
CGAL	Computational Geometry Algorithms Library
Delaunay	Delaunay Triangulation
FFD	Free-Form Deformation
ICP	Iterative Closest Point
KNN	k-Nearest Neighbors
LSCM	Least Squares Conformal Mapping
PCA	Principal Component Analysis
SVD	Singular Value Decomposition
UV	Texture Mapping Coordinate System
VXelements	Creaform's 3D Scanning Software

Symbols

Symbol	Definition	Unit
	Background - Alpha Shapes	
P	Finite set of points in $\mathbb{R}^n$	-
$\mathcal{T}$	Delaunay triangulation of P	-
α	Alpha shape parameter	-
$\mathcal{K}_\alpha$	Alpha complex	-
S_α	Alpha shape	-
$R(\sigma)$	Radius of the smallest circumscribing sphere of simplex σ	-
σ	Simplex (vertex, edge, or face) in the triangulation	-
	Background - Principal Component Analysis (PCA)	
n	Number of vertices per shape	-
m	Number of shapes in the dataset	-
$\mathbf{s}_i$	Flattened vector representation of shape i	$\mathbb{R}^{3n}$
$\mathbf{S}$	Data matrix of all shapes	$\mathbb{R}^{3n\times m}$
$\bar{\mathbf{s}}$	Mean shape	$\mathbb{R}^{3n}$
$\mathbf{S}_c$	Centered data matrix	$\mathbb{R}^{3n\times m}$
$\mathbf{U}$	Principal component matrix	$\mathbb{R}^{3n\times m}$
$\mathbf{\Sigma}$	Diagonal matrix of singular values	$\mathbb{R}^{m\times m}$
$\mathbf{V}$	Right singular vector matrix	$\mathbb{R}^{m\times m}$
$\mathbf{u}_j$	j-th principal component (column of $\mathbf{U}$)	$\mathbb{R}^{3n}$
w_j	Weight for the j-th principal mode	-
k	Number of principal components used	-
$\mathbf{s}'$	Deformed shape vector	$\mathbb{R}^{3n}$
	Background - Iterative Closest Point (ICP)	
P	Source point cloud: $P = \{p_1, p_2, ..., p_n\}$	-
Q	Target point cloud: $Q = \{q_1, q_2, ..., q_m\}$	-
R	Rotation matrix in the transformation	$\mathbb{R}^{3\times 3}$

Symbol	Definition	Unit
t	Translation vector in the transformation	$\mathbb{R}^3$
q_i^*	Closest point in Q to transformed p_i	$\mathbb{R}^3$
$E(R,t)$	Sum of squared errors between corresponding points	-
$\bar{p}$	Centroid of P	$\mathbb{R}^3$
$\bar{q}$	Centroid of Q (closest points)	$\mathbb{R}^3$
H	Covariance matrix for alignment	$\mathbb{R}^{3\times 3}$
U, Σ, V	Matrices from SVD decomposition of H	-
	Method - Plane Fitting and Alignment	
$\mathbf{C}$	Centroid of a shape	$\mathbb{R}^3$
$\mathbf{n}$	Normal vector of the fitted plane	$\mathbb{R}^3$
d	Offset of the plane equation	$\mathbb{R}$
M	Covariance matrix of the shape	$\mathbb{R}^{3\times 3}$
	Method - 2D Projection and Alpha Shape Extraction	
$\mathbf{v}_i'$	Projected vertex onto a fitted plane	$\mathbb{R}^2$
α	Alpha parameter for computing the alpha shape	$\mathbb{R}$
B_α	Alpha shape boundary of a projected object	$\mathbb{R}^{M\times 2}$
	Method - Boundary Sorting and Correspondence	
$\text{SBV}_{\text{Shape 1}}$	Sorted boundary vertices of Shape 1	$\mathbb{R}^{n\times 2}$
$\text{SBV}_{\text{Shape 2}}$	Sorted boundary vertices of Shape 2	$\mathbb{R}^{m\times 2}$
$d(v_i, v_j)$	Euclidean distance between two vertices	$\mathbb{R}$
i^*	Index of the closest boundary point match	$\mathbb{N}$
	Method - Boundary Alignment	
$B_{\text{Mesh 1}}$	Initial boundary vertices of Shape 1	-
$B'_{\text{Mesh 1}}$	Improved boundary vertices of Shape 1 (connected)	-
$B_{\text{Mesh 2}}$	Initial boundary vertices of Shape 2	-
$B'_{\text{Mesh 2}}$	Improved boundary vertices of Shape 2 (connected)	-
$B''_{\text{Mesh 2}}$	Newly added boundary vertices in Shape 2	-
$B''_{\text{Mesh 1}}$	Corresponding new boundary vertices in Shape 1	-
	Method - Unit Parameterization	
L	Set of user-defined landmarks along the boundary	-
u_i	Unit parameterization coordinate of boundary vertex i in Shape 1	-
u_j'	Unit parameterization coordinate of boundary vertex j in Shape 2	-
dp_i	Relative unit distance along the boundary curve	-
d_i	Corresponding unit parameterization distance in Shape 1	-
	Method - Projection and 3D Lifting	
$\mathbf{C}$	Centroid of a shape's vertices	$\mathbb{R}^3$
$\mathbf{n}$	Normal vector of the fitted plane	$\mathbb{R}^3$
$\bar{z}$	Average z-coordinate of the shape	$\mathbb{R}$
$\mathbf{P}_i$	Lifted 3D point from the 2D boundary curve	$\mathbb{R}^3$
$\mathbf{P}_i'$	Projected 3D point onto the mesh surface	$\mathbb{R}^3$
$\mathbf{P}_j$	Closest vertex in the mesh to $\mathbf{P}_i'$	$\mathbb{R}^3$
	Method - UV Mapping	
UV_1	UV map of Shape 1	$\mathbb{R}^{2N}$
UV_2	UV map of Shape 2	$\mathbb{R}^{2M}$
LSCM(D, E, F)	LSCM parameterization function, mapping Shape D using boundary indices E and vertices F	-
	Method - Shape Correspondence	

Symbol	Definition	Unit
$(u, v)_i$	UV coordinates of vertex v_i in Shape 1	$\mathbb{R}^2$
$\mathbf{v}'_i$	Corresponding 3D position in Shape 2 for vertex v_i in Shape 1	$\mathbb{R}^3$
λ_j	Barycentric weight of vertex j in the UV triangle of Shape 2	-
	Method - PCA Shape Generation	
$\mathbf{g}_{\text{mean}}$	Mean shape in 3D space	$\mathbb{R}^{3N}$
$\mathbf{p}_{\text{mean}}$	Mean shape in 2D (projected)	$\mathbb{R}^{2N}$
$\mathbf{G}$	Matrix of PCA eigenvectors in 3D	$\mathbb{R}^{3N \times k}$
$\mathbf{P}$	Matrix of PCA eigenvectors in 2D	$\mathbb{R}^{2N \times k}$
$\mathbf{s}$	User-provided outline points (flattened)	$\mathbb{R}^{2M}$
i^*	Index of the closest mean shape point for each outline point	-
$\mathbf{w}$	Weight vector controlling shape reconstruction	$\mathbb{R}^k$
$\mathbf{J}$	Jacobian matrix used in optimization	$\mathbb{R}^{2M \times k}$
$\mathbf{H}$	Hessian matrix used in optimization	$\mathbb{R}^{k \times k}$
$\mathbf{r}$	Residual vector (difference between outline and shape)	$\mathbb{R}^{2M}$
$\mathbf{p}_{\text{reconstructed}}$	Reconstructed 2D shape	$\mathbb{R}^{2N}$
$\mathbf{s}'$	Reconstructed 3D shape	$\mathbb{R}^{3N}$
λ	Regularization parameter for smoothing	$[0, 1]$
$\mathbf{s}''$	Reconstructed 3D shape With regularization	$\mathbb{R}^{3N}$
	Method - Iterative Closest Point (ICP)	
$\mathbf{V}$	Set of vertices in the shape being deformed	$\mathbb{R}^{N \times 2}$
$\mathbf{S}$	Set of selected points for matching	$\mathbb{R}^{M \times 2}$
$x_{\min}, x_{\max}$	Bounding box limits in the x-direction	$\mathbb{R}$
$y_{\min}, y_{\max}$	Bounding box limits in the y-direction	$\mathbb{R}$
	Method - Free-Form Deformation (FFD)	
s_x, s_y	Normalized grid coordinates of a vertex	$\mathbb{R}$
N_x, N_y	Number of control points along x and y axes	$\mathbb{N}$
u, v	Local coordinates within the FFD grid cell	$\mathbb{R}$
$w_{00}, w_{10}, w_{01}, w_{11}$	Bilinear interpolation weights	$\mathbb{R}$
$\Delta v_{i,0}, \Delta v_{i,1}$	Deformation displacements in x and y	$\mathbb{R}$
$\mathbf{v}'_i$	Deformed vertex position after FFD	$\mathbb{R}^2$

1 Introduction

Boomerangs may appear simple and planar at first glance, but their aerodynamic performance depends on a combination of finely tuned features: wing twists, elbow angles, local airfoil variations, and subtle thickness changes. These features are often shaped by hand, guided by intuition and experience rather than parametric models or formal design tools. This makes them part of a broader class of "structured but irregular" geometries-objects that are largely planar but contain essential three-dimensional variation, or what might be called (mostly) 2.5D shapes.

Despite the craftsmanship involved in their design, there is currently no computational tool that allows throwers, designers, or researchers to systematically compare boomerang shapes, analyze their aerodynamic structure, or explore design variations based on existing examples. This raises several important questions: How can we systematically define and compare the geometry of handcrafted aerodynamic shapes? What makes one boomerang fly better than another? How can small changes in outline or curvature be traced and understood in a meaningful way? And more importantly, can this understanding be used to model new shapes based not on trial-and-error, but on structured, data-driven analysis?

As a competitive thrower, I've experienced how minor design changes - often made by intuition can lead to dramatic differences in flight performance. Yet the lack of formal tools for comparison or modeling based on real-world boomerangs leaves these insights largely unquantified.

To develop such a tool, a key computational challenge emerges: shape correspondences. Accurately matching points and features across a set of handcrafted boomerangs is essential for any data-driven analysis or modeling. However, this is not a trivial task. Most existing correspondence methods are optimized for controlled settings, assuming dense datasets, consistent sampling, or global shape similarity. These methods perform well on benchmarks, but they fail on objects like boomerangs.

Boomerangs highlight the limitations of current methods. While no two are exactly alike, most share a recognizable structural layout: a central elbow connecting two wings and a continuous boundary curve. This shared but flexible structure offers a pathway to develop correspondences, even in the face of high global variation.

This thesis investigates whether it is possible to define a shape correspondence pipeline for this specific class of shapes-structured, largely planar, and empirically derived. But beyond establishing correspondences, the ultimate goal is modeling: enabling new boomerang designs to be generated from example shapes with awareness of both structure and aerodynamic performance.

Though focused on boomerangs, the methods developed in this thesis aim to contribute more broadly to the analysis and modeling of irregular yet structured 3D shapes, helping bridge the gap between traditional craftsmanship and modern computational tools.

1.1. Contribution

To address these restrictions, this research proposes a novel computational framework specifically designed to analyze, parameterize, and reconstruct boomerang geometries systematically.

The core steps of the method involve:

1. *Correspondence via Structural Landmarks*: Given the inadequacy of traditional global shape-matching methods for asymmetrical and handcrafted objects, structural control points (such as wing tips, elbows, and prominent geometric features) are introduced.
2. *Boundary Curve Extraction using Alpha Shapes*: Rather than relying on standard geometric approximations, alpha shapes combined with boundary curve lifting from 2D to 3D is used, and re-projection to systematically capture boomerang outlines.
3. *Consistent Parameterization via Least Squares Conformal Mapping (LSCM)*: To systematically analyze the shapes, LSCM is applied to achieve consistent UV parameterizations. This technique minimizes angular distortion, preserving critical local aerodynamic features while enabling accurate shape comparisons across different boomerangs.
4. *Shape reconstruction via PCA and gradient optimization, and free-form deformation*: New boomerang geometries are generated using a PCA-based statistical model as a template. Gradient descent is used to find a nearby shape in PCA space, followed by free-form deformation to better match a target outline.

1.2. Key Academic Contributions

This thesis provides several distinct academic contributions to computational geometry and aerodynamic shape analysis:

- *A 3D Scanned Dataset with Established Correspondences*: Over twenty high-performance competition boomerangs were 3D scanned and post-processed to create a comprehensive dataset. This dataset includes structured correspondences across multiple boomerangs, providing a valuable resource for further research and benchmarking correspondence techniques in related studies.
- *Introduction of a Specialized Framework for Aerodynamic Shape Correspondences*: It explicitly addresses the limitations of conventional shape-matching methodologies by integrating structural considerations-such as airfoil geometry into the shape correspondence framework.
- *Bridging Empirical Craftsmanship and Digital Geometry*: This research is the first to stimulate computational formalization of empirically-crafted boomerangs, enabling their systematic analysis, reconstruction, and optimization.
- *A Novel Approach to Boundary Definition and Parameterization*: This framework demonstrates how integrating alpha shapes with a 3D boundary curve for LSCM parameterization can yield realistic results. It showcases the effectiveness of applying a restricted boundary condition within an otherwise unrestricted parameterization method, offering precise control over seam creation and ensuring consistency in shape correspondences.
- *Statistical Shape Reconstruction, Optimization and Reconstruction*: Leveraging PCA combined with gradient-based optimization and a free-form deformation aids in going from a set of 2D points to a 3D shape.

Ultimately, this thesis provides a computational foundation for analyzing, reconstructing, and innovating boomerang designs, bridging gaps between empirical craftsmanship, computational geometry and 3D modelling.

Beyond immediate applications in competitive boomerang design, the methods presented herein may extend to shape analysis and aerodynamic modeling in related fields where similar challenges regarding empirical and intricate geometries persist.

1.3. Report Structure

Chapter 2 reviews related work, highlighting existing approaches in boomerang design, shape correspondences, parameterization, and statistical modeling.

Chapter 3 provides essential technical background on 3D scanning, anatomical boomerang features, and mathematical concepts such as conformal mapping, Principal Component Analysis (PCA), and Iterative Closest Point (ICP) alignment.

Chapter 4 describes the methodology in detail, including dataset creation, boundary extraction, alignment, UV mapping for correspondences, and PCA-based shape generation with Free-Form Deformation (FFD).

Chapter 5 outlines evaluation metrics assessing parameterizations and correspondences through distortion and distance measures, discussing practical interpretations.

Chapter 6 presents a series of experiments exploring the effects of parameter choices and methodological decisions, ranging from alpha shape tuning to PCA behavior and deformation performance.

Chapter 7 reflects on the results, discussing key findings, limitations of the current approach, and potential directions for future work. Chapter 8 concludes the thesis with a summary of contributions and final insights.

An appendix is included to cover supplementary material, such as methods not incorporated into the main pipeline, relevant nomenclature, and documentation of the created dataset.

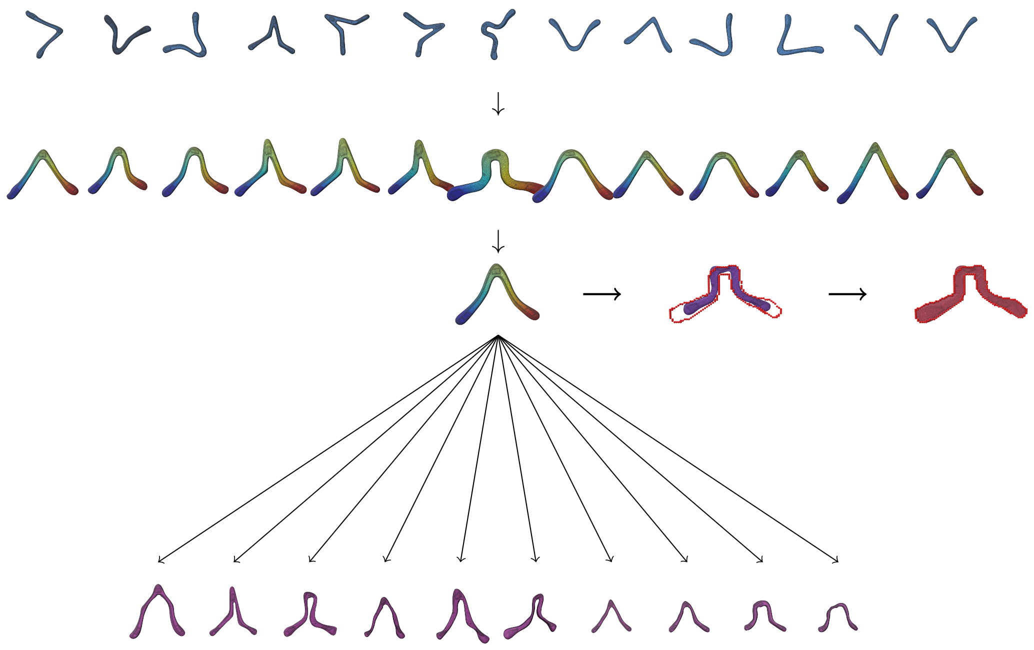


Figure 1.1: Pipeline of the Methodology

2 Related Work

2.1. Boomerang Design and Aerodynamics

Although the physics of boomerangs has been studied extensively [18, 42, 16, 15, 41], the integration of computational tools for their design and analysis remains limited. Most designs still rely on manual crafting or basic 3D modeling techniques. While technologies like 3D printing and mold-based manufacturing highlight the potential of computational geometry to streamline the design process, they also emphasize the difficulty of achieving aerodynamic precision and structural integrity.

2.2. Shape Correspondences

Establishing correspondences between shapes is a known problem in geometry processing. Whether for shape interpolation, recognition, or statistical modeling, finding meaningful mappings between points or regions of different shapes is a key enabler across domains like computer graphics, vision, and medical imaging.

A wide range of techniques has been developed to tackle this challenge. For instance, recent work focuses on dense correspondences between genus-zero shapes, addressing the complexities of aligning entire surfaces in a robust and automated manner [30]. Surveys such as [23] offer a structured overview of these methods, organizing them by the types of assumptions they make, the features they rely on, and their optimization strategies. Across the works, several recurring challenges emerge: managing large deformations, handling partial data, and coping with complex or inconsistent topologies.

Different methodological choices tried to tackle these issues. Some approaches rely on segmenting shapes into perceptually meaningful parts and then matching these parts across objects [28]. Others take a deformation-driven route, iteratively refining correspondences by minimizing deformation energy-a powerful strategy when shapes undergo non-rigid transformations or articulations [47]. A further direction leverages spectral analysis, using Laplacian eigenfunctions to define correspondences in a domain that is more invariant to noise and topological differences [22].

While many shape correspondence techniques are effective in general-purpose scenarios, applying them to boomerangs proved challenging. Their assumptions-such as clean topology, global symmetry, or elastic deformations-did not align well with the nature of handcrafted boomerangs. In particular, methods operating in the spectral domain struggled with directional ambiguity, making it unclear which regions of one shape should correspond to which regions of another.

These difficulties motivated the development of a dedicated correspondence pipeline tailored to boomerangs. Rather than treating their semi-regular, (mostly) 2.5D-like geometry as a challenge, the approach in this work approach leverages it as a structural advantage - incorporating domain knowledge through control points and boundary-aware mappings. This allows us to construct correspondences that are not only geometrically meaningful but also aligned with the functional design of each object.

2.3. Parametrization

In the context of this thesis, parameterization plays a central role in establishing correspondences. By flattening 3D surfaces to 2D domains in a consistent and conformal way, you can enable shape comparisons that preserve local geometric features and make meaningful alignment between structures possible.

Mesh parameterization is widely used in texture mapping, simulation, and shape optimization. A comprehensive theoretical and practical guide to parameterization techniques is available in [19]. Foundational work introduced Least Squares Conformal Maps (LSCM) for automatic texture atlas generation, which has since become a standard method in the field [31].

Recent advances include techniques that combine intrinsic triangulations with parameterization to improve robustness and efficiency [1], as well as spectral conformal parameterization methods that effectively preserve conformal properties [35]. An efficient method for computing geodesic distances, known as the heat method, provides a powerful tool for smooth and accurate parameterizations in complex domains [7].

Further contributions propose local/global optimization approaches to improve parameterization quality [33], and methods that address free-boundary linear parameterization under constraints [24]. Intrinsic parameterizations of surface meshes have also been explored [11], and efficient boundary-based techniques have been introduced to improve surface flattening [37].

A broad overview of historical developments and state-of-the-art methods can be found in comprehensive surveys such as [13].

2.4. Deep Learning Approaches

In recent years, deep learning has emerged as a powerful tool for shape reconstruction, representation, and parameterization. Several methods explore how neural networks can learn meaningful surface representations directly from raw 3D data. For example, planar parameterization has been used to drive symmetric shape reconstruction using neural architectures [21]. Other techniques, such as those inspired by atlas-based learning, use deformable surface elements or learned patches to represent collections of 3D shapes in an interpretable way, enabling both reconstruction and parameterization [10].

Networks like SpiderCNN extend convolution operations to irregular point clouds, extracting local geometric features for classification and segmentation tasks [46]. Hybrid models combine implicit surfaces with geometric primitives to support both free-form design and engineering constraints in shape modeling [43]. Furthermore, learning-based parameterization has been proposed specifically for aerodynamic shape optimization, using either latent space embeddings or direct vertex mappings to enable gradient-based design pipelines [44].

Despite these advances, several limitations persist when applying deep learning methods to the boomerang shapes considered in this thesis. Most deep models rely on large training datasets and perform best when shape variability is either global or follows smooth deformations. In contrast, prior to this work, no dataset of boomerangs existed. These shapes are locally irregular and shaped by function rather than data-driven priors. Moreover, neural parameterization techniques often lack geometric guarantees such as conformality, and their outputs can be difficult to control or interpret in contexts where precise correspondence is essential.

While some methods show promise in generalizing across collections of shapes, they are not yet robust or accurate enough for the kind of high-fidelity, correspondence-driven modeling required in this work. Additionally, many require retraining or extensive tuning to adapt to new classes of shapes. Given these constraints, this thesis focuses instead on a geometry-driven pipeline, grounded in explicit structural cues and conformal parameterization. Nevertheless, the growing body of work in deep shape learning provides valuable inspiration and may become increasingly relevant as such models mature.

2.5. Shape Reconstruction and Modelling with Statistical Models

Statistical models have previously been used in shape reconstruction and modeling, particularly in medical imaging, vision, and design applications. These models leverage prior knowledge about geometric structure to improve accuracy, robustness, and interpretability-especially in ill-posed or under-constrained problems.

One classical approach to shape reconstruction is atlas-based reconstruction, where a predefined statistical model guides the recovery of a 3D shape - an approach closely aligned with the modeling methodology used in this thesis. For instance, [27] demonstrates how a PCA-based anatomical prior can be used to reconstruct 3D shapes from sparse 2D X-ray views. Similarly, statistical shape models have been applied to tasks such as 3D/2D registration and segmentation. In [2], vertebrae are registered using a statistical model of spinal anatomy, while [17] employs a vertebra template with non-rigid transformations to perform mesh warping for vertebra segmentation. These techniques are particularly useful in the context of noisy medical imaging data.

Other methods have extended this to nonrigid shape alignment. For instance, [12] introduces a framework for nonrigid image registration, which enhances alignment between shapes from multiple views. Likewise, [32] integrates statistical priors into motion and geometry estimation pipelines to reduce ambiguity in monocular 3D reconstruction. Statistical models have also enabled real-time applications, such as the facial animation system in [5], which uses online modeling to track and reconstruct expressions in real time.

More recently, some approaches have combined statistical shape modeling with deep learning. These include using latent shape spaces for parameterizing aerodynamic designs [25], or fusing CNNs with statistical body models for robust 3D pose estimation from images [36]. While these methods show promise for high-level reconstruction tasks, they often require large training datasets and lack fine control over localized geometric features.

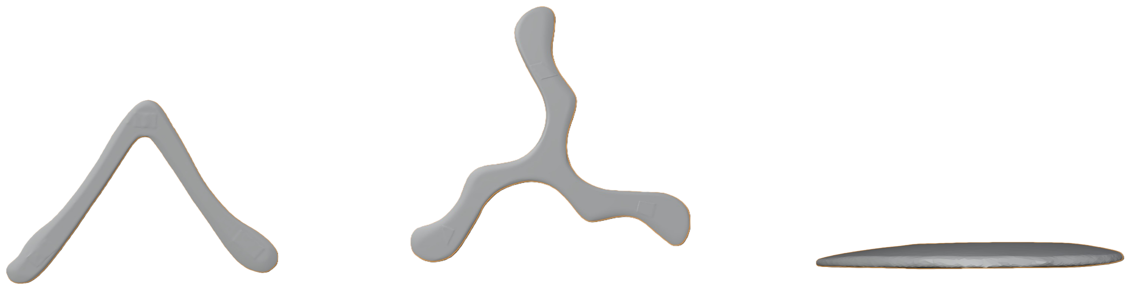
In contrast, traditional modeling tools such as Free-Form Deformation (FFD) remain highly relevant for intuitive and locally-controllable shape modeling. Originally proposed in [38] and later extended in [6], FFD provides a way to manipulate shapes using embedded control lattices, offering both global and local deformation capabilities. Other refinement techniques such as As-Rigid-As-Possible modeling [20] preserve fine-scale geometric detail during deformation and enable fast, interactive manipulation. Related work also explores sketch-based modeling using annotated 2D inputs to produce structured 3D geometry [14], providing semantic control over complex shapes.

This thesis draws from both classical and modern modeling. A 3D PCA-based statistical shape model is constructed from a dataset of boomerangs to define a low-dimensional, structured shape space. This enables new boomerang shapes to be explored and optimized in a controlled and meaningful way. Additionally, Free-Form Deformation from a 2D (sketch-based) perspective is used to provide geometric flexibility when refining shapes toward target boundaries - preserving global structure while allowing local variation.

3

Background

3.1. Anatomy of Boomerang



(a) 2-wing boomerang **(b)** 3-wing boomerang **(c)** Airfoil

Figure 3.1: Comparison of wing boomerangs and an airfoil.

Figure 3.1 illustrates two types of (right-handed) boomerangs: a classic two-wing (here: ”V”-like shape) design and a more modern three-wing variant, along with an airfoil for reference. A boomerang generally consists of multiple wings and a central section, known as the elbow, where the wings intersect.

- **Wings/Arms:** The long, curved sections of the boomerang, known as wings or arms, generate lift during flight. The images depict both a two-wing and a three-wing boomerang, demonstrating how the number of wings influences stability and flight dynamics. For right-handed boomerangs, the left arm is called the dingle wing and the right arm the leading wing.
- **Elbow:** The elbow is the central junction where the wings connect. It plays a crucial role in maintaining the boomerang’s balance and rotation.
- **Airfoil Shape:** Each wing’s cross-sectional profile resembles an airfoil, which is designed to optimize lift. The airfoil (Figure 3.1c) provides a reference for understanding how wing shape affects aerodynamic performance.

3.1.1. Boomerang airfoil

Each boomerang wing has specific airfoils that influence its aerodynamic properties:

- **Leading Edge:** The front edge of the wing that first contacts the airflow. It is typically rounded and plays a key role in generating lift. On the right in Figure 3.1c.
- **Trailing Edge:** The rear edge of the wing where the airflow separates. It is often tapered or sharp to control airflow and reduce drag. On the left in Figure 3.1c.
- **Chord Line:** An imaginary straight line connecting the leading and trailing edges. The angle of attack, which affects lift and stability, is measured relative to this line.

