

Learning Spectral Descriptors for Shape Analysis

Oguzhan Yigit

Doctor of Philosophy

UNIVERSITY OF YORK
COMPUTER SCIENCE

December 2025

Abstract

The Laplace-Beltrami Operator (LBO) is a cornerstone of geometry processing, valued for invariance under isometric transformations, a discrete orthonormal spectrum, and its ability to encode intrinsic manifold geometry via geodesic distances, proving effective for point-to-point correspondence on recent benchmarks. Yet it is as fragile as it is powerful: even near-isometric deformations can significantly alter its eigenbasis, and few methods backpropagate loss through it, given its complexity.

This thesis introduces a learning-based framework that constructs mesh operators by modifying the system matrices governing the LBO eigenbasis, yielding a data-adaptive LBO robust to near- and non-isometric deformations. The work unfolds in three stages: a neuroevolutionary approach probing mass-matrix weighting; LBONet, a learnable architecture adjusting the system matrices for task-dependent spectral behaviour; and LBONet embedded unsupervised, via functional maps and ZoomOut, yielding a fully intrinsic correspondence framework without supervision.

Operating mainly in the intrinsic domain, with rotation-invariant extrinsic measures at the input, the pipeline is largely unaffected by the embedding and invariant to rigid transformations, properties coordinate-based approaches typically approximate via data augmentation. Inverted orientations may thus lead such models to misread features as noise, whereas intrinsic methods offer a more stable representation, motivating a notion of shape understanding invariant to embedding and rigid transformations.

Custom sparse routines accelerate backpropagation through the eigenbasis, and combined with growing Graphics Processing Unit (GPU) memory on consumer hardware, make the framework practical and broadly applicable. Experiments on global tasks (SHREC*, SHAPENET) and local tasks (SHAPESEG, FAUST, SMAL, DT4D) show consistent improvements over existing methods.

To Ceren,
My truest love,
You are the plankton of my heart,
The pulse that nourishes every part.

Acknowledgements

I must thank various Vision, Graphics and Learning Research Group (VGL) members at the Department of Computer Science at the University of York for their insightful comments and inspiration during the presentation of parts of this accumulated work at our group meetings and other venues. Furthermore, I would like to thank my direct predecessor, Frederico Limberger, who laid the foundation for the presented work through his excellent publications and PhD thesis, and who kindly provided me with his code, which formed the basis of the earlier stages of this research. I am also grateful to Nick Pears, who gave important directions at specific checkpoints and challenged the presented work to ensure its methodological precision. I thank my supervisor, Richard C. Wilson, who guided me throughout the years, taught me to become an independent researcher, and inspired me to approach problems with uncompromising thoroughness.

Lastly, I must thank both Wolfgang Neumann and Cornelia Scheuring for having a lasting impact on my early career, inspiring me to pursue research and to remain hungry for discovering new knowledge.

Declaration

I declare that this thesis is a presentation of original work and I am the sole author. This work has not previously been presented for a degree or other qualification at this University or elsewhere. All sources are acknowledged as references.

Parts of this thesis has been published in journals and submitted for review,

- Yigit, Oguzhan, and Richard C. Wilson. "LBONet: Supervised Spectral Descriptors for Shape Analysis." Institute of Electrical and Electronics Engineers (IEEE) Transactions on Pattern Analysis and Machine Intelligence (2025).
- Yigit, Oguzhan, and Richard C. Wilson. "IntrinsicFM: Unsupervised Intrinsic Operator Learning for Functional Maps." **Submitted.**

Contents

List of Figures	xv
------------------------	-----------

List of Tables	xxix
-----------------------	-------------

1 Introduction	1
1.1 Problem Statement	1
1.2 Objective	5
1.3 Intrinsic Shape Understanding	6
1.4 Contributions	8
1.5 Thesis structure	11
2 Background	13
2.1 Mathematical Background	13
2.1.1 Laplace–Beltrami Operator	13
2.1.2 Spectral Signatures	16
2.1.3 Triangle Mathematics	19
2.1.4 Trigonometric gradients	20
2.1.5 Triangle Positioning Under Diffusion Constraints	21
2.1.6 Nelson’s Method	24
2.1.7 Functional Maps	28
2.2 LBO Methods	30
2.2.1 Anisotropic LBO	31
2.2.2 Kinetic LBO	32
2.2.3 Riemannian metric optimization	34
2.2.4 HodgeNet	34

2.3	Deep Learning Methods	35
2.3.1	Spectral Convolutional Neural Networks (CNNs)	35
2.3.2	Spatial CNNs	35
2.3.3	Feature Learning	37
2.3.4	DeepFM	38
2.3.5	ZoomOut	39
3	Neuroevolutionary Laplace-Beltrami Operator (LBO)	41
3.1	Introduction	41
3.2	System Overview	46
3.2.1	Overall Aim of the Technology	46
3.2.2	Inputs and Outputs	47
3.2.3	How It Is Trained	47
3.2.4	Evaluation in Inference	48
3.3	Manifold Geometry and Feature Extraction	49
3.4	Neuroevolution	53
3.5	ELBO	57
3.6	ELBO-Net	59
3.7	Evaluation	60
3.7.1	ELBO	63
3.7.2	ELBO-Net	74
3.7.3	SHREC'14	75
3.7.4	SHREC'15	76
3.8	Conclusion	77
4	LBONet: Supervised Spectral Descriptors for Shape Analysis	81
4.1	Introduction	81
4.2	System Overview	84
4.2.1	Overall Aim of LBONet	84
4.2.2	Inputs and Outputs	84
4.2.3	Training Phase (Supervised End-to-End)	85

Contents

4.2.4	Evaluation and Inference Pipeline	85
4.3	Approach	86
4.3.1	Building blocks	86
4.4	Deep Geometric Processing Layout	89
4.5	Intrinsic Feature Engineering and Invariance	90
4.5.1	Sensitivity Analysis of Spectral Descriptors	92
4.5.2	Mass and Stiffness matrix	93
4.5.3	Learning on mesh edges	93
4.5.4	Learning on mesh faces	96
4.5.5	Learning on mesh vertices	103
4.5.6	LBONet Architecture	103
4.5.7	Learning Stability and Regularization	105
4.5.8	Implementation	106
4.6	Evaluation	108
4.6.1	Retrieval	108
4.6.2	Classification	110
4.6.3	Segmentation	110
4.6.4	Functional Correspondence	113
4.6.5	Ablation Study	114
4.7	Conclusion	117
5	IntrinsicFM: Unsupervised Intrinsic Operator Learning for Func-	
	tional Maps	121
5.1	Introduction	121
5.2	System Overview	122
5.2.1	Overall Aim of IntrinsicFM	122
5.2.2	Inputs and Outputs	123
5.2.3	How It Is Trained (Unsupervised Iterative Loop)	123
5.2.4	Evaluation and Inference Pipeline	124
5.3	Background	124
5.4	Related Work	129

5.5	Background	130
5.6	Near-Isometry and Non-Isometry	131
5.7	Approach	132
5.7.1	Building Blocks	134
5.7.2	Estimating Functional Maps	136
5.7.3	Intrinsic Learning for Functional Maps	144
5.7.4	Accurate Functional Maps	145
5.7.5	Refining Functional Maps	146
5.7.6	LBONet	148
5.7.7	RefineNet	148
5.8	Evaluation	149
5.8.1	Near-Isometric Shape Correspondence	151
5.8.2	Non-Isometric Shape Correspondence	151
5.9	Supervised vs. Unsupervised Learning	152
5.10	Conclusion and Limitations	161
6	Conclusions	165
6.1	Intrinsic and Extrinsic Features for Spectral Learning	165
6.2	Optimization in Non-Smooth Spectral Landscapes	166
6.3	Contributions	167
6.4	Results and Insight	168
6.5	Limitations	169
6.6	Future Work	169
Appendices		
A	Gradient Verification	173
A.1	Numerical-Analytical Gradient Verification	173
A.1.1	Operators Check	174
A.1.2	Eigenbasis Check (Nelsons Method)	177
A.1.3	Spectral Descriptor Check	180

Contents

Abbreviations	183
Glossary	185
References	187

List of Figures

1.1	The School of Athens by Raphael [3]. In the center, Aristotle points towards the earth, implying that everything is worldly and takes on many different forms, whereas Plato, on the left, points upwards, alluding to his Theory of Ideas, where behind every concept, a pure form or shape is manifested.	2
1.2	Penrose [4] illustrating Platos Theory of Ideas. The LBO eigenbasis can be interpreted as analogous to the Platonic world; however, near- and non-isometric deformations disrupt this correspondence. Traditional deep learning primarily addresses relationship 2, learning the physical variations of an object and mapping them to a class in the Mental world. In contrast, the LBO eigenbasis inherently captures an objects intrinsic nature. Learning within the LBO eigenbasis, therefore, focuses on relationship 3, aiming to restore the objects intrinsic structure to its archetype by mitigating factors that disrupt this correspondence.	4
1.3	(a) An Rocket-Propelled Grenade (RPG) example where the handle and scope, correctly part of the body, are confused with rocket blades by coordinate-based methods. (b) A rocket instance where reliance on spatial coordinates leads to false feature attribution. (c) Distance from LBONet’s Heat Kernel Signature (HKS) descriptors to the centroid of the "blades" region, compared to the baseline. (d) Distance to the main body of the manifold. (eh) Corresponding plots on a second manifold, confirming the consistency of LBONets intrinsic representations.	7

2.1	Voronoi cell on vertex i , which is outlined in red. The Voronoi cell can be calculated using the illustrated angles and the distances shown.	14
2.2	Third eigenvector ϕ_3 of the LBO eigenbasis on two manifolds. Note that one eigenvector is flipped.	17
2.3	A sample triangle $\triangle(i, j, k)$ scaled with its centroid (red dot).	19
2.4	A degenerate triangle $\triangle(i, j, k)$ with its centroid (red dot).	19
2.5	A side (ij) of triangle $\triangle(i, j, k)$ is scaled and an angle (α) changes.	20
2.6	A side (ij) of triangle $\triangle(i, j, k)$ is scaled and another angle (β) changes.	21
2.7	A triangle $\triangle(i, j, k)$ scaled uniformly by its centroid (red dot).	21
2.8	A triangle $\triangle(i, j, k)$ scaled non-uniformly starting with edge ik .	22
2.9	A triangle $\triangle(i, j, k)$ scaled non-uniformly starting with edge ij .	23
2.10	Three triangles $\triangle(i, j, k)$ scaled in different order ij , ik and jk , resulting in triangles with different coordinates.	24
2.11	Differentiation through the LBO eigenbasis produces a gradient with respect to the Riemannian metric g_{ij} . Blue indicates negative values, while red indicates positive values.	28
2.12	Functional matrix between an intra-class manifold pair (left blue and top blue) and a inter-class manifold pair (left blue and top purple). One can observe two things, first, the functional matrix deviates from the diagonal even with mild non-isometries and second, that with greater deformation, the matrix deviates even further.	29
3.1	Rastrigin function shown as a filled contour plot (red: high values, blue: low values). The global minimum at $(x = 0, y = 0)$ is challenging to locate for standard optimization techniques due to the functions highly nonconvex landscape.	42

List of Figures

3.2	Rosenbrock function with added Gaussian noise, visualized to represent a curved, multi-modal optimization landscape. Although the landscape appears simpler than the Rastrigin function, the large, narrow valley makes locating the global minimum challenging. Gaussian noise is introduced to simulate eigenvalue crossings and other artifacts encountered in practical optimization scenarios. . . .	43
3.3	Top-down view of the Rosenbrock function with Gaussian noise. The default LBO eigenbasis basin is situated within many local optima along the valley. Although better basins exist, they are difficult to reach due to the presence of numerous intervening local minima. . .	45
3.4	ELBO-Net Architecture. The architecture consists of two stages: optimization of the spectral descriptor (orange) and feature encoding (blue). Vertex-wise features are first computed on the input mesh and forwarded into an MLP, which is evolved using the NeuroEvolution of Augmenting Topologies (NEAT) algorithm. This stage produces a weighting term for each vertex, which is applied to the corresponding Voronoi cell, followed by the computation of a spectral descriptor. The descriptor is then passed through a neural network that encodes the features point-wise into higher-dimensional representations. These features are aggregated using second-order pooling and subsequently undergo dimensionality reduction, resulting in the final k -dimensional descriptor.	47
3.5	Mean curvature visualized on two shapes from the DT4D benchmark. It highlights near-isometric deformations around joint regions, such as the intersections between legs and torso. However, as a non-intrinsic feature, mean curvature reduces the invariance property of the LBO eigenbasis embedding.	50

3.6	Gaussian curvature visualized on two shapes from the DT4D benchmark. As an intrinsic feature, it preserves all properties of the LBO eigenbasis. Although it varies between poses, the observed changes primarily indicate near-isometric deformations rather than bending.	51
3.7	Standard heat kernel signature at the first timeframe for two shapes from the DT4D benchmark. Based on the LBO eigenbasis, it is fully intrinsic and preserves all properties and benefits of the LBO embedding.	52
3.8	Different extrinsic and intrinsic features of a manifold of the Shape Retrieval Challenge (SHREC)'14 Shape retrieval of non-rigid 3D human models [51] training set. From left to right: Maximum curvature, minimum curvature, Gaussian curvature, Shape Index, and Curvature Index deviation.	57
3.9	Boxplot showing ELBO values over successive generations trained using benchmark metrics on SHREC'10. The median quickly exceeds the baseline LBO by the fifth generation, demonstrating rapid improvement, and the optimal ELBO is reached around the sixteenth generation.	61
3.10	Boxplot of ELBO values across generations for networks trained with contrastive loss using landmarks on SHREC'14 (real). The median reaches the baseline LBO by the seventeenth generation. The observed performance exhibits a wider range, reflecting the reduced regularization compared to training with full benchmark metrics. . .	61

List of Figures

- 3.11 Evolved network topology and weights obtained via NEAT on SHREC'10.
The network uses a sigmoid activation function, and the output neuron performs regression. Notably, skip connections are extensively employed in the neuroevolutionary design. One could argue that because descriptors are already achieving a good performance on this benchmark, that the topology remains trivial. However, since the descriptors already achieve strong performance on this benchmark, the resulting topology remains relatively simple. 64
- 3.12 Evolved network topology and weights obtained via NEAT on SHREC'14 (real). The network uses a sigmoid activation function, and the output neuron performs regression. Multiple hidden layers and numerous skip connections are observed, reflecting the increased complexity required to achieve strong performance on this benchmark. 65
- 3.13 Evolved network topology and weights obtained via NEAT on SHREC'14 (synthetic). The network uses a sigmoid activation function, with a regression output neuron. Multiple hidden layers and extensive skip connections are present. Notably, the fifth input feature passes through three activation functions before reaching the output neuron. 66
- 3.14 Side-by-side comparison of ELBO Scale-Invariant Heat Kernel Signature (SI-HKS) and standard LBO SI-HKS on the SHREC'14 Real benchmark. ELBO clearly enhances intra-class similarity while preserving inter-class discriminability. Shapes belonging to the same class are highlighted with red borders. To enhance clarity, the heatmap is shown with 11 classes, each containing 6 representative shapes. 68
- 3.15 Standard heat kernel signature at 1st timeframe on two shapes from the DT4D benchmark. The standard heat kernel signature is based on the LBO eigenbasis, which makes it fully intrinsic and retains all LBO eigenbasis benefits. 70

3.16	ELBO HKS at the first timeframe on submatrix shapes from SHREC'14. Differences in the superimposed submatrix, notably between shapes 4 and 6 and shapes 1 and 6, are reduced compared to the standard HKS, with ELBO partially mitigating discrepancies caused by near- isometric deformations around the shoulders.	71
3.17	Drilldown of the two most discrepant shapes within a class, comparing their standard and ELBO HKS. The ELBO descriptor not only appears visually more stable but also reduces the numerical distance between the shapes, resulting in improved performance.	72
3.18	Features evolved via ELBO for weighting the mass matrix. Specific areas, notably around the armpits, are weighted to enhance intra-class similarity across shapes.	73
4.1	LBONet comprises multiple modules introduced in this thesis, which operate directly on the input mesh. RiemannNet weights the Rie- mannian metric on a manifold, ALBONet introduces anisotropy by rotating and scaling the diffusion speed, and VoronoiNet weights the Voronoi cells. These modules affect the Laplace-Beltrami operator in several different ways, allowing it to adapt to the given data and alleviate performance breakdowns incurred by non-isometric deformations.	83

4.2	LBONet Architecture. LBONet takes features on mesh edges, vertices, and faces as input of its three building blocks, RiemannNet, ALBONet and VoronoiNet. Each block consists of one EdgeConv which is creating a local neighbourhood graph as in [6] and learns an neighbourhood function. Then the features are passed through multiple shared MLPs to predict either the Riemannian metric on an edge, the anisotropy on a face or the Voronoi cell weighting at a vertex. These predictions are applied directly on the discretized LBO, modifying the stiffness and mass matrix, which we detail in Section 4.5 onwards. Its eigenbasis is then used to form a spectral descriptor, which then can be passed into a "backend" for further processing.	84
4.3	Functional matrix between an intra-class manifold pair (left blue and top blue) and a inter-class manifold pair (left blue and top purple). One can observe two things, first, the functional matrix deviates from the diagonal even with mild non-isometries and second, that with greater deformation, the matrix deviates even further.	87
4.4	Edge flap surrounding edge ij . Adjusting the Riemannian metric, which is highlighted in red, has an impact on all the angles and distances shown and are hence part of the gradients introduced. . .	94
4.5	A degenerate triangle $\triangle(i, j, k)$	95
4.6	A degenerate triangle $\triangle(i, j, k)$	96
4.7	Effect of varying the neighborhood size in RiemannNet. The numbers above indicate the specific neighborhood size used for each visualization. Yellow denotes a high degree of stretching, whereas blue indicates the shortening of the corresponding edge.	96

4.8	Orthonormal reference frame on a face depicted by v_m and v_M based on the principal curvature directions, which is highlighted red. Scaling the directions v_m and v_M by a can change the anisotropy on the given face, which results in scaling dot product, between the corresponding edge and the direction.	97
4.9	Orthonormal reference frame on a face depicted by v_m and v_M based on the principal curvature directions, which is highlighted blue. It is rotated by θ , which is depicted red. Besides, scaling the directions v_m and v_M rotating the directions has a direct impact on the used dot products.	98
4.10	Effect of varying neighborhood size in ALBONet Anisotropy factor 1 (part of LBONet). The numbers above indicate the neighborhood size used in each visualization. Red denotes an acceleration of anisotropy, whereas blue indicates a deceleration.	99
4.11	Effect of varying neighborhood size in ALBONet Anisotropy factor 2 (part of LBONet). The numbers above indicate the neighborhood size used in each visualization. Red denotes an acceleration of anisotropy, whereas blue indicates a deceleration.	99
4.12	Plot of the hyperbolic tangent function scaled by π : $y = \pi \tanh(x)$. This allows the learning of the rotation in both directions effectively and flattens out as it approaches half a revolution.	101
4.13	Effect of varying neighborhood size in ALBONet Rotation parameter θ (part of LBONet). The numbers above indicate the neighborhood size used in each visualization. Red denotes an acceleration of anisotropy, whereas blue indicates a deceleration. Arrow illustrates the Θ	101
4.14	Voronoi cell on vertex i , which is outlined in red. The Voronoi cell can be scaled and has only impact on the local Voronoi cell.	103

List of Figures

4.15	Effect of varying neighborhood size in VoronoiNet (part of LBONet). The numbers above indicate the neighborhood size used in each visualization.	104
4.16	HKS at selected points evaluated with the standard LBO and with the LBONet trained on the SHREC11 benchmark. Values are given as the difference of the heat kernel at the highlighted points.	112
4.17	Texture transfer experiment on the FAUST test set. Source illustrates the texture applied to the source shape, and target the texture mapped via the ground-truth correspondence to the target shape. Followed by the illustrations of the predictions of DiffusionNet, Unsupervised, and LBONet.	114
4.18	Ground-truth segmentation of a manifold and learned features on a test manifold from the human segmentation benchmark by Rie- mannNet (second), ALBONet (third and fourth), and VoronoiNet (fifth). Interesting patterns emerge along the segmentation bound- aries, which demonstrates that individual modules are stronger in certain aspects.	117
4.19	HKS features obtained from the ablation study done in chapter 5. Top shows the HKS evaluated with the standard LBO and bottom the HKS evaluated with LBONet. The features were reduced from 16 dimensions to 2D with t-SNE.	118
4.20	Precision/Recall curve of LBONet, Baseline, LBONet HKS, and the LBO HKS evaluated on the SHREC'14 'real' benchmark (top) and SHREC'14 'synthetic' benchmark (bottom). Distances were calculated using Euclidean distance. The performance of the spectral signatures for both LBONet (HKS) and LBO (HKS) were determined by using average pooling.	119

- 5.1 **IntrinsicFM Overview.** Given shapes X and Y , IntrinsicFM learns deformation-aware spectral operators in a fully unsupervised closed-loop framework. **Operator Learning (Left):** LBONet predicts adapted stiffness and mass matrices yielding eigenbases Φ, Ψ ; RefineNet augments these with spectral sensitivities $(\partial\Phi/\partial\mathbf{v}, \partial\Psi/\partial\mathbf{v})$ for robust vertex matching. **Initialization (Top):** ZoomOut estimates an initial functional map $\hat{C}_n$. **Eigenbasis Learning (Bottom):** Filtered correspondences construct refined maps (T_{12}, T_{21}) that supervise spectral alignment via bidirectional projection error, forming an iterative cycle that adapts the LBO to the correspondence objective without ground-truth supervision. 122
- 5.2 **Error Analysis: Spectral Truncation vs. Non-Isometry.** Top/bottom rows show correspondence error before/after IntrinsicFM. *Spectral Truncation* ($k = 64$): projection error from missing high-frequency components remains spatially structured and moderate. *Non-Isometry*: geometric deviations cause significantly larger spectral misalignment and widespread geodesic error. IntrinsicFM reduces error in both regimes, with pronounced gains under non-isometric deformation where classical LBO assumptions break down. Colorbar: geodesic loss (None $\rightarrow$ High). 125
- 5.3 **Geodesic Error.** Fox and horse in the IntrinsicFM LBO eigenbasis (left) and the standard LBO eigenbasis (right). Geodesic error is computed as the geodesic distance between the predicted vertex correspondence and the ground-truth correspondence. 133

List of Figures

5.4	Geodesic error between two manifolds with varying eigenfrequency. The left column shows the prediction of an unseen manifold using our approach, while the right column shows the standard LBO eigenbasis. The color scale ranges from blue (indicating smaller error) to red (indicating greater error). Both manifolds share the same color scale for precise comparison. The diagram below illustrates performance based on the number of eigenfrequencies used. Our approach significantly outperforms the baseline, achieving better results with only 64 eigenfrequencies, compared to the baseline LBO, which uses 128 eigenfrequencies. Furthermore, it is evident that learning transfers across frequencies: as more frequencies are used, the accuracy improves. Even with just 64 eigenfrequencies, the majority of the relevant information is already captured.	134
5.5	Non-isometric Shape pair used in the plotted functional maps. . . .	137
5.6	Side-by-side comparison of naive Procrustes functional map estimation versus ground truth for a non-isometric pair.	139
5.7	Side-by-side comparison of enhanced naive procrustes functional map estimation versus ground-truth of a non-isometric pair.	140
5.8	Side-by-side comparison of enhanced naive procrustes functional map estimation versus ground-truth of a non-isometric pair.	140
5.9	Side-by-side comparison of enhanced naive procrustes functional map estimation versus ground-truth of a non-isometric pair.	141
5.10	Further non-isometric functional map estimations on SMAL.	142
5.11	Further non-isometric functional map estimations on DT4D.	143

- 5.12 Pruning effects on the point-to-point map. To iteratively refine the correspondence and exploit improvements in the LBO eigenbasis, we identify drifting points, i.e., points whose spectral distance increases despite the overall enhancement of the eigenbasis. This apparent contradiction provides a useful signal: such points can be selectively pruned from the point-to-point map and reassigned to alternative matching candidates, enabling a more stable and accurate correspondence. The standard ZoomOut performance is shown in purple. The refined correspondence produced by IntrinsicFM is indicated by the red curve. The blue curve ($\dagger$) represents the spectrum induced by the initial ZoomOut point-to-point map under the updated LBO eigenbasis. It can be clearly observed that this prediction is no longer compatible with the modified eigenbasis, exposing false correspondences. 147
- 5.13 **Near-Isometric Qualitative Results.** IntrinsicFM trained on the SCAPE train set and evaluated on the SCAPE test set. The top row shows the IntrinsicFM Eigenbasis prediction, while the bottom row shows the standard LBO Eigenbasis. Pairs are rendered side by side for comparison. IntrinsicFM consistently improves correspondence accuracy across poses and manifolds. Topological changes, such as intersections, remain challenging (see the second figure from the right), likely due to reliance on spatial proximity during training, which can introduce false positives in intersecting meshes. Preprocessing the training data could help mitigate this limitation. 150
- 5.14 **Near-isometric matching on (Left) SCAPE and (Right) FAUST.** Our method substantially outperforms existing state-of-the-art methods. 152
- 5.15 **Non-isometric matching on (Left) SMAL and (Right) DT4D-H.** Our method substantially outperforms existing state-of-the-art methods. 153

List of Figures

5.16 Spectral sensitivity computed by RefineNet. Although this is a high-dimensional feature, it provides valuable insight into the richness and complexity of the representation. Spectral sensitivity addresses the question of how infinitesimal changes in vertex positions affect the LBO eigenbasis. This information can be leveraged to refine functional maps, assist in their upscaling, and improve initial functional map estimation. While many regions on a manifold may appear spectrally similar, their influence, meaning the gradient of the eigenbasis with respect to vertex perturbations, can differ significantly. Capturing this sensitivity is therefore crucial for disambiguating such regions and enabling more precise correspondence localization. . . .	154
5.17 RefineNet vs. ZoomOut. When RefineNet is employed to further enhance k-nearest neighbor search, it consistently outperforms the standard ZoomOut method.	155
5.18 Comparison of supervised and unsupervised Voronoi results.	155
5.19 Comparison of supervised and unsupervised Anisotropy factor α_1 results.	156
5.20 Comparison of supervised and unsupervised Anisotropy factor α_2 results.	157
5.21 Comparison of supervised and unsupervised Anisotropy Rotation θ results.	158
5.22 Comparison of supervised and unsupervised geodesic error.	159
5.23 Comparison of supervised and unsupervised Voronoi weighting results	160
5.24 Comparison of supervised and unsupervised Anisotropic factor α_1 results.	161
5.25 Comparison of supervised and unsupervised Anisotropy factor α_2 results.	162
5.26 Comparison of supervised and unsupervised Anisotropy Rotation θ results.	163
5.27 Comparison of supervised and unsupervised geodesic error.	164

A.1	Sample mesh used to validate the introduced gradients. From left to right: VoronoiNet, RiemannNet, and ALBO+Net. Red indicates higher values while blue indicates lower values. All gradients are computed using an LBO eigenbasis with 32 frequencies to simulate a realistic benchmarking scenario.	182
-----	---	-----

List of Tables

3.1	Comparison of curvature quantities, Shape Index, Curvature Index, and Heat Kernel Signature, highlighting type and invariance.	54
3.2	Results on the SHREC'10 benchmark. ELBO and KLBO refer to the respective weighting schemes. Features were encoded using a super vector and either SI-HKS or Wave Kernel Signature (WKS) descriptors.	63
3.3	Results on the SHREC'14 (real) benchmark in comparison to State of the Art (SOTA) methods. ELBO and KLBO refer to the respective weighting schemes. Features were encoded using a super vector and either SI-HKS or WKS descriptors.	66
3.4	Results on the SHREC'14 (synthetic) benchmark in comparison to SOTA methods. ELBO and KLBO refer to the respective weighting schemes. Features were encoded using a super vector and either SI-HKS or WKS descriptors.	67
3.5	Results on SHREC'14 Real Dataset (in %)	75
3.6	Results on the SHREC'14 (synthetic) dataset (in %). ELBO-Net does not reach state-of-the-art performance, primarily due to the disconnected training between the ELBO and its backend network.	76
3.7	Results on the SHREC'15 dataset (in %). ELBO-Net underperforms compared to KLBO, suggesting that while the descriptor may be more discriminative, its backend does not achieve the same level of performance.	77
4.1	Backpropagation runtimes on varying mesh resolution on a RTX 4090 Graphics Processing Unit (GPU) given in seconds.	107

4.2	Results on the SHREC'11 dataset (in %). Metrics are computed using the evaluation protocol introduced in the Princeton Shape Benchmark [50].	109
4.3	Results on SHREC'14 'real' Dataset (in %). Metrics are computed using the evaluation protocol introduced in the Princeton Shape Benchmark [50].	109
4.4	Results on SHREC'14 'synthetic' Dataset (in %). Metrics are computed using the evaluation protocol introduced in the Princeton Shape Benchmark [50].	110
4.5	Results on SHREC'15 Dataset (in %). Metrics are computed using the evaluation protocol introduced in the Princeton Shape Benchmark [50].	110
4.6	Results on SHREC'11 Dataset (in %)	111
4.7	Results on SHREC'15 Dataset (in %)	111
4.8	Segmentation accuracy on humans and COSEG benchmarks (in %)	112
4.9	Results on the FAUST [78] benchmark. Values are given in geodesic mean error x 100. Result marked with † serves as the baseline method.	113
4.10	Results on the human segmentation benchmark. The table reports the performance of the proposed modules, both individually and in combination. A ✓ symbol indicates that the corresponding module was used to obtain the reported result.	115
5.1	Geodesic Mean Error x 100 on the FAUST and SCAPE benchmark. All benchmarks were run on the remeshed versions of the meshes with ca. 5k vertices. Results below the dashed line show the ground-truth correspondence error with the given basis.	152
5.2	Geodesic Mean Error x 100 on SMAL and DT4D-H. Results below the dashed line show the ground-truth correspondence error with the given basis.	153

"Wir müssen wissen. Wir werden wissen." [1]

David Hilbert

1

Introduction

1.1 Problem Statement

In recent years, the computational modeling of 3D shapes has attracted considerable attention for several reasons. First, the increase in computational processing power has enabled the handling of higher-resolution models, which are now common in simulations and video games. Second, the growing accessibility of affordable consumer-grade 3D printers has raised public awareness and engagement with 3D modeling. Third, the surge of interest and investment in virtual and augmented reality, often grouped under the term metaverse, has further accelerated developments in this field. Beyond entertainment, 3D modeling has become essential in disciplines such as chemistry, biology, and medicine. Moreover, recent advances in transformer models, such as Trellis [2], have reignited excitement around generative modeling. Although most research has focused on understanding 3D on a deeper level, many tasks involving 3D remain challenging, especially tasks such as correspondence or segmentation, which require a highly precise local descriptor. Near- and Non-isometry, which describes a change in the surface, (from the Greek *isos*, meaning ‘same’), which occurs as soon as an object is stretched and therefore the distances on the surface altered. Near-isometry describes changes which are limited and can still

be attributed to the same class of objects, e.g. a female body would be near-isometric to a male body, no matter how thin or corpulent the bodies are, they remain close in their surface length. Non-isometry defines a much bigger change on the surface, s.t. the object at hand is significantly different from the compared object. Big enough to be classified as non-isometric, meaning not equal length. Both result in a difficult problem to solve as descriptors need to be aware of these transformations a manifold can undergo. Furthermore, descriptors must be invariant to the embedding of the shapes as well, as their orientation might not necessarily be fixed.



Figure 1.1: The School of Athens by Raphael [3]. In the center, Aristotle points towards the earth, implying that everything is worldly and takes on many different forms, whereas Plato, on the left, points upwards, alluding to his Theory of Ideas, where behind every concept, a pure form or shape is manifested.

Spectral descriptors leverage the spectrum of the shape and are therefore well suited to isometric deformations, where the shape and its parts (such as legs and arms) may rotate without altering intrinsic geometry. However, they struggle when deformations deviate from isometry and become increasingly non-isometric. This results in a drastic change in the spectrum of the shape, hence a different descriptor, losing its descriptive power. Further challenges are raised by topological deformations, i.e. if the surface changes s.t. new connections are formed, e.g. a

1. Introduction

human body touching its head. The touch connects, the hand to the head allowing the spectrum to be connected and resulting in an overall different descriptor.

Geometric descriptors however, do not suffer as much from these weaknesses, but have their own downsides, such as lacking global context or poor generalization. Lastly, *learned* descriptors can be trained to account for these changes in surface, however, it boils down to exhaustively training the framework to capture, the near- and non-isometry and topological changes. Furthermore, it needs to be trained excessively to have a decent enough performance, as learned descriptors do not have a notion of any intrinsic invariancy unless penalized in the loss function, as opposed to the isometry invariance in the Laplace-Beltrami Operator (LBO).

We can draw an analogy to the confrontation between Aristotle and Plato (Fig. 1.1), where their ideas that everything takes a unique shape and appears in many different forms, and that behind every concept a shape is manifested, respectively mirror spectral and geometric/learned descriptors. Spectral descriptors are based on the LBO, which is isometric and describes the shape in a more theoretical world, whereas geometric and learned operators pick up how the manifold appears in a worldly manner. This is a trade-off, and both ideas are powerful when executed properly, and this thesis will demonstrate that Platos Theory is powerful enough to achieve competitive results on challenging benchmarks (See Fig. 1.2, where the respective learning is taking place).

The discipline these algorithms are part of is coined *non-rigid shape analysis*. Non-rigid captures both near- and non-isometry and deformations tasks, where precise descriptors are required to execute the task at hand. The tasks are diverse and require different traits in the descriptor; correspondence and segmentation require more local descriptors, whereas retrieval and classification more global. While geometric and spectral descriptors used some kind of dictionary such as bag-of-words to pool down the descriptor for global tasks, learning based frameworks use pooling mechanisms, such as max-, average-, or 2nd order-pooling.

The idea of equipping spectral descriptors with learning capabilities has been widespread, and many approaches focus on addressing their limitations by using

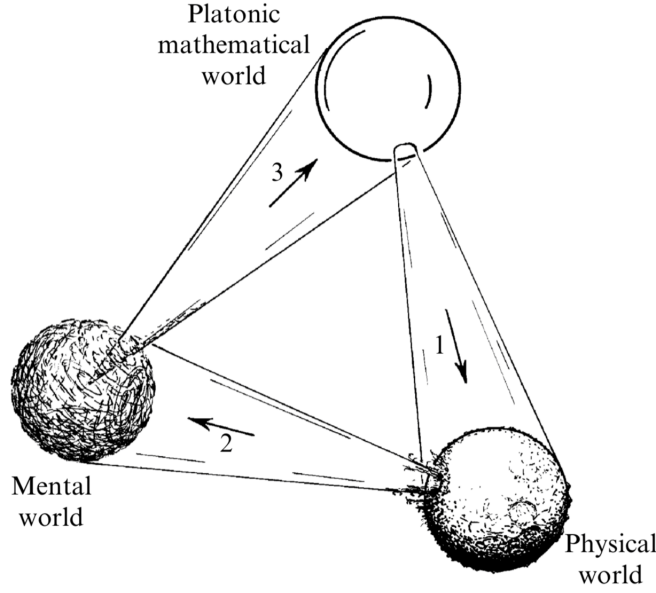


Figure 1.2: Penrose [4] illustrating Platos Theory of Ideas. The LBO eigenbasis can be interpreted as analogous to the Platonic world; however, near- and non-isometric deformations disrupt this correspondence. Traditional deep learning primarily addresses relationship 2, learning the physical variations of an object and mapping them to a class in the Mental world. In contrast, the LBO eigenbasis inherently captures an objects intrinsic nature. Learning within the LBO eigenbasis, therefore, focuses on relationship 3, aiming to restore the objects intrinsic structure to its archetype by mitigating factors that disrupt this correspondence.

spectral descriptors as input and learning different operators; see PointNet [5], Dynamic Graph Convolutional Neural Network (DGCNN) [6], and DiffusionNet [7] for reference. Some even attempt to learn on the operator level, modifying the operator used to construct the spectrum of the shape. This modification is far more fundamental than learning on top of the produced descriptor, but also much more difficult as the presented thesis will explore. The methods modifying the nature of the operator are however limited to a few examples and their application is limited. The presented corpus is focusing on how an ideal operator can be learned for all tasks with the given non- or near-isometric deformations, but the performance edge even goes beyond, as results in segmentation tasks demonstrate. Learning a spectral operator, which is not just invariant to isometry, but also near-isometry and to non-isometry, comes with difficult challenges. Firstly, the eigenbasis used to construct a spectral descriptor is differentiable, but complex and expensive in computation. Secondly, learning in the spectral domain is not generalizable as in

1. Introduction

the spatial domain, due to various reasons, one of which is the drastic changes near- and non-isometry causes in the spectrum of a shape. Another reason is that the eigenbasis itself has various weaknesses, such as the swapping of eigenvectors, the order in which the eigenvectors can vary from pose to pose, let alone from shape to shape. Furthermore, the sign of the eigenvectors are non-deterministic, meaning that the sign can flip, from one computation to the other. All these issues have been tackled and solutions were proposed partially, but nothing adequately solving the problem of non-rigid shape analysis end-to-end.