- **An-/Dihedral Angle:** The slight downward/upward tilt of the wings, which contributes to the boomerang's returning flight path.

3.2. 2D Alpha Shapes

Alpha shapes [3] provide a formal framework for reconstructing the "shape" of a set of unorganized points in 2D or 3D space. The construction is based on Delaunay triangulation [29] and a parameter α, which governs the inclusion of simplices (vertices, edges, and faces). A simplex is included in the alpha shape if its circumscribing sphere has a squared radius less than α and does not contain any other points from the dataset. Varying α allows the shape to transition from the convex hull ($\alpha \to \infty$) [26] to the point set itself ($\alpha \to 0$) .

Let $P \subset \mathbb{R}^n$ be a finite set of points and $\mathcal{T}$ its Delaunay triangulation. The alpha complex $\mathcal{K}_\alpha$ is defined as:

$$\mathcal{K}_\alpha = \{\sigma \in \mathcal{T} \mid R(\sigma) \leq \sqrt{\alpha} \text{ and } \sigma \text{ is conflict-free}\} \tag{3.1}$$

where $R(\sigma)$ is the radius of the smallest circumscribing sphere of the simplex σ, and "conflict-free" means that no other points of P lie inside this sphere.

The corresponding alpha shape S_α is the union of the simplices in $\mathcal{K}_\alpha$, representing a polytope that can capture both topological and geometric properties of the original dataset. Alpha shapes can approximate an object's surface with provable guarantees, provided the point set satisfies certain sampling conditions. These conditions ensure that the reconstructed alpha shape is homeomorphic to the original object and that the approximation error is bounded.

Since the alpha-shape construction can produce a large number of edges and simplices, especially for small values of α, a filtering step is often applied to extract only the most relevant structures. The approach used in this thesis is to classify edges and retain regular and singular edges, which define the primary (outside) boundary of the shape: on the boundary of the α-shape, but no incident to a triangle of the α-complex. Additionally, small isolated components or noise artifacts can be removed by filtering out short edges or components with low persistence in the alpha filtration. This post-processing step helps ensure that the extracted border is a meaningful representation of the underlying shape while reducing unwanted noise.

3.3. Conformal Mapping

Conformal mapping is a technique that transforms one surface onto another while preserving local angles. Although this may alter the overall scale, it ensures that small features retain their shape. This property makes conformal maps useful in areas like texture mapping, mesh generation, and geometry processing. In 3D graphics, they are particularly valuable for creating UV parameterizations that reduce distortion while maintaining detail. In this work, conformal mapping will be applied with a defined boundary as input.

3.3.1. UV Mapping

UV mapping refers to the process of projecting a 3D surface onto a 2D plane by assigning each vertex a coordinate in a 2D texture space. This mapping allows for accurate texture placement on complex models. A good UV layout aims to minimize distortion, preserve the shape of features, and prevent overlapping regions. While many traditional approaches rely on manual tweaking to balance stretch and uniformity, automated methods - especially those using conformal techniques - can significantly improve the quality of the mapping.

3.3.2. Least Squares Conformal Maps (LSCM)

Lévy et al. [31] introduced Least Squares Conformal Maps (LSCM) as an efficient parameterization method that minimizes angular distortion while maintaining computational efficiency. Unlike harmonic maps or energy-based methods, LSCM formulates the parameterization problem as a least-squares approximation of the Cauchy-Riemann equations. This formulation leads to the following advantages:

- **Angle Preservation:** LSCM minimizes non-uniform scaling and shear, maintaining local structures.

- **Automatic Boundary Handling:** Unlike earlier methods that require boundary conditions (e.g., fixed convex boundaries), LSCM allows free-form boundary constraints. This property is heavily used in this work.
- **Robustness to Mesh Resolution:** The resulting parameterization is independent of the resolution of the mesh, ensuring consistent UV layouts across different levels of detail.
- **Numerical Stability:** The objective function is quadratic and can be solved efficiently using numerical optimization techniques such as the Conjugate Gradient method.

3.3.3. Mathematical Formulation

The LSCM method aims to find a conformal parameterization of a 3D surface by minimizing angle distortions. Given a triangulated surface with vertices V and faces T, a mapping $(x, y, z) \to (u, v)$ that locally preserves angles is sought.

Conformality is measured using a complex-valued function $U = u + iv$, representing the 2D parameterization in the complex plane. The conformality condition states that the Cauchy-Riemann equations should hold:

$$\frac{\partial U}{\partial x} + i\frac{\partial U}{\partial y} = 0. \tag{3.2}$$

Since an exact conformal map is not always possible for arbitrary surfaces, LSCM minimizes the deviation from this condition in a least-squares sense:

$$C(U) = \sum_{T \in \mathcal{T}} \int_T \left| \frac{\partial U}{\partial x} + i\frac{\partial U}{\partial y} \right|^2 dA. \tag{3.3}$$

Here,

$\mathcal{T}$ is the set of all triangles in the mesh,

(x, y) are the local coordinates of a point in a triangle,

(u, v) are the 2D texture coordinates of that point,

$U = u + iv$ is the complex representation of the parameterization,

dA is the differential area element of the triangle.

Minimizing $C(U)$ results in a parameterization that is as conformal as possible while ensuring computational efficiency. Unlike other methods that require the boundary to be mapped to a convex shape, LSCM is more flexible in its handling of boundaries. It requires only two fixed vertices to ensure a well-posed system, but can also incorporate arbitrary boundary constraints if desired. In this work, full boundary constraints are provided to guide the parameterization and ensure consistent alignment across shapes.

The whole algorithm is outlined in Section two of [31].

3.4. Principal Component Analysis (PCA)

Principal Component Analysis (PCA) is a dimensionality reduction technique used to find a low-dimensional representation of high-dimensional data while preserving the most significant variations. In this case, PCA is applied to a set of 3D shapes to identify the principal modes of variation.

3.4.1. Data Representation

Each shape in the dataset consists of n vertices, each with 3D coordinates. Here, each shape is represented as a vector:

$$\mathbf{s}_i = \begin{bmatrix} x_1 & y_1 & z_1 & x_2 & y_2 & z_2 & \cdots & x_n & y_n & z_n \end{bmatrix}^\top \in \mathbb{R}^{3n} \tag{3.4}$$

Given a collection of m such shapes, construct a data matrix:

$$\mathbf{S} = \begin{bmatrix}\mathbf{s}_1 & \mathbf{s}_2 & \cdots & \mathbf{s}_m\end{bmatrix} \in \mathbb{R}^{3n\times m} \tag{3.5}$$

where each column represents a shape.

3.4.2. Mean Shape and Centering

The mean shape is computed by averaging the vertex positions across all shapes:

$$\bar{\mathbf{s}} = \frac{1}{m}\sum_{i=1}^{m}\mathbf{s}_i \tag{3.6}$$

To center the data, subtract the mean shape from each shape:

$$\mathbf{S}_c = \mathbf{S} - \bar{\mathbf{s}}\mathbf{1}^T \tag{3.7}$$

where $\mathbf{1}$ is a vector of ones.

3.4.3. Singular Value Decomposition (SVD)

Instead of computing the covariance matrix $\mathbf{S}_c\mathbf{S}_c^T$, which is large and computationally expensive, perform Singular Value Decomposition (SVD) directly on the centered data:

$$\mathbf{S}_c = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \tag{3.8}$$

where:

- $\mathbf{U} \in \mathbb{R}^{3n\times m}$ contains the principal components (eigenvectors).
- $\mathbf{\Sigma}$ is a diagonal matrix with singular values.
- $\mathbf{V}$ contains the right singular vectors.

The principal components (columns of $\mathbf{U}$) represent the modes of shape variation.

3.4.4. Shape Deformation using PCA Modes

A new shape can be generated by adding a weighted sum of principal components to the mean shape:

$$\mathbf{s}' = \bar{\mathbf{s}} + \sum_{j=1}^{k} w_j\mathbf{u}_j \tag{3.9}$$

where:

- $\mathbf{s}'$ is the deformed shape.
- $\mathbf{u}_j$ is the j-th principal mode (column of $\mathbf{U}$).
- w_j is a weight controlling the influence of the j-th mode.
- k is the number of principal components used (typically much smaller than $3n$).

Similarly, starting from an input shape $\mathbf{s}_i$, a deformation can be applied using:

$$\mathbf{s}' = \mathbf{s}_i + \sum_{j=1}^{k} w_j\mathbf{u}_j. \tag{3.10}$$

By adjusting the weights w_j, new plausible shapes can be synthesized while preserving the dominant geometric variations observed in the dataset.

3.4.5. Justification

Principal Component Analysis (PCA) is particularly suited to this study due to its effectiveness in capturing dominant shape variations from empirical data without relying on predefined parametric models. PCA allows the extraction of statistically significant deformation modes (deformations from a template), providing a foundation for generating boomerang shapes based on observed geometric variance. Alternative methods, such as neural networks or parametric regression, often require large datasets or predefined parameters, neither of which aligns well with empirically derived, small-scale boomerang data. By employing PCA, this research delivers computationally feasible models.

3.5. Iterative Closest Point (ICP)

The Iterative Closest Point (ICP) algorithm is a widely used method for aligning two point clouds or surfaces by iteratively minimizing the distance between corresponding points. It is commonly applied in 3D shape registration, object tracking, and reconstruction tasks. Given an initial estimate of the transformation, ICP refines the alignment by iteratively finding closest point correspondences and updating the transformation.

3.5.1. Algorithm Overview

Given two sets of points:

- $P = \{p_1, p_2, ..., p_n\}$ (the source point cloud)
- $Q = \{q_1, q_2, ..., q_m\}$ (the target point cloud),

ICP seeks a rigid transformation (R, t), where R is a rotation matrix and t is a translation vector, such that:

$$R^*, t^* = \arg\min_{R,t} \sum_i \|Rp_i + t - q_i^*\|^2, \tag{3.11}$$

where q_i^* is the closest point in Q to $Rp_i + t$.

The ICP algorithm follows these iterative steps:

1. **Find correspondences:** For each point $p_i \in P$, find the closest point $q_i^* \in Q$.
2. **Compute transformation:** Solve for the optimal rotation R and translation t that minimize the mean squared error.
3. **Update and iterate:** Apply the transformation to P and repeat until convergence (i.e., when the change in error falls below a threshold).

3.5.2. Computing the Optimal Transformation

To compute R and t, the problem is formulated as minimizing the sum of squared errors:

$$E(R, t) = \sum_i \|Rp_i + t - q_i^*\|^2. \tag{3.12}$$

This is solved using the Singular Value Decomposition (SVD) approach:

1. Compute the centroids of P and Q:

$$\bar{p} = \frac{1}{n} \sum_i p_i, \quad \bar{q} = \frac{1}{n} \sum_i q_i^*. \tag{3.13}$$

2. Construct the covariance matrix:

$$H = \sum_i (p_i - \bar{p})(q_i^* - \bar{q})^T. \tag{3.14}$$

3. Perform SVD on H as $H = U\Sigma V^T$.
4. Compute the optimal rotation as $R = VU^T$.
5. Compute the optimal translation as $t = \bar{q} - R\bar{p}$.

4 Methods

The implementation developed as part of this thesis is publicly available at:

`https://github.com/NilsvVeen/MSc-Thesis-TUD---boomerang-correspondence`

The dataset used in the experiments, consisting of scanned and parameterized boomerangs, is accessible at: `https://graphics.tudelft.nl/NilsVanVeen`

4.1. Dataset Collection

To build a dataset of boomerangs for this research, a range of competition-grade , non-standard "V"-shaped and multi-blade boomerangs were scanned using a Creaform Go!SCAN device. This handheld 3D scanner offers an accuracy of up to 0.05 mm and a resolution of 0.1 mm [8]. Compared to alternatives such as photogrammetry [34], it obtains 3d models faster and with a more reliable capture of fine details, critical for modeling the intricate twists, curves, and thickness variations that influence boomerang aerodynamics.

Since the Go!SCAN device cannot capture an entire boomerang in a single pass, each boomerang was scanned from multiple static orientations to ensure comprehensive surface coverage. The resulting partial scans were then merged into a cohesive 3D model using VXelements software [9], which supports user-guided alignment based on markers or reference points. Although the exact alignment algorithm used in VXelements is not disclosed, it is likely based on an Iterative Closest Point (ICP) method [4] for minimizing point-to-point distances in overlapping regions. The software also includes noise filtering and outlier rejection mechanisms that help achieve robust and accurate alignment.

After the scans were merged, additional post-processing was applied: small holes were filled, surfaces were smoothed, and local artifacts introduced during scanning or alignment were corrected. In most cases, the resulting meshes were decimated (in Blender) to reduce complexity and improve computational efficiency in later stages of the pipeline.

Some boomerangs presented scanning challenges due to thin airfoils, reflective surfaces, or transparent materials. To improve scan quality, these were treated with a temporary chalk spray to enhance surface visibility. Scanning was conducted with an accuracy setting between 0.2 mm and 0.4 mm. While the resulting 3D models reliably captured overall geometry, minor deviations were observed. For instance, boomerangs with a measured thickness of 3–3.4 mm were occasionally reconstructed at slightly higher values (3.5–3.8 mm), likely due to smoothing and surface interpolation during the alignment and meshing process.

The scanned and processed boomerangs form the foundation of the dataset used throughout this thesis.

4.2. Boundary Condition

A fundamental component of the proposed correspondence pipeline is the parameterization of boomerang surfaces. Due to the structured yet non-standard nature of boomerang geometries - largely planar but exhibiting critical three-dimensional variations - boundary-based parameterization offers a foundation for establishing shape correspondences.

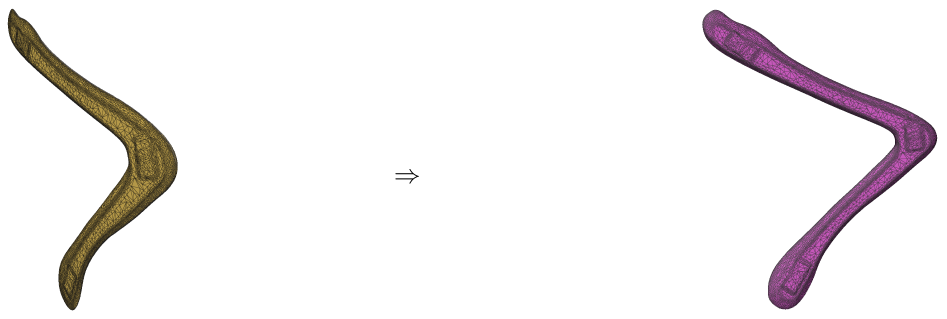
It is important to note that boomerangs, as scanned 3D objects, are typically closed surfaces and do not possess true geometric boundaries. The boundaries referred to in this work are introduced algorithmically: they are artificial seams created by projecting the shape from a top-down view and extracting a representative 2D outline. This outline, derived from the surface silhouette of the projected mesh, acts as a boundary condition for conformal parameterization. In this way, a consistent structure is imposed across shapes that otherwise lack a natural edge.

To parameterize the surfaces, Least Squares Conformal Mapping (LSCM) is used. This technique minimizes angular distortion when flattening a 3D surface into a 2D domain. The result is a UV coordinate system that places each shape into a shared parametric space.

While LSCM typically requires only two fixed vertices to anchor the parameterization, it also supports additional positional constraints. In this work, the flexibility of LSCM is utilized by constraining the entire (artificially introduced) boundary of each shape rather than relying solely on minimal anchors. This approach enforces shared boundary conditions across different boomerangs, resulting in compatible UV maps and correspondences in the parameter domain.

In this approach, two shapes are aligned by first matching their extracted boundary curves, then using one as a reference for the other. Specifically, Shape 1 is projected from a top-view to define a 2D boundary outline and this is used as the shared boundary constraint for Shape 2. Establishing a one-to-one mapping between the outlines allows both shapes to be parameterized using LSCM with a consistent boundary, ensuring meaningful correspondences across their surfaces.

4.2.1. Step 1: Align objects in world space



(a) Boomerang input **(b)** Boomerang Top-View-Aligned

Figure 4.1: Plane fitted alignment

Both shapes are rotated in the world space so their tops or bottoms face the same direction. This is done by fitting a plane to each shape and ensuring the plane's normal vector points in the desired direction, e.g., $(0, 0, 1)$.

The plane fitting in is achieved by minimizing the perpendicular distances of the vertices to the plane. Instead of solving the full optimization problem:

$$\min_{\mathbf{n},d} \sum_{i=1}^{N} (\mathbf{n} \cdot \mathbf{v}_i + d)^2 , \quad \text{where } \|\mathbf{n}\| = 1 \tag{4.1}$$

a simplified version is used by assuming the plane passes through the centroid of the points $(\mathbf{C})$. The centroid is calculated as:

$$\mathbf{C} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{v}_i \tag{4.2}$$

where $\mathbf{v}_i$ are the vertices of the shape.

The covariance matrix of the points, computed as:

$$M = \sum_{i=1}^{N} (\mathbf{v}_i - \mathbf{C})(\mathbf{v}_i - \mathbf{C})^\top \tag{4.3}$$

captures the variance in all directions. The plane normal vector $\mathbf{n}$ is then found as the eigenvector corresponding to the smallest eigenvalue of M.

Thus, minimizing:

$$\min_{\mathbf{n}} \sum_{i=1}^{N} \left(\mathbf{n}^\top (\mathbf{v}_i - \mathbf{C})\right)^2 \tag{4.4}$$

where the plane offset is implicitly given by $d = -\mathbf{n}^\top \mathbf{C}$.

Users can manually flip the object if needed by rotating it $180°$ around a chosen axis to ensure consistent orientation, aligning the normal vector to the desired direction, e.g., $(0, 0, 1)$.

4.2.2. Step 2: Project onto a parallel plane

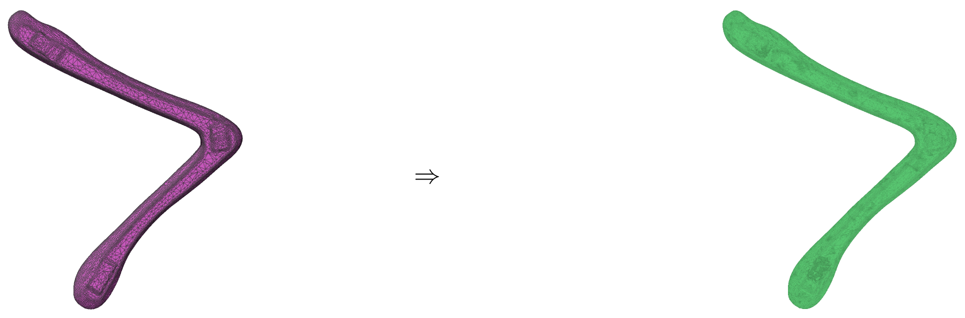
Once aligned, each shape is projected onto a plane parallel to the fitted plane. In the general case, the projection is computed as:

$$\mathbf{v}'_i = \mathbf{v}_i - (\mathbf{n} \cdot (\mathbf{v}_i - \mathbf{c}))\mathbf{n} \tag{4.5}$$

where $\mathbf{c}$ is a point on the plane, and $\mathbf{n}$ is the plane normal. This formula ensures that the points are projected orthogonally onto the fitted plane.

However, if the plane normal $\mathbf{n}$ aligns with the z-axis (e.g., after alignment in Step 1), the projection simplifies to simply discarding the z-coordinate of each vertex:

$$\mathbf{v}'_i = (x_i, y_i) \tag{4.6}$$



(a) Boomerang input **(b)** Projection

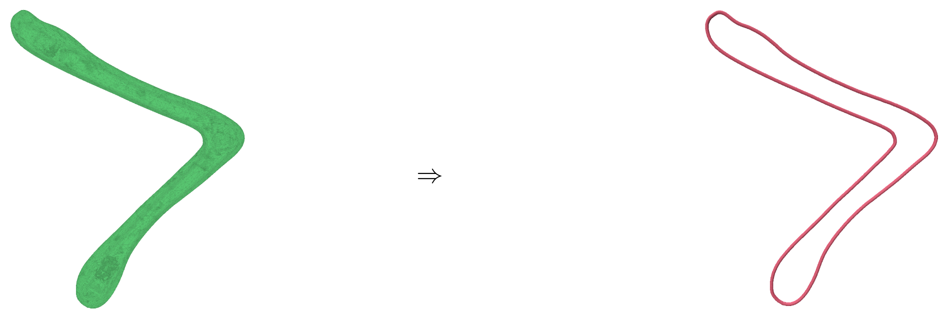
Figure 4.2: Projection from a 3D shape to a 2D Shape

4.2.3. Step 3: Compute the 2D outline using Alpha Shapes

Using the 2D projections, the 2D alpha shape is computed for each object.

The α parameter is adjusted iteratively by the user to balance capturing the outer boundary while minimizing internal sub-boundaries. Care must be taken not to use an α that is too large, as it may introduce gaps in sharper corners, reducing accuracy.

How α is picked, is discussed in section 6.1.



(a) 2D projection **(b)** 2D α shape

Figure 4.3: From a projected 2D shape to a 2D α outline

4.2.4. Step 4: Sort the boundary curve

The algorithm sorts the boundary vertices

$$\{v_1, v_2, \ldots, v_n\} \quad \text{with } v_i \in \mathbb{R}^2 \tag{4.7}$$

using a nearest-neighbor approach. The steps are as follows:

Start with an arbitrary vertex, say v_{i_0}. Mark it as visited, and initialize the sorted sequence:

$$s_1 = v_{i_0} \tag{4.8}$$

The Euclidean distance between two vertices v_i and v_j is given by:

$$d(v_i, v_j) = \sqrt{\sum_{k=1}^{d} \left(v_i^{(k)} - v_j^{(k)}\right)^2} \tag{4.9}$$

For each subsequent step $l = 2, 3, \ldots, n$, from the current vertex s_{l-1}, select the next vertex s_l from the set of unvisited vertices

$$S \setminus \{s_1, s_2, \ldots, s_{l-1}\} \tag{4.10}$$

such that:

$$s_l = \underset{v \in S \setminus \{s_1, s_2, \ldots, s_{l-1}\}}{\arg\min} \; d(s_{l-1}, v) \tag{4.11}$$

Mark s_l as visited and update the current vertex to s_l.

After all vertices have been visited, the sorted list for Shape 1 is:

$$\text{SBV}_{\text{Shape 1}} = \{s_1, s_2, \ldots, s_n\} \tag{4.12}$$

Similarly, for Shape 2, the sorted boundary vertices are given by:

$$\text{SBV}_{\text{Shape 2}} = \{s'_1, s'_2, \ldots, s'_m\} \tag{4.13}$$

Since the choice of the starting vertex affects the traversal direction (clockwise or counterclockwise) in the XY-plane, a post-processing step is applied. If the user-specified landmarks are provided in a clockwise order and the computed curve is counterclockwise, the order of the vertices is reversed to ensure consistency.

4.2.5. Step 5: User-defined landmarks

Since the outlines (curves) have different dimensions (number of points), users can select landmarks along the boundaries that should correspond to each other. These landmarks L act as anchor points for the parametrization. These points are the first correspondences. An example can be observed in Figure 4.4 where there are four landmarks (at the tips of the wings, inner and outer elbow) between two shapes.

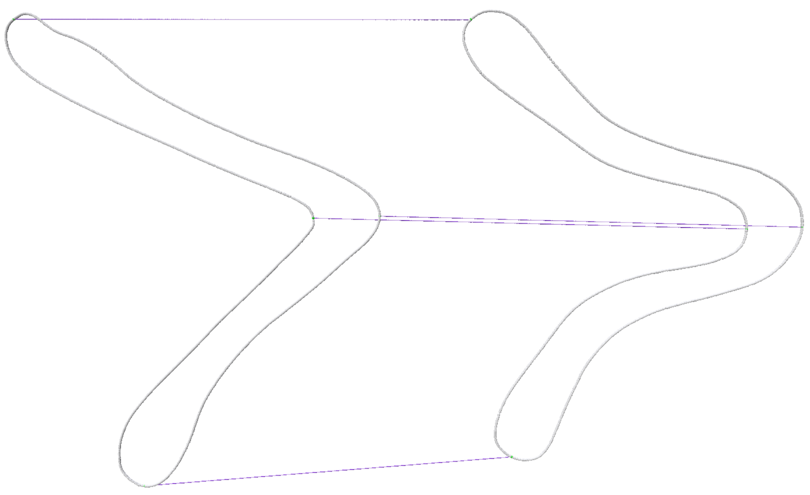


Figure 4.4: User-defined landmarks on two shapes

4.2.6. Step 6: Apply constant speed parametrization over the unit interval between landmarks

Between each pair of corresponding landmarks, constant speed parametrization over the unit interval is applied to both shapes.

Denote the n landmarks as:

$$\text{Landmarks } L = \{l_1, l_2, \ldots, l_n\} \tag{4.14}$$

Equalizing Between Landmarks

In the implementation of this work, the number of vertices between corresponding landmarks in Shape 1 and Shape 2 is balanced (equalized). This is necessary because the two shapes may have different vertex distributions, meaning one shape could have more points between landmarks than the other.

$$|\{s_t, s_{t+1}, \ldots, s_q\}| \quad \gtrless \quad \left|\{s'_p, s'_{p+1}, \ldots, s'_r\}\right| \tag{4.15}$$

To formalize this, let's define:

$L_1 = \{s_t, s_{t+1}, \ldots, s_q\}$ as the set of vertices between two consecutive landmarks in Shape 1, where t and q are the indices of the landmarks in Shape 1.

$L_2 = \{s'_p, s'_{p+1}, \ldots, s'_r\}$ as the corresponding set of vertices in Shape 2, where p and r are the indices of the landmarks in Shape 2.

These indices are determined based on landmark positions:

t and q are the indices of the start and end landmarks in Shape 1. p and r are the indices of the corresponding start and end landmarks in Shape 2.

The goal is to ensure

$$|L_1| \quad \approx \quad |L_2| \tag{4.16}$$

While omitting this step would not necessarily break the process, it helps prevent an excess of points in Shape 1 (or vice versa) from having too few corresponding matches in Shape 2. This imbalance could introduce inaccuracies when lifting the curve in later stages. Equalizing beforehand ensures a more consistent mapping and reduces unnecessary computations.

Parameterization on Shape 1

For Shape 1, consider the sorted boundary vertices $\{\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_N\}$ along the curve. The constant speed parametrization over the unit interval between landmarks l_l and l_{l+1} is defined as:

$$u_i = \frac{\sum_{j=1}^{i} \|\mathbf{s}_j - \mathbf{s}_{j-1}\|}{\sum_{j=1}^{N} \|\mathbf{s}_j - \mathbf{s}_{j-1}\|}, \quad u_i \in [0,1] \tag{4.17}$$

where $\mathbf{s}_j$ represents the jth vertex in the sorted boundary of Shape 1, and the index j runs over the vertices between landmarks l_l and l_{l+1}.