1.2 Objective

Spectral descriptors are leveraged to compensate for their inherent shortcomings, resulting in more descriptive representations that are applicable to a broader range of tasks, including brain mapping and shape segmentation. However, a spectral descriptor, which would bring a deeper understanding of the shape and its potential near- and non-isometric deformations, would allow follow-up frameworks to learn quickly the gap left which the operator cannot cover. With a standard spectral descriptor, the learning framework is required to handle all near- and non-isometric variations, effectively placing the full burden of modeling such deformations on the backend and thereby limiting its applicability and coverage. The objective of the presented work is to learn an operator based on features derived from the input shapes, which accounts for near- and non-isometry as much as possible. In this way, the operator could become much more powerful and would allow the operator to be adjusted to the data and task at hand. This would enable to adapt the operator to the scale of the problem, be it local or global and to the data at hand. Manifolds can come in varying mesh resolution and triangulation, which can also affect the generalization, which this work also aims to eliminate. By this it would be possible to derive an operator which is more suited for any given task and invariant to the given discretizations of the shape. This would allow heavy back ends to further exploit the given data improving performance at any given task.

1.3 Intrinsic Shape Understanding

As the LBO is intrinsic, it describes the shape independently of its embedding. When the LBO is trained with intrinsic or semi-intrinsic features, the learned LBO remains intrinsic. Conversely, when using Extrinsic measures, i.e., raw coordinates of a manifold, the learned LBO is no longer fully intrinsic and suffers from the same phenomenon that frameworks using raw coordinates do: spatial bias can lead to false predictions, such as when a training example is flipped in orientation or when hallucinating segments that are not actually present. See figure 1.3 for a demonstration of spatial bias and how the presented method is able to avoid this by avoiding using coordinates as features. When the feature space is observed in 1.3 (a) and (c), it is evident that the distance to the segment, which depicts the blades of the rockets (green) in (b) and (f) has been increased and conversely the feature distance in (d) and (f) to the main body (blue) of the rocket has been decreased. The baseline is the default Heat Kernel Signature (HKS) and it is intrinsic to the manifold as it is derived directly from the LBO. However, as a feature retrieval network consuming the HKS it would need to learn to differentiate between the different attachments on the object, for example the blades in (b) and the handle and scope in (a). With the presented method however, this differentiation becomes easier for the feature retrieval network as the descriptor itself encodes these semantics and the HKS transitions from a default spectral descriptor to a highly-descriptive learned descriptor.

Frameworks using coordinates try to circumvent this by exhaustively learning to rotate and shift the manifolds; however, this leads to significantly longer training times. This technique reduces the problem, however the benefits of staying closer to intrinsic learning are clear. Therefore, we can conclude that intrinsic shape understanding excels precisely where intricate frameworks struggle the most and cannot reliably predict, as extrinsic measures, which break rotation or positional invariance give conflicting signals. Extrinsic signals offer shortcuts for frameworks, where positions are hard-wired to predictions. For example, in segmentation tasks,

1. Introduction

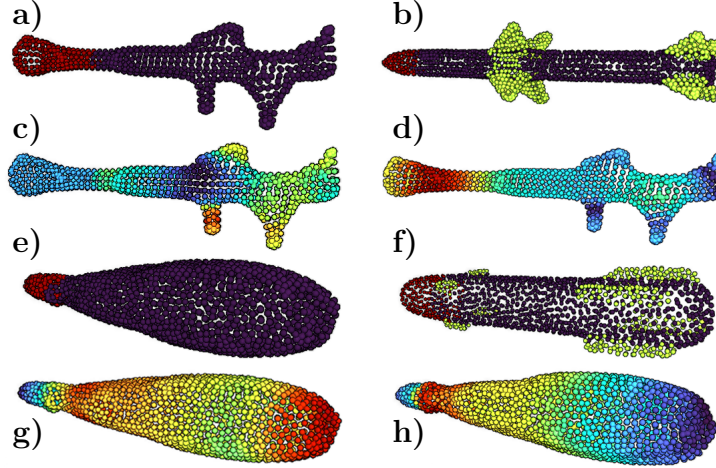


Figure 1.3: (a) An Rocket-Propelled Grenade (RPG) example where the handle and scope, correctly part of the body, are confused with rocket blades by coordinate-based methods. (b) A rocket instance where reliance on spatial coordinates leads to false feature attribution. (c) Distance from LBONet’s HKS descriptors to the centroid of the "blades" region, compared to the baseline. (d) Distance to the main body of the manifold. (eh) Corresponding plots on a second manifold, confirming the consistency of LBONet’s intrinsic representations.

the network can rely on position by reasoning: if this coordinate is high up, it’s most likely the top of the object. However, when the network is confronted with a rotated or flipped version of the object, it becomes confused. Likewise, when these external signals are used in this intrinsic setting, it leads to shortcut learning, where the trained LBO is not adding any meaningful value beyond raw coordinates, but when it is stripped of extrinsic features, it has nothing but intrinsic features, which describe the shape more fundamentally. Extrinsic features however are very descriptive when compared to intrinsic measures, therefore it is crucial to find the right balance between intrinsic and extrinsic measures, in order to give the neural network the ideal ground to learn effective features, without having to relearn positional and rotational invariance.

This thesis furthermore highlights the importance and role of a backend and its ability to backpropagate meaningful gradients. Most backends have an intrinsic notion of geometry, e.g. PointNet++ [8] divides the manifold into multiple segments and runs PointNet in each of them, DGCNN constructs a local neighbourhood graph, etc. This allows the network to deliver strong gradients for intrinsic learning.

Another prerequisite for successful learning is the establishment of appropriate regularization and the careful design of constrained loss functions, ensuring that the neural network learns meaningful representations rather than converging to spurious solutions that exploit random quirks. For instance, if the spectral descriptor were not normalized, the optimization process would likely be dominated by the tendency to increase the descriptors magnitude, rather than focusing on learning intrinsic and useful features. For instance, a random quirk could manifest as a significant increase in the Riemannian metric to boost the descriptors magnitude, inadvertently suppressing other valuable geometric patterns that the metric could encode. Intrinsic shape understanding is inherently more challenging to achieve, as many extrinsic features provide more immediate and seemingly stronger learning signals. However, once intrinsically grounded frameworks are able to capture meaningful geometric representations, they can easily surpass their extrinsic counterparts and exhibit greater robustness with respect to variations in the objects embedding.

1.4 Contributions

In this work, we analyze the spectrum of a shape and observe how changes in the system matrices (stiffness and mass) affect the LBO eigenbasis. We implement Nelson’s method for all variations, where all combinations of the system matrices are used depending on the operator at hand. We demonstrate the learning potential by introducing trivial but powerful operators, which allow us to adjust the eigenbasis to the task at hand. This allows us to overcome State of the Art (SOTA) methods and points towards an underexplored area in the spectral descriptor literature. The eigenvector swapping and sign-flipping problem has rendered the LBO eigenbasis too fragile to learn from as even on the same mesh, with minimally different weighting of the system matrices, the eigenbasis can look "different". A sign flip for example might encourage the network to ping-pong between weights and eigenvector swap might encourage the network to make dissimilar parts similar. Therefore, a large part of the focus has been put into solving this problem, by aligning the eigenvectors and sign-correcting them. However, with increasing non-isometry, this relation gets

1. Introduction

weaker to such a degree that it might not even be possible to align eigenvectors in higher frequency between an elephant and a lion, making learning more difficult in such cases. Moreover, learning in higher frequencies is challenging, as much of the higher frequencies blend noise with fine details on the manifold, but with increasing precision, this frontier gets more interesting, as algorithms work up to such detail. We apply multiple techniques to deal with these problems, allowing to learn effectively, both explicitly, when the gradients are given directly wrt. eigenvectors and implicitly when they are given wrt. spectral descriptors. Our results demonstrate that, even when learning is only done with a range of $1 \cdots \lambda_{64}$, it can outperform the standard LBO with a range of $1 \cdots \lambda_{128}$, meaning that with half the spectrum it is possible to do a more precise mapping in correspondence tasks. This alleviates the need to go to very high frequencies to a certain degree, as the same precision or even more is possible with fewer frequencies. Keeping the eigenbasis compact has also consequences for down-stream frameworks, as they can operate on a much smaller basis and learn more effectively. Even with a better learned LBO eigenbasis, the need to go into higher frequencies arises and in this work we will show that learning gets more tricky the higher we go in the frequencies. As the optimization problem itself is non-trivial, the need to differentiate through the Generalized Eigenvalue Problem (GEP) consists of multiple steps and has dense non-sparse matrix operations, we explore the problem first as a black box problem and optimize it with a neuroevolutionary approach (NeuroEvolution of Augmenting Topologies (NEAT)), which eliminates the need for having a differentiable loss. We show that it is possible to outperform the standard LBO in classification and retrieval tasks, however the computational cost is high as many offsprings need to be evaluated in parallel to achieve decent solutions. Furthermore, with added complexity in weighting the matrices, the solution space gets incredibly large, making the problem more difficult to optimize. Moreover, reproducibility is another key concern, as the evolution is non-deterministic. Later, we switch to a fully supervised approach, differentiating the problem using Nelsons method and implementing it into PyTorch in a scalable, vectorized and sparse way.

With so many variables in the pipeline from geometry up to a prediction of a class, segment, corresponding vertex etc. it is becoming increasingly important to ablate the performance of any gain achieved by a component. As our contribution itself is mainly based on a shared neural network architecture, the additional parameters introduced are negligible when compared with an overall pipeline. In the default setting, the introduced network contains around 41k trainable parameters, and if those are added to the baseline comparison they do not result in any gain. Therefore, we always omit to add this additional complexity from the baseline comparison.

Key contributions of the presented work that allow us to derive the aforementioned operators are:

- a fully-vectorized, scalable, and sparse backpropagation routine incorporating both stiffness and mass matrices, allowing the fast computation of the loss on the LBO eigenbasis with respect to edges, faces, and vertices.
- an operator learning framework that is able to learn operators on the edge, face, and vertex of a manifold by shared neural networks.
- a framework that is able to utilize the previous framework to learn a spectrum, resulting in a more precise functional maps paradigm.

As these contributions are on a fundamental level, they have an accumulative effect; hence, they can be used in conjunction with other frameworks to further improve performance.

Moreover, we present in-depth analyses on:

- how much learning potential is unexploited in spectral descriptors,
- how far learning can extend frequency-wise,
- generalization across frequencies and manifolds,
- what the limiting factors are,
- and how much capacity is needed to exploit this potential.

1. Introduction

These analyses give important insights into what future research can steer into, to further unleash the power of spectral descriptors. For the sake of clarity and consistency, this thesis is restricted to triangular meshes. This focus facilitates a clearer exposition of the proposed techniques, while readers are invited to generalize the methods to other mesh types when applicable.

1.5 Thesis structure

The structure of this thesis mirrors the evolution of my doctoral research. It begins with the initial idea of modifying the LBO eigenbasis to meet a specific objective and gradually delves deeper, introducing additional tasks and learning capabilities to extend the proposed framework. This organization enables readers from diverse backgrounds to follow the research journey and better appreciate its results. Experienced readers may prefer to proceed directly to sections of particular interest, although noteworthy findings appear throughout the thesis.

Chapter 2 reviews the fundamental mathematics and computer science concepts underlying this thesis, spanning topics from the LBO to geometric Convolutional Neural Networks (CNNs).

Chapter 3 examines neuroevolutionary approaches to optimizing the LBO, framing the mass-matrix weighting as a black-box optimization problem. This strategy bypasses the costly backpropagation routine and enables learning through a shortcut, albeit with notable trade-offs

Chapter 4 presents LBONet, which enables learning of various operators to optimize the system matrices through backpropagation. All equations are derived using calculus, and Nelsons method is employed to obtain the eigenbasis gradients. This foundational approach integrates naturally with frameworks that use spectral descriptors. Although it involves different trade-offs compared with the neuroevolutionary attempt, it ultimately proves superior.

Chapter 5 extends LBONets learning capabilities to the functional maps framework, where eigenbasis learning is linked to the point-to-point map. This alignment of spatial and spectral domains reduces geodesic loss and enables explicit, rather than

descriptor-based, eigenbasis learning. The results show consistent improvements over state-of-the-art methods on all non-isometry benchmarks.

Finally, Chapter 6 delves into the core contribution of this thesis, examining the limits of learning capability and capacity. Furthermore, it concludes the thesis by outlining future research directions, identifying the challenges that must be addressed for widespread adoption of LBO eigenbasis learning, and summarizing this work's contributions and significant findings in shape-analysis research.

A thorough low-level gradient check to verify the correctness of all gradients introduced in this thesis is presented in the Appendix.

"Alle Messungen der Welt wiegen keinen Satz auf, durch den die Wissenschaft der ewigen Wahrheiten wirklich gefördert wird." [9]

Carl Friedrich Gauss

2

Background

This chapter lays the foundation for the methods and tools used throughout this thesis and requires the reader to be familiar with these concepts. First, the mathematical background is presented. Second, advances in feature learning are discussed, which enable the extraction of relevant features from the given descriptors. Next, we examine spectral descriptors and the learning methods associated with them. Finally, we review approaches that attempt to learn the LBO itself. This progression leads to the central contribution of this work: an effective method for learning the LBO directly.

2.1 Mathematical Background

This thesis assumes that the reader is familiar with the fundamentals of calculus, including concepts such as the chain rule and the product rule. Consequently, this section focuses on more advanced mathematical notions that are required for the subsequent developments. For each topic, references for further reading are provided.

2.1.1 Laplace–Beltrami Operator

Let (M, g) be a smooth Riemannian manifold. The Laplace–Beltrami operator is the natural generalization of the Euclidean Laplace operator to Riemannian

manifolds and is defined intrinsically as

$$\Delta f = \operatorname{div}(\nabla f), \quad (2.1.1.1)$$

where ∇f denotes the Riemannian gradient of a smooth function $f \in C^\infty(M)$ and div is the divergence operator associated with the Riemannian metric g .

The gradient ∇f is defined implicitly by the relation

$$g(\nabla f, X) = X(f) \quad \text{for all vector fields } X, \quad (2.1.1.2)$$

while the divergence is defined via the Riemannian volume form vol_g as

$$\mathcal{L}_X \operatorname{vol}_g = (\operatorname{div} X) \operatorname{vol}_g, \quad (2.1.1.3)$$

where $\mathcal{L}_X$ denotes the Lie derivative with respect to X .

In local coordinates, the Laplace–Beltrami operator admits the following expression:

$$\Delta f = \frac{1}{\sqrt{|g|}} \partial_i \left(\sqrt{|g|} g^{ij} \partial_j f \right), \quad (2.1.1.4)$$

where g^{ij} are the components of the inverse metric tensor and $|g|$ denotes the determinant of the metric tensor.

It was introduced first into shape analysis with ShapeDNA [10] and utilized the eigenvalues of the operator on a manifold to form an invariant embedding.

Spectral Theorem The LBO is a self-adjoint, positive semi-definite operator and therefore admits an orthonormal eigensystem, which can be retrieved by solving the eigenvalue problem:

$$\Delta \phi = \lambda \phi. \quad (2.1.1.5)$$

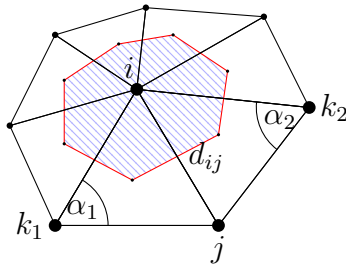


Figure 2.1: Voronoi cell on vertex i , which is outlined in red. The Voronoi cell can be calculated using the illustrated angles and the distances shown.

2. Background

As we mainly deal with discrete approximations of manifolds such as meshes, point clouds, and voxels, we need to discretize the LBO. Several discretization methods have been proposed, but the cotangent weight method [11] has established itself for meshes, and is defined as:

$$\Delta f = Lf = A^{-1}Wf, \quad (2.1.1.6)$$

where W is referred to as *stiffness* matrix and its elements are defined by:

$$W(i, j) = \begin{cases} \frac{\cot \alpha_{ij} + \cot \beta_{ij}}{2}, & \text{if } (i, j) \in E \\ -\sum_{k \neq i} W(i, k), & \text{if } i = j \\ 0, & \text{otherwise.} \end{cases} \quad (2.1.1.7)$$

A is referred to as *mass* matrix and is a diagonal matrix and defined by the *Voronoi* cell at each vertex,

$$A(i, j) = \begin{cases} \frac{1}{8} \sum_{j \in N_1(i)} (\cot \alpha_{ij} + \cot \beta_{ij}) \|d_{ij}\|^2, & \text{if } i = j \\ 0, & \text{otherwise.} \end{cases} \quad (2.1.1.8)$$

Another popular method to calculate the *mass* matrix is using barycentric cells, however it is more prone to be mesh-dependent, which makes it an inferior choice. Finally, we derive the generalized eigenvalue problem:

$$W\phi = \lambda A\phi. \quad (2.1.1.9)$$

which leads us to the LBO eigenbasis $\{\phi, \lambda\}$. The LBO eigenbasis possesses several remarkable properties that distinguish it from other descriptors. First, it is fully isometric, making it invariant to the manifolds embedding. This is a significant advantage over coordinate-based methods, which must learn such invariance explicitly through exhaustive training involving rotations and translations of the manifold. Second, this isometric property also holds under articulated deformations for example, when a robots arm is raised provided that the deformation remains near-isometric and does not introduce substantial stretching. However, this descriptor fails under several conditions. One such case occurs when near- or non-isometric stretching is present, causing a loss of descriptiveness and discriminative power. Another arises when the manifold contains holes or represents only a partial

shape. Both challenges have been the focus of ongoing research, with numerous methods proposed to mitigate their effects.

Another difficulty when dealing with the LBO eigenbasis is that eigenvectors are defined only up to a sign, which can lead to sign flips even in the absence of perturbations. Furthermore, when consecutive eigenvalues are close in magnitude, their corresponding eigenvectors may interchange positions. These two properties pose challenges for directly applying eigenvectors in downstream applications.

2.1.2 Spectral Signatures

We refer to any descriptor that utilizes the spectrum of the LBO eigenbasis as a spectral signature. Although spectral signatures are fundamentally derived from the eigenvalues and eigenvectors of the LBO, numerous formulations have been proposed to capture different characteristics of the underlying manifold. New variations and improvements continue to appear regularly in the literature. In the following, we present the foundational spectral signatures and refer the reader to a comprehensive list of further formulations at the end of this section.

Global Point Signature (GPS) Following ShapeDNA’s [10] example, Rustamov [12] introduced the GPS, which used both eigenvalues and eigenvectors of the LBO eigenbasis,

$$\text{GPS}(x) = \left(\frac{1}{\sqrt{\lambda_1}}\phi_1(x), \frac{1}{\sqrt{\lambda_2}}\phi_2(x), \dots, \frac{1}{\sqrt{\lambda_n}}\phi_n(x) \right) \quad (2.1.2.1)$$

It was the first to combined both elements of the spectrum, however still suffered from the aforementioned drawbacks, such as the flipping of eigenvectors, as eigenvectors are only defined up to a sign or swapping of eigenvectors. This phenomenon is depicted in figure 2.2. This lead to the introduction of the HKS, which is based on the heat equation.

2. Background

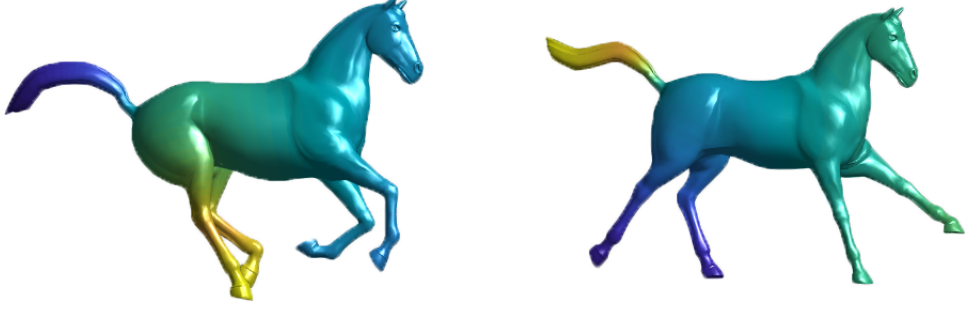


Figure 2.2: Third eigenvector ϕ_3 of the LBO eigenbasis on two manifolds. Note that one eigenvector is flipped.

HKS Motivated by aforementioned drawbacks, Sun et al. [13] introduced a signature based on the heat diffusion process, and is relevant in most shape retrieval methods. The heat diffusion process is given by

$$\Delta u(x, t) = -\frac{\partial u(x, t)}{\partial t}. \quad (2.1.2.2)$$

Given initial heat distribution f and heat distribution $H_t(f)$ at time t . By the *informative theorem* [13], the heat kernel can be restricted to the temporal domain by restricting it to the diagonal elements:

$$h_t(x, x) = \sum_{k>0} e^{-t\lambda_k} \phi_k(x)^2. \quad (2.1.2.3)$$

As HKS varies with the scale of a manifold, a scale-invariant variant of HKS was proposed in [14].

Wave Kernel Signature (WKS) Although very similar, but inducing different dynamics, Aubry et al. [15] proposed to use the wave equation rather than the heat diffusion process:

$$\text{WKS}(E, x) = \sum_{k=1}^{\infty} \phi_k(x)^2 f_E(\lambda_k)^2, \quad (2.1.2.4)$$

where f is the wave equation and E the energy to be varied.

This leads to a descriptor based on the probability measuring a particle with an energy distribution at point x , resulting in a band-pass descriptor, which has better feature localization than HKS, but in return has difficulties describing large-scale features.

Enhancements Many methods were proposed to finetune these descriptors to enhance their performance such as improved WKS [16], which tweaks the scaling of the eigenvalues to given data, heuristic WKS [17], which follows a similar approach. These enhancements offer significant advantage for known descriptors and can be used to further tweak their performance.

Optimal Spectral Descriptor (OSD) With the introduction of OSD [18], the authors note that the major descriptors such as the HKS or the WKS only differ in their filter responses. Hence, they parameterize the descriptor:

$$\mathbf{p}(x) = \sum_{k>1} \mathbf{f}(v_k) \phi_k^2(x), \quad (2.1.2.5)$$

where $\mathbf{f}$ is the filter response, v the eigenvalue, and ϕ the eigenvector. In order to derive the HKS or the WKS, one can use the wave function and the heat diffusion function as $\mathbf{f}$. As both signatures have their own strengths and weaknesses, we can now take the aim to learn an optimal $\mathbf{f}$ for a given task. To make use of the learning, the eigenvalues need to be fixed with a basis function. Hence, the function $\mathbf{f}$ is split into

$$\mathbf{f}(v) = \mathbf{A}\mathbf{b}(v) \quad (2.1.2.6)$$

where $\mathbf{A}$ is an $n \times m$ matrix with the learned coefficients acting on the basis functions. By substituting Eq. 2.1.2.6 back into Eq. 2.1.2.5 we obtain

$$\mathbf{p}(x) = \mathbf{A}\mathbf{b}(v) \phi_k^2(x) \quad (2.1.2.7)$$

Isolating the coefficient matrix $\mathbf{A}$ results in the well known *geometry vector* g at point x .

$$\mathbf{g}(x) = \mathbf{b}(v) \phi_k^2(x) \quad (2.1.2.8)$$

The authors propose a metric learning approach to optimize coefficient matrix A for given tasks such as localization, discrimination, and invariancy.

2. Background

2.1.3 Triangle Mathematics

As we delve deeper into the study of triangles, it is important to establish the fundamental concepts and constraints governing their geometry. Consider a sample triangle $\triangle(i, j, k)$, illustrated in Figure 2.3.

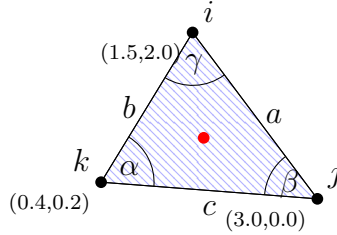


Figure 2.3: A sample triangle $\triangle(i, j, k)$ scaled with its centroid (red dot).

The lengths of its sides are constrained by the triangle inequality,

$$c \leq a + b, \quad (2.1.3.1)$$

which states that the sum of the lengths of any two sides of a triangle must be greater than or equal to the length of the remaining side. When equality holds, the triangle is referred to as degenerate, or sometimes as a collinear triangle, as all three vertices lie on a single straight line as depicted in Figure 2.4.

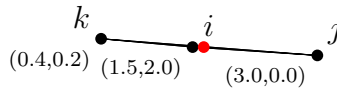


Figure 2.4: A degenerate triangle $\triangle(i, j, k)$ with its centroid (red dot).

Although a degenerate triangle is technically valid, it is generally undesirable as it introduces several issues. In a mesh structure, such a triangle poorly represents the shape, effectively behaving as an edge of neighboring triangles due to its near-zero area. More critically, numerical quantities derived from such a triangle, such as gradients, can become unstable or even divergent, as the triangle is close to being invalid and certain computations may be undefined.

The area of a triangle is calculated with:

$$T = \frac{1}{2}bc \sin \alpha. \quad (2.1.3.2)$$

2.1.4 Trigonometric gradients

Since this work heavily relies on angle gradients, it is important to clarify some of the equations used throughout this thesis. We are mainly interested in how the quantities used to calculate the discretized LBO behave when an edge is being scaled. In this particular case, we are interested how the angles behave.

To derive this we start with the Law of Cosines, which states,

$$\cos \alpha = \frac{b^2 + c^2 - a^2}{2bc}. \quad (2.1.4.1)$$

From this, we differentiate both sides wrt. side a ,

$$-\sin \alpha \frac{\partial \alpha}{\partial a} = \frac{\partial}{\partial a} \frac{b^2 + c^2 - a^2}{2bc}, \quad (2.1.4.2)$$

which simplifies to,

$$-\sin \alpha \frac{\partial \alpha}{\partial a} = -\frac{a}{bc} \quad (2.1.4.3)$$

and finally,

$$\frac{\partial \alpha}{\partial a} = \frac{a}{bc \sin \alpha} = \frac{a}{2T}. \quad (2.1.4.4)$$

With this gradient at hand, we can now run a sensitivity analysis on angle α when its opposite edge is scaled, as illustrated in Figure 2.5.

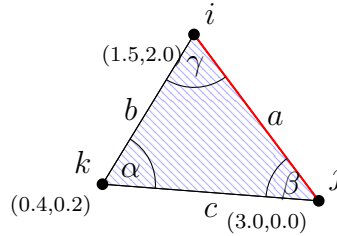


Figure 2.5: A side (ij) of triangle $\triangle(i, j, k)$ is scaled and an angle (α) changes.

We use this property similarly to calculate the gradient of the adjacent edge instead of the opposite edge,

$$\frac{\partial \beta}{\partial a} = \frac{b \cos \beta}{\sin \gamma bc \sin \alpha} = \frac{b \cos \beta}{\sin \gamma 2T}. \quad (2.1.4.5)$$

2. Background

as illustrated in Figure 2.6.

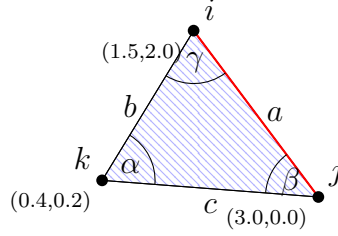


Figure 2.6: A side (ij) of triangle $\triangle(i, j, k)$ is scaled and another angle (β) changes.

Many gradients will be introduced later, but they are fundamentally variations of this gradient. The reader is expected to keep this in mind, as it will not be explicitly highlighted each time.

2.1.5 Triangle Positioning Under Diffusion Constraints

In this work, we encounter an intriguing problem that arises when both the geometry and diffusion of a manifold are altered simultaneously. In the following, we will present the problem along with a proposed approach to address it.

Let a triangle $\triangle(i, j, k)$ undergo uniform scaling along its edges (ij, jk, ik) with respect to the triangle's centroid. The resulting vertex positions satisfying the scaling constraints are well-defined, as illustrated in Figure 2.3.

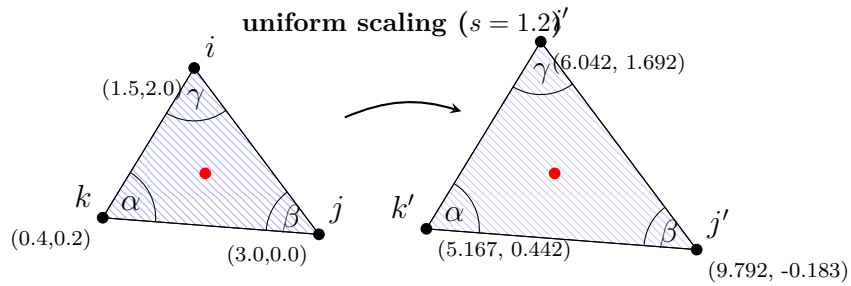


Figure 2.7: A triangle $\triangle(i, j, k)$ scaled uniformly by its centroid (red dot).

This problem is well-defined, as the triangle $\triangle(i, j, k)$ undergoes uniform scaling, resulting in a smaller or larger triangle while maintaining the relative proportions of its edges, which is reflected in the unchanged angles between them.

The situation, however, changes when non-uniform scaling is applied. Let $\triangle(i, j, k)$ undergo non-uniform scaling along edges (ij, jk, ik) . The set of possible vertex positions (i, j, k) satisfying the scaling constraints is infinite, and the choice of vertex positions affects any position-dependent operators $D(\triangle)$. For instance, one could scale down the edge ik while simultaneously scaling up the edges ij and jk , altering the shape of the triangle as in Figure 2.8.

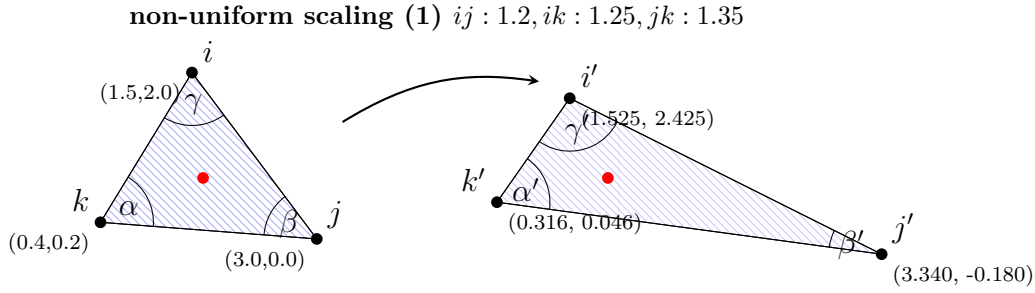


Figure 2.8: A triangle $\triangle(i, j, k)$ scaled non-uniformly starting with edge ik .

One can see that not only the edge lengths but also the angles have changed noticeably under this transformation. In this scenario, we encounter an under-determined problem: the final positions of the vertices depend on the order in which the scaling is applied. Scaling first along ik , jk , or ij yields three distinct vertex configurations. Since all three edges are altered, the triangle rotates depending on the sequence of operations. Consequently, there exists an infinite number of triangles that satisfy the scaling constraints, rendering the problem under-determined. The centroid used in the non-uniform scaling cannot be applied in this case, as its position changes with the transformation. If we begin scaling along another edge, say ik , ij , or jk , we obtain the same triangle but rotated slightly in space as illustrated in Figure 2.9,

2. Background

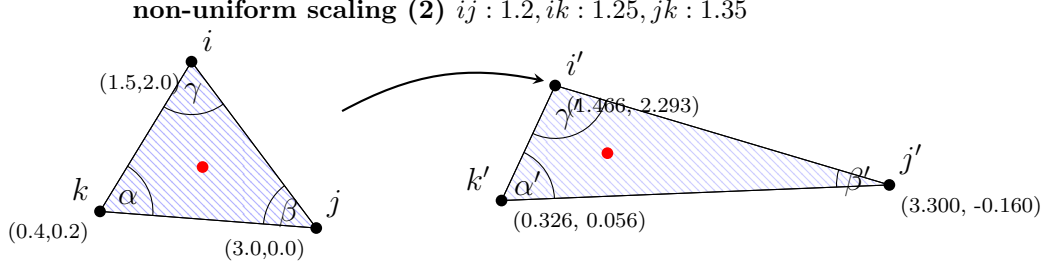


Figure 2.9: A triangle $\triangle(i, j, k)$ scaled non-uniformly starting with edge ij .

As observed in the previous section, the discretization proposed by [11] is invariant under the position of the triangles, since it relies solely on angles and edge lengths. The under-determinacy discussed earlier affects only the vertex positions; therefore, any modification of the Riemannian metric g_{ij} and any choice of scaling order can be applied without altering the discretization of the system matrices. However, the situation changes when we consider components that depend on vertex positions, such as vectors. The standard LBO is isotropic, meaning that diffusion on the manifold is distributed evenly in all directions. Suppose we use the principal curvature directions v_1 and v_2 to form an orthonormal frame on the triangle $\triangle(i, j, k)$. By default, the principal curvature directions are orthogonal and normalized to unit length, which corresponds to isotropic diffusion. Once we scale and rotate this frame, certain directions become preferred over others. Consequently, as soon as we modify the diffusion, it depends on the positions of the triangle vertices. This is precisely where the under-determinism introduced earlier becomes problematic: which vertex positions of the scaled triangle should we consider correct? The issue becomes more pronounced when the weights applied to the Riemannian metric increase, as the rotation of the triangle is amplified. A naive approach to mitigate this problem is to compute all three possible rotations of the scaled triangle, align them by their centroid and take their mean rotation, as illustrated in Figure 2.10, to determine the triangles position.

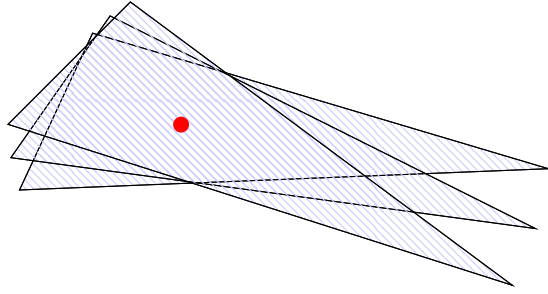


Figure 2.10: Three triangles $\triangle(i, j, k)$ scaled in different order ij , ik and jk , resulting in triangles with different coordinates.

The true position of the triangle can be determined in several ways. One approach is to first apply a non-uniform scaling to the triangle and then perform modifications to the diffusion. In this setting, we can compute the gradient both numerically and analytically. The numerical and analytical gradients will coincide only if the triangle is positioned exactly as the diffusion operator expects. A vertex positioning algorithm could be realized using Iterative Closest Point (ICP), although this might incur a substantial computational overhead.

The true extrinsic position of the triangle is crucial because diffusion operators may depend on principal curvature directions, which change with the triangles rotation. Even when considering a single triangle, maintaining consistency with the overall manifold geometry is essential. For instance, if we were to arbitrarily fix every triangle at $(0, 0)$ to simplify computations, the manifold structure would collapse, making meaningful diffusion-based learning impossible.

2.1.6 Nelson’s Method

As mentioned earlier, the LBO eigenbasis consists of two components: the eigenvectors ϕ and the eigenvalues λ , where each eigenvector ϕ_i has an associated eigenvalue λ_i that controls its magnitude. Spectral methods use these as a basis and effectively apply filters over them. For example, the HKS acts as a low-pass filter, while the WKS acts as a band-pass filter, each extracting different parts of the spectrum. Consequently, the HKS emphasizes global geometric properties of the manifold, whereas the WKS captures finer local details. Many existing approaches build upon this idea, focusing on what information to extract from the given LBO eigenbasis.

2. Background

The earlier mentioned method OSD [18] exactly provides a framework at this point to learn an "optimal" spectral descriptor.

However, when we invert the question and ask instead what geometry is required to form a target-specific LBO eigenbasis, we must compute the gradients of the eigenbasis itself. As discussed in previous sections, gradients with respect to geometry are straightforward to obtain, but differentiating eigenvectors and eigenvalues is non-trivial and has been an active research topic for several decades. Existing methods typically impose strong constraints to make this feasible. Moreover, popular deep learning frameworks such as PyTorch, at the time of writing this thesis, do not support gradient computation when both system matrices are sparse. One widely used approach is Nelson's method [19], which is conceptually simple but introduces non-sparse intermediate matrices. This unavoidably ties the methods computational cost to the number of vertices, faces, or edges in the mesh, leading to a scalability bottleneck.

The method must also operate on the given form. For instance, if we aim to learn operators acting on the vertices, we need to compute gradients for all vertices in order to train an operator that can leverage the global gradient information across the entire mesh. This results in an operator capable of assigning different weights to individual vertices, thereby allowing it to adjust the LBO eigenbasis with respect to a given loss function.

Assuming a genus of zero for a triangular mesh, meaning that the mesh has no holes, we can start from Eulers polyhedral formula

$$V - E + F = 2. \tag{2.1.6.1}$$

For a closed triangular mesh, each face consists of three edges and each edge is shared by two faces, yielding

$$3F = 2E. \tag{2.1.6.2}$$

Combining this relation with Euler's formula gives

$$E = 3V - 6, \tag{2.1.6.3}$$

$$F = 2V - 4. \tag{2.1.6.4}$$

Thus, the numbers of vertices, edges, and faces are linearly related and therefore grow at the same asymptotic rate. However, the edge dimension is the largest by a constant factor, with approximately $E \approx 1.5F$ and $E \approx 3V$ for large meshes. Consequently, algorithms operating on edges typically incur a higher computational cost than equivalent vertex- or face-based formulations.