Parameterization on Shape 2

For Shape 2, assume the sorted boundary vertices are denoted by $\{\mathbf{s}'_1, \mathbf{s}'_2, \ldots, \mathbf{s}'_M\}$. To establish a correspondence with Shape 1, adjust the parameterization of Shape 2 to match that of Shape 1 as follows:

Step 1: Compute the constant speed parametrization over the unit interval for Shape 2

$$u'_j = \frac{\sum_{k=1}^{j} \|\mathbf{s}'_k - \mathbf{s}'_{k-1}\|}{\sum_{k=1}^{M} \|\mathbf{s}'_k - \mathbf{s}'_{k-1}\|}, \quad u'_j \in [0,1] \tag{4.18}$$

Step 2: Find the Corresponding Interval For each parameter u_i from Shape 1, identify two adjacent parameters u'_k and u'_{k+1} in Shape 2 such that:

$$u'_k \leq u_i < u'_{k+1} \tag{4.19}$$

This step ensures that the precise interval on Shape 2 is located where u_i falls in Shape 1, allowing interpolation between $\mathbf{s}'_k$ and $\mathbf{s}'_{k+1}$. By doing so, you can guarantee that the corresponding point on Shape 2 is positioned at a relative distance along its boundary that mirrors the distance traveled along Shape 1. In other words, it preserves the proportionality of the boundary walk between the two shapes.

Step 3: Interpolate the New Point on Shape 2 Once the correct interval is identified, compute the interpolation factor:

$$t = \frac{u_i - u'_k}{u'_{k+1} - u'_k} \tag{4.20}$$

Then, the new corresponding point on Shape 2 is obtained by:

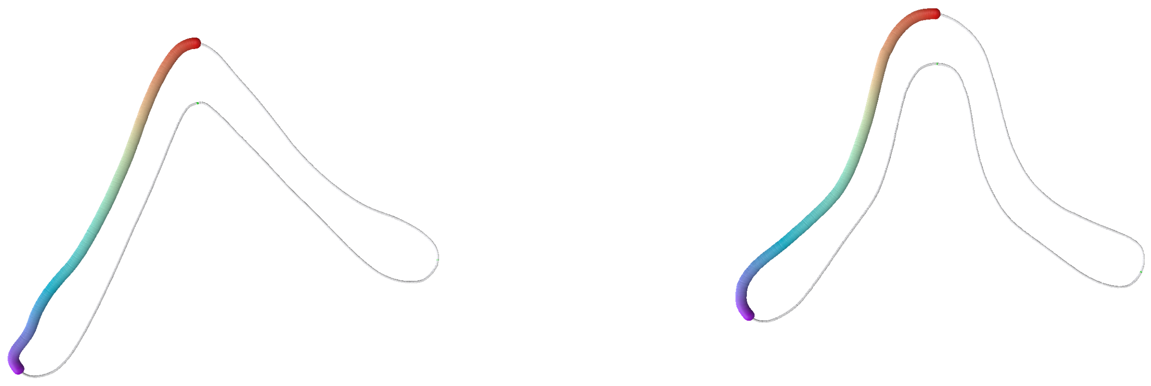
$$\mathbf{s}''_i = (1-t)\,\mathbf{s}'_k + t\,\mathbf{s}'_{k+1} \tag{4.21}$$

This yields a new set of points on Shape 2 that match the parameterization of Shape 1.

Note that in computations, there is one edge case where a point may be indexed across the boundary indices. Specifically, if u_i falls near the end of the parameterization in Shape 1, its corresponding interval in Shape 2 may wrap around, requiring special handling to ensure continuity.

In Figure 4.5, the constant speed parametrization over the unit interval between two landmarks is illustrated. The parameterization is color-coded in a rainbow gradient, allowing for an intuitive visual representation of how the parameter values progress along the shape. In particular, you can observe a one-to-one mapping between $\{\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_N\}$ and $\{\mathbf{s}''_1, \mathbf{s}''_2, \ldots, \mathbf{s}''_N\}$.

This step is necessary because simply assuming a one-to-one mapping between existing vertices would be incorrect. The vertices in Shape 2 may have varying distances between neighbors, meaning the total length of the boundary traversal could differ from that of Shape 1. By applying constant speed parametrization over the unit interval, it ensures that longer sections receive proportionally larger spacing, while shorter sections have smaller spacing, preserving the relative distribution of points along the shape. Additionally, this step is helpful to minimize small inaccuracies at the boundary level, which magnify during parameterization and reconstruction stages, potentially resulting in substantial deviations in predicted shapes.



(a) Unit parametrization between two landmarks on Shape 1

(b) Unit parametrization between two landmarks on Shape 2

Figure 4.5: Unit parametrization between two landmarks on two shapes. Color encoded

4.2.7. Step 7: Lift the 2D curve to 3D

The 2D boundary curves are mapped back into 3D space. For both shapes, first calculate the average z-value (thickness) of the entire mesh:

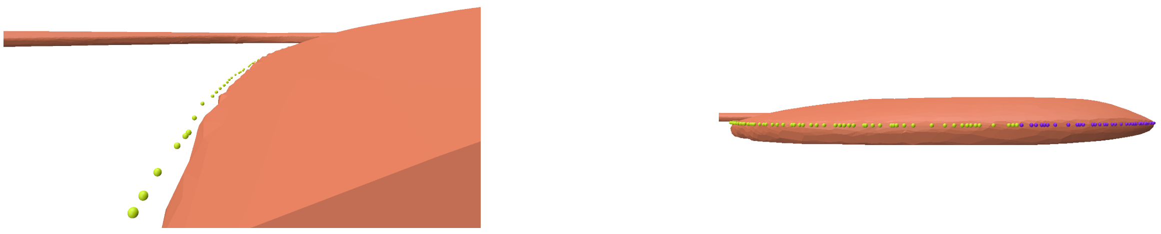
$$\bar{z} = \frac{1}{N} \sum_{i=1}^{N} z_i \tag{4.22}$$

where z_i are the z-coordinates of the mesh vertices and N is the total number of vertices in the mesh.

Next, use the average z-value to lift the 2D boundary curve to a plane:

$$P_i = (\mathbf{p}_{x_i}, \mathbf{p}_{y_i}, \bar{z}) \quad \text{for each} \quad (\mathbf{p}_{x_i}, \mathbf{p}_{y_i}) \in \text{2D curve} \tag{4.23}$$

where $(\mathbf{p}_{x_i}, \mathbf{p}_{y_i})$ represents the coordinates of each point on the 2D curve, and $\bar{z}$ is the average z-value computed across the entire mesh. Note here that for shape 1 $(\mathbf{p}_{x_i}, \mathbf{p}_{y_i}) = (s_{i,x}, s_{i,y})$ and for shape 2 $(\mathbf{p}_{x_i}, \mathbf{p}_{y_i}) = (s''_{i,x}, s''_{i,y})$. An example is show in 4.6.



(a) Lifted curve to a plane in 3D, side view

(b) Lifted curve to a plane in 3D, view from tip

Figure 4.6: Lifted curve to a plane in 3D, two perspectives

Once the 2D points are lifted to 3D-positioned in the middle of the boomerang's height on a plane, each (x, y, z) point of the boundary curve is projected onto the nearest point on the mesh surface. This nearest point, denoted by $\mathbf{P}'_i$, is computed by finding the point on the mesh that minimizes the Euclidean distance to the lifted point $\mathbf{P}_i$:

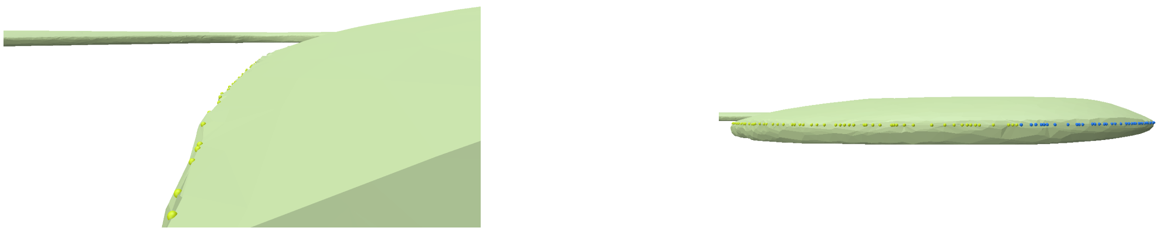
$$\mathbf{P}'_i = \arg \min_{\mathbf{Q} \in \text{mesh surface}} \|\mathbf{P}_i - \mathbf{Q}\| \tag{4.24}$$

Note that $\mathbf{P}'_i$ may lie on a face or an edge of the mesh rather than exactly at one of its vertices.

To ensure the boundary curve aligns precisely with the mesh, identify the mesh vertex closest to $\mathbf{P}'_i$ by computing

$$\mathbf{P}_j = \arg \min_{\mathbf{P}_j \in \text{mesh vertices}} \|\mathbf{P}'_i - \mathbf{P}_j\| \tag{4.25}$$

Once the closest vertex $\mathbf{P}_j$ is determined, its coordinates are updated to match those of $\mathbf{P}'_i$, thereby aligning the mesh with the boundary curve.


(a) Lifted curve to a plane in 3D, side view

(b) Lifted curve to a plane in 3D, view from tip

Figure 4.7: Lifted curve to a plane in 3D, two perspectives

Both the projected boundary curve and the updated mesh are shown in Figure 4.7. In Figure 4.6, the boundary points are initially positioned at a uniform height, often above the mesh, giving the appearance that they are ”floating” in space. After projection, as seen in 4.7, these points are precisely aligned with the mesh surface. The boundary points typically follow the most prominent features of the airfoil geometry, particularly near the leading and trailing edges, where the curvature is steepest. Specifically, they align with the uppermost regions of the leading and trailing edges, just before the surface curves downward, essentially near the points (but still above) where the chord line intersects these edges.

4.2.8. Making the 3D Curve edge-connected

In cases where multiple vertices are closest to a single projected point, several boundary vertices are discarded to ensure that each mesh vertex is only shifted once. A side effect of this simplification is the loss of some correspondences between boundary vertices.

Using Least Squares Conformal Mapping (LSCM) requires at least two boundary vertices to function correctly. During experimentation (see Section 6.7), it was observed that LSCM performs sub-optimally when the boundary vertices are not directly edge-connected. To address this, the boundary vertices are connected as follows:

1. For Shape 1: For each boundary vertex $B_{i,\text{Mesh 1}}$ on the 3D mesh, check if the next boundary vertex shares an edge. If an edge exists, continue to the next vertex. Otherwise, calculate the shortest path between the current boundary vertex and the neighboring boundary vertex using Dijkstra’s algorithm. The vertices along this shortest path are added to the set of boundary vertices, resulting in an updated boundary $B'_{\text{Mesh 1}}$.

2. For Shape 2: Repeat the same process for Shape 2, resulting in an updated boundary $B'_{\text{Mesh 2}}$.

Initially, there is a one-to-one correspondence between the original boundary vertices of the two meshes:

$$B_{i,\text{Mesh 1}} \leftrightarrow B_{i,\text{Mesh 2}} \tag{4.26}$$

However, for the newly added boundary vertices, $B'_{\text{Mesh 1}}$ and $B'_{\text{Mesh 2}}$, no such correspondence exists because the two meshes may have different topologies. To establish a correspondence between the new boundary vertices of Shape 2 and Shape 1, proceed as follows:

1. Identify Newly Added Vertices: Identify vertices in $B'_{\text{Mesh 2}}$ that are not part of the original boundary $B_{\text{Mesh 2}}$:

$$B''_{\text{Mesh 2}} = B'_{\text{Mesh 2}} \setminus B_{\text{Mesh 2}} \tag{4.27}$$

2.Find Neighbors: For each new boundary vertex $B''_{i,\text{Mesh 2}}$, find its two neighboring vertices in $B_{\text{Mesh 2}}$, denoted as $B_{m,\text{Mesh 2}}$ and $B_{n,\text{Mesh 2}}$. Let their indices in $B_{\text{Mesh 2}}$ be m and n, respectively:

$$B'_{k,\text{Mesh 2}} = B_{m,\text{Mesh 2}}, \quad B'_{l,\text{Mesh 2}} = B_{n,\text{Mesh 2}} \tag{4.28}$$

3. Compute Shortest Paths: Calculate the shortest edge path between $B'_{k,\text{Mesh 2}}$ and $B'_{l,\text{Mesh 2}}$. Also, calculate the shortest edge path between $B'_{k,\text{Mesh 2}}$ and the new boundary vertex $B''_{i,\text{Mesh 2}}$.

4. Calculate Unit Distance: Compute the relative parameterization distance percentage dp_i of $B''_{i,\text{Mesh 2}}$ along the shortest path from $B'_{k,\text{Mesh 2}}$ to $B'_{l,\text{Mesh 2}}$:

$$dp_i = \frac{||B''_{i,\text{Mesh 2}} - B'_{k,\text{Mesh 2}}||}{||B'_{l,\text{Mesh 2}} - B'_{k,\text{Mesh 2}}||} \tag{4.29}$$

where $||X||$ is the distance walking over the edges.

5. Calculate Distance to Walk in Mesh 1: Using dp_i, calculate the distance d_i to walk along the corresponding path between $B_{m,\text{Mesh 1}}$ and $B_{n,\text{Mesh 1}}$:

$$d_i = dp_i \cdot ||B_{n,\text{Mesh 1}} - B_{m,\text{Mesh 1}}|| \tag{4.30}$$

This d_i represents the scaled distance along the path in Mesh 1. To find the exact point on Mesh 1, you "walk" along the edges of the connected graph between $B_{m,\text{Mesh 1}}$ and $B_{n,\text{Mesh 1}}$, summing edge lengths until reaching the desired distance d_i.

Since this involves traversing edges of the connected graph, the process is as follows: Start at $B_{m,\text{Mesh 1}}$. Traverse the edges along the shortest path toward $B_{n,\text{Mesh 1}}$, keeping a running sum of the edge lengths. Once the cumulative distance surpasses d_i, identify the two neighboring vertices connected by the edge containing the target point. Interpolate the position of the target point along this edge based on the remaining distance.

6. Unit Parameterize with Shape 1:

$$B''_{i,\text{Mesh 1}} = (1 - \alpha) \cdot B_{p,\text{Mesh 1}} + \alpha \cdot B_{q,\text{Mesh 1}} \tag{4.31}$$

where $B_{p,\text{Mesh 1}}$ and $B_{q,\text{Mesh 1}}$ are the two neighboring vertices, and α is the interpolation factor calculated as:

$$\alpha = \frac{\text{Remaining Distance}}{||B_{q,\text{Mesh 1}} - B_{p,\text{Mesh 1}}||} \tag{4.32}$$

7. Repeat for All New Points: Perform this process for all new points in $B''_{\text{Mesh 2}}$ to obtain the corresponding updated boundary $B''_{\text{Mesh 1}}$. Note that there is now a correspondences $B''_{i,\text{Mesh 1}} \leftrightarrow B''_{i,\text{Mesh 2}}$, where $B''_{\text{Mesh 2}}$ has vertices in the original mesh of shape 2 while $B''_{\text{Mesh 1}}$ has no vertices in the original mesh of shape 1.

4.2.9. Step 8: Calculate UV Maps

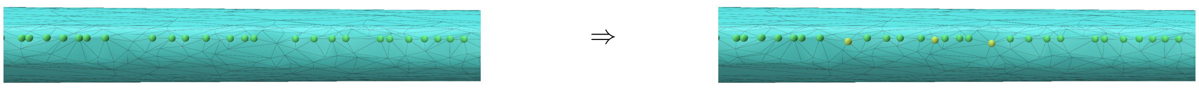
You want the boundary of the first shape to be used as the connected boundary. Therefore, for shape 1, use the connected vertices of itself, i.e., $B'_{\text{Mesh 1}}$. For the second shape, you can use the indices from $B''_{\text{Mesh 2}}$ to map to $B''_{\text{Mesh 1}}$. To goal here is to map from a vertex to a boundary vertex location in the UV map.

Hence,

$$UV_1 = \text{LSCM}(\text{Shape 1}, \text{Indices}(B'_{\text{Mesh 1}}), B'_{\text{Mesh 1}}) \tag{4.33}$$

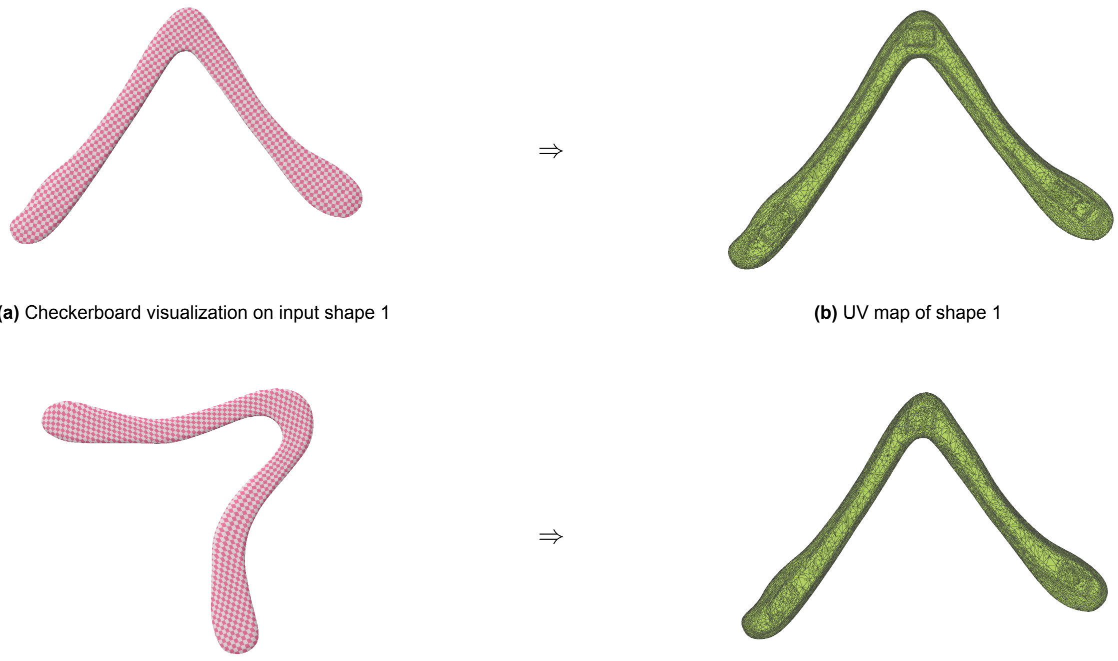
$$UV_2 = \text{LSCM}(\text{Shape 2}, \text{Indices}(B''_{\text{Mesh 2}}), B''_{\text{Mesh 1}}) \tag{4.34}$$

where $LSCM(D, E, F)$, means for Shape D and the indices E_i you map to vertices F_i.



(a) Boundary curve before improvement **(b)** Boundary curve after improvement

Figure 4.8: Before and after view of the boundary curve improvement, observe three added vertices to make all vertices neighbor-connected



(a) Checkerboard visualization on input shape 1 **(b)** UV map of shape 1

(c) Checkerboard visualization on input shape 2 **(d)** UV map of shape 2

Figure 4.9: Input meshes with their checkerboard visualization to show UV distortion, and their UV maps

By using the boundary of Shape 1 as a shared constraint, a consistent UV parameterization is ensured between the two shapes in the following steps.

Note here, if there are exact vertex-to-vertex mappings, then you might have to use both correspondences ($\circ$: Concatenation), i.e.

$$UV_2 = \text{LSCM}(\text{Shape 2}, \text{Indices}(B''_{\text{Mesh 2}} \circ B_{\text{Mesh 2}}), B''_{\text{Mesh 1}} \circ B_{\text{Mesh 1}}) \tag{4.35}$$

Figure 4.9 shows the results.

4.3. 3D Shape Correspondence

With UV maps for both shapes, a one-to-one vertex correspondence is established as follows:

1. Boundary-guided segmentation: Using the previously computed boundaries (i.e., $B'_{\text{Mesh 1}}$ and $B'_{\text{Mesh 2}}$), split each shape into two parts: top and bottom.

2. Vertex correspondence: For each vertex v_i in Shape 1:

Identify the UV coordinate $(u, v)_i$ of the vertex and determine whether it lies in the top or bottom region using the boundary information, i.e., side classification like in Figure 4.10 . The algorithm used is described in Section A.1.

(a) Shape 1 side view **(b)** Shape 1 front view **(c)** Shape 1 back view **(d)** Shape 2 side view **(e)** Shape 2 front view **(f)** Shape 2 back view

Figure 4.10: Side, front and back view for input shapes. White is one side and pink is another side.

Locate the corresponding face in Shape 2's UV map that contains $(u, v)_i$. Since the UV map contains faces for both regions of the object, first look for the face on the same side (top or bottom) as v_i in Shape 1. If $(u, v)_i$ is close to the boundary in UV space, a mismatch may occur, then also check the other side.

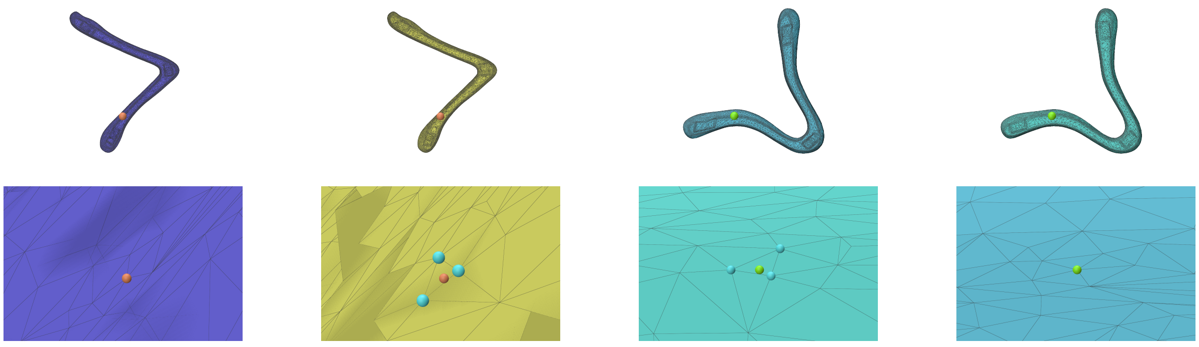
In cases where $(u, v)_i$ does not lie in any UV face of Shape 2 (which should theoretically be rare), project $(u, v)_i$ to the closest face in UV space. This provides a reasonable approximation but may result in minor inaccuracies. Such cases can usually be avoided by addressing boundary alignment issues as described in Step 4.2.8.

Compute the barycentric coordinates for $(u, v)_i$ in the corresponding face:

$$\mathbf{v}'_i = \sum_{j=1}^{3} \lambda_j \mathbf{v}_j, \quad \sum_{j=1}^{3} \lambda_j = 1 \tag{4.36}$$

where λ_j are the barycentric weights for the vertices $\mathbf{v}_j$ of the triangle in Shape 2 containing $(u, v)_i$.

The interpolated position $\mathbf{v}'_i$ in Shape 2 becomes the correspondence for $\mathbf{v}_i$ in Shape 1.



(a) $(u, v)_i$ in UV map of Shape 1, and zoomed in

(b) $(u, v)_i$ with $(u, v)_j$'s around it in UV map of input Shape 2, and zoomed in

(c) $\mathbf{v}'_i$ with $\mathbf{v}_j$'s around it of input Shape 2, and zoomed in

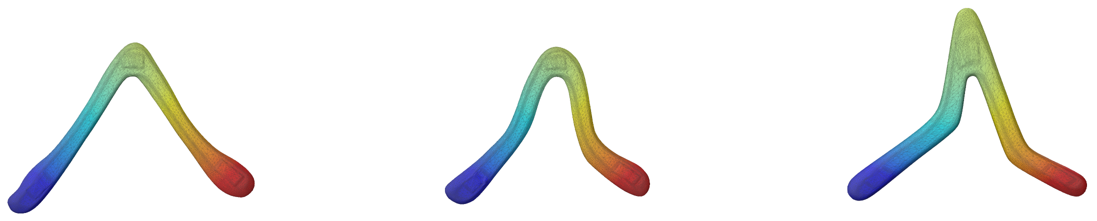
(d) $\mathbf{v}'_i$ of output Shape 2, and zoomed in

Figure 4.11: Barycentric mapping using UV maps with similar outlines.

An example of the one-to-one correspondences between the models is shown in Figure 4.12. In this visualization, a color mapping is used to highlight the matching locations, with each color representing a specific pair of points, facilitating the identification of regions with high similarity across the models. While assessing the overall quality of the correspondences can be challenging at first glance, specific regions provide more clarity. For instance, note the increased green area in Figure 4.12c compared to Figures 4.12a and 4.12b. This is justified, as the elbow in Figure 4.12c is larger, which accounts for the difference.

4.4. Generating New Shapes

Based on the dataset of corresponding shapes, a PCA model is created, serving as a template to deviate from. The detailed procedure for performing PCA is described in Section 3.4. Using this PCA



(a) Shape 1 **(b)** Shape 2 one-to-one to Shape 1 **(c)** Shape 3 one-to-one to Shape 1

Figure 4.12: One-to-one mappings between input Meshes

model, you can generate new shapes by adjusting the weight vector $\mathbf{w}$. The goal is to find the optimal weight vector such that the 2D projection of the reconstructed shape best matches a given set of outline points. These outline points are provided by the user and represent a desired shape outline. Once the optimal weight vector is found, it can be used in the 3D model for further shape generation.

4.4.1. Mean Shape Projection

The mean shape $\mathbf{g}_{\text{mean}}$ is given as a $(3N, 1)$ vector, where each triplet corresponds to a 3D point. To match this to the 2D outline points, only the x- and y-coordinates are extracted of the mean shape, effectively ignoring the z-coordinate for the 2D projection. This allows us to form a 2D representation of the shape, which is useful for matching the reconstructed shape to the user-provided outline points.

The 2D projected mean shape $\mathbf{p}_{\text{mean}}$ is represented as:

$$\mathbf{p}_{\text{mean}} = \begin{bmatrix} x_1 & y_1 & x_2 & y_2 & \dots & x_N & y_N \end{bmatrix}^T \in \mathbb{R}^{2N} \tag{4.37}$$

where N is the number of points in the shape. This vector includes only the x and y components of each point in the mean shape, arranged sequentially.

4.4.2. Eigenvector Projection

The eigenvectors $\mathbf{G}$ represent the principal components of the shape variation in 3D space. The eigenvectors are obtained after performing PCA, which models the variation in the dataset of shapes. Each eigenvector corresponds to a direction of maximum variance in the shape data.

For generating new shapes, project the eigenvectors onto the 2D plane by extracting only their x- and y-components. This allows us to use the 2D projections of the eigenvectors in the reconstruction of the 2D projected shapes. The eigenvectors are reshaped into a matrix $\mathbf{P}$ of size $2N \times k$, where k is the number of eigenvectors used.

$$\mathbf{P} = \begin{bmatrix} G_{11} & G_{12} & \dots & G_{1k} \\ G_{21} & G_{22} & \dots & G_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ G_{(2N)1} & G_{(2N)2} & \dots & G_{(2N)k} \end{bmatrix} \in \mathbb{R}^{2N \times k} \tag{4.38}$$

Here, G_{ij} represents the j-th component of the i-th eigenvector, and the matrix $\mathbf{P}$ is the 2D projection of the eigenvectors of the shape model.