To learn an optimal LBO eigenbasis, one needs to differentiate the generalized eigenvalue problem (2.1.1.9) wrt. a given learnable parameter p of the manifold, where p could, for example, represent the edge length of a triangle:

$$\frac{\partial W}{\partial p_{ij}}\phi + W \frac{\partial \phi}{\partial p_{ij}} = \frac{\partial \lambda}{\partial p_{ij}}A\phi + \lambda \frac{\partial A}{\partial p_{ij}}\phi + \lambda A \frac{\partial \phi}{\partial p_{ij}}. \quad (2.1.6.5)$$

Differentiating the stiffness W and mass matrix A is trivial, whereas the eigenbasis involves multiple steps. Nelson's method [19] has established itself for differentiating eigenvalues and eigenvectors and has been used widely throughout the literature. For the mass matrix, rather than using barycentric cells as in [20][21], we opt for using Voronoi cells introduced by [11], as they prove to be more robust to meshing structure and is recommended by [11], due to having a provably tight error bound. Depending on whether the task at hand impacts the stiffness and/or mass matrix, Nelson's method needs to be adjusted accordingly. We first start with the default method, stiffness & mass matrix, and then follow with stiffness only, and finally mass only.

Let A and W be the stiffness and mass matrices of the system, respectively. Given a small perturbation in the parameter p_{ij} , we denote the corresponding changes in A and W by ΔA and ΔW . Then, from equation 2.1.6.5, the derivative of the eigenvalue λ_k with respect to p_{ij} is given by

$$\frac{\partial \lambda_k}{\partial p_{ij}} = \phi_k^T (\Delta A - \lambda_k \Delta W) \phi_k, \quad (2.1.6.6)$$

and the derivative with respect to the eigenvector ϕ_k can be computed similarly.

$$\frac{\partial \phi_k}{\partial p_{ij}} = \mu_{ij} + c \phi, \quad (2.1.6.7)$$

where μ is the result of the following equation,

2. Background

$$(W - A)\mu_{ij} = F_i, \quad (2.1.6.8)$$

and F_i is defined as,

$$F_i = \phi^T \left(\frac{\partial W_k}{\partial p_{ij}} - \lambda_i \frac{\partial A_k}{\partial d_{ij}} \right) \phi A \phi - \left(\frac{\partial W_k}{\partial p_{ij}} - \lambda_i \frac{\partial A_k}{\partial p_{ij}} \right) \phi \quad (2.1.6.9)$$

and c as

$$c = -\mu A \phi - \frac{1}{2} \phi^T \Delta A \phi. \quad (2.1.6.10)$$

As we have seen from the gradients of the LBO earlier, a change in the Riemannian metric g_{ij} affects only the immediate neighborhood of the mesh. It is important to note that while the LBO *eigenbasis* is a global construct and therefore has an influence across the entire mesh, the geometric effect of locally modifying g_{ij} remains confined to its nearby region, as we will demonstrate in the chapter 4.1.

Mathematically, this implies that both ΔA and ΔW are sparse and can be computed within the localized range where the influence of the metric change propagates. For instance, from the formulation of the eigenvalue gradient, we can see that it is evaluated within this subspace and only projected onto the eigenbasis in the final operation. However, from the gradient of the eigenvectors, we observe an early transition into dense operations, which constitutes the main computational bottleneck of this method.

On the other hand, the method can be fully vectorized across both the dimensions of the mesh (vertices, faces, edges) and the frequency range of the spectrum. This essentially reduces the computation to few large dense operations on the Graphics Processing Unit (GPU), resulting in fast runtime. A gradient with respect to the Riemannian metric g_{ij} on a "Centaur" manifold is shown in Figure 2.11. Note that multiple frequencies exhibit activated regions. As we move to higher frequencies, smaller structures such as the hands or the neck, also show activation. This gradient can be used to weight the Riemannian metric g_{ij} , s.t. a given loss is reduced.



Figure 2.11: Differentiation through the LBO eigenbasis produces a gradient with respect to the Riemannian metric g_{ij} . Blue indicates negative values, while red indicates positive values.

When the operator in question affects only the mass matrix, Nelson’s method is slightly modified, as in this case the stiffness matrix does not contribute any gradients.

$$\frac{\partial \lambda_k}{\partial m_{ij}} = \phi_k^T (\Delta A) \phi_k, \quad (2.1.6.11)$$

$$F_i = \phi^T \left(-\lambda_i \frac{\partial A_k}{\partial d_{ij}} \right) \phi A \phi - \left(\lambda_i \frac{\partial A_k}{\partial d_{ij}} \right) \phi \quad (2.1.6.12)$$

Similarly, when there are no changes in the mass matrix but modifications occur in the stiffness matrix, the formulation of Nelson’s method adapts accordingly to account for these variations.

$$\frac{\partial \lambda_k}{\partial p_{ij}} = \phi_k^T (-\lambda_k \Delta W) \phi_k, \quad (2.1.6.13)$$

$$F_i = \phi^T \left(\frac{\partial W_k}{\partial p_{ij}} \right) \phi A \phi - \left(\frac{\partial W_k}{\partial p_{ij}} \right) \phi \quad (2.1.6.14)$$

For instance, for a mesh with 32 frequencies and 1024 vertices, the runtime on an RTX 5090 GPU is less than 20ms. Given continued improvements in hardware, this method is expected to become increasingly accessible.

2.1.7 Functional Maps

The LBO has established itself in numerous areas since its inception, one of which is serving as a basis for mapping correspondences on manifolds, as it satisfies

2. Background

many criteria of an elegant basis. It is compact and can be adapted to the desired level of precision, and it provides high accuracy, although increasing non-isometric deformations can degrade this property, among other benefits. As this work focuses primarily on improving the LBO eigenbasis and enhancing its stability against such deformations, we extend the approach to functional maps. Functional maps were first introduced by [22] and continue to be an active area of research. Recent methods have also demonstrated the ability to learn arbitrary bases for mapping functions between manifolds, however the LBO eigenbasis still remains interesting as it brings many other properties with itself against arbitrarily learned bases.

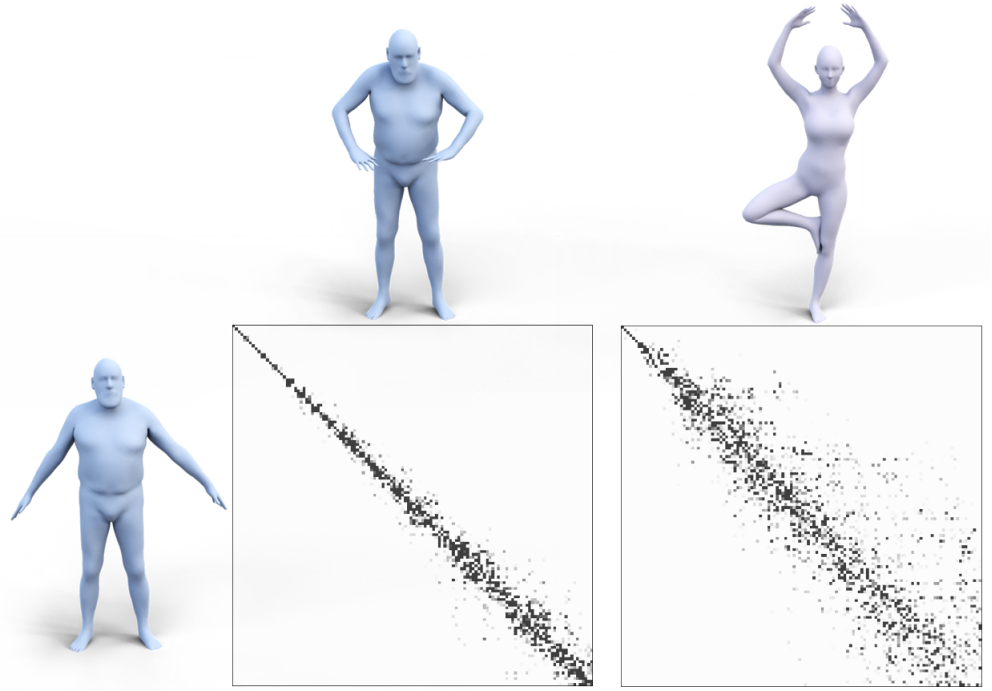


Figure 2.12: Functional matrix between an intra-class manifold pair (left blue and top blue) and a inter-class manifold pair (left blue and top purple). One can observe two things, first, the functional matrix deviates from the diagonal even with mild non-isometries and second, that with greater deformation, the matrix deviates even further.

Specifically, functional maps map functions on manifold X to functions on manifold Y via a linear operator

$$T : L^2(X) \rightarrow L^2(Y). \quad (2.1.7.1)$$

In this framework, functions defined on one manifold are mapped to functions on the other. This correspondence can be represented by a matrix $\mathbf{C}$, with

coefficients defined as

$$c_{ji} = \langle \phi_j, T\psi_i \rangle_Y, \quad (2.1.7.2)$$

where $\{\psi_i\}$ and $\{\phi_j\}$ are orthogonal bases on manifolds X and Y , respectively. This formulation naturally leads to the following expansion:

$$Tf = \sum_{i,j \geq 1} \langle \psi_i, f \rangle_X c_{ji} \phi_j. \quad (2.1.7.3)$$

In the context of functional maps, the LBO eigenbasis is an ideal choice, as it allows truncating the eigenbasis at frequency λ_k , leading to a compact representation. The LBO eigenbasis performs well for isometric shapes and provides good correspondence quality. However, in the presence of non-isometric deformations, the LBO eigenbasis can become less effective and, in extreme cases, unusable. This is illustrated in Figure 2.12, where a near-isometric deformation (blue manifolds and left matrix) results in only a small deviation from the diagonal and remains relatively sparse, in contrast to a non-isometric deformation (blue and purple manifolds and right matrix), which is more scattered and requires a combination of many more eigenvectors to map functions back and forth.

The term “Isospectralization” is commonly used for methods that aim to make the spectra of two shapes equal. However, it is important to note that a more diagonal map does not necessarily imply a more precise mapping. The precision ultimately depends on how the point-to-point map is constructed. As we will see in subsequent chapters, the methods presented in this work can achieve highly diagonal maps, but this can come at the expense of losing the connection to the point-to-point map and therefore reducing spatial precision. Consequently, the off-diagonal elements are necessary for a precise map, and the connection between the spectral and spatial domains must be carefully considered.

2.2 LBO Methods

In this section, we review several methods that aim to learn and fundamentally modify the LBO eigenbasis. This overview is not exhaustive but provides the reader

2. Background

with a clear understanding of the various ways the LBO can be adapted and shaped through different approaches, which are relevant to understand the presented work.

2.2.1 Anisotropic LBO

The next step towards the introduction of machine learning in non-rigid shape retrieval is marked by the approach introduced by [23]. The main idea boils down to the fact that heat does not spread completely isotropically. In real-world media such as organic matter, crystals, and engineered materials, heat rarely propagates isotropically. Instead, thermal diffusion is governed strictly by the composition and geometric structure of the underlying manifold. This directional dependence is modeled mathematically by introducing a thermal conductivity tensor into the Laplacian, thereby yielding the anisotropic Laplacian.

$$\Delta_{\alpha\Theta}f = -\text{div}(R_{\Theta}D_{\alpha}(x)R_{\Theta}^T\nabla f(x)) \quad (2.2.1.1)$$

where α controls the degree of anisotropy and Θ the rotation w.r.t. the principal curvature direction. However, these parameters can only be trained when a differentiable cost function is at hand. The eigenspectrum is a solution to the generalized eigenvalue problem and therefore prevents designing a simple differentiable cost function. To mitigate this problem, the authors precalculate all the possible combinations of anisotropy degree α and rotation Θ and pass the precalculated spectrum through the spectral filter layer, which picks up the right amount of each spectrum for a given task. This mitigation is however at the cost of precalculating the spectrum for every combination for each manifold, which gets computationally expensive with increasingly growing learning data.

The spectral filter takes the eigensystems sampled on predetermined anisotropy degree $\alpha_1, \dots, \alpha_A$, and equally-spaced angles $\Theta_1, \dots, \Theta_L$ and is defined as:

$$f_{ijm}(x) = \sum_{k=1}^K \tau_m(\lambda_{\alpha\Theta}) \phi_{\alpha\Theta k}^2(x) \quad (2.2.1.2)$$

We can see the relation to Eq. 2.1.2.5 and in fact, the only difference is the type of learning the descriptor. OSD used metric learning, whereas Anisotropic Diffusion

Descriptor (ADD) makes use of neural network learning. The spectral filter stage consists of a filter bank of M filters, which takes the $A * L$ eigensystems as input and filters them producing an $A * L$ output. It is then passed through a dimension reduction stage resulting a Q -dimensional descriptor. The problem, however, arises when descriptors are precomputed with fixed anisotropy degrees and angles, and the network learns only on top of these precomputed values to determine what is most useful for a given task. If certain features or benefits lie between the sampled angles or degrees of anisotropy, the network cannot capture them with higher fidelity, as it is constrained by the resolution of the precomputation.

2.2.2 Kinetic LBO

Another interesting way of modifying the LBO was introduced by [24]. Based on classical field theory, the authors discount the kinetic term of joint regions, which leads to a non-rigid invariant operator. The Lagrangian density of a system is the difference of the kinetic and potential energy:

$$\mathcal{L}(\phi, \nabla\phi, \dot{\phi}, x, t) = T - V, \quad (2.2.2.1)$$

where ϕ is a field defined over the space. The action of the Lagrangian density is given by:

$$S(\mathcal{L}) = \int \mathcal{L} dx dt \quad (2.2.2.2)$$

By Hamilton's principle we can recover the dynamics of the system which results in the well known Euler-Lagrange equation:

$$\frac{\partial \mathcal{L}}{\partial \phi} - \nabla \frac{\partial \mathcal{L}}{\partial \nabla \phi} - \frac{\partial}{\partial t} \left(\frac{\partial \mathcal{L}}{\partial \dot{\phi}} \right) = 0. \quad (2.2.2.3)$$

This Partial Differential Equation (PDE) can be solved by the general scaled Lagrangian

$$\mathcal{L} = \frac{1}{2} \dot{\phi} \dot{\phi}^* + \frac{1}{2} (\phi \dot{\phi}^* - \phi^* \dot{\phi}) - \nabla \phi \nabla \phi^* - \phi^* - \phi^* V \phi \quad (2.2.2.4)$$

2. Background

and by choosing appropriate terms we can derive the aforementioned HKS or WKS. We choose the second and third term

$$\mathcal{L} = \frac{1}{2}(\phi\dot{\phi}^* - \phi^*\dot{\phi}) - \nabla\phi\nabla\phi^* \quad (2.2.2.5)$$

for deriving the dynamics for the family of HKSs.

$$\nabla^2\phi = \dot{\phi}. \quad (2.2.2.6)$$

To reduce object articulations, a weighting term is added to the wave Lagrangian and heat Lagrangian

$$\mathcal{L} = \frac{k}{2}C(x)(\phi\dot{\phi}^* - \phi^*\dot{\phi}) - \nabla\phi\nabla\phi^* \quad (2.2.2.7)$$

resulting in

$$\nabla^2\phi = kC(x)\dot{\phi}. \quad (2.2.2.8)$$

After discretizing we get

$$L\Phi = \lambda K\Phi \quad (2.2.2.9)$$

where K is a diagonal matrix determined by the weighting function $C(x)$. Combining this with Eq. 2.1.1.9 results in the following generalized eigenvalue problem

$$W\Phi = \lambda AK\Phi. \quad (2.2.2.10)$$

After careful observation the authors find a relation between the positive volume of a local surface patch and their joints. They define the function as

$$C(\mathbf{x}) = \max(\epsilon, H(\mathbf{x})) \quad (2.2.2.11)$$

where $H(\mathbf{x})$ boils down to the mean curvature at any given point. This generic approach has many upsides, but it is clear that this approach does not make any use of class-specific features and does not learn to highlight or to disregard them, but demonstrates that weighting the Voronoi cell can be highly beneficial.

2.2.3 Riemannian metric optimization

In a follow-up work to Conformal Metric Optimization on Surfaces (CMOS) [20], the authors of Riemannian Metric Optimization on Surfaces (RMOS) [21] learn brain mappings by optimizing the Riemannian metric on a manifold through an iterative algorithm. They first establish a coarse correspondence between two manifolds and then iteratively refine the Riemannian metrics on both until the energy function converges. This optimization produces a weighting that makes the resulting eigenbases nearly identical, even under near-isometric deformations between the two manifolds. This work demonstrates that the Riemannian metric g_{ij} can be optimized such that the resulting spectral descriptors bridge the gap caused by structural variations between different brains.

2.2.4 HodgeNet

HodgeNet learns the 0-form and 1-form Hodge star matrices from vertices and edges, respectively, and approximates the corresponding gradients rather than computing them exactly. This approximation, however, requires the authors to compute twice as many frequencies to achieve the desired level of accuracy. Although the method is theoretically well-grounded and, in principle, should be capable of outperforming even the approach presented in this work, it achieves only modest results across various benchmarks. This raises the question of whether the performance degradation stems from the gradient approximation itself or from the network learning weak geometric features. Unfortunately, the authors do not provide a standardized baseline for comparison. For instance, on the human segmentation benchmark, our method introduced in Chapter 4 outperforms HodgeNet [25] significantly, even when relying solely on RiemannNet without any backend, where we define backend as the framework consuming the features to predict a given task.

2.3 Deep Learning Methods

In this section, we take a deeper look at methods that apply deep learning techniques, including CNNs and more sophisticated architectures such as transformers. The goal is to provide the reader with a clear understanding of what we refer to as the backend, but also some techniques we use to design the learning capacity to learn an optimal LBO eigenbasis.

2.3.1 Spectral CNNs

As CNNs learn from the hierarchical structure of the underlying data, an analogous concept is required for the patch operator [26] in the spectral domain. [27] proposes the use of a vertex-frequency analysis framework, also known as the Windowed Fourier Transform (WFT), which encapsulates the local context of a point on the manifold within the spectral domain. This can be interpreted as a *patch*. The method generalizes all previously introduced spectral descriptors and can, in principle, only improve upon them. The first fully connected layer can reproduce any earlier spectral descriptor when the weights are fixed accordingly. The WFT layer, acting as the convolutional layer, demonstrates that the number of WFT windows corresponds to the number of dimensions in the given input. The resulting output is a descriptor, referred to by the authors [27] as the Localized Spectral CNN (LSCNN) descriptor. The loss function aggregates pairs of points that are known to be similar or dissimilar across manifolds. It is worth noting that this network architecture has since become a standard foundation in many spectral and spatial CNNs, with subsequent works introducing novelty primarily in the convolutional layer design.