4.4.3. Outline Points and Closest Matches

The outline points $\mathbf{s}$ are provided by the user and are represented as a set of 2D points. These points are flattened into a single vector $\mathbf{s}$ of size $2M$, where M is the number of outline points. This step is necessary to match the dimensionality of the outline points with the projected mean shape and the eigenvector projections.

$$\mathbf{s} = \begin{bmatrix} x'_1 & y'_1 & x'_2 & y'_2 & \dots & x'_M & y'_M \end{bmatrix}^T \in \mathbb{R}^{2M} \tag{4.39}$$

For each outline point (x'_j, y'_j), find the closest point in the projected mean shape $\mathbf{p}_{\text{mean}}$ by minimizing the squared Euclidean distance between the outline point and the mean shape points:

$$i^* = \arg\min_i \left((x_i - x'_j)^2 + (y_i - y'_j)^2\right) \tag{4.40}$$

This ensures that each outline point corresponds to the closest point in the projected mean shape.

4.4.4. Gradient Descent with Jacobian

To find the 'optimal' weight vector $\mathbf{w}$, gradient descent can be used. The weights are initialized as:

$$\mathbf{w} = \mathbf{0} \in \mathbb{R}^k \tag{4.41}$$

The residual vector $\mathbf{r}$ is computed as the difference between the outline points and the corresponding projected mean shape points:

$$\mathbf{r} = \begin{bmatrix} x'_1 - x_{i^*_1} \\ y'_1 - y_{i^*_1} \\ x'_2 - x_{i^*_2} \\ y'_2 - y_{i^*_2} \\ \vdots \\ x'_M - x_{i^*_M} \\ y'_M - y_{i^*_M} \end{bmatrix} \in \mathbb{R}^{2M} \tag{4.42}$$

Note the x_{i*} points are calculated from the latest $p_{\text{reconstruct}}$ instead of just p_{mean}.

The Jacobian matrix $\mathbf{J}$ is constructed by collecting the eigenvector components corresponding to the selected mean shape points. It represents the linear change in the projected shape with respect to the weights:

$$\mathbf{J} = \begin{bmatrix} P_{i^*_1 1} & P_{i^*_1 2} & \dots & P_{i^*_1 k} \\ P_{i^*_1 1} & P_{i^*_1 2} & \dots & P_{i^*_1 k} \\ P_{i^*_2 1} & P_{i^*_2 2} & \dots & P_{i^*_2 k} \\ P_{i^*_2 1} & P_{i^*_2 2} & \dots & P_{i^*_2 k} \\ \vdots & \vdots & \ddots & \vdots \\ P_{i^*_M 1} & P_{i^*_M 2} & \dots & P_{i^*_M k} \\ P_{i^*_M 1} & P_{i^*_M 2} & \dots & P_{i^*_M k} \end{bmatrix} \in \mathbb{R}^{2M \times k} \tag{4.43}$$

The gradient ∇E of the energy function is computed as:

$$\nabla E = \mathbf{J}^T \mathbf{r} \tag{4.44}$$

The Hessian matrix $\mathbf{H}$ is the product of the Jacobian and its transpose:

$$\mathbf{H} = \mathbf{J}^T \mathbf{J} \tag{4.45}$$

The weight update step in gradient descent is:

$$\mathbf{w} \leftarrow \mathbf{w} + \eta \mathbf{H}^{-1} \nabla E \tag{4.46}$$

where η is the learning rate.

4.4.5. Stopping Condition

Gradient descent stops when the update step is sufficiently small, indicating that the algorithm has converged to a solution:

$$\|\mathbf{H}^{-1}\nabla E\| < \epsilon \tag{4.47}$$

where ϵ is a small threshold that determines when convergence is reached.

4.4.6. Reconstruction

The final reconstructed shape is obtained by adding the weighted sum of the eigenvector projections to the projected mean shape:

$$\mathbf{p}_{\text{reconstructed}} = \mathbf{p}_{\text{mean}} + \sum_{j=1}^{k} w_j \mathbf{P}_j \tag{4.48}$$

where $\mathbf{P}_j$ is the j-th eigenvector projection. The reconstructed shape is a 2D representation that best fits the user-provided outline points.

Additionally, the shape can be reconstructed in 3D using the full PCA model as:

$$\mathbf{s}' = \mathbf{s}_i + \sum_{j=1}^{k} w_j \mathbf{u}_j \tag{4.49}$$

where $\mathbf{s}_i$ represents a shape in the original dataset, and $\mathbf{u}_j$ are the eigenvectors of the PCA model.

4.4.7. Regularization

Overfitting can occur when the parameterization adapts too closely to local variations in the shape, leading to instability or excessive deformation. To mitigate this, a regularization term can be introduced, λ, where $0 \leq \lambda \leq 1$.

Regularization helps smooth the solution by balancing the influence of the original shape and the deformation applied. A lower λ enforces stronger adherence to the initial structure, while a higher λ allows for greater flexibility in the deformation. This results in a more stable parameterization that avoids excessive distortion while preserving essential geometric features.

The regularized model is defined as:

$$\mathbf{s}'' = \mathbf{s}_i + \lambda \sum_{j=1}^{k} w_j \mathbf{u}_j \tag{4.50}$$

4.4.8. Non-Rigid ICP with Free-Form Deformation

Due to the intrinsic limitations of PCA, certain shapes are difficult to represent accurately, especially when they lie far from the mean shape or involve complex deformations. This happens because PCA relies on linear combinations of principal components, which may not capture the non-linear nature of certain deformations or the full variability of the shapes. As a result, directly applying PCA to generate these shapes might lead to unrealistic or poorly fitted results.

To overcome this, a hybrid approach is used. First, estimate the shape as closely as possible using the PCA model and the input shapes. This provides a rough approximation of the target shape in the PCA space. Then, to refine the shape and better account for local deformations, apply a Free-Form Deformation.

To align the selected points with the target shape, employ a non-rigid Iterative Closest Point (ICP) algorithm using Free-Form Deformation (FFD). This method allows for smooth, flexible deformations by representing the transformation as a weighted combination of control point displacements. Unlike rigid transformations, which preserve distances and angles, FFD provides greater flexibility by allowing localized shape adaptations while maintaining smoothness across the entire domain.

Bounding Box Computation

Before computing deformation influences, establish a bounding box that includes both the shape $\mathbf{V}$ and the selected points $\mathbf{S}$:

$$x_{\min} = \min\big(\min_i V_{i,0}, \min_j S_{j,0}\big), \quad x_{\max} = \max\big(\max_i V_{i,0}, \max_j S_{j,0}\big), \tag{4.51}$$

$$y_{\min} = \min\big(\min_i V_{i,1}, \min_j S_{j,1}\big), \quad y_{\max} = \max\big(\max_i V_{i,1}, \max_j S_{j,1}\big). \tag{4.52}$$

Since the deformation is primarily constrained to the XY- plane, a 2D bounding box is used instead of a 3D one. This choice simplifies the process by focusing on the projected view where depth variations are irrelevant, reduces unnecessary degrees of freedom by preserving structure along the z-axis, and improves efficiency by lowering computational cost and avoiding overfitting from an overly complex 3D control grid.

To avoid numerical issues when the bounding box is too small, a minimum size is enforced:

$$x_{\max} = x_{\min} + \max(1.0, x_{\max} - x_{\min}), \quad y_{\max} = y_{\min} + \max(1.0, y_{\max} - y_{\min}). \tag{4.53}$$

Why Free-Form Deformation (FFD)?

Free-Form Deformation (FFD) is selected over alternatives like Thin-Plate Splines, Radial Basis Functions or a nonlinear deformation term, due to its ability to produce smooth yet localized deformations, while keeping it simple. By using a structured control point grid, FFD offers a regular and efficient parameterization, making it well-suited for iterative optimization. This grid-based approach allows deformation to be guided locally while maintaining smooth global transitions, preserving important shape features without introducing unnecessary complexity. Moreover, the use of bilinear interpolation ensures a stable transformation that avoids the overfitting and oscillations often seen in higher-degree based methods.

FFD Influence Weights

Each vertex $\mathbf{v}_i \in \mathbf{V}$ is influenced by a set of four neighboring control points in the deformation grid. To determine these influences, map each vertex into a normalized grid space:

$$s_x = \frac{v_{i,0} - x_{\min}}{x_{\max} - x_{\min}}(N_x - 1), \quad s_y = \frac{v_{i,1} - y_{\min}}{y_{\max} - y_{\min}}(N_y - 1), \tag{4.54}$$

where N_x and N_y denote the number of control points along the x and y dimensions, respectively.

Each vertex falls within a grid cell, indexed as $(\lfloor s_x \rfloor, \lfloor s_y \rfloor)$. Let $u = s_x - \lfloor s_x \rfloor$ and $v = s_y - \lfloor s_y \rfloor$ denote the relative position within the grid cell. The four influencing control points and their bilinear weights are:

$$w_{00} = (1-u)(1-v), \qquad w_{10} = u(1-v), \tag{4.55}$$

$$w_{01} = (1-u)v, \qquad w_{11} = uv. \tag{4.56}$$

The displacement of each vertex is then computed as a weighted sum of the control point displacements:

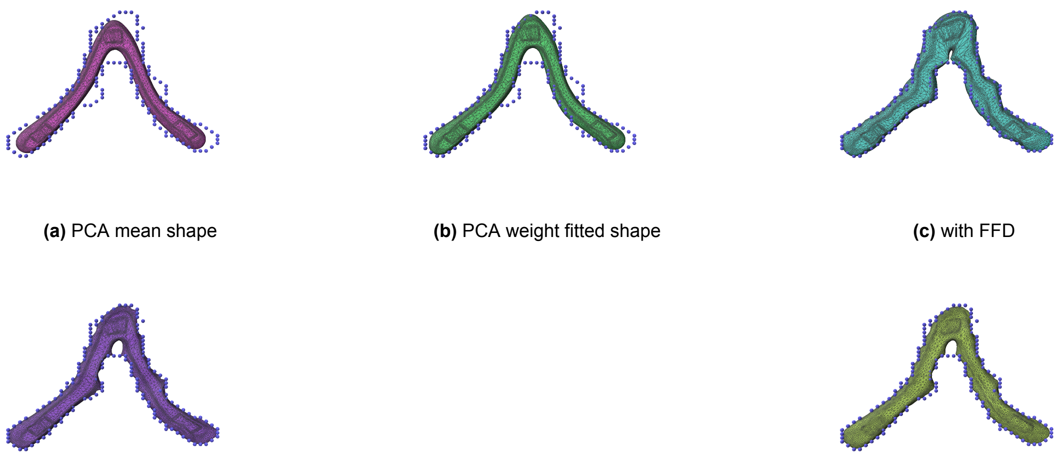
$$\Delta v_{i,0} = w_{00}\Delta x_{00} + w_{10}\Delta x_{10} + w_{01}\Delta x_{01} + w_{11}\Delta x_{11}, \tag{4.57}$$

$$\Delta v_{i,1} = w_{00}\Delta y_{00} + w_{10}\Delta y_{10} + w_{01}\Delta y_{01} + w_{11}\Delta y_{11}. \tag{4.58}$$

Thus, the deformed vertex positions are given by:

$$v'_{i,0} = v_{i,0} + \Delta v_{i,0}, \quad v'_{i,1} = v_{i,1} + \Delta v_{i,1}. \tag{4.59}$$

Result



(a) PCA mean shape **(b)** PCA weight fitted shape **(c)** with FFD

(d) with ARAP **(e)** Laplacian smoothed ARAP

Figure 4.13: Visualization of PCA-based deformations using FFD and ARAP+Laplacian.

In Figure 4.13, different stages of shape construction can be observed. Figure 4.13a represents the mean shape, as described in Section 4.4.1. Figure 4.13b shows the reconstructed shape without regularization, corresponding to the discussion in Section 4.4.6. Note that this example does not exhibit major variations; more examples can be seen in the experiments Section.

Figure 4.13c presents the result after applying Free-Form Deformation (FFD), as detailed in Section 4.4.8. Finally, Figures 4.13d and 4.13e illustrate an alternative deformation approach, which is further explored in the Experiments section (Section 6.13).

5 Evaluation

5.1. Evaluation Metrics

To assess the accuracy and quality of the shape correspondence framework, the results are evaluated using four key metrics: *area distortion*, *shear distortion*, *Hausdorff distance* and *Chamfer distance*. These metrics were selected to comprehensively measure both geometric accuracy and deformation quality, ensuring that the correspondence method preserves essential structural properties.

5.1.1. Chosen Metrics

Area Distortion Area distortion quantifies how much surface areas have changed between the input and output meshes. This is particularly important in applications where preserving local proportions is crucial. If significant area distortion occurs, the correspondence method may introduce unintended stretching or compression, altering the aerodynamic behavior of the boomerang.

Shear Distortion Shear distortion measures changes in angles within the mesh, indicating whether elements have undergone excessive shearing. This metric is critical in ensuring that deformations remain physically logical, preserving the overall structural integrity of the boomerang. High shear distortion values indicate that certain regions of the shape may have been deformed disproportionately, leading to unrealistic geometric transformations.

Hausdorff Distance The Hausdorff distance measures the worst-case deviation between two shapes by computing the largest minimum distance between points on one surface to the closest points on the other surface. This metric was chosen because it captures localized mismatches and outliers, which are particularly relevant when aligning geometries, where even small deviations can significantly change the function of a boomerang.

Chamfer Distance Chamfer distance provides a more balanced measure of overall shape similarity by computing the average squared distance between corresponding points on two surfaces. Unlike Hausdorff distance, which is sensitive to outliers, Chamfer distance considers the distribution of differences, making it useful for evaluating global shape accuracy. This is particularly relevant in ensuring that the overall geometry is well-matched while tolerating small local variations that may arise due to noise or minor surface inconsistencies.

5.1.2. Alternative Evaluation Metrics

While the chosen metrics provide a well-rounded evaluation of shape correspondence, alternative measures could also be considered to further validate and refine the approach.

Geodesic Distances Instead of measuring Euclidean distances between corresponding points, geodesic distances could be used to evaluate correspondences along the surface of the shape. This would be particularly useful in assessing whether boundary correspondences follow natural surface paths rather than direct shortest-path mappings.

Bending Energy Bending energy measures the smoothness of deformation by evaluating how much the curvature of the shape changes after transformation. This could be useful in determining whether the correspondence method introduces unnatural warping, which may be undesirable for aerodynamic analysis.

Volume Preservation While area distortion captures changes in 2D projections, volume preservation could be used to evaluate whether the 3D nature of the shape is maintained. This would be particularly relevant if future work extends the framework to boomerangs with more complex, non-flat geometries.

5.2. Distortion

To evaluate the quality of the correspondences, both **area distortion** and **shear distortion** are analyzed. These metrics measure how much the geometry of the mesh changes when mapped to another shape.

5.2.1. Area Distortion

The area distortion quantifies how much the area of each triangle changes between the original and transformed meshes. It is defined as:

$$\text{Area Distortion} = \frac{A_{\text{original}}}{A_{\text{mapped}}} \tag{5.1}$$

where:

- A_{original} is the area of a triangle in the original mesh.
- A_{mapped} is the area of the corresponding triangle in the transformed mesh.

The area of a triangle given its three vertices v_1, v_2, v_3 is computed as:

$$A = \frac{1}{2} \|(v_2 - v_1) \times (v_3 - v_1)\| \tag{5.2}$$

where $\times$ denotes the cross product.

5.2.2. Shear Distortion

Shear distortion measures how much the relative lengths of triangle edges change between the original and transformed meshes. The shear distortion for each edge is given by:

$$SH_i = \frac{L_{i,\text{mapped}}}{L_{i,\text{original}}} \tag{5.3}$$

where:

- $L_{i,\text{original}}$ is the length of edge i in the original mesh.
- $L_{i,\text{mapped}}$ is the length of edge i in the transformed mesh.

The length of an edge between two vertices v_1 and v_2 is:

$$L = \|v_2 - v_1\| \tag{5.4}$$

The **average shear distortion** per triangle is computed as the mean of the distortions across all three edges:

$$\text{Average Shear Distortion} = \frac{SH_1 + SH_2 + SH_3}{3} \tag{5.5}$$

5.2.3. Global Distortion Metrics

To assess the overall quality of correspondences, you can compute the average area and shear distortions across all triangles in the mesh:

$$\text{Average Area Distortion} = \frac{\sum_{i=1}^{N} \text{Area Distortion}_i}{N} \tag{5.6}$$

$$\text{Average Shear Distortion} = \frac{\sum_{i=1}^{N} \text{Average Shear Distortion}_i}{N} \tag{5.7}$$

where N is the total number of triangles. These metrics provide a quantitative measure of how well the correspondence preserves the geometric structure of the mesh.

5.2.4. Interpretation

The distortion values provide a quantitative measure of how well the geometry of the original mesh is preserved during the mapping process:

Area Distortion: A value of 1.0 indicates that the triangle's area is perfectly preserved (no distortion). A value greater than 1.0 indicates that the triangle in the mapped mesh is larger than its corresponding triangle in the original mesh, representing stretching. A value less than 1.0 indicates that the triangle in the mapped mesh is smaller than its corresponding triangle in the original mesh, representing compression. The closer the average area distortion is to 1.0, the better the correspondence.

Shear Distortion: A value of 1.0 for any edge indicates perfect preservation of the edge length ratio (no shear distortion). A value greater than 1.0 or less than 1.0 for an edge indicates a deviation from the original edge length ratio, representing angular or shape distortion. The average shear distortion across all edges and triangles should ideally be as close to 1.0 as possible for high-quality correspondences.

The overall metrics of Average Area Distortion and Average Shear Distortion provide a global assessment of the mapping's geometric consistency. Lower deviations from 1.0 in these metrics indicate higher fidelity in the correspondence mapping.

5.3. Distance to Original Shape

To further assess the quality of the mesh correspondences, two additional metrics are computed that measure the discrepancy between the original and transformed shapes: the Hausdorff distance and the Chamfer distance. These metrics quantify the deviation between two sets of points, providing complementary information about the worst-case and average-case differences, respectively. Note here that given two shapes the first mesh will stay the same, while the second shape will end up with the same amount of points as the first shape, thus a potential loss of information.

5.3.1. Hausdorff Distance

The **Hausdorff distance** [40] is defined as:

$$d_H(A,B) = \max\left\{\sup_{a\in A}\inf_{b\in B}\|a-b\|,\ \sup_{b\in B}\inf_{a\in A}\|a-b\|\right\}, \tag{5.8}$$

where A and B denote the sets of points on the original and transformed meshes, respectively. sup denotes the supremum operator, inf the infimum operator. The Hausdorff distance is the greatest distance from a point in one set to the closest point in the other set. It captures the worst-case deviation between the two shapes, ensuring that every point of one shape is within a certain distance of some point in the other shape. A lower Hausdorff distance implies a better overall alignment and a closer correspondence between the two meshes.

Since the computed distances are in units of the mesh (world space), it is more informative to express them as a dimensionless ratio relative to the overall size of the mesh. There are several options for normalization, but it was chosen to normalize by the diagonal length of the mesh's bounding box. That is, the normalized Hausdorff distance is given by

$$\widetilde{d}_H(A,B) = \frac{d_H(A,B)}{D} \tag{5.9}$$

Where D is the length of the diagonal of the bounding box of the mesh (e.g., the original mesh).

Let $\mathbf{b}_{\min}$ and $\mathbf{b}_{\max}$ be the minimum and maximum corner points of the axis-aligned bounding box (AABB) of the mesh. Then, the diagonal length D is given by:

$$D = \|\mathbf{b}_{\max} - \mathbf{b}_{\min}\| \tag{5.10}$$

where $\|\cdot\|$ denotes the Euclidean norm. Explicitly, in three dimensions:

$$D = \sqrt{(b^x_{\max} - b^x_{\min})^2 + (b^y_{\max} - b^y_{\min})^2 + (b^z_{\max} - b^z_{\min})^2} \tag{5.11}$$

This diagonal length is used to normalize the Hausdorff and Chamfer distances, ensuring that the values are scale-invariant and comparable across meshes of different sizes.

This normalization yields a dimensionless value that represents the error as a fraction of the overall size of the mesh. For example, a normalized distance of 0.01 indicates that the error is 1% of the mesh's bounding box diagonal.

5.3.2. Chamfer Distance

The **Chamfer distance** [45] provides a more averaged measure of discrepancy between two point sets. It is computed as:

$$d_C(A,B) = \frac{1}{|A|}\sum_{a\in A}\min_{b\in B}\|a-b\| + \frac{1}{|B|}\sum_{b\in B}\min_{a\in A}\|a-b\|, \tag{5.12}$$

where $|A|$ and $|B|$ represent the number of points in the original and transformed meshes, respectively. The Chamfer distance calculates the average of the minimum distances from points in one set to the other, thereby being less sensitive to outliers compared to the Hausdorff distance. It effectively measures the overall fidelity of the mapping by averaging the discrepancies across the entire mesh.

And, the normalized Chamfer distance:

$$\widetilde{d}_C(A,B) = \frac{d_C(A,B)}{D}. \tag{5.13}$$

5.3.3. Interpretation:

Together, these metrics provide a comprehensive assessment of the transformation:

- The Hausdorff distance focuses on the maximum deviation, highlighting the worst-case error.
- The Chamfer distance offers an average error measure, which can be more robust in the presence of a few misaligned points.

A successful correspondence mapping should yield both a low Hausdorff distance (ensuring that no part of the mesh is drastically misaligned) and a low Chamfer distance (indicating overall good agreement).

5.4. Evaluation and Analysis

Evaluations on a subset of the dataset are presented in Tables 5.1 and 5.2. In this section, some of the key findings are discussed.

5.4.1. Distance Metrics

The ideal scenario is to achieve area and shear distortion values close to 1.0, with minimal standard deviation. Similarly, Hausdorff and Chamfer distances should be as close to 0.0 as possible.

From Table 5.2, it is evident that the Hausdorff and Chamfer distances are relatively low, as expected. Since these values are measured in world space, normalized values are more informative. Specifically, all normalized Hausdorff distances remain below 3.5×10^{-3}, meaning the worst-case deviation between the initial and output meshes is less than 0.35%. Similarly, Chamfer distances are well below 8.5×10^{-5}, indicating an average misalignment of less than 0.0085%. These results suggest that the output mesh closely resembles the original.

While the Hausdorff distances highlight some localized areas with larger distortion, these values remain within acceptable boundaries. Notably, these distortions tend to occur near the lifted boundary of the mesh. One possible explanation is the displacement of vertices, as discussed in Section 4.2.7. Additionally, challenges near the boundary may stem from UV mapping inaccuracies. Matching boundary points can complicate the mesh reconstruction process, potentially misplacing vertices on either side of the shape. Consequently, the parameterization of boundary points in the UV map may be less precise.

Regardless of the cause, it is evident that points near the boundary are the most difficult to align - though the overall deviations remain acceptable.

5.4.2. Distortion Metrics

Table 5.1 presents distortion metrics. The results indicate that when the mean area distortion is below 1.0, which typically signifies compression in the mesh, the mean shear distortion tends to be higher. This correlation arises from the interplay between shape geometry and the effects of compression.

When the initial area of Shape 1 is smaller than that of Shape 2, the mean area distortion is often below 1.0, implying compression. Compression naturally reduces the shape's area but may introduce localized deformations, leading to increased shear distortion. As the shape compresses, different regions may undergo varying degrees of stretching, causing non-uniform deformation.

Conversely, when models have similar initial areas, both area and shear distortions tend to be lower. In such cases, the transformations applied to the shape are more uniform, leading to less localized variation in the mesh structure.

However, when Shape 1 has a lower area than Shape 2, the area distortion often exceeds 1.0. This suggests that the shape is being expanded to match the larger reference shape. As the area increases, localized stretching may introduce additional distortions, leading to greater shear distortion. This effect is particularly noticeable when the mesh undergoes uneven transformations.

5.4.3. Implications

These results highlight that the relationship between area and shear distortion is inherently tied to the geometric properties of the shapes involved. While general trends can be predicted, the precise amount of distortion is difficult to determine in advance due to the complex interplay of shape geometry, mesh matching, and transformation processes.

One possible approach to mitigate area distortion would be to scale shapes to match their areas before processing. However, such an approach has trade-offs. Specifically, scaling would alter intrinsic shape characteristics, such as thickness, which is often a key design feature. For instance, in aerodynamic applications, thickness directly influences lift. Thus, while scaling could reduce area distortion, it might also compromise the intended functionality of the shape.

Another factor that may have influenced these results is the selection of landmarks. For example, row five exhibits significantly worse results than other shapes. However, this shape is also the most distinct: It is larger, thicker, and has a more blunt airfoil.

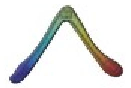
A possible contributing factor to the variation in results is the difference in the number of vertices and faces generated by the decimation process for each mesh. While some variation is expected, certain shapes showed significant discrepancies in mesh resolution. Notably, row 1 - where the input vertex counts were nearly identical - exhibited the lowest area and shear distortion. Although this may be coincidental, it suggests that maintaining similar input resolutions across meshes could improve the consistency and accuracy of the results.

While the selection of landmarks was consistent - targeting key locations such as the convex hull, tips, outer elbow, and inner elbow - the definition of the ”elbow” is somewhat flexible. In this approach, it is defined as the connecting point of the wings. However, for certain shapes (e.g., rows two, three, and four), the elbow could alternatively be interpreted as starting from the midpoint of the wings. This discrepancy may have influenced the results, particularly for larger shapes with proportionally bigger elbows. A larger elbow region increases the potential for parameterization distortions, as the affected area becomes more extensive.

5.4.4. Conclusion

In summary, the observed distortions are influenced by multiple factors, including differences in initial shape areas, compression-induced shear effects, and landmark selection. While scaling could mitigate some of these distortions, it would come at the cost of altering fundamental shape properties. Similarly, landmark placement plays a critical role, especially for shapes with significant structural variations. Additionally, inconsistencies in mesh resolution due to decimation may also impact the results - meshes with more closely matched vertex counts tended to show reduced distortion, suggesting that maintaining similar input resolutions can contribute to more stable correspondences.

These findings suggest that while certain trends in distortion behavior can be anticipated, the precise impact of shape differences remains highly context-dependent. Variations in mesh resolution, landmark placement, and initial shape properties all contribute to the complexity of the outcome. Future work could investigate alternative landmark selection strategies, more consistent mesh simplification methods, or adaptive scaling techniques that minimize distortion while preserving the essential characteristics of each shape.

Shape 1: 

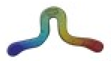
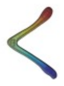
	Area Distortion Min	Area Distortion Max	Area Distortion Mean	Area Distortion StdDev	Shear Distortion Min	Shear Distortion Max	Shear Distortion Mean	Shear Distortion StdDev	Area Shape 1	Area Shape 2	Shape 2
0	0.0	1503.51	1.28	9.24	0.0	3.09	0.96	0.10	23210.8	21447.0	
1	0.0	115.96	1.09	1.66	0.0	3.49	1.02	0.11	23209.4	23344.3	
2	0.0	1371.84	0.95	7.28	0.0	9.23	1.19	0.32	23208.6	29088.9	
3	0.0	421.12	0.98	2.91	0.0	4.18	1.15	0.30	23209.6	28544.0	
4	0.0	444.34	0.98	3.01	0.0	13.26	1.13	0.33	23210.8	27342.1	
5	0.0	510.99	0.55	2.94	0.0	7.74	1.47	0.27	23211.9	47923.5	
6	0.0	307.79	0.89	1.93	0.0	4.39	1.15	0.14	23208.9	28687.4	
7	0.0	8827.27	1.32	44.99	0.0	5.67	1.03	0.16	23208.6	23731.3	
8	0.0	2207.87	1.11	11.44	0.0	2.97	1.05	0.12	23207.0	24587.3	
9	0.0	243.12	1.29	2.63	0.0	3.68	0.94	0.14	23206.3	20222.9	
10	0.0	97.63	0.92	1.13	0.0	3.60	1.13	0.13	23210.7	27291.2	
11	0.0	432.48	1.28	3.14	0.0	3.85	0.97	0.11	23208.2	19708.5	

Table 5.1: Distortion results.