2.3.2 Spatial CNNs

As noted earlier, spectral CNNs generally struggle to generalize across different manifolds. We have also discussed an alternative approach to address this limitation. In parallel to spectral methods, spatial methods have begun to emerge, such as ShapeNet [28]. This method makes use of the patch operator introduced by [26], where patches are defined in terms of geodesic neighborhoods on the manifold,

$$(D(x)f)(p, \Theta) = \int_X v_{p,\Theta}(x, x') f(x') dx' \quad (2.3.2.1)$$

$$v_{p,\Theta}(x, x') = \frac{v_p(x, x') v_\Theta(x, x')}{\int_X v_p(x, x') v_\Theta(x, x') dx'} \quad (2.3.2.2)$$

the authors define the geodesic convolution as:

$$f_{\Delta\Theta, q}^{out}(x) = \sum_{p=1}^P (f_p * a_{\Delta\Theta, qp})(x). \quad (2.3.2.3)$$

Drawbacks of this approach include the arbitrary rotation of patches, which forces the convolutional layer to evaluate all possible rotations and retain only the maximum response. Moreover, this type of learning is restricted to meshes only.

Later, [29] introduced a *patch operator* based on the anisotropic heat kernel. The patch is defined as:

$$(D(x)f)(\Theta, t) = \frac{\int_X h_{\alpha\Theta t}(x, \xi) f(\xi) d\xi}{\int_X h_{\alpha\Theta t}(x, \xi) d\xi}, \quad (2.3.2.4)$$

and intrinsic convolution

$$(f * a)(x) = \int a(\Theta, t) (D(x)f)(\Theta, t) dt d\Theta. \quad (2.3.2.5)$$

We saw from [30] how this method compares to its spectral counterparts and that it cannot compete in certain tasks. An important upside however, is the higher ability to generalize across manifolds.

Rather than using a handcrafted patch operator, [31] parameterized the *patch operator* with Gaussian Mixture Model (GMM), allowing the *patch operator* to adapt to given data:

$$D_j(x)f = \sum w_j(u(x, y)) f(y), \quad (2.3.2.6)$$

alongside the spatial generalization of convolution

$$(f * g)(x) = \sum_{j=1}^J g_j D_j(x) f. \quad (2.3.2.7)$$

2. Background

As shown in the previous chapter many descriptors aim to tackle the problem of non-rigid shape retrieval. In order to anticipate with the machine learning trend in computer vision, the scientific literature has put a strong focus on the learning aspects of such descriptors in recent years. The beginning of this era is marked by the OSD introduced in [18], which fixed the basis of the descriptors on a sufficiently wide interval and thus making the representation independent of a shapes eigenvalues. This step allowed to learn from data systematically and task-driven.

2.3.3 Feature Learning

In PointNet[5] the authors propose a framework which is able to learn from unordered point clouds. For this they utilize pointwise perceptrons projecting input features to higher dimensions, which then is followed by a pooling operation. This results in a global descriptor, which then can be used to learn a task-specific network. In their second paper, PointNet++ [8], the authors conduct an experiment on non-rigid shape retrieval on the Shape Retrieval Challenge (SHREC)'15 benchmark. By switching input features from XYZ coordinates to spectral descriptors (HKS and WKS), they gain an accuracy increase by 30%. This proves that not only the network is able to utilize spectral descriptors, but also that non-rigid shape retrieval is a very hard problem if no spectral descriptors are used. This shows that there might be potential unused within the spectral descriptors, which we did not utilize until today. DGCNN [6], can be seen as generalization of the PointNet approach. The authors, do not just use the information on the input point cloud, but they construct a local neighbourhood graph for each point and learn a neighbourhood function, which is able to propagate local neighbourhood information to the given point. More recent transformer-based methods [32, 33] have further advanced feature extraction, albeit at a significant increase in computational cost. A HKS-based method hierarchically constrains the transformer using heat diffusion [34]. Since this work primarily focuses on learning optimal features for downstream prediction, it is crucial that the backpropagated gradients maintain sufficient fidelity.

If excessive learning capacity is absorbed by the downstream module, the feature-learning module may converge to trivial representations, resulting in only modest improvements over standard spectral descriptors.

In Chapter 6, we provide an in-depth analysis of how backend complexity affects gradient propagation in earlier layers. Furthermore, the presented results demonstrate the substantial learning capacity inherent in the LBO eigenbasis itself. Even when using traditional backends, such as a simple MLP, the learned eigenbasis can outperform deeper architectures like PointNet++ [8]. Nonetheless, the choice of backend remains valuable when combined with an optimally learned LBO eigenbasis, the overall system exhibits a synergistic improvement rather than redundancy.

2.3.4 DeepFM

The concept of functional maps has been further extended into deep learning [35], enabling descriptors that predict the functional map between two manifolds. The functional map, $\mathbf{C}^*$, is solved by the following optimization problem:

$$\mathbf{C}^* = \arg \min_{\mathbf{C}} \|\mathbf{C}\hat{\mathbf{F}} - \hat{\mathbf{G}}\| \quad (2.3.4.1)$$

where $\hat{\mathbf{F}}$ and $\hat{\mathbf{G}}$ are features F and G projected onto the eigenbasis of the corresponding shapes. The dense point-to-point map is recovered using a nearest neighbor approach. However, even when the ground-truth map $\mathbf{C}_{gt}$ is available, there is no guarantee that the nearest neighbor corresponds to the actual point, or points in the 1-ring, 2-ring, or 3-ring spatial neighborhood. This issue is further exacerbated by near-isometric or non-isometric deformation between the compared manifolds as illustrated in Figure 2.12. As a result, functional maps and the LBO eigenbasis are limited to applications with near-isometric deformations. Once more significant deformations occur, the functional map becomes decoupled from the point-to-point correspondence. This framework enables the learning of descriptors that extract the relevant information from the spectrum to predict functional maps with high precision. Moreover, it allows for learning arbitrary bases [36], further improving correspondence accuracy.

2. Background

2.3.5 ZoomOut

Another significant method is ZoomOut [37], which is able to spectrally upsample a functional map. Given a coarse 4×4 map, it starts estimating a point-to-point map from the functional map C_0 for input $k_M \times k_N$ using knn-search and with this the functional map is extended spectrally to

$$C_1 = (\phi_M^{k_M+1})^T A_M \Pi \phi_N^{k_N+1}. \quad (2.3.5.1)$$

Consequently, this method allows a noisy coarse correspondence to be refined to a highly accurate functional map. This is especially significant as estimating a coarse map is a much easier task than estimating a highly precise map. As it can deal with varying amount of vertices and requires no further input during the refinement process makes this method especially unique. Furthermore in most cases where a correspondence can be established, the first few eigenvectors correspond very closely to the diagonal (see Fig. 2.12). For stronger non-isometries, such as in the case of [38], this is even true for pairs such as hippo and a lion. However, in these cases ZoomOut struggles due to the given strong non-isometry. However, this method allows one to obtain a highly accurate mapping with a very coarse estimation of a functional map. When using ZoomOut in a pipeline, the problem of accurate correspondence reduces to two. One, the accurate estimation of the a small functional map, e.g. as low as 4×4 and two, to the accurate upsampling by the ZoomOut method. Both are points, which we will cover in the last main chapter for setting up an unsupervised correspondence learning pipeline.

"Ουκ ἐστὶ βασιλικὴ πρὸς γεωμετρικὰν ὁδὸν." [39]

Euclid

3

Neuroevolutionary LBO

3.1 Introduction

The Travelling Salesman Problem (TSP) and the Airline Crew Scheduling Problem (ACSP) are famous long-standing examples for which evolutionary algorithms are particularly effective. Beyond such canonical cases, evolutionary approaches have also produced unconventional solutions, such as highly efficient antenna designs [40].

The underlying reason is that these problems are inherently combinatorial, which makes the formulation of a differentiable loss function difficult. Instead, a fitness function is employed to quantify the quality of a candidate solution and to generate new *offspring*, under the assumption that improved solutions lie in the vicinity of existing good ones. Coevolutionary mechanisms and stochastic elements help prevent the algorithm from becoming trapped in a single *species*, which could otherwise limit its ability to discover better solutions. Moreover, such approaches evolve both weights and network topology, enabling recurrent structures, skip connections, and connection pruning forms of architectural flexibility that are challenging to realize with backpropagation-based optimization.

Closely related to the difficulty of defining a differentiable loss function is the fact that the search space itself is often non-smooth and highly non-linear. Rastrigin

introduced a deliberately challenging objective function (see Figure 3.1) in [41], which has since become a standard benchmark for evolutionary optimization. While later work extended this construction to higher dimensions [42], we restrict ourselves here to the two-dimensional variant for illustrative purposes.

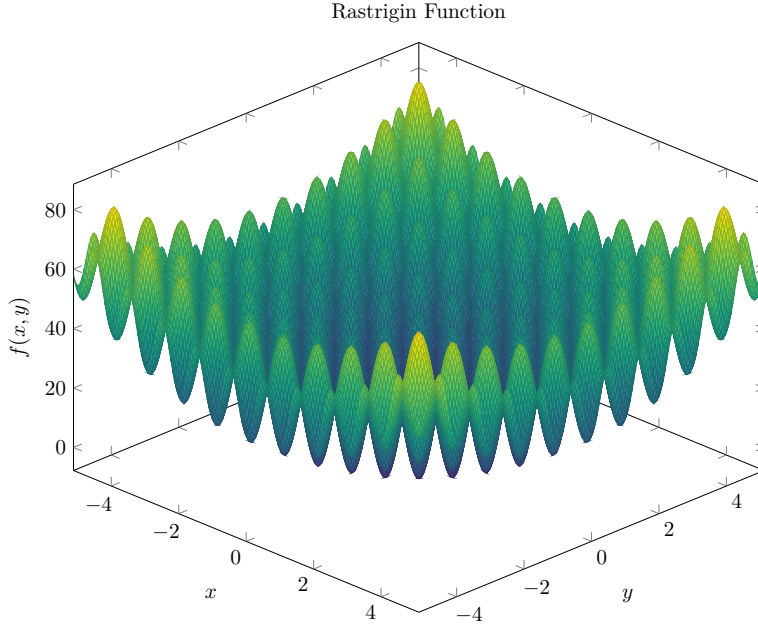


Figure 3.1: Rastrigin function shown as a filled contour plot (red: high values, blue: low values). The global minimum at $(x = 0, y = 0)$ is challenging to locate for standard optimization techniques due to the functions highly nonconvex landscape.

As one can observe, the function attains its global minimum at $(x = 0, y = 0)$. However, reaching this point is particularly difficult due to the presence of many local minima and the large extent of the search space. Since modern deep learning methods rely on variants of gradient descent to backpropagate the loss, they are prone to becoming trapped in a local minimum basin. In Chapters 5 and 6, we will refer back to this observation to justify the use of a sufficiently large learning rate to escape local minima, which corresponds to the baseline LBO. Small modifications to the LBO, on the other hand, do not allow the optimization to escape the baseline LBO basin in order to reach a lower optimum.

Recalling the previously introduced generalized eigenproblem and introducing

3. Neuroevolutionary LBO

a set of learnable parameters θ ,

$$W(\theta)\phi = \lambda M(\theta)\phi, \quad (3.1.0.1)$$

we restrict the learning process to valid elliptic, self-adjoint differential operators. Since the eigenbasis is determined by this equation, and under mild constraints on θ , it yields a smooth and stable eigenbasis. Consequently, for any point in the search space, there exists a valid modification of the underlying geometry that induces a corresponding eigenbasis. Compared to directly learning the eigenbasis, this formulation is less flexible, as it is regularized by the geometry itself. This restriction is particularly useful, as the learned solution remains physically meaningful: even when the outcome is difficult to interpret, there exists a geometric explanation for why, for example, classification performance improves when certain weights are applied to the manifold geometry.

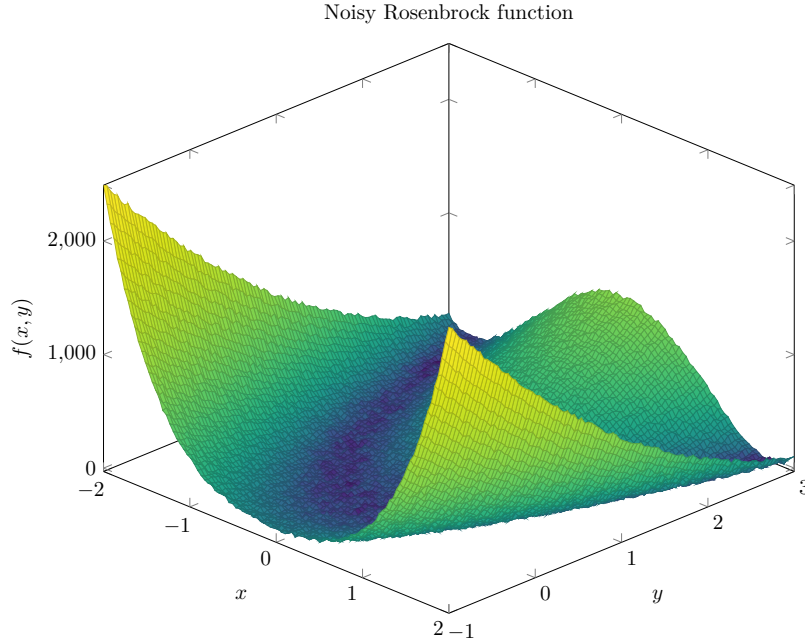


Figure 3.2: Rosenbrock function with added Gaussian noise, visualized to represent a curved, multi-modal optimization landscape. Although the landscape appears simpler than the Rastrigin function, the large, narrow valley makes locating the global minimum challenging. Gaussian noise is introduced to simulate eigenvalue crossings and other artifacts encountered in practical optimization scenarios.

Nevertheless, the LBO eigenbasis is highly sensitive to input parameters: eigenfunctions can change sign arbitrarily, and eigenvalues may swap when they are close

to each other. From this, one can conclude that the associated search space is highly non-linear and scattered with discontinuities, as eigenvalue crossings induce abrupt changes. One may nevertheless argue that the default LBO eigenbasis forms a large local optimum, and that the underlying input parameters are strongly coupled, such as edge weights, Voronoi cell weighting, or anisotropy factors.

In contrast to the function plotted in Figure 3.1, which is separable and whose parameters independently contribute to the oscillatory behavior, the present search space more closely resembles a Rosenbrock function [43] with additional Gaussian noise applied. This noise can be interpreted as modeling eigenvalue swapping and related spectral artifacts, as illustrated in Figure 3.3. As experiments presented later in chapters 4 and 5 confirm, the default LBO eigenbasis indeed forms a plateau-like local optimum (Figure 3.3: Default LBO eigenbasis basin). Small parameter perturbations have little effect on performance or on the qualitative behavior of the LBO eigenbasis, particularly in the low-frequency block, which captures highly global features.

In contrast, modifying higher-frequency components leads to more rapid changes, which can be interpreted as the steep sides of the Rosenbrock function. Such perturbations tend to push the optimization process laterally, rather than guiding it along the valley. The global optimum, however, lies within the same valley as the default LBO eigenbasis, but is separated by small humps and flat regions caused by eigenvalue crossings and other spectral artifacts. Hence, despite the strongly constrained nature of the problem, the resulting search space remains extremely difficult to navigate, both for evolutionary algorithms and for deep learning approaches. While many good solutions exist in the vicinity of the baseline LBO, only a small number of superior solutions reside in narrow regions that are particularly hard to reach.

There is substantial evidence that solutions outperforming the default LBO eigenbasis do exist, as demonstrated by numerous works such as [44][21][20][45][25], which employ either heuristic modifications or learning-based strategies to achieve improvements over the baseline.

3. Neuroevolutionary LBO

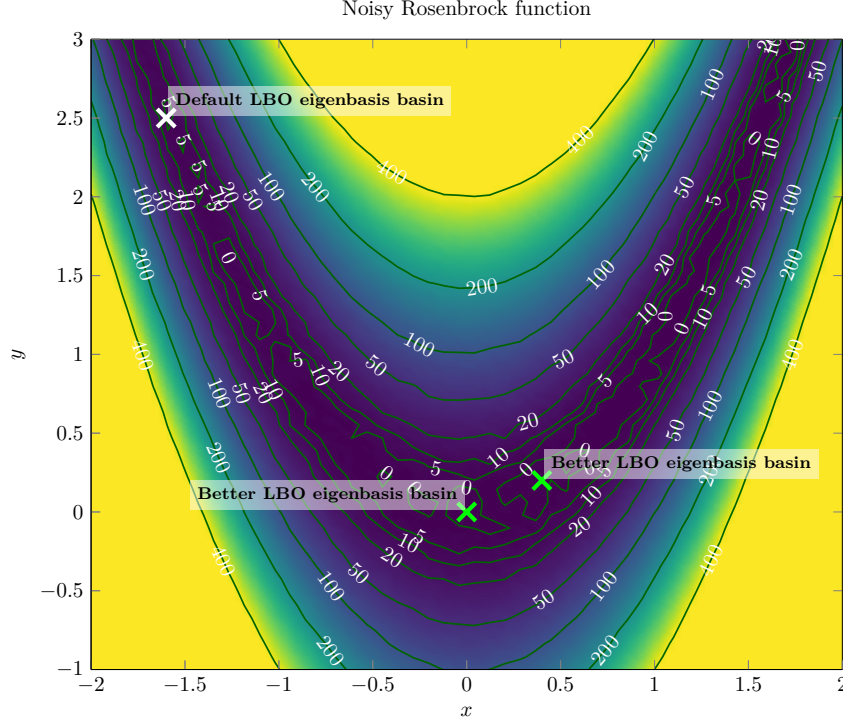


Figure 3.3: Top-down view of the Rosenbrock function with Gaussian noise. The default LBO eigenbasis basin is situated within many local optima along the valley. Although better basins exist, they are difficult to reach due to the presence of numerous intervening local minima.

Moreover, we need to clarify what this superiority represents and how it is achieved. The default LBO eigenbasis already provides substantial advantages, such as isometry invariance and robustness to near-isometries. It forms a solid backbone that can be leveraged for downstream tasks, either to derive highly expressive descriptors or to perform shape correspondence. Consequently, it is important first to consider the specific task being performed on the LBO eigenbasis.