Shape 1: 

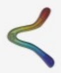
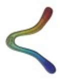
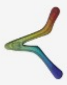
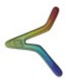
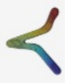
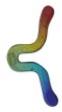
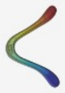
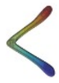
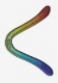
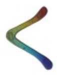
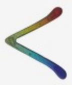
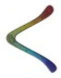
	Hausdorff Distance	Chamfer Distance	Normalized Hausdorff Distance	Normalized Chamfer Distance	Shape 2
0	0.54	0.01	1.82e-03	4.33e-05	
1	0.33	0.01	1.05e-03	4.03e-05	
2	0.85	0.02	2.37e-03	5.33e-05	
3	0.60	0.02	1.76e-03	5.25e-05	
4	0.59	0.02	1.75e-03	5.65e-05	
5	0.84	0.03	1.89e-03	6.70e-05	
6	0.48	0.02	1.28e-03	5.40e-05	
7	0.82	0.02	2.43e-03	6.12e-05	
8	0.48	0.02	1.46e-03	5.01e-05	
9	0.51	0.02	1.73e-03	6.61e-05	
10	0.38	0.02	9.98e-04	4.59e-05	
11	0.41	0.02	1.27e-03	5.67e-05	

Table 5.2: Distance results.

5.4.5. Challenges in Comparing Other Methods

While the aim was to compare the results with other open-source implementations, significant difficulties were encountered when setting up alternative projects. Although several repositories exist, most had dependencies that proved challenging to resolve. Despite extensive efforts, these limitations prevented successful benchmarking against several existing solutions.

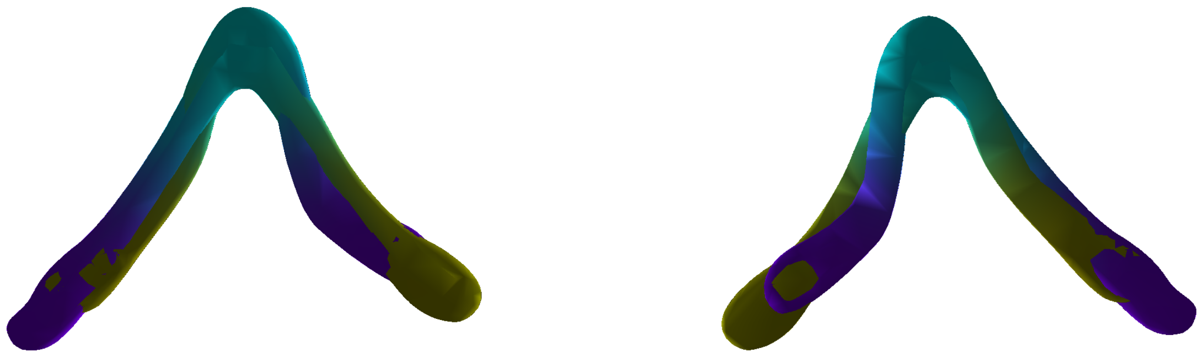
Robust 3D Shape Correspondence in the Spectral Domain

`https://github.com/cheapl/SpectralCorrespondence`

The method proposed by [22] operates in the spectral domain. The original implementation was developed in MATLAB, and the output visualizations were used directly. To ensure compatibility with the framework, the output shapes from the correspondences were decimated by an additional factor of twenty to make them processable.

Experiment: Spectral Correspondences Tthe spectral domain correspondence method was evaluated on various shapes. Figures 5.1 and 5.2 show the resulting correspondences produced by this approach. One challenge in interpreting these results is the visual overlap of the meshes, which can make the correspondences harder to assess. While the method generally captures the shape similarities well, a critical issue arises: the sides of the shapes are flipped. This results in incorrect mappings, where the front of one shape is matched with the back of another. As seen in Figures 5.2c and 5.2d, the spectral domains appear visually similar despite this flip. This phenomenon - known as intrinsic symmetry ambiguity is a well-documented challenge in computing correspondences.

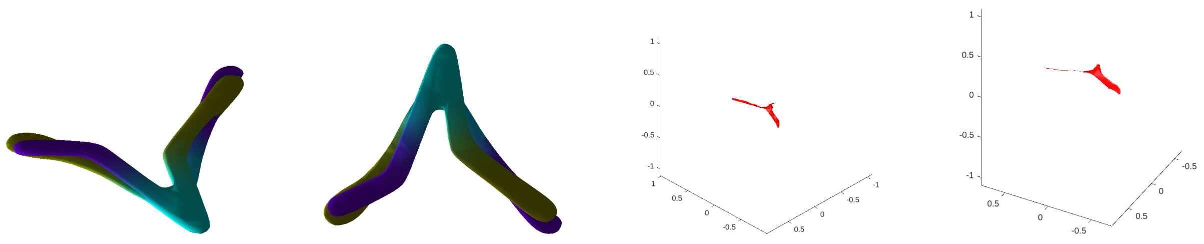
Given that the mappings consistently flip to the opposite side, a direct comparison with this method is not meaningful. Especially since the proposed pipeline from this work approach provides more reliable and intuitive correspondences.



(a) Front view of correspondence

(b) Back view of correspondence

Figure 5.1: Spectral correspondence of a pair of shapes



(a) Front view of correspondence

(b) Back view of correspondence

(c) Spectral domain of first shape

(d) Spectral domain of second shape

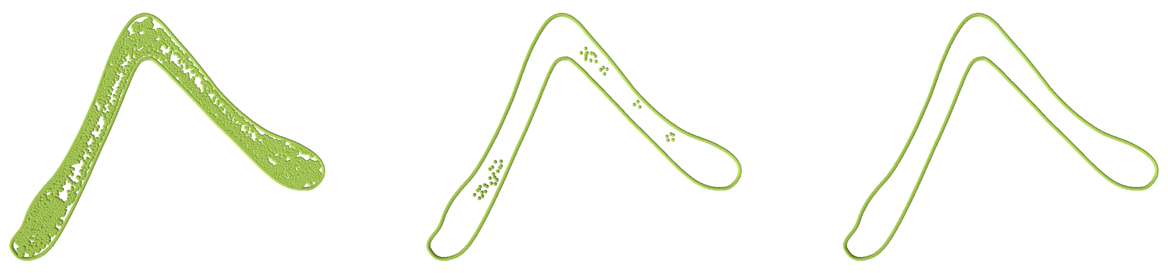
Figure 5.2: Spectral correspondence of another pair of shapes

6 Experiments

In developing a correspondence method for boomerang shapes, several experimental factors play a crucial role in shaping the accuracy and reliability of the results. These include the choice of alpha shape parameters for boundary extraction, sensitivity to landmark placement, and the handling of 2D boundary representations. Additionally, UV parameterization techniques such as LSCM and ARAP affect distortion characteristics. Finally, model behavior under PCA and Free-Form Deformation (FFD) settings determines how well shape variation and deformation are captured. Careful evaluation of these factors is essential for achieving meaningful and consistent correspondence across varying boomerang geometries.

6.1. Affect on α selection

Preferably, the resulting shape should have a smooth boundary. Since the 3D mesh if first projected to 2D before computing the alpha shape, this is not a major concern because the projection yields more points than are strictly on the boundary. However, the choice of α is crucial. The input meshes were decimated by a factor of ten, which inevitably leads to some data loss. Experiments showed that an α value of 1 worked well for the original, non-decimated meshes, but for decimated meshes, the value of α needs to be adjusted dynamically.



(a) Boomerang with $\alpha = 1$ **(b)** Boomerang with $\alpha = 8$ **(c)** Boomerang with $\alpha = 13$

Figure 6.1: Comparison of alpha selection. Higher α reduces noise inside the shape but results in the loss of boundary points.

In Figure 6.1, you can see that if α is set too low, the outline of the shape is not well defined and contains considerable noise. As α gradually increases, the points that are not on the outside boundary disappear and it will become less dense.

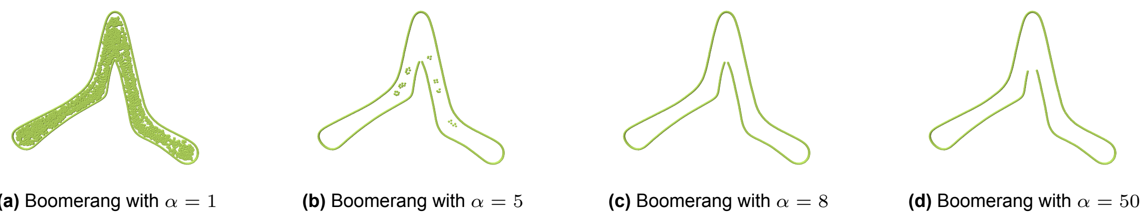

(a) Boomerang with $\alpha = 1$ **(b)** Boomerang with $\alpha = 5$ **(c)** Boomerang with $\alpha = 8$ **(d)** Boomerang with $\alpha = 50$

Figure 6.2: Comparison of alpha selection. Higher α reduces noise inside the shape, but results in boundary loss near sharp curves, such as the inner elbow.

In Figure 6.2, a similar pattern emerges, but with an additional issue: at higher α, the sharp-angled inner elbow region (as opposed to in Figure 6.1 where it is rounded) loses points on the boundary. This effect becomes evident between Figures 6.2b and 6.2c, where the elbow region is already degraded. In Figure 6.2d, the boundary loss is even more pronounced. Although the effect in Figure 6.2c is subtle, it can complicate the parameterization process in later stages, potentially leading to visually suboptimal results.

This illustrates the delicate balance required in choosing α: it must be low enough to preserve the boundary's detail without introducing excessive noise, yet high enough to ensure a complete and smooth outline of the shape.

In step 4.2.7 the projection is from the average mean. Projecting in high curvature areas may project into the wrong direction of the curve. While preferably there are no gaps such as in Figure 6.2d, it also means that there are less points to project in problematic areas. A shortest Dijkstra path traversal to get a valid 3D boundary for the boundary gap will do. This path is more jagged than a smooth traversal when there are no gaps.

In Step 4.2.7, the lifting of the 2D boundary to 3D is performed by projecting each point along the averaged normal direction. However, in regions of high curvature, this approach can become unreliable, occasionally resulting in misaligned projections or lifting in the wrong direction relative to the surface. While it is preferable to avoid boundary discontinuities-such as the gap illustrated in Figure 6.2d-the absence of points in such regions also reduces the risk of incorrect projections. To recover a valid 3D boundary across these gaps, a shortest-path traversal using Dijkstra's algorithm is applied. Although this produces a more jagged segment compared to the smooth boundary seen in well-sampled areas, it ensures completeness of the projected curve.

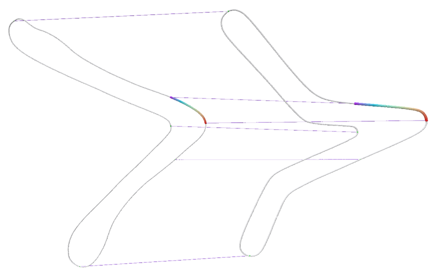
While the α parameter performs well for meshes grounded in real-world scale-such as those obtained from 3D scanners - it requires careful adjustment when applied to rescaled models or those not originating from real-world measurements. In tests using models from an external repository, the parameterization only succeeded when the meshes were first scaled to approximate real-world dimensions, highlighting the importance of consistent scaling when using α-based methods.

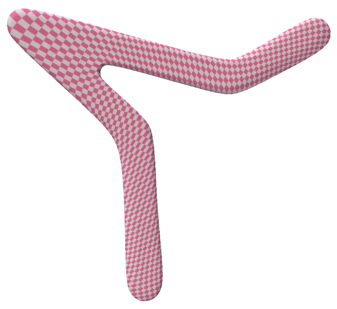
6.2. Affect on landmark selection

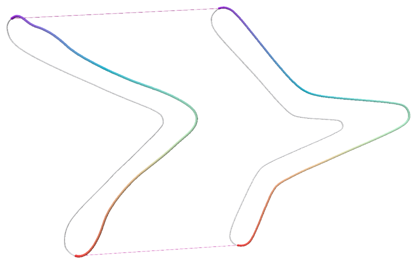
User-selected landmarks significantly impact the quality of correspondences in the non-lifted 2D boundary, and any inaccuracies at this stage propagate through subsequent steps. Since some models are larger than others-featuring wider wing chords, asymmetric wings, or variations in the elbow region-the choice of landmarks plays a crucial role in ensuring accurate UV mappings.

For instance, Fuzzies (B) exhibit a combination of these shape variations. When matching a TyphoonXL120% with a Fuzzy Phoenix, the landmarks directly influence the UV map quality, as illustrated in Figure 6.3.

In Figure 6.3, two different landmark configurations are compared. Figure 6.3c shows a naive approach where only two landmarks are set. While the rainbow mapping does not immediately reveal inconsistencies, the other input shape (similar to Figure 4.5a) has a noticeably less stretched elbow region.


(a) Five landmarks set by the user


(b) Checkerboard visualization resulting from the UV map


(c) Two landmarks set by the user

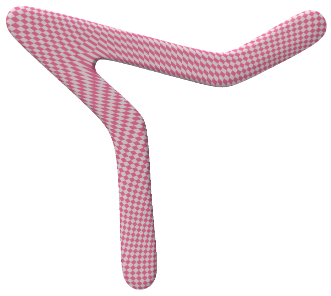

(d) Checkerboard visualization resulting from the UV map

Figure 6.3: Effect of landmark selection on the UV map quality. In 6.3d, the squares appear more stretched compared to 6.3b.

Ignoring this aspect during landmark selection results in a distorted UV map, as seen in Figure 6.3d, where the checkerboard pattern exhibits noticeable stretching.

A simple yet effective improvement is to place additional landmarks in stretched areas, as shown in Figure 6.3a, with the corresponding UV result in Figure 6.3b. While some distortion remains, the stretching is significantly reduced, leading to a more evenly distributed mapping.

6.3. 2d Boundary

There are various methods for selecting a boundary condition, and in certain shapes, the seams (which define the outer boundary) may lie in areas where the object experiences significant changes, such as a pronounced curve. In the case of boomerangs, a 2D boundary was chosen for two key reasons:

Visibility and Input Definition: Later in the process, input points need to be defined where the boomerang should change. The projection of the angle, where most of the surface is visible, facilitates this more effectively than viewing the side of the boomerang. A 2D representation allows for clearer input point definitions that are easier to manage and interpret.

Airfoil Dominance in Shape Variations: Boomerangs have a distinctive structure, with shape variations mostly occurring in the airfoils, not in the general design of the boomerang itself. By focusing on a 2D boundary, you can more easily match these variations and reduce complexity in the matching process.

However, this raises the question of why this work did not approach the problem directly in 3D, instead of mapping the shape to 2D and then projecting it back into 3D. The decision was based on the fact that working in 2D simplifies the problem and allows us to focus on the key features - primarily the airfoils - without being distracted by the more complex 3D details. Additionally, mapping the boomerang's surface to a 2D plane makes it easier to visualize and manipulate the boundary conditions before reintroducing them into the 3D model. This two-step process helps ensure accuracy while maintaining manageable complexity.

6.3.1. Alternatives for Boundary Condition Representation

While a 2D boundary condition was chosen for the reasons mentioned above, several alternative methods could have been explored:

Isolines: One alternative to using a 2D boundary is the use of isolines, which represent curves where a specific value (such as height or a scalar field) is constant across the surface. Isolines are often used in topography, but they could have been employed here to define boundaries where the scalar field (e.g., surface height or curvature) stays consistent. This would have allowed for a more flexible representation of the boundary, particularly in regions where the surface curvature changes significantly. However, isolines may not provide as intuitive or easily defined control points as a 2D projection. Additionally, the leading and trailing edges have different heights, which would be hard to navigate via isolines.

Another possibility to extract a 2D curve is using a flood fill approach instead of relying on 2D alpha shapes on the projected mesh. This method could better capture complex concave regions-such as inner elbow sections-which may be missed or oversimplified by alpha shape-based boundaries. Flood fill would adapt more dynamically to the local geometry, offering improved coverage in regions with tight curvature. However, implementing flood fill in this context can be challenging, as the spacing between points is not explicitly defined. Without a reliable notion of local connectivity or distance, the fill may 'leak' into unintended regions, resulting in inaccurate boundary extraction. boundaries.

3D Boundaries: Another potential alternative would have been to directly define the boundary condition in 3D. This approach would involve working with the boomerang's surface in its full 3D form, but it would introduce additional complexity in terms of defining control points and projecting them onto the surface. While this could provide a more detailed representation of the shape, it also risks increasing the computational burden and making the matching process more challenging. And, the initial mapping of which part of the wing should corresponds with the other wing (other shape) would have to be tackled via a similar route as in this work.

Slicing and Cross-sections: Another method might involve slicing the 3D object into a series of cross-sections, each representing a 2D slice of the object at a specific height or depth. These cross-sections could be treated individually as 2D boundaries, with their relationship to one another captured through the slicing plane. This approach is similar to the one used, but it focuses on different 2D representations from multiple planes of the object. While this approach might offer more precision in capturing the object's internal structure, it could also result in a loss of global context and require more complex algorithms to stitch the 2D slices together.

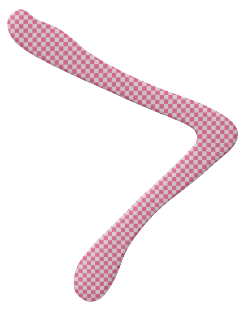
An alternative approach that was initially tested was slicing the boomerang at the cross-section with the maximum area. This method aimed to capture a representative 2D boundary while preserving the 3D structure. Given that boomerangs are generally flat, this approach seemed promising, as it would directly provide the 3D boundary points, eliminating the need for additional projection steps. However, one major drawback became apparent: while most boomerangs are relatively flat, many incorporate minor twists or tuning adjustments, cambering or aggressive airfoils (e.g. large undercuts). These subtle variations affect the overall shape, and relying on a single maximum-area slice would mean losing control over precisely where the boundary is defined. For example, consider a flat boomerang with no undercuts - meaning the lower part of the airfoil is blunt rather than shaped. In this case, the maximum-area slice could lie somewhere between the lower surface of the boomerang and the point where the trailing edge changes its angle. If this were used as the reference boundary, attempting to match it with a boomerang that does have undercuts (where the airfoil extends beneath the main body) would lead to inconsistencies. Specifically, the 3D boundary points would not align properly, causing mismatches when establishing correspondences between different boomerangs. While this method offered the advantage of reducing processing steps, the loss of boundary control and the potential for mismatches made it unsuitable for the intended purposes.

6.4. Parameterization

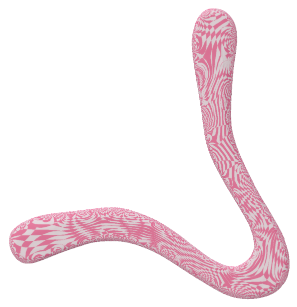
While, several options could have been used, it was decided to use a single patch, fixed boundary approach. One of the downsides of that is these algorithms usually produce high-distortion maps due to the fixed boundary. The goal is to use a fixed boundary while minimizing distortion.

One example of a fixed-boundary, single-patch algorithm known for its smooth interpolation and absence of internal crossings is harmonic mapping. However, its effectiveness diminishes when constrained by a fixed boundary - such as the projected outlines used in the approach - particularly if the shape's intrinsic geometry deviates significantly from the prescribed boundary.

This issue is particularly evident in Figure 6.4, where you can observe a clear distinction in how the checkerboard pattern is mapped, using harmonic paramaterization:



(a) Harmonic parameterization of Shape 1

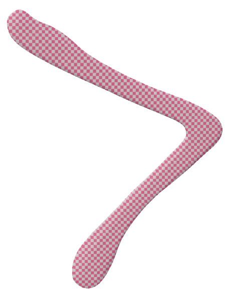


(b) Harmonic parameterization of Shape 2 with Shape 1's boundary

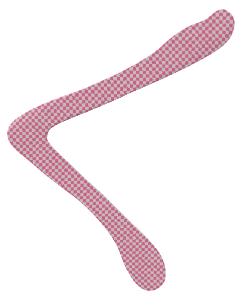
Figure 6.4: Comparison of harmonic parameterization on two different shapes. While Shape 1 maintains a well-structured checkerboard pattern, Shape 2 suffers from severe distortions when constrained by Shape 1's boundary.

Least Squares Conformal Mapping (LSCM) is well-suited for the given case. It does not require a fixed boundary, operates as a single patch, and allows for self-intersecting borders, which is acceptable since you can track their locations on the original mesh. Therefore, providing an input boundary remains a viable option.

Another free-boundary alternative is As Rigid As Possible (ARAP) parameterization, initialized with harmonic parameterization. When using the same boundary as LSCM, ARAP produces ok results for both Shape 1 and Shape 2, however, only on the front side of the boomerang. Distortions appear on the back sides, as shown in Figure 6.5.



(a) ARAP parameterization of Shape 1, front side



(b) ARAP parameterization of Shape 1 back side

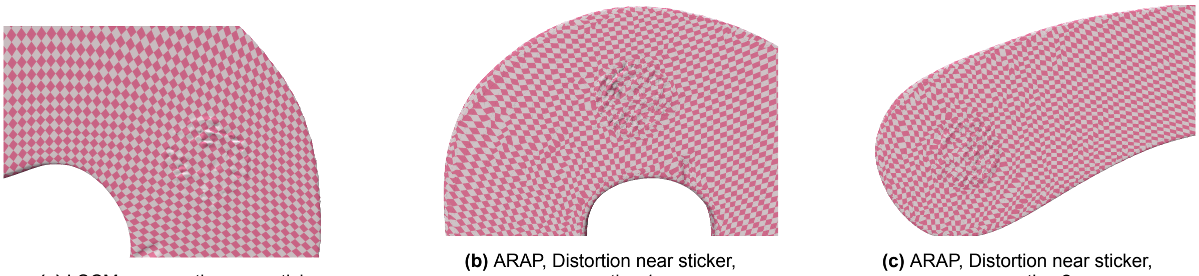
Figure 6.5: Front and back side of UV parameterization using ARAP with a fixed boundary. Observe the front side has a clear checkerboard pattern, while the back has distortion.

While 3D scans are ideally expected to be flawless and independent of external factors, real-world scanning conditions introduce challenges. The equipment performed well overall, but certain surfaces posed difficulties-particularly reflective surfaces and thin airfoils.

6.5. Affect on 3D Scan quality when Parameterizing

When parameterizing with LSCM, challenging surfaces-such as reflective stickers-are still handled correctly, even though the triangulation near these areas fluctuates. Despite this, LSCM effectively preserves the overall shape.

In contrast, ARAP exhibits noticeable distortions, not only near the sticker but also in surrounding areas. This suggests that ARAP struggles to maintain local rigidity, whereas LSCM provides a more stable mapping, even in the presence of surface variations.



(a) LSCM, perspective near sticker

(b) ARAP, Distortion near sticker, perspective 1

(c) ARAP, Distortion near sticker, perspective 2

Figure 6.6: A checkerboard pattern visualized after parametrization on Shape 2. ARAP has trouble handling near a reflective sticker, while LSCM performs well

6.6. Boundary Shape

In the suggested approach, one of the 3D shapes is projected onto a 2D plane to serve as a UV map. However, an alternative is to let Least Squares Conformal Mapping (LSCM) determine the optimal boundary shape instead of enforcing a predefined structure. This raises a critical question: How does constraining the UV boundary to a specific shape affect the overall mapping quality?

(a) Checkerboard texture mapped onto the model using a circular UV domain

(b) The circular UV parameterization

Figure 6.7: A visualization of a 3D shape mapped to a circular UV domain. Observe the distortions in the checkerboard pattern due to boundary constraints.

In Figure 6.7a, a checkerboard pattern is visualized on the 3D model when its UV map is constrained to a circular boundary, as shown in Figure 6.7b. The circular parameterization is a common choice due to its well-defined boundary constraints and ease of implementation. However, this approach introduces distortions, particularly in regions where the original shape deviates significantly from a disk-like structure.

A key observation is that the checkerboard pattern in Figure 6.7a exhibits noticeable stretching and shearing. This is a direct consequence of forcing the model's boundary to conform to a perfect circle, disregarding its intrinsic geometric properties. While the circular mapping preserves continuity, it sacrifices local conformality, leading to uneven distortions across the surface.

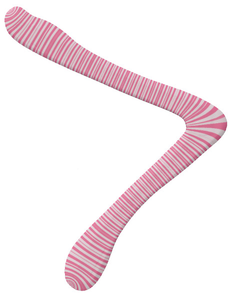


Figure 6.8: Checkerboard pattern under an unconstrained LSCM mapping with only two fixed boundary points

An alternative approach is to relax boundary constraints and allow LSCM to determine the UV layout dynamically. Instead of enforcing a specific shape, provide only the minimal necessary constraints-two fixed boundary points-and let the algorithm compute an optimal mapping. However, as seen in Figure 6.8, this results in a highly distorted parameterization where the structure of the original shape is lost entirely. The checkerboard pattern, which should retain uniform spacing, is completely deformed, indicating severe non-uniform scaling and warping. Similar findings were found for harmonic and ARAP without an input boundary.

Another important observation is that LSCM exhibits instability when boundary vertices are duplicated at the same location. For example, if in step 4.2.7, indices reference multiple vertices that occupy the same position in 2D space, LSCM struggles to compute a well-behaved parameterization. While the overall UV map may appear reasonable, these duplicate boundary constraints cause certain regions to collapse to a single point. As a result, the solver places those points at extreme distances, leading to large spikes in the UV map and introducing severe noise.

6.7. Boundary smoothness

In Figure 6.9, two UV maps are shown-one with and one without the boundary correction from Section 4.2.8. Notice that in Figure 6.9b, the UV map exhibits jagged edges, whereas in Figure 6.9a, the boundary is smooth and well-connected.

This difference arises from how Least Squares Conformal Mapping (LSCM) handles the input boundary. While LSCM does not explicitly require a structured boundary, it relies on fixed boundary vertices to compute an accurate UV parameterization. In cases where the boundary appears jagged, certain key boundary points were missing from the constraint set, leading to distortions. To address this, refine the boundary by including all intermediate points between existing boundary vertices, ensuring a continuous and properly connected edge structure.

(a) UV map with boundary correction **(b)** UV map without boundary correction

Figure 6.9: Comparison of UV maps with and without boundary correction, as described in Section 4.2.8.

While the UV maps without guaranteed edge-connections yield suboptimal results, the discrepancies are primarily noticeable only upon close inspection of the shape itself. Specifically, the boundary regions of the boomerangs experience the worst distortions. This is evident in the final outcome, where a one-to-one correspondence is established between shape 1 and shape 2. In Figure 6.10, shape 2 is displayed with this correspondence, showing that the UV map without boundary correction leads to noticeable jaggedness in the 3D model, as seen in 6.14b. This is undesirable, as smoothness is preferred for consistency. In contrast, 6.10a shows the result from the improved boundary, which remains smooth and free of distortions.

Figure 6.11 illustrates another issue: shape 2's UV map struggles to fit the points from shape 1, which are contained within shape 1's UV map. As a result, the points fall outside of shape 2's UV map, complicating the mapping process. To address this, projecting the points onto the UV map provides an effective estimate, though this is not an ideal solution.

This estimation approach is measured by the results in Table 6.1 and one can observe that the Mean Area Distortion and Shear Distortion values are closer to 1.0, indicating improved consistency, and the Distance Metrics are reduced, approaching 0.0 - an ideal scenario for minimizing distortion. Though,

(a) One-to-one mapping from shape 2 to shape 1 using the corrected UV map (6.9a).

(b) One-to-one mapping from shape 2 to shape 1 using the uncorrected UV map (6.9b).

Figure 6.10: Comparison of shape correspondence results (in 3D) using UV maps with and without boundary correction. The improved UV map produces a more accurate one-to-one mapping.

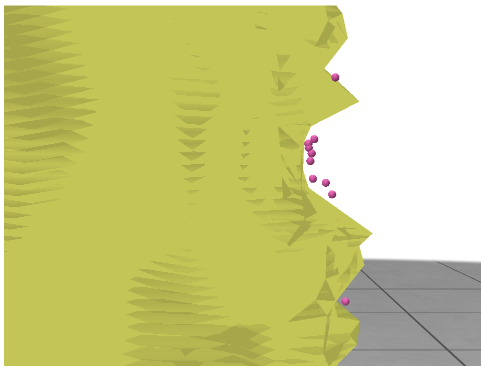


Figure 6.11: UV map of shape 2 without boundary correction, points from shape 1 that fit in the UV map of shape 1. Observe that the points are outside the UV map.

visually this can be best confirmed for correctness.

6.8. Lifting Curve

In this work, the 2D boundary points is placed in a plane at the average thickness of the boomerang, which corresponds roughly to the midpoint between the airfoils. While other options could have been explored, this approach worked well because each airfoil is unique. Given that the lowest point of the trailing edge slope is typically lower than the leading edge. To account for this, project the mean lifted 2D coordinate onto the closest point on the surface near the trailing edge. This guarantees that the point will be lower than the mean, as the trailing edge slopes downward. The hypothesis is that this point will coincide with the location where the slope angle changes most significantly in the trailing edge, which has proven to be the case in tests.

The same logic applies to the leading edge, though with more uncertainty, as this part of the airfoil is typically more blunt and has less of a defined slope compared to the trailing edge. However, through experimentation, it was found that the projected point tends to fall somewhere between the invincible mean line and the lower part of the sloped edge, which provides sufficiently accurate results.

While this method works well, exploring alternative approaches could be considered in the future. Additionally, this step inherently forces the boomerang to be flat, though ways to make this process more flexible were considered, to account for certain parameters like the angle of attack or adjustments for anhedral/dihedral angles. These improvements are considered possible, but it is suggested they be explored in future work.

An alternative to using the mean might involve taking the mean over a neighborhood of points or using a cross-section of the airfoil at that location. This could help account for variations in height, manufacturing inaccuracies, or lower-quality 3D scans.

Table 6.1: Comparison of UV Mapping Distortions with and without Boundary Improvement - for TyphoonXL120% and 3d3

Metric	Without Improvement	With Improvement
Area Distortion		
Min	0.00	0.00
Max	381.01	599.67
Mean	1.13	1.11
StdDev	3.07	3.76
Shear Distortion		
Min	0.00	0.12
Max	4.43	3.41
Mean	1.03	1.02
StdDev	0.15	0.12
Distance Metrics		
Hausdorff Distance	0.43	0.40
Chamfer Distance	0.01	0.01
Normalized Hausdorff	1.35e-03	1.24e-03
Normalized Chamfer	4.40e-05	4.08e-05

6.8.1. Projecting Lifted Curve

When the lifted curve is projected back onto the surface (see Section 4.2.7), the closest vertices are adjusted (6.12a), which can lead to some faces overlapping if vertices belong to multiple faces.

While the current approach retained this method - since the final correspondences yielded statistically better results and potential overlaps can be undone in post-processing if desired - several ways to mitigate this issue were explored.

One option that worked, but gave a jagged boundary is in 6.12b, where the projected point is shifted toward the closest vertex. Note that this point can also be shifted several times if there are several projected points closest to it.

A logical approach would be to avoid shuffling vertices and instead split faces where the projected points lie, i.e Figure 6.12c. This method proved effective, however a decrease in accuracy due to an increased number of faces was observed, which in turn made the UV maps denser near the boundaries (where they were already denser compared to regions such as the midpoint of the wings).

A potential improvement to this approach could involve modifying a step in the pipeline: rather than using the in-between shape to determine UV correspondences, use the original input shape while keeping track of the split faces to backtrack their original positions. This adjustment should be feasible since each triangle splits into three new faces that remain within the same plane.

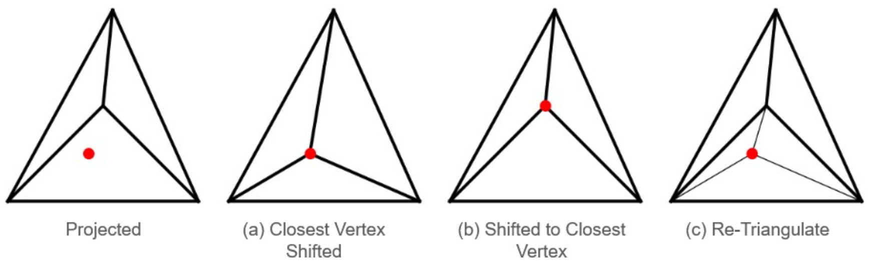
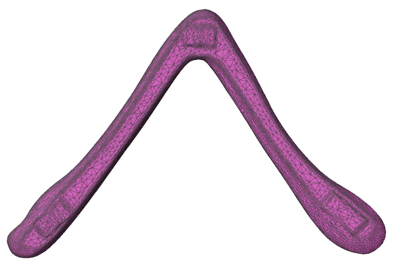


Figure 6.12: Alternatives when lifting Curve

6.9. Effect on Base shape

To ensure consistency when mapping to different shapes, it is crucial to select a single base shape. Ideally, the base shape should be simple and not predisposed to excessive stretching in specific regions.


(a) Shape 1

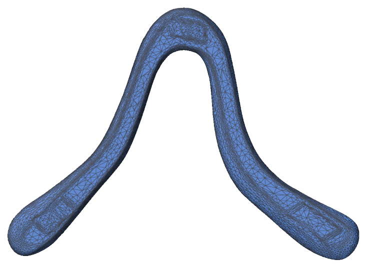

(b) Shape 2

Figure 6.13: Input shapes

Shapes that inherently stretch more than others will introduce bias, as most mappings would require shrinking rather than balanced transformations.

Evaluating the Results

The results do not definitively suggest an optimal base shape, but the goal is to minimize mean area distortion while ensuring that:

- Shear distortion remains close to **1.0**, as this should be enforced by Least Squares Conformal Mapping (LSCM)
- Hausdorff and Chamfer distances remain as low as possible, ensuring minimal deviation between mapped shapes.

Metric	Shape 1 → Shape 2	Shape 2 → Shape 1	Winner
Area Distortion			
Min	7.73e-04	5.89e-04	–
Max	115.96	536.55	–
Mean	1.09	1.13	Shape 1
StdDev	1.66	4.01	Shape 1
Shear Distortion			
Min	0.00	0.23	–
Max	3.49	3.67	–
Mean	1.02	1.01	Shape 2
StdDev	0.11	0.12	Shape 1
Area			
Object 1	2.32e+04	2.34e+04	–
Object 2	2.33e+04	2.32e+04	–
Distances			
Hausdorff Distance	0.33	0.43	Shape 1
Chamfer Distance	0.01	0.01	Shape 1
Normalized Hausdorff Distance	1.05e-03	1.23e-03	Shape 1
Normalized Chamfer Distance	4.03e-05	4.00e-05	Shape 2

Table 6.2: Comparison of mapping metrics between Shape 1 and Shape 2

Conclusion

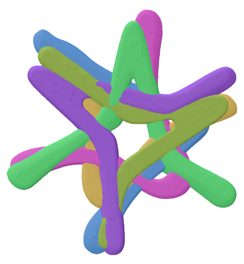
While the results in Table 6.2are close, **Shape 1 is the 'better' choice** for a base shape. It exhibits:

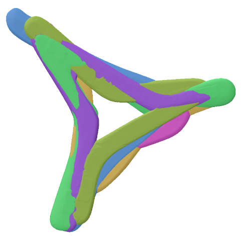
- Lower mean and standard deviation in area distortion, reducing unwanted stretching/shrinking.
- Lower Hausdorff and Chamfer distances, meaning closer shape matching.
- Slightly higher shear distortion mean, but within an acceptable range (close to 1.0).

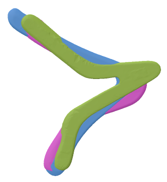
Given these factors, Shape 1 is the recommended base shape for consistent mapping. While this analysis is based on two shapes, it can be generalized to multiple shapes depending on your specific shape goals.

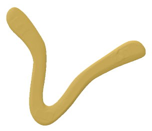
It's worth noting that the results from both input orders are visually similar, highlighting the potential influence of landmark selection. Consequently, while the statistics favor Shape 1 as a more reliable option, one cannot definitively state that it is inherently ”better” than the other. In this example, the statistical data strongly suggest that Shape 1 is the more dependable choice, but the final decision should also consider other factors such as application-specific requirements and the context of the shape data.

6.10. Alignment With(out) ICP


(a) Input shapes. Same mean, rotationally unaligned.


(b) Shapes after ICP.


(c) Shapes after ICP: Subset 1.


(d) Shapes after ICP: Subset 2.

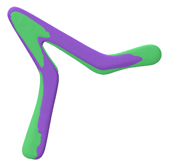
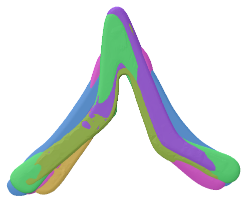

(e) Shapes after ICP: Subset 3.

Figure 6.14: ICP applied to shapes with only the mean location in common, rotationally unaligned. Note that the final result is not perfectly rotation-aligned, showing three possible perspectives of alignment.

Although pre-alignment (before PCA) might seem redundant, it is essential. Certain boomerangs possess features that make alignment more challenging. For example, some shapes (like fuzzies) have both wings shaped in a way that either can be interpreted as a three-blader, making automatic alignment ambiguous. Without proper pre-alignment, distinctive features such as the elbow may be misidentified as the leading or trailing arm, leading to incorrect orientation and affecting downstream processing. Figures 6.14 and 6.15 illustrate why manual rotational alignment is necessary in such cases.


(a) Input shapes. Same mean, manually rotationally aligned.

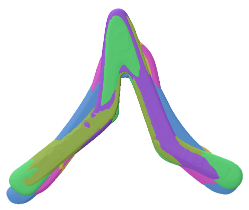

(b) Shapes after ICP.

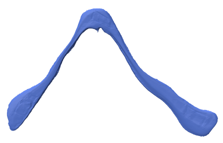
Figure 6.15: ICP performed on shapes where only the mean location is common and the shapes are manually rotationally aligned as a starting point. All boomerangs are now well aligned.

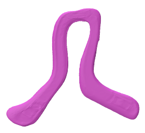
6.11. Affect of missing Boundary points in overall PCA

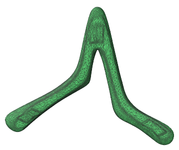
As discussed in Section 6.1, the choice of α is crucial. Although row two in Table 5.1 does not indicate any extreme irregularities, it exhibits a relatively high mean shear distortion, which could be due to inaccurate mappings. While it is known that the selection of α directly impacts the inner elbow regions, the selection of landmarks is believed to play a significant role in this issue.

Due to the chosen α, even when optimally selected, the border remains disconnected. As a result, the projection between the two points where the disconnect begins introduces estimation errors during reprojection onto the surface. This leads to a flawed reconstruction of the lifted curve in the inner elbow region.

Figure 6.16 illustrates that a shape with a sharp inner elbow angle exhibits a less desirable triangulation after a one-to-one mapping to Shape 1. When stretching a boomerang in PCA space, an unusual peak - visible in Figure 6.16a - often appears in the inner elbow, further highlighting these distortions. When leaving out this model in the PCA model, those noisy peaks disappear near that location.


(a) Shape deformed including the Shape in Figure 6.16c in PCA model


(b) Shape deformed excluding the Shape in Figure 6.16c in PCA model


(c) Boomerang with inner elbow triangulation issues

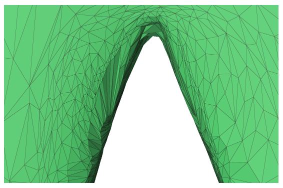

(d) Figure 6.16c zoomed in at inner elbow

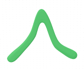
Figure 6.16: Boomerang deformations in PCA space, with and without a model that issues in the inner elbow

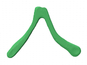
6.12. Overfitting PCA

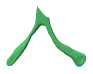
When using Principal Component Analysis (PCA) to model shape variation, it is important to consider the risk of overfitting - especially when working with a limited number of training examples or when the model is used for tasks like correspondence or deformation. Overfitting can lead to unrealistic shape reconstructions, reducing generalization to new or unseen shapes. In this section, a brief discussion is provided on how overfitting may arise in the given setting.

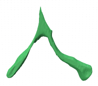
6.12.1. Overfitting with Two-Point Constraints: The Need for Regularization

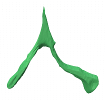
To illustrate how PCA attempts to reconstruct a given shape, two points near the wing tips were selected as constraints. Figure 6.17 shows the resulting deformations in PCA space as the regularization parameter λ varies from 0 to 1 in increments of 0.2. As λ increases, the shape becomes increasingly overfitted, losing its characteristic boomerang structure. This highlights a key limitation of the PCA space: while it optimizes for the best fit, it can fail to preserve structural integrity when regularization is insufficient.


(a) $\lambda = 0$


(b) $\lambda = 0.2$


(c) $\lambda = 0.4$


(d) $\lambda = 0.6$


(e) $\lambda = 0.8$

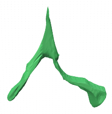

(f) $\lambda = 1$

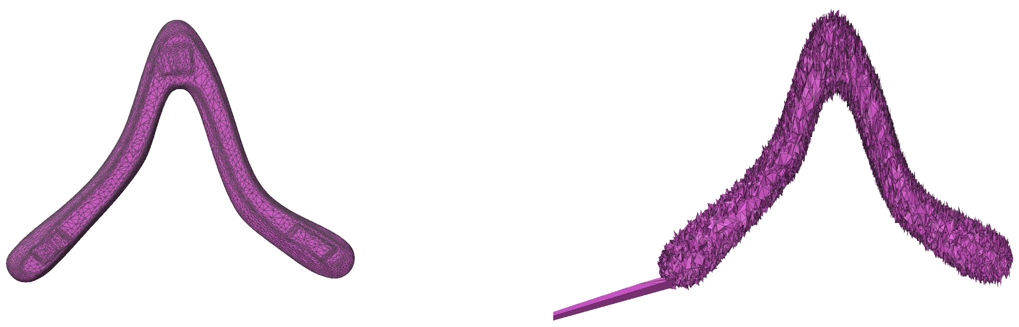
Figure 6.17: Deformations in PCA space with varying regularization. less regularization leads to overfitting, causing the shape to lose its structural integrity.

A similar experiment could have been conducted by selecting a third point near the elbow. However, another limitation arises due to the recalculation of closest points: as the shape deforms, it can shift in such a way that a different internal vertex is selected as the closest point. This allows the model to achieve an ”optimal weight” in PCA space while no longer maintaining correspondence with the original boundary. To mitigate this issue, it is sometimes preferable to keep the original vertex indices and minimize distances based on the fixed set of points rather than recalculating them dynamically.

6.12.2. Overfitting in PCA and the Role of Principal Components

As the number of input shapes increases, the number of principal components grows accordingly. This can lead to overfitting, where higher-order principal components receive disproportionately high weights. Notably, the first principal components capture the most variance in the data and should be prioritized, while the later components often represent noise or minor variations.

Figure 6.18 illustrates this effect. The mean shape (Figure 6.18a) appears smooth and undistorted, while applying only the last principal component (Figure 6.18b) introduces significant deformations, highlighting the risk of overfitting when excessive weight is assigned to less significant components.



(a) Mean shape without deformation

(b) Shape deformed using only the last principal component

Figure 6.18: Effect of overfitting in PCA. The mean shape remains well-defined, while applying only the last principal component results in unrealistic deformations.

6.12.3. Effect of Less Standard Airfoils in PCA

Certain boomerangs feature cambering, where material is removed from the bottom of the wing in a way that is not connected to one of the sides (otherwise, it would be classified as an undercut). While most shapes in the dataset do not exhibit this characteristic, cambering can still be partially captured by the PCA model as a feature.

The expected effect of this in the PCA space is a transition from a regular shape to one with cambering, or vice versa, where material is effectively added to the bottom of the wing. However, an interesting phenomenon was observed: in some cases, PCA-generated shapes exhibited cambering on the top of the wing instead, particularly in areas where cambering is naturally present in the dataset on the other side of the wing. This effect, though subtle, is visible near the wing tips in Figure 6.19.

A more extreme case occurs when certain principal components are weighted too heavily, either positively or negatively. Under these conditions, the reconstructed boomerang may wrap through itself, producing an unrealistic geometry - but if gone too far potentially realistic. Notably, this behavior was only present in cases where cambering was involved, suggesting that the PCA model struggles to generalize such non-standard airfoil features effectively.

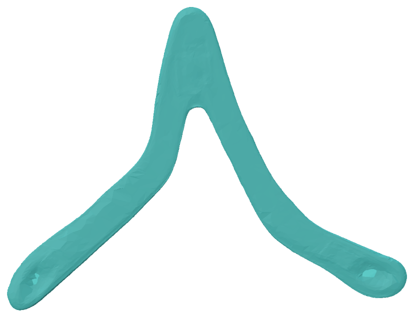


Figure 6.19: Effect of cambering in PCA. When one principal component is weighted too highly (either positive or negative), the object deforms unrealistically, wrapping through itself. Top View.

6.12.4. PCA Amount of Principal components

Figure 6.20 provides insight into how shape variations are distributed across different principal components. The mean shape acts as the foundation, capturing the essence of the dataset before any transformations occur. From there, each principal component reveals a unique mode of deformation, gradually introducing more intricate details.

The first few components drive the most noticeable changes, altering global structure-stretching, widening, or bending the shape in more predictable ways. As you move further down the hierarchy, the higher-order components refine these variations, encoding more localized transformations. This balance between broad structural shifts and fine-grained adjustments is what makes PCA so effective for understanding shape variability.

Another important aspect is the interplay between symmetric and asymmetric changes. Some components introduce uniform distortions, preserving overall balance, while others create asymmetries that push the shape in distinctly different directions. This behavior highlights the complexity of how structures can vary within a dataset.

Ultimately, the number of components retained determines the level of detail and generalization in any reconstruction. Keeping only the most influential components preserves the essential structure while eliminating noise. However, including too many can lead to overfitting, capturing irrelevant variations rather than meaningful patterns.

In the illustration, you can observe that the first three principal components show that the general structure of the boomerang moves smoothly. While the fourth until the sixth show more unpredictable movements, while the seventh until the eleventh show unpredictable changes and make the boomerang unbalanced and the changes are not symmetrically distributed across both wings. E.g. part of the wings turning inwards or outwards. The last principal component is noisy with only local changes.

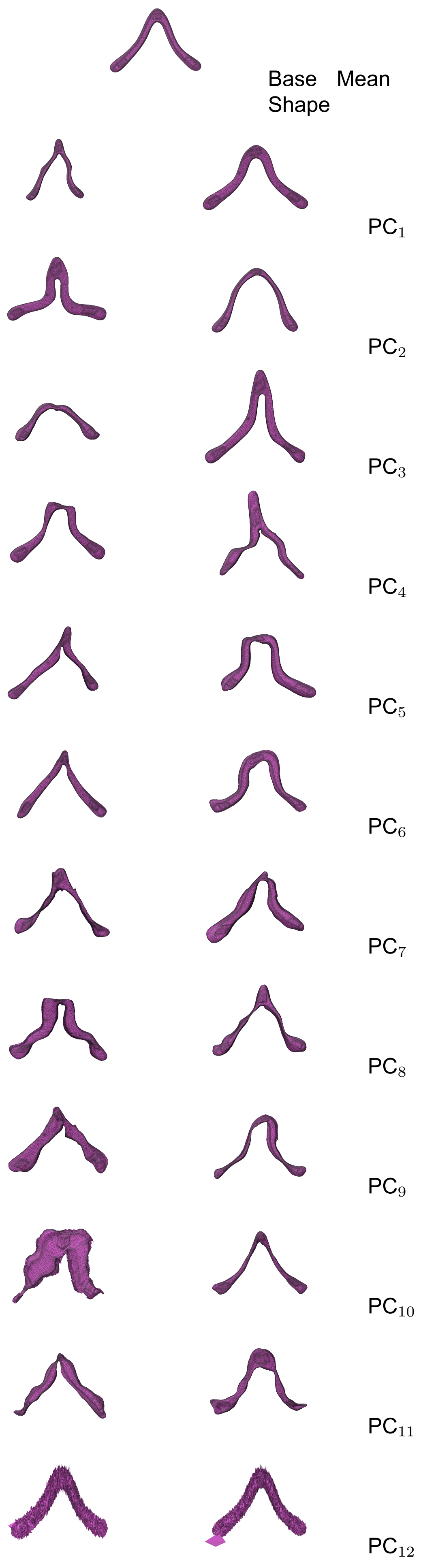


Figure 6.20: Visualization of PCA components. The base shape is at the top, with negative and positive variations side by side.

6.13. Deformation term

An alternative approach explored in place of the current bilinear Grid-based FFD involved using As-Rigid-As-Possible (ARAP) deformation. In this method, the closest vertices to the control points were directly manipulated, and Laplacian smoothing was applied to reduce sharp peaks and ensure smoother transitions across the surface.

ARAP

To refine the shape and achieve a more accurate fit to the target shape, adjust the coordinates of the points $(x_{i_1^*}, y_{i_1^*})$ to (x'_1, y'_1), $(x_{i_2^*}, y_{i_2^*})$ to (x'_2, y'_2), and so on, for each of the selected points. Mathematically, this can be represented as:

$$(x_{i_j^*}, y_{i_j^*}) \to (x'_j, y'_j) \quad \text{for} \quad j = 1, 2, \ldots, M \tag{6.1}$$

where x'_j and y'_j denote the boundary input coordinates of the point corresponding to the surface points that are closest to i_j^* in the PCA space. These adjustments are made to minimize the residuals between the PCA-based approximation and the target shape.

Smoothing

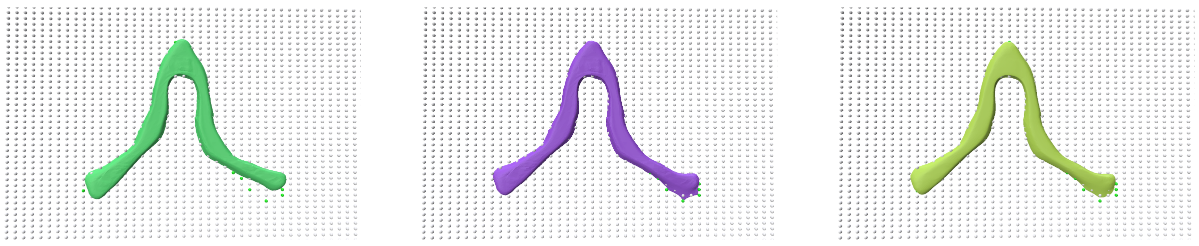
Since ARAP (As-Rigid-As-Possible) can produce spiky or overly deformed solutions when the points are not sufficiently close to each other, apply Laplacian smoothing afterward. This smoothing step helps to mitigate excessive deformations by enforcing local smoothness, ensuring that the points are not moved too drastically in relation to their neighbors. Laplacian smoothing redistributes the vertex positions by averaging the positions of neighboring vertices, which results in a smoother, more stable deformation that better conforms to the desired shape.

While Laplacian smoothing is beneficial for stabilizing the deformation and reducing spikiness, it also means that the smoothed points no longer lie exactly on the outline points. This trade-off occurs because the smoothing process alters the vertex positions to achieve a smoother, more cohesive shape, which may slightly deviate from the original outline. However, this deviation is often necessary to ensure that the overall shape maintains a natural, stable form without excessive distortions.

Results

While the results could work, the downside of this approach is that it is a local deformation instead of a global deformation. In other words, instead of stretching out the whole object to fit the target points, it only moves a small area toward a point. And, because the results look so spiky a smoothing term is needed. The smoothing term on the other hand could be handy as a regularization term.

An example is shown in Figure 6.21. Points near tips and outer elbow, and additional points near the right wing. Observe the local spiky behavior in 6.21b that gets smoothed out in 6.21c.



(a) Optimized Weights **(b)** Applying ARAP **(c)** Laplacian Smoothing

Figure 6.21: Results when using ARAP and Laplacian smoothing after PCA with optimized weights.

Another reason for choosing Bilinear Free-Form Deformation (FFD) over ARAP+Laplacian can be observed in Figure 4.13. Figures 4.13c and 4.13e appear visually similar; however, the FFD technique results in a more uniformly stretched airfoil, while the ARAP+Laplacian method primarily stretches the leading and trailing edges, keeping the main body relatively static.

This difference leads to an undesirable thinning effect in the ARAP+Laplacian approach, altering the airfoil's overall shape more significantly. Since preserving the structural integrity of the airfoil is crucial, FFD provides a more controlled and desirable deformation for the application.

While thinning of the airfoil might be design choice, a more challenging issue with ARAP arises when multiple target points are aligned perpendicularly in a straight line toward the deforming shape. In such cases, ARAP can become unstable, as illustrated in Figure 6.22, particularly in subfigure 6.22c. When this limitation is taken into account, the deformation behaves more predictably, as seen in subfigure 6.22g. In contrast, FFD remains stable in these scenarios, as it simply guides the surface toward the outermost target point, making it more robust to such alignment configurations.

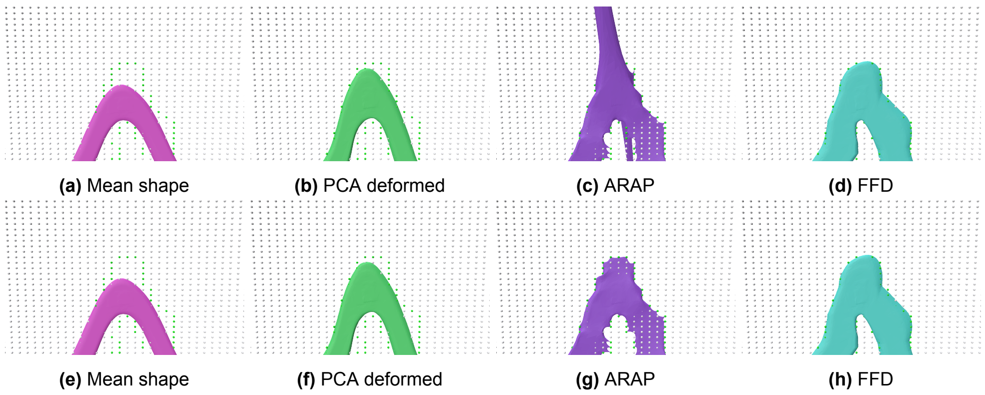


Figure 6.22: Comparison of deformation techniques on two different sets of input points. Each row shows the mean shape, PCA deformation, ARAP, and FFD results respectively.