Informally, the LBO eigenbasis has been referred to as the *Swiss army knife* of shape analysis, being effective in retrieval, classification, segmentation, and correspondence tasks. This versatility, however, comes at a cost: it performs all these tasks reasonably well but not optimally for any single one. Recent deep learning methods have begun to surpass LBO-based approaches in task-specific performance. Notably, architectures such as PointNet++ [8] and DiffusionNet [7] still incorporate the LBO to enhance performance on non-rigid tasks, demonstrating

its continued utility, although its relative advantage has begun to diminish.

Therefore, since all other deep learning tasks are trained with a specific task in mind, a learned LBO eigenbasis must also be approached from a task-specific perspective. Accordingly, we will consider a retrieval LBO eigenbasis, a classification LBO eigenbasis, a segmentation LBO eigenbasis, and a correspondence LBO eigenbasis. This ensures a fair comparison, as deep learning methods are trained in a task-specific fashion, whereas LBO-based approaches have performed limited learning on a fixed eigenbasis [17][18][16].

As a result, each task defines a distinct search space. Some search spaces are partially compatible: for example, classification and retrieval share the common goal of deriving a global descriptor. Segmentation, on the other hand, requires the spectrum to be more separable to capture part-specific information, while correspondence demands higher precision. Furthermore, once the network reaches a superior basin, additional improvement through exhaustive learning is minimal, as optimization primarily explores within the basin. This may also explain why, contrary to common belief, such networks generalize well across different datasets.

3.2 System Overview

This section provides a high-level overview of the ELBO and ELBO-Net framework, outlining its primary goals, data flow, training stages, and deployment behavior.

3.2.1 Overall Aim of the Technology

The goal of ELBO is to move away from fixed, hand-crafted geometric operators. Instead, it uses a neural network to automatically learn a custom, task-specific weighting function for a 3D shape’s mass matrix. The final aim of the complete ELBO-Net pipeline is to take a 3D mesh and generate a highly compact, discriminative k -dimensional global shape vector for accurate shape retrieval.

3. Neuroevolutionary LBO

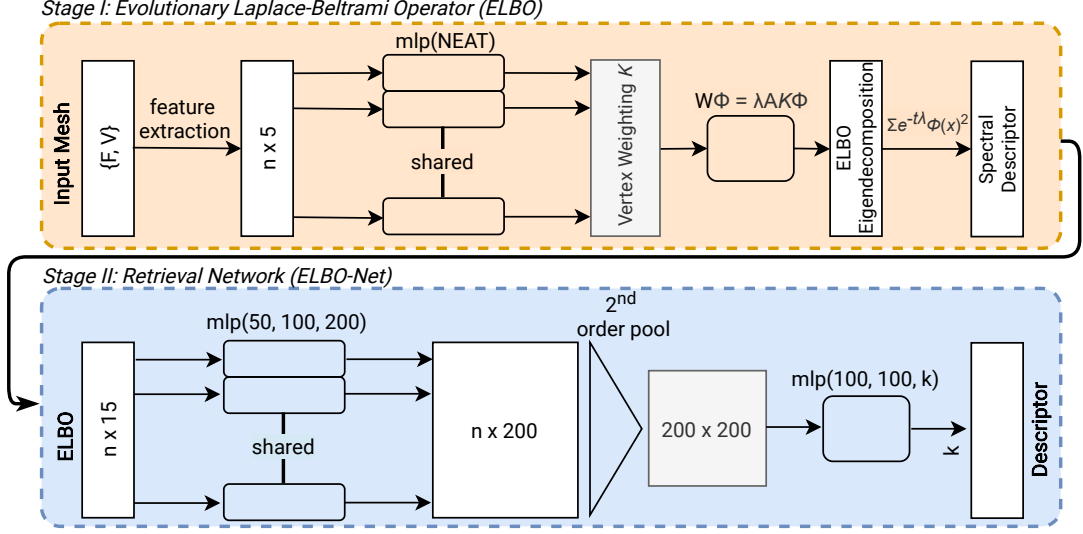


Figure 3.4: ELBO-Net Architecture. The architecture consists of two stages: optimization of the spectral descriptor (orange) and feature encoding (blue). Vertex-wise features are first computed on the input mesh and forwarded into an MLP, which is evolved using the NEAT algorithm. This stage produces a weighting term for each vertex, which is applied to the corresponding Voronoi cell, followed by the computation of a spectral descriptor. The descriptor is then passed through a neural network that encodes the features point-wise into higher-dimensional representations. These features are aggregated using second-order pooling and subsequently undergo dimensionality reduction, resulting in the final k -dimensional descriptor.

3.2.2 Inputs and Outputs

- **Inputs:** A raw 3D mesh. From this mesh, we extract a 7-dimensional local feature vector at every point (containing curvature measurements like Gaussian, Mean, and principal curvatures, along with HKS, SI, and CI).
- **Outputs:** The immediate output of Stage I is a modified, task-specific LBO eigenbasis (ϕ_i, λ_i) . The final output of Stage II is a single, global k -dimensional shape descriptor vector $\mathbf{z} \in \mathbb{R}^k$ representing the entire 3D object.

3.2.3 How It Is Trained

The system uses a two-stage offline training process:

- **Stage I (Neuroevolution):** Because optimizing network architectures and non-differentiable fitness metrics lacks an analytical gradient, we use an

evolutionary algorithm (NEAT) to evolve both the topology and weights of the mass-matrix weighting network. Once complete, this network is frozen.

- **Stage II (Feature Encoding):** With Stage I frozen, we train downstream MLPs and second-order pooling layers using standard gradient-based back-propagation. We minimize a Contrastive Loss function applied over pairs of matching and non-matching 3D shapes.

3.2.4 Evaluation in Inference

During testing and inference, the evolutionary search loops are completely removed. The pipeline operates as a clean, purely feed-forward system:

1. An unseen 3D shape has its point features extracted.
2. These features pass through the frozen, evolved network to instantly compute the custom LBO eigenbasis.
3. The resulting spectral descriptor is mapped point-wise by the Stage II network and aggregated via second-order pooling.
4. The final k -dimensional vector $\mathbf{z}$ is compared directly against other shapes using distance metrics to retrieve matching shapes.

To test the hypotheses, namely, to what extent the LBO eigenbasis is learnable, whether these basins are reachable, and if the learning is generalizable we focus on arguably simpler tasks such as retrieval, implementing a neuroevolutionary approach based on NEAT[46]. NEAT has gained popularity because it can evolve both the topology and the weights of a neural network for a given task. While setting up a differentiable loss function for these problems is possible, it is computationally expensive, the gradient derivation is non-trivial [19], and additional engineering is often required to make it robust enough for effective learning. These considerations alone justify the use of evolutionary methods.

Furthermore, it is unclear whether the global optimum can be reached via gradient descent. As performance improves, the eigenbasis can become unstable,

3. Neuroevolutionary LBO

suggesting that lower basins might be obstructed by intervening humps. In what follows, we refer to this approach simply as Neuro**E**volutionary **LBO** (ELBO).

To even further reduce the problem, we restrict the weighting to the mass matrix only,

$$W\phi = \lambda M(\theta)\phi. \quad (3.2.4.1)$$

As for the mass matrix, common choices are barycentric cells and Voronoi cells, the former involves off-diagonal elements, whereas the latter consists only of diagonal elements. Furthermore, they have some desired properties such as being more stable to the meshing of an object and having a tight error bound [11]. With this setup, we basically derived the Kinetic Laplace–Beltrami Operator (KLBO) [24], which were covered in the Background chapter earlier.

The modified generalized eigenvalue problem in [24] is,

$$W\Phi = \lambda AK\Phi, \quad (3.2.4.2)$$

where K is a diagonal matrix governed by the weighting function $C(x)$, which the authors base on heuristics with,

$$C(\mathbf{x}) = \max(\epsilon, H(\mathbf{x})) \quad (3.2.4.3)$$

where $H(\mathbf{x})$ is defined as the mean curvature at a given point.

3.3 Manifold Geometry and Feature Extraction

At this point, it is important to emphasize that while this thesis aims at an intrinsic understanding of shapes, mean curvature is an extrinsic quantity. It is rotation-invariant, which makes it appealing for this work and for related approaches, but it fundamentally depends on how a shape is embedded and bent in space. If a shape undergoes a near-isometric deformation, as illustrated in Figure 3.5, the mean curvature generally changes, even though intrinsic distances on the manifold remain largely preserved. Such deformations may include minor stretching components; however, these effects are typically negligible in comparison.

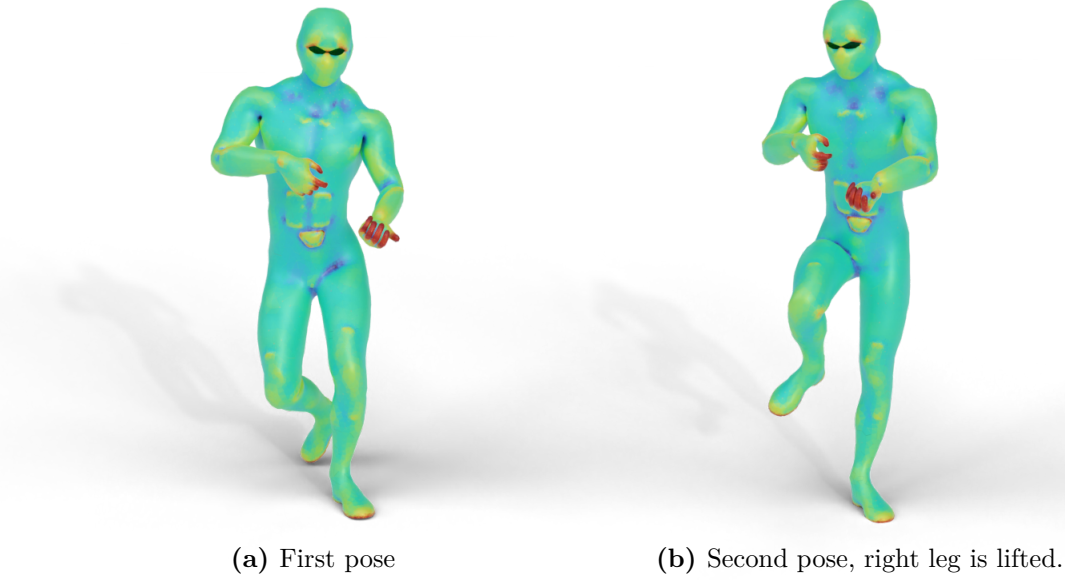


Figure 3.5: Mean curvature visualized on two shapes from the DT4D benchmark. It highlights near-isometric deformations around joint regions, such as the intersections between legs and torso. However, as a non-intrinsic feature, mean curvature reduces the invariance property of the LBO eigenbasis embedding.

Although mean curvature preserves rotation invariance, its use interferes with the embedding invariance of the LBO eigenbasis. When carefully weighted, as proposed in [24], the resulting performance gains can outweigh this loss. These gains arise because variations in mean curvature tend to localize in regions where bending occurs, which often coincide with areas affected by near-isometric deformations. In this sense, mean curvature provides complementary information that can partially compensate for the limitations of purely intrinsic representations.

It is important to recognize that any non-intrinsic modification applied to the LBO eigenbasis inevitably alters its intrinsic properties. For example, using extrinsic coordinates instead of mean curvature could yield larger improvements in segmentation tasks, as coordinates are highly expressive for part-based discrimination. However, this comes at the cost of losing both embedding and rotation invariance, which constitute primary advantages of the LBO eigenbasis. Since most benchmarks assume a fixed orientation, state-of-the-art results often implicitly benefit from this

3. Neuroevolutionary LBO

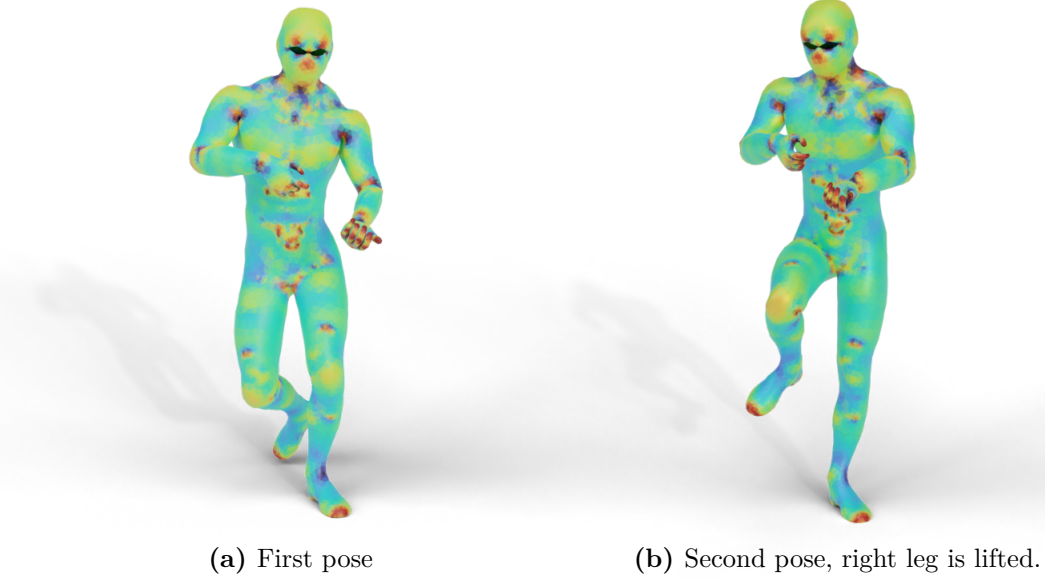


Figure 3.6: Gaussian curvature visualized on two shapes from the DT4D benchmark. As an intrinsic feature, it preserves all properties of the LBO eigenbasis. Although it varies between poses, the observed changes primarily indicate near-isometric deformations rather than bending.

assumption. A truly fair comparison would therefore require evaluation on randomly oriented datasets. Truly intrinsic features include, for example, the Gaussian curvature (Figure 3.6) and the heat kernel signature (Figure 3.7). Observable changes in these descriptors indicate an underlying modification of the manifold itself, rather than its embedding. In other words, they primarily capture near-isometric changes occurring on the manifold. We are given a two-dimensional differentiable manifold M . Each point $x \in M$ is locally homeomorphic to a two-dimensional tangent plane $T_x M$ in Euclidean space. The Riemannian metric, a 2×2 matrix, defines an inner product:

$$g_x : T_x M \times T_x M \rightarrow \mathbb{R}, \quad (3.3.0.1)$$

and is referred to as the *first fundamental form*. This metric allows us to measure intrinsic quantities, such as geodesics and angles between vectors. Any quantity derived from the first fundamental form is intrinsic.

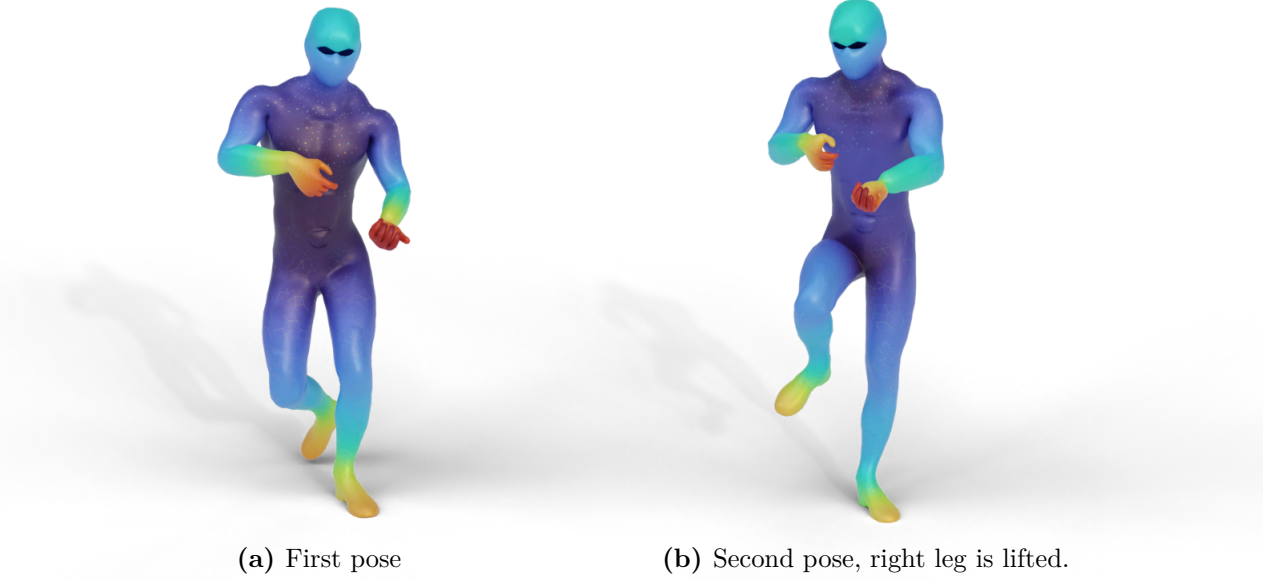


Figure 3.7: Standard heat kernel signature at the first timeframe for two shapes from the DT4D benchmark. Based on the LBO eigenbasis, it is fully intrinsic and preserves all properties and benefits of the LBO embedding.

The *second fundamental form*, also a 2×2 matrix, describes the curvature of the manifold. Its eigenvalues κ_1, κ_2 are called *principal curvatures*, and its eigenvectors v_1, v_2 are the corresponding *principal curvature directions*, forming an orthonormal basis on the tangent plane.

From Figures 3.5 to 3.7, it is evident that the first and second fundamental forms are crucial for learning an operator that aims to overcome the limitations of the LBO eigenbasis. However, it is important to understand the trade-offs involved. Extrinsic information, such as mean curvature, can provide additional discriminative power but must be used cautiously, as it can break the LBOs intrinsic properties. In their study of local shape descriptors, Heider et al. [47] demonstrate that curvature-based descriptors are stable and informative across benchmarks. Accordingly, we use these features [48], as listed in Table 3.1. To calculate the features, we utilize the Rusinkiewicz [49] curvature estimation method. An illustration of the remaining input features is provided in Figure 3.8. Upon calculating these features, we observed

3. Neuroevolutionary LBO

two major limitations. First, overall curvature estimation is highly sensitive to outliers, such as sharp edges. Second, triangles with nearly collinear vertices can introduce significant errors. Although our proposed approach is designed to generalize across benchmarks, in practice it does not, due to inconsistencies in curvature distributions between datasets. Consequently, it is necessary to tune ELBO individually for each benchmark.

3.4 Neuroevolution

As it has been detailed out, why certain methods manage to improve over the default LBO eigenbasis by using intrinsic and extrinsic features, it follows what methods can be used to optimize a given function, which takes these features as inputs.