6.14. 3-Bladers

The methodology was also evaluated on three-blader boomerangs. Due to the limited availability of 3D-scanned tribladers, the analysis was conducted on the few flat models available. The results align well with the findings from the two-blader experiments.

As seen in Figure 6.23, the checkerboard pattern exhibits some shearing, which is further confirmed by the values in Table 6.4. While some compression is present, the overall area distortion remains within reasonable limits.

The resulting correspondences, shown in Figure 6.24, reveal no unexpected discrepancies. Both Hausdorff and Chamfer distances remain low, comparable to those observed in the two-blader cases.

Figure 6.25 illustrates the results of PCA-based deformations. Given the limited number of shapes in PCA space, a direct match to the target outline points is unlikely. However, Free-Form Deformation (FFD) effectively bridges this gap. While ARAP + Laplacian deformation yields similar results, it tends to thin out the airfoil, as noted in previous discussions.

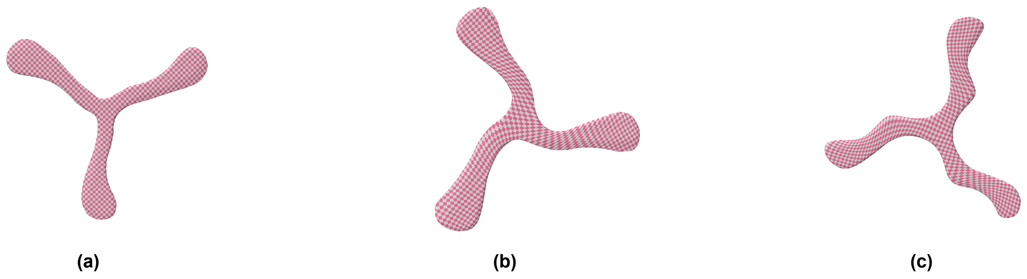
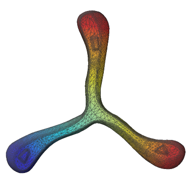
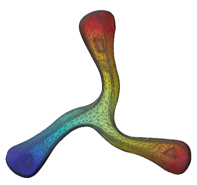


Figure 6.23: Checkboard pattern Visualization on thee models


(a)


(b)

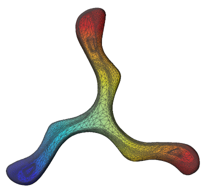
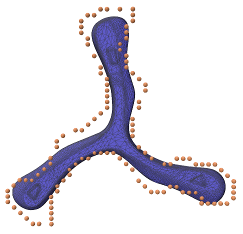
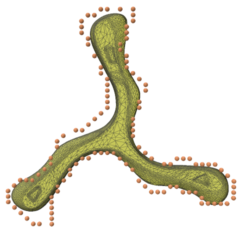
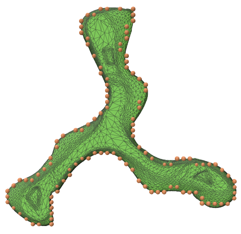
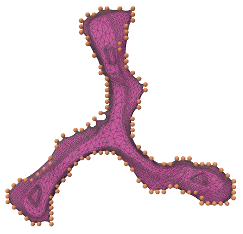

(c)

Figure 6.24: Correspondences visualized


(a) Mean Shape


(b) Deformed shape with optimized weights


(c) with FFD


(d) with ARAP

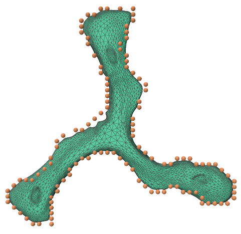

(e) with ARAP+Laplacian

Figure 6.25: Principal Component Analysis (PCA) with optimized weights, FFD and ARAP+Laplacian.

Shape 1:

	Area Distortion Min	Area Distortion Max	Area Distortion Mean	Area Distortion StdDev	Shear Distortion Min	Shear Distortion Max	Shear Distortion Mean	Shear Distortion StdDev	Area Shape 1	Area Shape 2	Shape 2
0	0.0	119.05	0.99	2.50	0.00	21.79	1.19	0.66	22086.9	26752.0	
1	0.0	380.66	0.93	2.54	0.38	3.86	1.15	0.18	22085.8	25558.0	

Table 6.3: Distortion results.

Shape 1:

	Hausdorff Distance	Chamfer Distance	Normalized Hausdorff Distance	Normalized Chamfer Distance	Shape 2
0	0.70	0.02	2.08e-03	4.72e-05	
1	0.46	0.01	1.31e-03	3.80e-05	

Table 6.4: Distance results

6.15. Affect on FFD paramaeters

The number of outline points significantly influences the required grid resolution in Free-Form Deformation (FFD). When fewer outline points are selected, a coarser grid is sufficient since only minimal deformation is needed. However, as the grid resolution increases, more outline points are required to prevent overfitting and ensure smooth deformations.

This effect is clearly visible in Figure 6.26, where grids larger than 11×11 begin to exhibit overfitting. At these higher resolutions, ”sensitive” areas develop peaks and distortions unless additional control points are introduced near critical regions.

The number of iterations is a less influential parameter. After a single iteration, FFD usually provides results very close to the expected shape. However, in cases where long wing tips or other elongated features are present, additional iterations may help refine the shape.

Figure 6.26 illustrates the impact of grid resolution in the X and Y directions on the final shape. If you use different setups, e.g. 14x2 or 2x14, similar behavior can be observed.

6.16. Comparison of PCA+FFD and FFD-Only Deformation

Combining Principal Component Analysis (PCA) with Free-Form Deformation (FFD) brings together global shape modeling and local flexibility. PCA captures the dominant modes of variation across training shapes, enabling the initialization to already approximate the target outline. As the number of training shapes increases, this initialization gets closer to the outline, reducing the deformation effort required during the FFD stage.

In contrast, using FFD alone - without prior PCA adjustment - starts from the mean shape and requires more parameter tuning to match the target. This results in a higher-dimensional, more sensitive optimization problem, often leading to less stable or less accurate deformations.

Figure 6.27 illustrates this difference. In particular, Figure 6.27c shows that using PCA followed by FFD with a 10×10 grid effectively adapts to the target outline. On the other hand, Figures 6.27d, 6.27e, 6.27f, 6.27g, and 6.27h demonstrate the limitations of using only FFD starting from the mean shape. Without PCA, FFD struggles to generalize across any tested grid resolutions: while higher grid sizes (e.g., 10×10) get closer to the target, they are computationally more expensive and prone to introducing artifacts unless additional target points are provided. Lower-resolution grids, conversely, fail to capture finer details, while higher-resolution grids resulted in ”spiky” behavior.

Additionally, the mesh topology is more severely affected when PCA is omitted. For example, surface features such as stickers located on the inner elbow, dingle, and leading arm of the boomerang are slightly stretched - as expected - in Figure 6.27c, while in Figures 6.27d, 6.27e, 6.27f, 6.27g, and 6.27h, these features are sheared and distorted more drastically. The airfoil lines (e.g., where the trailing and leading edges start from the top of the boomerang) also become disproportionately scaled, indicating that the airfoil is distorted as well.

In summary, for target shapes that deviate more significantly from the mean, applying PCA before FFD is crucial. It brings the initial shape closer to the target, simplifies the deformation task, reduces computational cost, and better preserves topological and structural details.

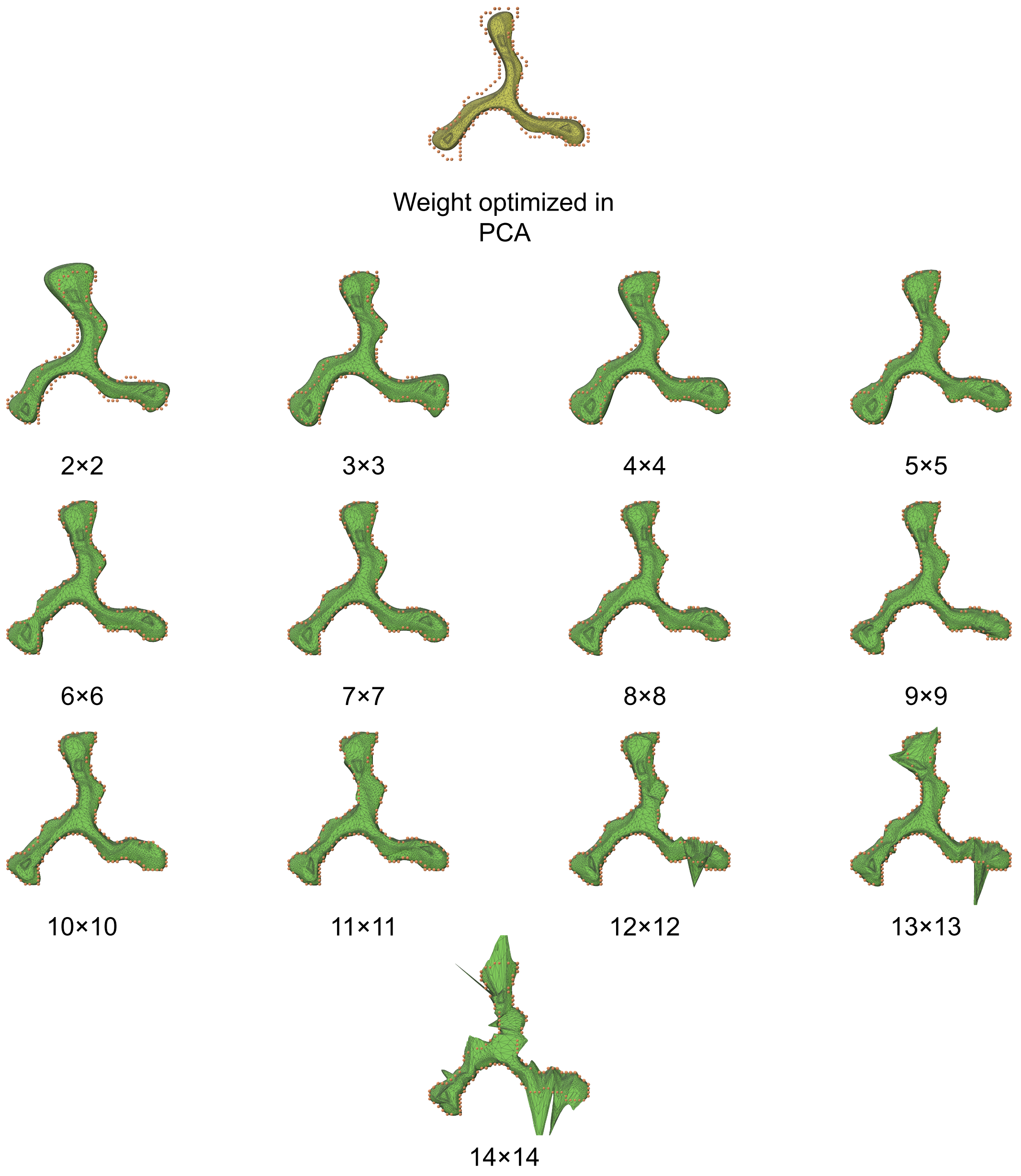
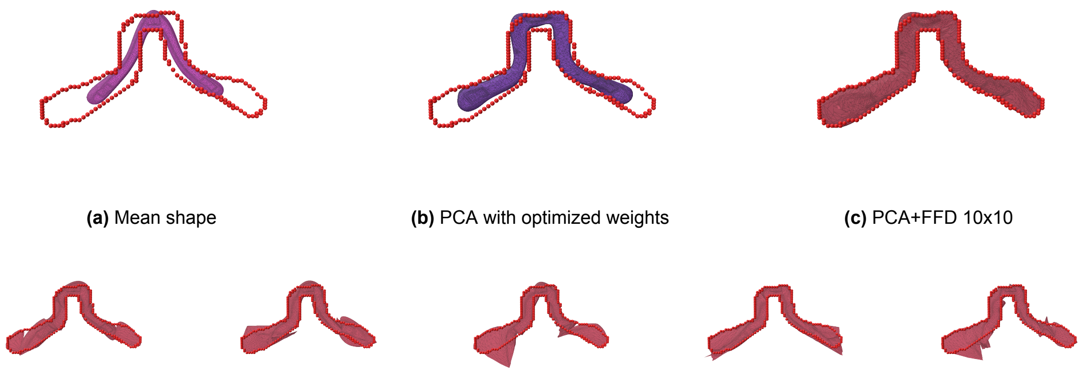


Figure 6.26: Comparison of different grid resolutions used in the analysis. Each subfigure represents a different grid size from 2×2 to 14×14.



(a) Mean shape **(b)** PCA with optimized weights **(c)** PCA+FFD 10x10

(d) FFD 6x6 **(e)** FFD 8x8 **(f)** FFD 10x10 **(g)** FFD 13x13 **(h)** FFD 15x15

Figure 6.27: Shape reconstructions using PCA+FFD versus solely FFD using different grid sizes.

7 Discussion

This thesis introduced a computational framework explicitly designed to analyze and generate new boomerang shapes by solving the correspondence problem across a dataset of 3D scanned models. To enable this, a dedicated dataset of over twenty boomerangs was created and post-processed. The framework combined structural landmark alignment, alpha-shape boundary extraction, curve parameterization, Least Squares Conformal Mapping (LSCM), and statistical modeling using PCA, further enhanced by gradient-based optimization and Free-Form Deformation (FFD). This pipeline allows for structured shape correspondence across boomerangs and enables the creation of new geometries through statistical shape reconstruction.

Scan Quality and Dataset Preparation 3D scanning proved effective for most boomerangs, but some challenges were encountered with transparent or reflective surfaces and particularly thin airfoils. Interestingly, models with less aggressive post-processing yielded better results, suggesting that minimizing manual editing helps preserve original shape quality. Preliminary comparisons on clay tablets conducted at TU Delft's NewMedia Center showed that both photogrammetry and micro-CT (μCT) scanning produced comparable results to the structured-light scanner, with μCT offering higher volumetric accuracy. This may be worth exploring in future datasets, especially for capturing intricate geometric details in thin or twisted regions.

Additionally, the decimation process in Blender may not be optimal, as it often produces jagged artifacts near the boundaries and results in inconsistent triangulation. Future work could explore alternative preprocessing techniques to reduce distortion and improve overall mesh quality.

Landmark Selection and Structural Alignment Manual landmark selection played a central role in the correspondence pipeline but introduced subjectivity and potential inconsistencies. In particular, additional landmarks in the elbow region were often required to preserve area consistency during parameterization. Without these, correspondences in high-curvature regions became stretched, reducing the quality of the resulting UV mappings. Automating or guiding landmark selection based on curvature, boundary salience, or structural heuristics could substantially improve repeatability and accuracy.

Boundary Extraction and Curve Lifting Alpha shape-based boundary extraction proved effective, but required careful tuning of the alpha parameter to avoid artifacts. While this method handles most boomerang geometries well, it occasionally fails to capture sharp elbow curvature, especially for inward curvatures. For such cases, alternative techniques such as flood-fill-based boundary detection or adaptive alpha selection could enhance robustness.

The curve lifting step currently relies on projecting boundary points from a globally averaged plane. While this works well for (mostly) flat boomerangs, it becomes less accurate for boomerangs with tuned (non-flat) wings. To better accommodate such cases, a more localized projection strategy could be adopted. For instance, computing a local cross-sectional average to project from or fitting a surface patch in the neighborhood of each projected point could yield more accurate lifting onto the 3D surface.

In Step 4.2.7, boundary curves are lifted from 2D to 3D by projecting each point along averaged normal directions. However, in regions of high curvature, this projection may fail-either by misalligning the vertex or projecting it in the wrong direction relative to the surface. Interestingly, alpha shapes can indirectly help here: highly curved regions may be excluded from the boundary entirely, resulting in a gap rather than risking an incorrect projection. To address such gaps, a shortest-path traversal using Dijkstra's algorithm is used to reconstruct a continuous 3D boundary. While this approach ensures topological completeness, the resulting paths are often more jagged and lose information about the precise location of the original boundary. As a result, the reconstructed path may diverge in different directions across shapes, potentially introducing inconsistencies. Incorporating geodesic paths instead could offer a more accurate and surface-aware alternative, preserving better alignment across corresponding shapes.

Parameterization Techniques LSCM was found to outperform alternative parameterization methods such as harmonic mapping and ARAP, particularly when applied with full boundary constraints. ARAP, while theoretically promising, introduced unwanted distortions on the boomerang's reverse side. Importantly, using the actual projected 3D boundary as the parameterization constraint significantly outperformed circular constraints, as expected due to greater shape consistency across samples. However, ensuring that the boundary is fully connected is crucial - disconnected boundaries led to severe inaccuracies in the UV maps.

LSCM performs best on surfaces with disk topology - that is, genus-0 surfaces without holes. Boomerangs with deliberately introduced holes or higher-genus structures were not considered in this work and are not directly supported by standard LSCM implementations. To extend LSCM to such cases, a practical approach is to preprocess the mesh by temporarily filling the holes to create a disk-like topology suitable for parameterization. The holes can then be reintroduced after the UV mapping is complete.

Correspondence Evaluation Quantitative evaluations confirmed the effectiveness of the proposed correspondence framework, yielding low Hausdorff and Chamfer distances across the dataset. Nonetheless, small local deviations-particularly in high-curvature or elbow regions-were observed. These are largely attributable to vertex shifts during projection and triangulation, further emphasizing the need for precise and curvature-aware boundary processing.

Distortion analysis across parameterizations suggested a possible inverse relationship between area distortion and shear distortion, where shapes experiencing compression tended to exhibit higher shear. While this trend was consistently observed across several examples, it remains a hypothesis that requires more systematic evaluation to confirm. Also, selecting a base shape that is geometrically similar to the target is important, ideally close to the dataset's mean shape. Landmark selection also played a major role in local distortion control. As such, both base shape choice and landmark placement significantly influence correspondence quality.

Scaling shapes to equal surface area before correspondence could reduce distortion, but this comes at the cost of altering physical properties-especially thickness-which may impact aerodynamic behavior. Furthermore, inconsistent mesh resolution across the dataset introduced additional complexity. Although all final correspondence meshes shared the same number of vertices, the input meshes did not. Future work may benefit from mesh resampling or remeshing to enforce consistent vertex densities.

This work was compared against an alternative correspondence method based on the spectral domain, as proposed in [22]. However, this approach was found to be unsuitable for boomerang shapes, as it mismatched corresponding sides.

Statistical Modeling and Shape Synthesis While PCA proved effective in modeling global shape variation, its inherent linearity limited its ability to capture fine-grained local details. Its performance depends heavily on the variation present in the input dataset; a broader or more diverse dataset would increase the expressiveness of the PCA space. Regularization was necessary to suppress the influence of noisy components - particularly from inner-elbow regions - and higher principal components were found to carry less meaningful information, often amplifying noise.

Combining PCA with Free-Form Deformation provided a powerful hybrid approach. The bilinear grid-based FFD worked as expected, but required careful tuning. Larger grid sizes allowed for finer deformation but increased the risk of introducing noise, while smaller grids produced smoother results at the cost of flexibility. Additionally, point density in the outline influenced the effectiveness of FFD-the denser the outline, the better the high-resolution grid performed.

An alternative deformation method, an ARAP-based grid, was also tested. While it produced valid results, it tended to over-thin airfoils and deviated more from the original shape than the bilinear FFD approach. Additionally, Laplacian smoothing was necessary to avoid spiky or irregular behavior in the deformed shapes, as ARAP alone occasionally introduced sharp local distortions. Using PCA before FFD (rather than relying on FFD alone) proved crucial: the PCA step moves the shape globally toward the target, while the FFD refines local structure. This combination resulted in more natural-looking outputs and minimized extreme deformations.

Incorporating a non-linear deformation term during PCA weight optimization is a promising direction for future work. While the current implementation uses gradient-based optimization with a Jacobian in the PCA space, the deformation model itself remains linear and prone to end in a local minima, as PCA defines a linear subspace of shape variation. Introducing non-linear deformation models-such as non-linear energy terms or constraint-based formulations-could further improve shape realism and reduce overfitting (or underfitting), particularly in regions with complex local geometry. Additionally, more targeted constraints could be integrated to enforce known physical or aerodynamic properties of boomerangs. For instance, a thickness constraint was implemented in this work, but no conclusive analysis was conducted on its impact. Future studies could explore such constraints more systematically to guide deformation while preserving critical aerodynamic features.

One limitation of the current FFD-based approach is that it minimizes the distance from points to the surface. However, when points already lie on the surface, this distance becomes zero-making them appear 'optimized' regardless of whether they correspond to the correct structural feature. As a result, shapes with a thinner wing chord than the mean shape may not be accurately reconstructed, since the outer geometry is under-constrained. A straightforward strategy to tackle this issue is to reparameterize the outline points with respect to the mean PCA shape using a constant-speed parameterization over the unit interval, similar to the method described in Section 4.2.6. Alternatively, one could modify the loss function to consider only the distance from the outermost points of the reconstructed shape to the target outline, ensuring that the outer geometry is explicitly preserved during optimization.

N-bladers The proposed method has been primarily evaluated on two-bladed boomerangs, with additional tests performed on a subset of three-bladed examples. The results were consistent across both cases, suggesting that the approach generalizes well. While explicit experiments on boomerangs with more than three blades (i.e., N-bladed designs) have not been conducted, the methodology is expected to extend naturally to such cases.

8 Conclusion

Summary and Contributions This thesis introduced a computational framework for analyzing and modeling boomerangs based on example shapes. Motivated by the lack of formal tools for understanding and designing handcrafted aerodynamic objects, the work addressed the shape correspondence problem as a foundational challenge. Boomerangs, with their subtle 3D features, asymmetries, and functional structure, fall outside the scope of standard shape matching pipelines. The developed method offers a tailored alternative by leveraging structural landmarks, alpha shape-based boundary extraction, boundary re-projection, Least Squares Conformal Mapping (LSCM), and statistical shape modeling using PCA, further refined through Free-Form Deformation (FFD).

By combining these components, the pipeline enables consistent correspondences across a set of scanned boomerangs and supports the generation of new designs grounded in geometric structure. Quantitative evaluation confirms the validity of the approach, and a curated dataset of over twenty scanned boomerangs serves as a practical foundation for further research. In doing so, this work bridges the gap between empirical craftsmanship and digital modeling, contributing to the digitalization of aerodynamic design tools for real-world objects.

Limitations and Insights While the framework performs well overall, several limitations remain. The reliance on manual landmark selection introduces subjectivity and variability, particularly in complex regions such as the elbow. Boundary extraction using alpha shapes is sensitive to parameter tuning and may struggle with sharp concavities or tuned geometries. The lifting process, while effective for flat shapes, is less accurate for boomerangs with curved or warped wings - highlighting the need for more localized projection strategies.

Parameterization using LSCM, though effective when constrained appropriately, assumes disk-topology surfaces and requires consistent, connected boundaries. While the modeling pipeline successfully combines PCA and FFD to balance global and local control, the linear nature of PCA limits its expressiveness, and FFD remains sensitive to grid resolution and outline density. Moreover, reconstructed shapes can fail to match target features when the outer geometry is under-constrained or poorly parameterized.

Broader Implications Beyond boomerangs, the methodology developed in this thesis has broader relevance for analyzing and modeling irregular but structured 3D shapes. In these contexts, the combination of structural guidance, data-driven modeling, and deformable reconstruction provides a powerful toolkit for shape exploration and comparison.

Future Work Several directions for future research emerge from this thesis. First, automating landmark selection using curvature, or learned feature detectors could improve robustness and consistency across shapes. Second, advancing boundary extraction through adaptive alpha tuning or geodesic-driven techniques could improve reliability in difficult cases. Third, extending LSCM to handle more complex topologies - through pre- and post-processing, or alternative parameterization strategies - would allow the framework to support a wider variety of shapes.

On the modeling side, incorporating non-linear deformation models could enhance fidelity, particularly in regions where PCA falls short. Introducing additional constraints - such as preserving average thickness - may further improve stability and ensure physically plausible deformations.

Closing Remarks This thesis contributes a practical, flexible, and extensible pipeline for analyzing and generating boomerang shapes based on real-world data. By treating handcrafted geometry not as a problem but as a structured input to be understood, the work offers new possibilities for design, comparison, and exploration - both within boomerang sports and in the broader field of computational shape modeling.

References

[1] Koray Akalin et al. “Mesh Parameterization Meets Intrinsic Triangulations”. In: *Computer Graphics Forum* 43 (July 2024). DOI: 10.1111/cgf.15134.

[2] Said Benameur et al. “3D/2D registration and segmentation of scoliotic vertebrae using statistical models”. In: *Computerized Medical Imaging and Graphics* 27.5 (2003), pp. 321–337. ISSN: 0895-6111. DOI: https://doi.org/10.1016/S0895-6111(03)00019-3. URL: https://www.sciencedirect.com/science/article/pii/S0895611103000193.

[3] Fausto Bernardini and Chandrajit Bajaj. “Sampling and Reconstructing Manifolds Using Alpha-Shapes”. In: *Proc. 9th Canad. Conf. Comput. Geom.* (July 1998).

[4] P.J. Besl and Neil D. McKay. “A method for registration of 3-D shapes”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 14.2 (1992), pp. 239–256. DOI: 10.1109/34.121791.

[5] Sofien Bouaziz, Yangang Wang, and Mark Pauly. “Online modeling for realtime facial animation”. In: *ACM Trans. Graph.* 32.4 (July 2013). ISSN: 0730-0301. DOI: 10.1145/2461912.2461976. URL: https://doi.org/10.1145/2461912.2461976.

[6] Sabine Coquillart. “Extended free-form deformation: a sculpturing tool for 3D geometric modeling”. In: *SIGGRAPH Comput. Graph.* 24.4 (Sept. 1990), pp. 187–196. ISSN: 0097-8930. DOI: 10.1145/97880.97900. URL: https://doi.org/10.1145/97880.97900.

[7] Keenan Crane, Clarisse Weischedel, and Max Wardetzky. “Geodesics in Heat”. In: *CoRR* abs/1204.6216 (2012). arXiv: 1204.6216. URL: http://arxiv.org/abs/1204.6216.

[8] Creaform. *Go!SCAN 3D Technical Specifications*. Accessed: 2024-11-27. 2024. URL: https://www.creaform3d.com/en/handheld-portable-3d-scanner-goscan-3d/technical-specifications.

[9] Creaform. *VXelements - 3D Software Platform*. https://www.creaform3d.com/en/metrology-solutions/3d-applications-software-platforms. Accessed: 2024-11-27. 2024.

[10] Théo Deprelle. “Learning surface elements for shape representation”. Theses. École des Ponts ParisTech, Oct. 2022. URL: https://pastel.hal.science/tel-03941345.

[11] Mathieu Desbrun, Mark Meyer, and Pierre Alliez. “Intrinsic parameterizations of surface meshes”. In: *Computer Graphics Forum* 21.3 (2002), pp. 209–218.

[12] M. Fleute and S. Lavallée. “Nonrigid 3-D/2-D Registration of Images Using Statistical Models”. In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI’99*. Ed. by Chris Taylor and Alain Colchester. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, pp. 138–147. ISBN: 978-3-540-48232-1.

[13] Michael S. Floater and Kai Hormann. “Surface Parameterization: A Tutorial and Survey”. In: *Computer Graphics Forum* 21.3 (2002), pp. 209–228.

[14] Yotam Gingold, Takeo Igarashi, and Denis Zorin. “Structured Annotations for 2D-to-3D Modeling”. In: *ACM Trans. Graph.* 28 (Dec. 2009). DOI: 10.1145/1618452.1618494.

[15] Prasad Gudem, Manuel Schütz, and Kyle Holland. “Flight Dynamics of Boomerangs: Impact of Reversal of Airflow, Reversal of Angle-of-Attack and Asymmetric Lift”. In: June 2019. DOI: 10.2514/6.2019-2826.

[16] Prasad Gudem et al. “Flight Dynamics of Boomerangs: Impact of Drag Force and Drag Torque”. In: June 2020. DOI: 10.2514/6.2020-2709.

[17] Geremy Heitz, Torsten Rohlfing, and Calvin Maurer. “Statistical Shape Model Generation Using Nonrigid Deformation of a Template Mesh”. In: *Proceedings of SPIE - The International Society for Optical Engineering* 5747 (Apr. 2005). DOI: 10.1117/12.594802.

[18] F. Hess. “Boomerangs, aerodynamics and motion”. Groningen University. PhD thesis. Groningen University, 1975.

[19] Kai Hormann, Bruno Eric Levy, and Alla Sheffer. “Mesh Parameterization: Theory and Practice”. In: *ACM SIGGRAPH Asia Courses*. 2008. DOI: 10.1145/1508044.1508091.

[20] Takeo Igarashi, Tomer Moscovich, and John F. Hughes. “As-rigid-as-possible shape manipulation”. In: *ACM Trans. Graph.* 24.3 (July 2005), pp. 1134–1141. ISSN: 0730-0301. DOI: 10.1145/1073204.1073323. URL: https://doi.org/10.1145/1073204.1073323.

[21] Hardik Jain, Manuel Wollhaf, and Olaf Hellwich. “Learning to reconstruct symmetric shapes using planar parameterization of 3D surface”. In: *Proceedings of the IEEE International Conference on Computer Vision Workshops*. 2019.

[22] Varun Jain and Hao Zhang. “Robust 3D Shape Correspondence in the Spectral Domain”. In: *IEEE Transactions on Visualization and Computer Graphics* 18.1 (2011), pp. 171–182.

[23] Oliver van Kaick et al. “A Survey on Shape Correspondence”. In: *Computer Graphics Forum* 30.6 (2011), pp. 1681–1707.

[24] Z Kami, Craig Gotsman, and Steven J Gortler. “Free-boundary linear parameterization of 3D meshes in the presence of constraints”. In: *International Conference on Shape Modeling and Applications 2005 (SMI’05)*. IEEE. 2005, pp. 266–275.

[25] Roman Klokov. “Deep Learning for 3D Shape Modelling”. PhD thesis. Dec. 2021.

[26] GRAHAM R. L. “An efficient algorithm for determining the convex hull of a finite planar set”. In: *Info. Proc. Lett.* 1 (1972), pp. 132–133. URL: https://cir.nii.ac.jp/crid/1570009750621882496.

[27] H. Lamecker, T.H. Wenckebach, and H.-C. Hege. “Atlas-based 3D-Shape Reconstruction from X-Ray Images”. In: *18th International Conference on Pattern Recognition (ICPR’06)*. Vol. 1. 2006, pp. 371–374. DOI: 10.1109/ICPR.2006.279.

[28] Longin Jan Latecki and Rolf Lakämper. “Shape Similarity Measure Based on Correspondence of Visual Parts”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence*. Vol. 22. 10. 2000, pp. 1185–1190.

[29] D. T. Lee and B. J. Schachter. “Two Algorithms for Constructing a Delaunay Triangulation”. In: *International Journal of Computer and Information Sciences* 9.3 (1980), pp. 219–242. ISSN: 0091-7036. DOI: 10.1007/BF00977785. URL: https://doi.org/10.1007/BF00977785.

[30] Sing Chun Lee and Misha Kazhdan. “Dense Point-to-Point Correspondences Between Genus-Zero Shapes”. In: *Computer Graphics Forum* 38 (Aug. 2019), pp. 27–37. DOI: 10.1111/cgf.13787.

[31] Bruno Eric Levy et al. “Least Squares Conformal Maps for Automatic Texture Atlas Generation”. In: *Proceedings of the 29th Conference on Computer Graphics and Interactive Techniques SIGGRAPH*. 2002, pp. 362–371.

[32] Hao Li et al. “Robust Single-View Geometry and Motion Reconstruction”. In: *ACM Trans. Graph* 28 (Dec. 2009). DOI: 10.1145/1618452.1618521.

[33] Ligang Liu et al. “A local/global approach to mesh parameterization”. In: *Computer Graphics Forum* 27 (2008), pp. 1495–1504.

[34] EM Mikhail. “Introduction to Modern Photogrammetry”. In: *John Williey & sons* (2001).

[35] Patrick Mullen et al. “Spectral Conformal Parameterization”. In: *Comp. Graph. Forum (Symp. Geom. Proc.)* 27 (July 2008). DOI: 10.1111/j.1467-8659.2008.01289.x.

[36] Mohamed Omran et al. *Neural Body Fitting: Unifying Deep Learning and Model-Based Human Pose and Shape Estimation*. 2018. arXiv: 1808.05942 [cs.CV]. URL: https://arxiv.org/abs/1808.05942.

[37] Rohan Sawhney and Keenan Crane. “Boundary First Flattening”. In: *ACM Transactions on Graphics* (2020).

[38] Thomas W. Sederberg and Scott R. Parry. “Free-form deformation of solid geometric models”. In: *Proceedings of the 13th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH ’86. New York, NY, USA: Association for Computing Machinery, 1986, pp. 151–160. ISBN: 0897911962. DOI: 10.1145/15922.15903. URL: https://doi.org/10.1145/15922.15903.

[39] Nicholas Sharp and Keenan Crane. “You Can Find Geodesic Paths in Triangle Meshes by Just Flipping Edges”. In: *ACM Transactions on Graphics (TOG)* 41.4 (2022), 112:1–112:12.

[40] Abdel Aziz Taha and Allan Hanbury. “An Efficient Algorithm for Calculating the Exact Hausdorff Distance”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37.11 (2015), pp. 2153–2163. DOI: 10.1109/TPAMI.2015.2408351.

[41] Justin Tahmassebpur et al. “Wind Tunnel Measurements of Non-Dimensional Lift Coefficient of a Boomerang and Comparison to Theory”. In: Aug. 2021. DOI: 10.2514/6.2021-2529.

[42] J. Vassberg. “Boomerang Flight Dynamics”. In: June 2012. ISBN: 978-1-62410-185-4. DOI: 10.2514/6.2012-2650.

[43] Subeesh Vasu et al. *HybridSDF: Combining Deep Implicit Shapes and Geometric Primitives for 3D Shape Representation and Manipulation*. 2022. arXiv: 2109.10767 [cs.CV]. URL: https://arxiv.org/abs/2109.10767.

[44] Zhen Wei, Pascal Fua, and Michaël Bauerheim. *Automatic Parameterization for Aerodynamic Shape Optimization via Deep Geometric Learning*. 2023. arXiv: 2305.02116 [cs.CV]. URL: https://arxiv.org/abs/2305.02116.

[45] Tong Wu et al. “Balanced Chamfer Distance as a Comprehensive Metric for Point Cloud Completion”. In: *Advances in Neural Information Processing Systems*. Ed. by M. Ranzato et al. Vol. 34. Curran Associates, Inc., 2021, pp. 29088–29100. URL: https://proceedings.neurips.cc/paper_files/paper/2021/file/f3bd5ad57c8389a8a1a541a76be463bf-Paper.pdf.

[46] Yifan Xu et al. *SpiderCNN: Deep Learning on Point Sets with Parameterized Convolutional Filters*. 2018. arXiv: 1803.11527 [cs.CV]. URL: https://arxiv.org/abs/1803.11527.

[47] Hao Zhang et al. “Deformation-Driven Shape Correspondence”. In: *Computer Graphics Forum* 27.5 (2008), pp. 1431–1439.

A
Additional Algorithms

A.1. Face Classification Using Flood-Fill

To segment the mesh into two regions, a flood-fill approach based on face connectivity can be used. Given:

- V_1 : The set of vertices of the mesh.
- F_1 : The set of triangular faces, where each face consists of three vertex indices.
- B' : The set of vertices forming the connected boundary.

Identifying the Seed Face

A face $f \in F_1$ is considered adjacent to the boundary if at least one of its vertices is close to a boundary vertex:

$$\exists v \in f \quad \text{such that} \quad v \approx b, \quad \forall b \in B' \tag{A.1}$$

where v is a vertex of face f, and b is a vertex in the boundary B'. The first such face found is selected as the seed face.

Determining Face Adjacency

Two faces f_i, f_j are considered adjacent if they share at least two vertices:

$$|\{v \in f_i\} \cap \{v \in f_j\}| \geq 2 \tag{A.2}$$

This ensures that only faces connected by a shared edge (not just a vertex) are considered neighbors.

Flood-Fill Region Expansion

Starting from the seed face, iteratively classify faces using:

$$F_A = \{f \in F_1 \mid f \text{ is reachable from the seed face through shared edges and } |f \cap B'| < 2\} \tag{A.3}$$

$$F_B = F_1 \setminus F_A \tag{A.4}$$

where:

- F_A is the set of faces connected to the seed face via shared edges, allowing at most one boundary vertex.

- F_B is the set of remaining faces, forming the opposite region.
- Each edge appears in exactly one of F_A or F_B, ensuring that no boundary edge is shared between both regions.

Since each edge in a manifold triangular mesh appears in exactly two faces, ensure that:

$$\forall e \in B', \quad e \text{ belongs to exactly one face in } F_A \text{ or } F_B. \tag{A.5}$$

Thus, a face f is classified as:

$$f \in F_A \quad \text{if it is reachable from the seed face and } |f \cap B'| < 2 \tag{A.6}$$

$$f \in F_B \quad \text{otherwise} \tag{A.7}$$

This ensures that the segmentation follows mesh connectivity while preventing any edge from appearing in both F_A and F_B, which would lead to ambiguity.

A.2. As-Rigid-As-Possible (ARAP) Deformation

As-Rigid-As-Possible (ARAP) deformation is a widely used technique in geometry processing that allows flexible deformation of shapes while preserving local rigidity. The method is especially effective in mesh editing, where it is desirable to maintain the intrinsic structure of a shape under transformation.

ARAP aims to minimize local distortion by computing a deformation that is as close as possible to a rigid transformation at each local neighborhood of the mesh. That is, each local patch is allowed to rotate and translate but not scale or shear excessively. The global deformation is then formed by blending these local rigid transformations in a way that maintains consistency across the mesh.

Formally, given a mesh with vertex positions $\mathbf{v}_i \in \mathbb{R}^d$ (where $d = 2$ or 3), and a deformed version with positions $\mathbf{v}'_i$, the ARAP energy is defined as:

$$E_{\text{ARAP}} = \sum_i \sum_{j \in \mathcal{N}(i)} w_{ij} \left\| (\mathbf{v}'_i - \mathbf{v}'_j) - R_i(\mathbf{v}_i - \mathbf{v}_j) \right\|^2, \tag{A.8}$$

where:

- $\mathcal{N}(i)$ is the set of neighboring vertices of vertex i,
- w_{ij} are weights (often cotangent weights) that control the influence of each edge,
- $R_i \in \mathrm{SO}(d)$ is the best local rotation matrix for vertex i,
- $\mathrm{SO}(d)$ denotes the Orthogonal group of degree d, i.e., the set of all $d \times d$ orthogonal matrices with determinant 1.

In particular:

- For 2D deformations, $R_i \in \mathrm{SO}(2)$, which includes all 2D rotation matrices:

$$R(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \tag{A.9}$$

- For 3D deformations, $R_i \in \mathrm{SO}(3)$, which includes all 3D rotation matrices that preserve orientation:

$$R^T R = I, \quad \det(R) = 1 \tag{A.10}$$

The deformation is solved through an iterative optimization process: alternating between estimating the optimal rotations R_i (local step), and updating the vertex positions $\mathbf{v}'_i$ to best match those rotations (global step). This alternating minimization converges to a deformation that maintains geometric detail while satisfying high-level constraints, such as control points or boundary conditions.

B Additional nomenclature

Symbol	Definition	Unit
	Appendix - Face Classification Using Flood-Fill	
V_1	Vertex positions of the mesh	$\mathbb{R}^{n\times 3}$
F_1	Set of triangular faces (indices into V_1)	$\mathbb{N}^{m\times 3}$
B'	Set of boundary vertices forming a connected loop	-
f	A triangular face in F_1	-
v	A vertex in a face f	-
F_A	Subset of faces reachable from seed face with < 2 boundary vertices	-
F_B	Complement of F_A; faces on the opposite side of the boundary	-
e	Mesh edge defined by a pair of vertex indices	-
$\|f \cap B'\|$	Number of boundary vertices in face f	-
	Appendix - As-Rigid-As-Possible (ARAP) Deformation	
$\mathbf{v}_i$	Original position of vertex i	$\mathbb{R}^d$
$\mathbf{v}'_i$	Deformed position of vertex i	$\mathbb{R}^d$
$\mathcal{N}(i)$	Set of neighbors of vertex i	-
w_{ij}	Edge weight between vertices i and j (e.g., cotangent weight)	-
R_i	Optimal local rotation for vertex i	$\mathrm{SO}(d)$
$\mathrm{SO}(d)$	Special orthogonal group (set of $d \times d$ rotation matrices with determinant 1)	-
E_{ARAP}	ARAP energy measuring deviation from local rigidity	-

C Dataset Description

This dataset was created as part of the MSc. thesis ”Shape Correspondences and Example-Based Modelling for Boomerang Design” by Nils van Veen and supervised by K. Hildebrandt and R. Marroquim, conducted at TU Delft. The research presents a shape correspondence pipeline tailored for irregular but structured aerodynamic boomerang shapes, focusing specifically on handcrafted competition boomerangs.

C.1. 3D Scanning and Processing

To construct the dataset, over twenty competition-grade boomerangs were 3D scanned using a Creaform Go!SCAN device. Each boomerang was scanned from multiple static orientations and merged into a complete 3D model with VXelements software, using an Iterative Closest Point (ICP) algorithm for alignment.

Post-processing included:

- Noise filtering and outlier rejection to eliminate isolated patches,
- Surface smoothing,
- Small hole filling,
- Correction of local artifacts from alignment or scanning,
- Mesh decimation (10%) in Blender to improve computational efficiency.

To improve scan quality for thin airfoils or reflective/transparent surfaces, a temporary chalk spray was used to enhance surface visibility. Scans were captured using a resolution between 0.2 mm and 0.4 mm. In some cases, original boomerang thicknesses of 3–3.4 mm were reconstructed as slightly thicker (3.5–3.8 mm), likely due to smoothing and surface interpolation effects during reconstruction.

C.2. Dataset Format and Access

The dataset consists of 3D mesh files in ‘.obj‘ format. Each mesh is aligned, decimated, and preprocessed for use with the shape correspondence pipeline.

- **Number of shapes:** 16
- **File format:** OBJ (.obj)
- **Average resolution**:˜10,000-50,000 vertices
- **Correspondences**:˜20,000 point-to-point matches per shape
- **Naming convention:** `boomerang_XX.obj`
- **Correspondence-ready:** All meshes are including correspondences.
- **Access link:** `https://graphics.tudelft.nl/NilsVanVeen`

- **Thesis code link:** `https://github.com/NilsvVeen/MSc-Thesis-TUD---boomerang-correspondence`

Additional models are added for future research.

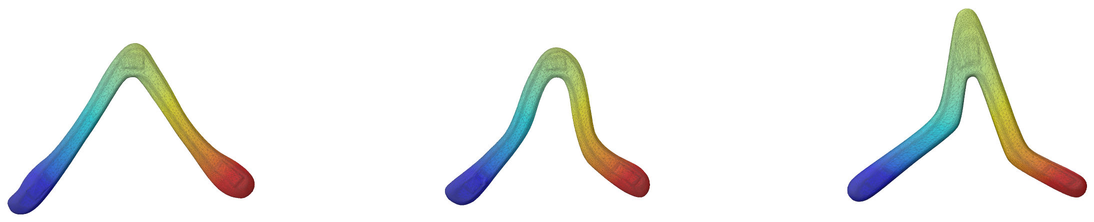
C.3. Shape Correspondence Pipeline

The dataset serves as the foundation for establishing shape correspondences using a novel pipeline that combines conformal mapping and structural constraints. The key idea is to parameterize each boomerang into a shared 2D UV space using Least Squares Conformal Mapping (LSCM), guided by a shared 2D boundary outline.

While LSCM typically requires only two fixed vertices to anchor the map, this work leverages its flexibility by constraining a full boundary curve. This ensures consistent UV mappings across different boomerangs.

The correspondence process involves the following steps:

1. **Align boomerangs in world space:** A plane is fitted to each mesh, and the mesh is rotated such that the plane normal aligns against the z-axis.
2. **Project onto a parallel plane:** The mesh is orthogonally projected to a 2D plane (typically the XY plane).
3. **Compute the 2D alpha shape:** A user-defined alpha parameter is used to extract the 2D outline of the projected shape using α-shapes.
4. **Sort the boundary curve:** The boundary vertices are sorted using a nearest-neighbor traversal to ensure consistent vertex ordering.
5. **Obtain user-defined landmarks:** Key landmarks (e.g., wing tips, elbow) are manually selected along the boundary.
6. **Apply constant-speed parameterization:** The boundary is parameterized between the landmarks using constant speed parametrization over the unit interval.
7. **Lift the 2D boundary to 3D:** The 2D curve is lifted back to 3D by projecting onto surface near the average z-coordinate and then shift the closest vertices to the projected points.
8. **Make the 3D boundary neighbor-connected:** To allow LSCM to operate properly, the lifted curve is connected into a watertight boundary loop.
9. **Compute UV maps with LSCM:** Using the shared boundary as a constraint, each mesh is mapped to 2D space using Least Squares Conformal Mapping.
10. **Compute 3D shape correspondences:** For each UV coordinate in Shape 1, barycentric interpolation is used in the UV triangle of Shape 2 to obtain the corresponding 3D point.



C.4. Citation

If you use this dataset, please cite the accompanying thesis:

> N. van Veen. *Shape Correspondences and Example-Based Modelling for Boomerang Design*. MSc. Thesis, Delft University of Technology, 2025.

Abbreviations of Boomerangs

Abbreviation	Definition
AR	Australian Round
FC	Fast Catch
END	Endurance
ACC	Accuracy
TC	Trick Catch
LD	Long Distance
MTA	Maximum Time Aloft
Misc.	Miscellaneous

Additional Information on Boomerangs Dataset

Most models can be viewed in the gallery on `www.boomerangsbynils.com/product-category/nils-collection/`

Name	Type	Thickness	Distance	Landmarks	Included	File	Comment
Two-bladed boomerangs							
Typhoon-XL 120%	AR	3mm	45m	Variable	Yes (base)	boomerang_01.obj	
3D3	AR	3mm	45m	4	Yes	boomerang_02.obj	
3DX	AR	3mm	40m	4	Yes	boomerang_03.obj	
Fuzzy Frido	AR	6mm	50m	6	Yes	boomerang_04.obj	α-shapes inaccuracy in inner elbow
Fuzzy Mark	AR	5mm	50m	6	Yes	boomerang_06.obj	
Fuzzy Phoenix	AR	5.3mm	55m	6	Yes	boomerang_05.obj	
Woodie	Misc.	6mm	30m	6	Yes	boomerang_07.obj	Significantly bigger and bad area and shear distortion
Dragonfly 2	AR	3mm	60m	4	Yes	boomerang_08.obj	
Kick A2	AR	3mm	50m	4	Yes	boomerang_09.obj	
Ayr Adj	AR	3mm	50m	4	Yes	boomerang_10.obj	
Pisang	AR	3mm	50m	4	Yes	boomerang_11.obj	
Hummingbird	AR	3mm	50m	4	Yes	boomerang_12.obj	
Deevee	AR	3mm	50m	4	Yes	boomerang_13.obj	
Fuji	AR	3mm	55m	4	No	unused/Fuji.obj	High avg. Area distortion due to one outlier
Marathon	AR	3mm	65m	4/6	No	unused/Marathon.obj	UV mapping issues near wing tips, more landmarks is better
Nabab	AR	3mm	55m	4	No		Heavily post-processed. Failed when UV mapping due to hole(s)
Acongua	FC	3mm	25m		No		Noisy Scan
Three-bladed boomerangs							
Rotor	AR	3mm	50m	6	Yes	boomerang_1.obj	
Rotor II	AR	3mm	50m	6	Yes	boomerang_2.obj	
Deborah	AR	3mm	50m	6	Yes	boomerang_3.obj	
Twisted or Non-Genus-0 Boomerangs							
Manu-Big Triblader	MTA	2mm	60m		No		Not flat
Braket	MTA	2mm	45m		No		Not flat
Beppy	MTA	2mm	20m		No		Not flat
Alex Triblader	MTA	2mm	65m		No		Not flat
Manu Flat	MTA	2mm	70m		No		Scanning inaccuracy near thin airfoil
Ninja Mark	END	4mm	22m		No		Not flat, not Genus 0
Guillem Injected	END	5mm	22m		No		Not flat, not Genus 0
IceRunner	END	4mm	22m		No		Not flat, not Genus 0
Aurora	FC	5mm	19m		No		Not Flat, note Genus 0

D Implementation Insights and Debugging Notes

D.1. Overfitting the Template Shape Due to Incorrect Correspondences

Even when input shapes are aligned using ICP, computing a mean shape via PCA can overly smooth localized variations-especially in the thickness direction-resulting in an averaged shape that is noticeably flatter than any of the individual inputs. This effect becomes more pronounced when thickness is the most variable dimension, or when correspondences are incorrectly matched in symmetric regions of the shape (e.g., one shape's correspondence lies halfway along the wing surface while another lies closer to the top, or when the correspondences are incorrectly UV mapped).

A practical test to verify the correctness of correspondences is to examine the resulting mean thickness. During the UV mapping stage, if the side classification is accidentally flipped (i.e., the top and bottom sides are reversed), the PCA mean shape may exhibit a thickness that is thinner than the minimum thickness of any individual shape in the dataset. This is physically implausible: the mean thickness should logically fall between the minimum and maximum values of the inputs.

Interestingly, similar issues are rarely mentioned in related works that rely on template-based shape models, such as [27, 2, 17, 5]. However, since averaging is intrinsic to PCA, such artifacts are expected not to occur assuming the correspondences are consistent.

Another key observation is that increasing the dataset size tends to amplify this over-smoothing effect. As more shapes with inconsistent or noisy correspondences are introduced, the mean shape tends to flatten further and deviate from realistic geometry. A particularly problematic consequence is the emergence of unrealistic cambering near the trailing edge of airfoils-resulting in shapes with structural features that no real boomerang in the dataset exhibits.

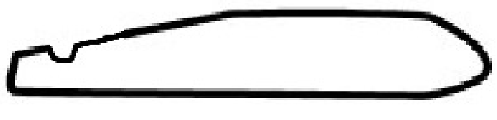


Figure D.1: Trailing edge cambering artifact in the PCA template shape.

This kind of deformation is especially visible near the wingtips, where the geometry is highly sensitive to small correspondence misalignments. Unlike Figure 6.12.3, where the camber is well-distributed and consistent with realistic airfoil shapes, the distortion in Figure D.1 is localized and unrepresentative.

In this thesis implementation, the issue has been resolved by enforcing consistent side classification during UV mapping, to avoid mapping a point from the top the the bottom of a boomerang. Nevertheless, this serves as an important debugging strategy and a diagnostic tool: when the mean thickness drops below the expected range, it likely indicates incorrect or flipped correspondences.

D.2. Convergence of α-Shapes Under Different Definitions

There are multiple ways to implement Alpha Shapes. In this work, the implementation from CGAL was used. However, when testing a custom implementation, it became clear that the conditions for determining whether a point is α-exposed can vary. For example, many approaches subtract circles from the shape outline that do not contain any data points (Figure D.2). Alternatively, a method based on the Delaunay triangulation can be used, where an edge is considered part of the alpha shape if it lies on a circle that passes through its two endpoints and contains no other points (Figure D.3).

Both approaches produce different results but converge reasonably well in practice—each following a slightly different geometric interpretation.

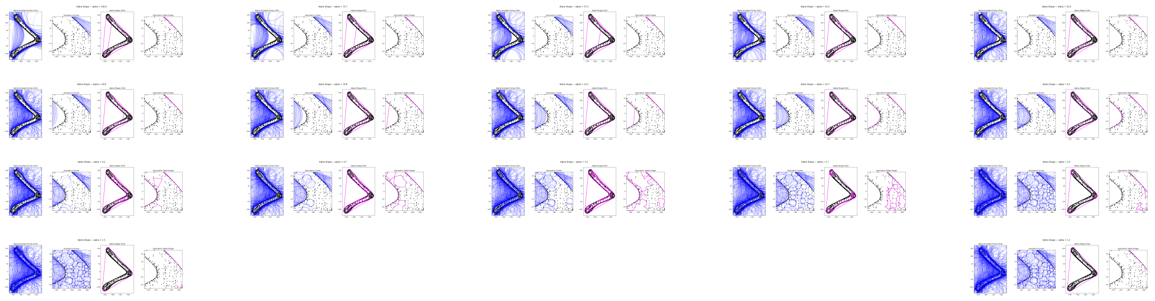


Figure D.2: Alpha shape Convergence when using method 1: subtract circles without any points in it

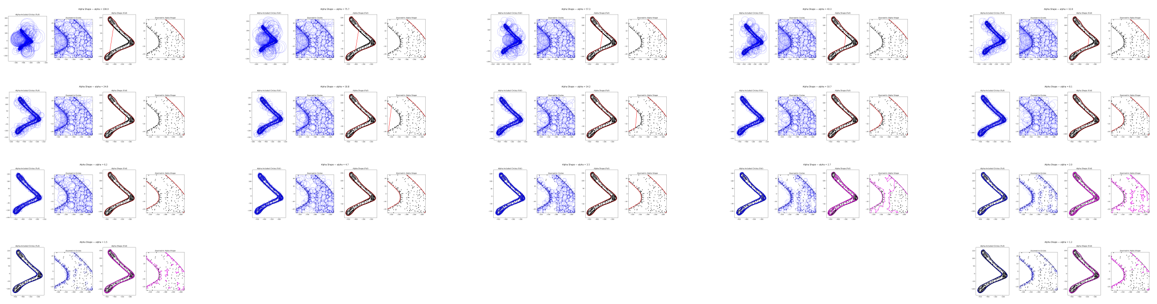


Figure D.3: Alpha shape Convergence when using method 2: subtract circles without 2 points of the Delaunay-triangulated edges in the boundary of the circle, and no other