Let M be a two-dimensional differentiable manifold, and let $\mathbf{x}_p$ denote the feature vector extracted at point $p \in M$ from Table 3.1:

$$\mathbf{x}_p = [K, H, \kappa_{\max}, \kappa_{\min}, \text{HKS}_t, \text{SI}, \text{CI}]_p. \quad (3.4.0.1)$$

We aim to learn a parametrization of the LBO mass matrix $M(\theta)$ using a neural network f_θ with parameters θ . The eigenvectors ϕ_i of the generalized eigenproblem

$$W\phi_i = \lambda_i M(\theta)\phi_i \quad (3.4.0.2)$$

are modified indirectly through changes in $M(\theta)$, while keeping W fixed. The task-specific descriptor is then

$$\mathbf{d}_p = [\phi_1(p), \phi_2(p), \dots, \phi_N(p)], \quad (3.4.0.3)$$

where the $\phi_i(p)$ now implicitly depend on θ via $M(\theta)$.

The general optimization problem can be written as:

$$\theta^* = \arg \max_{\theta} \mathcal{F}(\{\mathbf{d}_p\}_{p \in M}), \quad (3.4.0.4)$$

where $\mathcal{F}$ is a task-specific fitness function, such as:

Quantity	Definition / Formula	Type	Invariance / Notes
Gaussian curvature K	$K = \kappa_1 \kappa_2$	Intrinsic	Invariant under isometries; ignores bending
Mean curvature H	$H = \frac{\kappa_1 + \kappa_2}{2}$	Extrinsic	Sensitive to bending; depends on embedding
Maximum curvature $\kappa_{\max}$	$\kappa_{\max} = \max(\kappa_1, \kappa_2)$	Extrinsic	Sensitive to bending; embedding-dependent
Minimum curvature $\kappa_{\min}$	$\kappa_{\min} = \min(\kappa_1, \kappa_2)$	Extrinsic	Sensitive to bending; embedding-dependent
HKS	$h_t(p) = \sum_{i=0}^{\infty} e^{-\lambda_i t} \phi_i(p)^2$	Intrinsic	invariant under isometries; ignores embedding
Shape Index (SI)	$SI = \frac{2}{\pi} \arctan\left(\frac{\kappa_2 + \kappa_1}{\kappa_2 - \kappa_1}\right)$	Extrinsic	Maps local shape type; sensitive to bending
Curvature Index (CI)	$CI = \sqrt{\frac{\kappa_1^2 + \kappa_2^2}{2}}$	Extrinsic	Measures curvature magnitude; embedding-dependent

Table 3.1: Comparison of curvature quantities, Shape Index, Curvature Index, and Heat Kernel Signature, highlighting type and invariance.

3. Neuroevolutionary LBO

- **Retrieval:** maximizing similarity for similar pairs and minimizing for dissimilar pairs
- **Classification:** maximizing log-likelihood of correct class labels
- **Segmentation or correspondence:** task-specific loss measuring region or point matching

When using NEAT, each offspring θ represents a specific parameterization of $M(\theta)$. The network evolves over generations to maximize $\mathcal{F}$, thereby indirectly shaping the LBO eigenvectors to better suit the task. A key advantage of NEAT is that it evolves both weights and topology, preventing the network architecture from becoming a bottleneck. The evolved topology can later guide the architecture of the methods introduced in chapters 4 and 5.

It is important to note that the neural network is shared across all points $p \in M$, meaning each point passes through the same network. Consequently, the network can remain relatively small, as even a few layers and neurons are sufficient to learn complex functions. This design reduces the dimensionality of the search space discussed earlier, alleviating the difficulty of optimization to a manageable number of variables. With the introduction of the Princeton Shape Benchmark [50], complex performance metrics for shape retrieval have been established, including Nearest Neighbour (NN), First Tier (FT), Second Tier (ST), E-Measure (E), and Discounted Cumulative Gain (DCG), each evaluating different properties of the resulting *distance matrix*. We treat these metrics as objectives that assess ranking quality from different perspectives. We use a linear combination of these metrics as our fitness function:

$$f(\theta) = \sum w_i m_i(\theta), \quad (3.4.0.5)$$

where the optimal weights w_i and the choice of performance metrics m_i are hyperparameters. This approach has the advantage that benchmarks typically contain a diverse set of manifolds, improving the generalization of a learned

descriptor across datasets. However, the computational cost of evaluating a single offspring on the entire benchmark limits the feasible population size.

An alternative to class-based learning is to leverage known similar and dissimilar points on the manifolds. Given a set of similar pairs $P_+ = (x, x_+)$, we compute the Euclidean distance between their descriptors:

$$d^2(\mathbf{x}, \mathbf{x}_+) = \sum_{i=1}^N \sqrt{(\sigma(x_i) - \sigma(x_{i+}))^2}, \quad (3.4.0.6)$$

where $\sigma(\cdot)$ denotes the chosen signature and i indexes the frequency. Hand-labeled landmarks (knees, chest, and arms) on few (4 pairs) manifolds, were used for training. These points were selected strategically as landmarks reflect whether corresponding body parts across manifolds exhibit similar dynamics. This approach increases the invariance of the LBO descriptors under near-isometric deformations or noise.

To balance invariance with discriminativity, we also consider dissimilar points. Let $P_- = (x, x_-)$ be a dissimilar pair, defined as a random point on manifold a paired with a random point on manifold b . Using both similar and dissimilar pairs, the loss function is defined as:

$$\mathcal{L}(\theta, \mu, d) = (1 - \mu) d_+^2 + \mu d_-^2, \quad (3.4.0.7)$$

where μ controls the tradeoff between invariance and discriminativity. This approach is similar to methods previously adopted in [18, 27] to derive task-specific descriptors.

We propose two approaches for training the neural network. First, by evolving a network to weight manifold features using a benchmark as the training set, with benchmark metrics serving as components of the fitness function (Eq. 3.4.0.5). Second, by training on hand-labelled pairs of known similar and dissimilar points on the manifolds (Eq. 3.4.0.7). We demonstrate in the Evaluation section that both approaches yield similar weighting schemes and achieve comparable performance improvements over the standard LBO.



Figure 3.8: Different extrinsic and intrinsic features of a manifold of the SHREC’14 Shape retrieval of non-rigid 3D human models [51] training set. From left to right: Maximum curvature, minimum curvature, Gaussian curvature, Shape Index, and Curvature Index deviation.

3.5 ELBO

We now use the neuroevolutionary approach to build a complete pipeline, as depicted in Figure 3.4. In the orange box, features are first extracted according to Table 3.1. These features are then passed through the shared neural network, which is evolved using NEAT. The resulting output provides a weighting that is applied to the mass matrix M and used to solve the GEP in Equation 3.4.0.2. With the task-specific LBO eigenbasis obtained in this way, we can compute the spectral descriptor of choice; in this work, we use the HKS. Extensive experimentation has shown that the choice of spectral descriptor used for training has little to no effect on performance. ELBO is agnostic to the specific spectral descriptor: networks trained with WKS achieved similar results when evaluated using HKS or Scale-Invariant Heat Kernel Signature (SI-HKS), and vice versa. Ideally, the entire pipeline would operate end-to-end and synchronously. However, since no differentiable loss function is defined for the NEAT-evolved part, it is not possible to train a downstream neural network jointly with NEAT embedded in the pipeline.

Algorithm 1 Stage I: Neuroevolutionary Learning of the LBO (ELBO)

Require: Mesh dataset $\mathcal{M}$, feature extractor Φ , population size P , generations G **Ensure:** Optimized mass matrix weighting function $M(\theta)$

```

1: Initialize NEAT population  $\{\theta_1, \dots, \theta_P\}$ 
2: for generation = 1 to  $G$  do
3:   for each individual  $\theta$  in population do
4:     for each mesh  $M \in \mathcal{M}$  do
5:       Extract point-wise features  $\mathbf{f} = \Phi(M)$ 
6:       Compute weights  $w = \text{NN}_\theta(\mathbf{f})$ 
7:       Construct weighted mass matrix  $M(w)$ 
8:       Solve generalized eigenproblem:
           
$$W\phi = \lambda M(w)\phi$$

9:       Compute spectral descriptor  $\sigma(M(w))$ 
10:    end for
11:    Evaluate fitness  $\mathcal{F}(\theta)$  using task-specific objective
12:  end for
13:  Apply NEAT operators (selection, mutation, crossover, speciation)
14: end for
15: return best individual  $\theta^*$ 

```

Consequently, NEAT must be trained offline, and the resulting weights are then used to train the downstream network. This approach has significant downsides, which are important to highlight, as they motivate the subsequent chapters. As discussed previously, neuroevolutionary methods are powerful, but the disconnect between NEAT evolution and downstream training is critical to understand. Even if the fitness function used for NEAT and the loss function for the downstream network align, training them separately reduces learning effectiveness.

This ineffectiveness arises because the backend of the neural network plays a crucial role: projecting the descriptors into the task-specific space. The backend can absorb and adapt the descriptors, learning additional features during training. One might assume that providing better features from NEAT should improve performance; however, in this context, better must be defined precisely. It means the descriptors must supply exactly what the backend needs. Even if the descriptors are evolved using a task-aligned fitness function, they may not perfectly match the backends requirements, since the backend itself modifies and backpropagates

3. Neuroevolutionary LBO

loss during training.

While we will be introducing a custom backend to extract features effectively from spectral descriptors, first experiments were carried out in the framework specified by [24]. The backend is probabilistic and based on GMMs using fisher or super vector, which we will be introducing briefly, but refer the reader here to the Limbergers PhD thesis [44] for a more detailed derivation.

3.6 ELBO-Net

Recent works on learning from unordered point clouds, such as PointNet [5], PointNet++ [8], and DGCNN [6], have achieved significant success. The ability to supply point clouds with additional features, such as normals, color, or spectral descriptors, makes them particularly suitable for non-rigid shape retrieval. Moreover, the introduction of 2nd-order pooling for non-rigid retrieval tasks, as demonstrated by [52], has proven superior to standard average or max pooling. Experiments with the aforementioned approaches showed that simple pooling strategies cannot match the performance of 2nd-order pooling for non-rigid shape retrieval. Similarly, constructing a k -NN graph, as in [6], did not yield significant improvements. Consequently, we opted for an architecture that combines the most promising components for non-rigid shape retrieval. The final architecture can be observed in the blue box depicted in Figure 3.4.

Point Sampling: Since points are typically not uniformly distributed in a point cloud, uniform sampling is necessary to achieve optimal outcomes. As we also work with a subset of all available points, the selection of which points to include in the learning process becomes an important consideration.

Our goal is to derive a stable k -dimensional descriptor for shape retrieval tasks. The loss function in Equation 3.6.0.1 can be directly applied, yielding the Contrastive Loss function [53]:

$$\mathcal{L}(\theta, \mu, d) = (1 - \mu) d_+^2 + \mu d_-^2, \quad (3.6.0.1)$$

Algorithm 2 Stage II: ELBO-Net Feature Encoding and Aggregation

Require: Spectral descriptors $\Phi \in \mathbb{R}^{n \times d}$ (from Stage I), output dimension k **Ensure:** Global shape descriptor $\mathbf{z} \in \mathbb{R}^k$

- 1: Initialize shared MLP parameters θ_{enc}
- 2: Initialize projection MLP parameters θ_{proj}

3: **Point-wise feature encoding**4: **for** $i = 1, \dots, n$ **do**5: $\mathbf{h}_i \leftarrow \text{MLP}_{\theta_{\text{enc}}}(\Phi_i)$ 6: **end for**7: Stack features: $\mathbf{H} \leftarrow [\mathbf{h}_1, \dots, \mathbf{h}_n] \in \mathbb{R}^{n \times 200}$ 8: **Second-order pooling**9: $\mathbf{G} \leftarrow \frac{1}{n} \mathbf{H}^\top \mathbf{H} \in \mathbb{R}^{200 \times 200}$ 10: **Dimensionality reduction**11: Vectorize pooled features: $\mathbf{g} \leftarrow \text{vec}(\mathbf{G})$ 12: $\mathbf{z} \leftarrow \text{MLP}_{\theta_{\text{proj}}}(\mathbf{g})$ 13: **return** $\mathbf{z}$

where μ is a binary indicator distinguishing similar ($\mu = 0$) and dissimilar ($\mu = 1$) pairs. For this, we generate sets of similar and dissimilar pairs of manifolds, which are then used to learn the descriptor.

3.7 Evaluation

In this section, we evaluate the proposed approach and assess its ability to improve upon the standard LBO. It is important to note that this work generalizes and extends KLBO [24], and should be understood as a continuation thereof. Consequently, our initial experiments are conducted within the same experimental framework as KLBO. However, instead of employing a heuristics-based weighting scheme, the weighting function is evolved, allowing us to directly assess whether accuracy improvements can be achieved through learning.

Specifically, we employ Stage I (orange box) of the pipeline shown in Figure 3.4. Rather than using the Stage II (blue box) neural backend, we adopt a supervector-based aggregation [54], as previously used in [44]. By keeping the pipeline as close as

3. Neuroevolutionary LBO

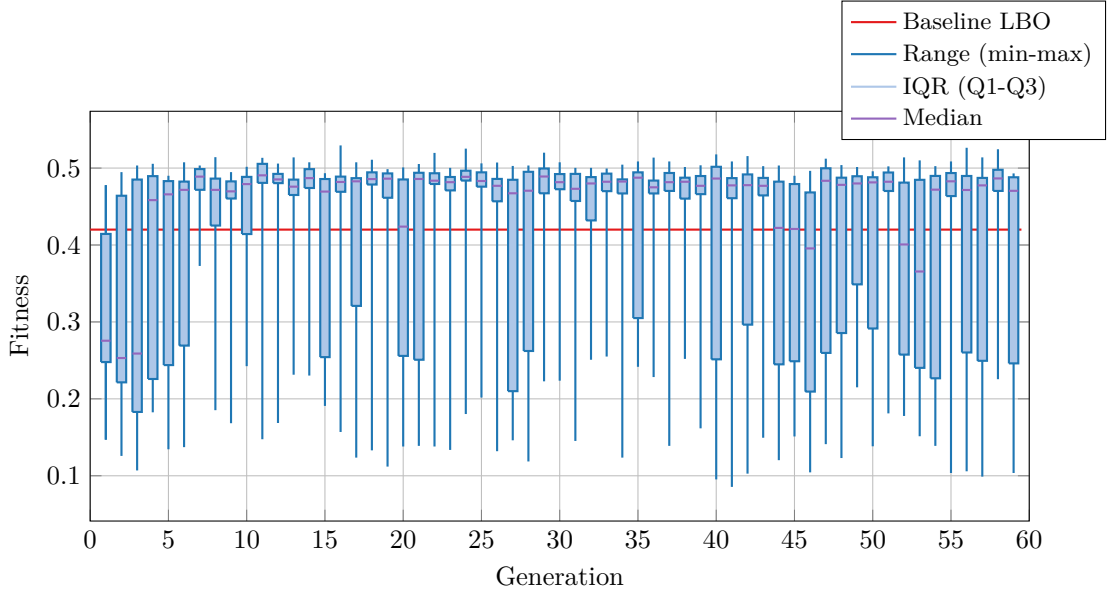


Figure 3.9: Boxplot showing ELBO values over successive generations trained using benchmark metrics on SHREC'10. The median quickly exceeds the baseline LBO by the fifth generation, demonstrating rapid improvement, and the optimal ELBO is reached around the sixteenth generation.

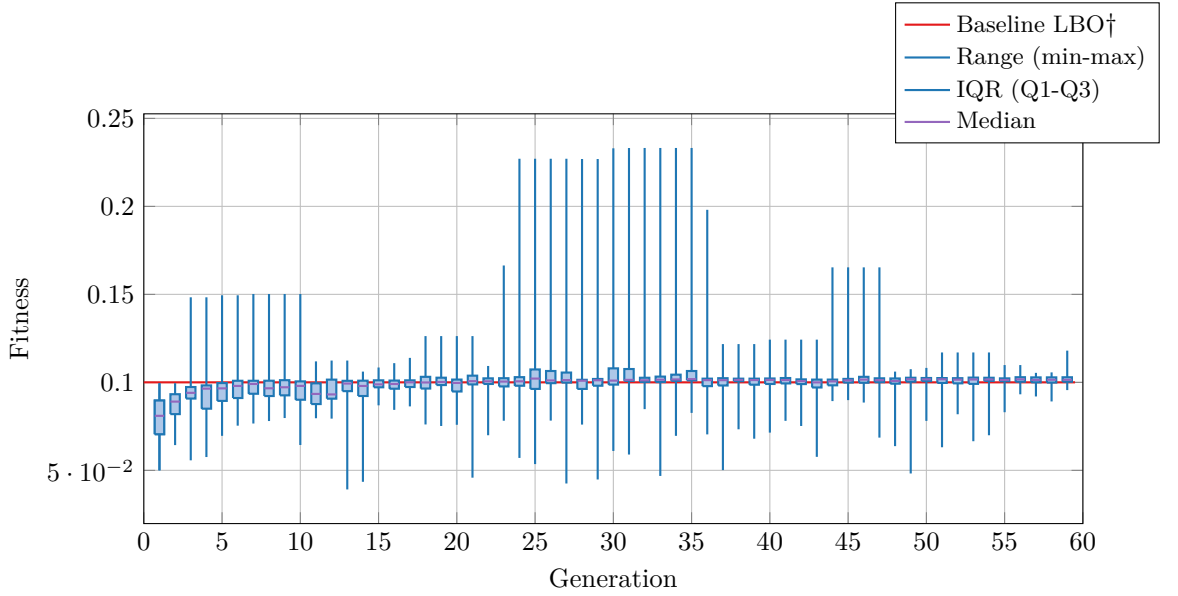


Figure 3.10: Boxplot of ELBO values across generations for networks trained with contrastive loss using landmarks on SHREC'14 (real). The median reaches the baseline LBO by the seventeenth generation. The observed performance exhibits a wider range, reflecting the reduced regularization compared to training with full benchmark metrics.

possible to the original KLBO setup and modifying only the mass matrix weighting, any observed performance gains, even if modest, provide strong evidence that a task-specific, superior LBO eigenbasis can indeed be learned. Furthermore, we use KLBO as the baseline for any improvements achieved.

After confirming these results, we proceed to experiments using the full architecture shown in Figure 3.4, incorporating Stage II (blue box) as the backend. This enables more competitive performance across benchmarks and allows us to fully exploit the benefits of the proposed ELBO framework. We further evaluate each learning strategy introduced earlier. First, we optimize performance directly on SHREC’10 using the aggregated metric defined in Eq. 3.4.0.5. Second, we employ contrastive learning based on pairs of similar and dissimilar points across manifolds (for the remainder of the benchmarks), using the loss defined in Eq. 3.4.0.7, in order to demonstrate the potential of this alternative supervision signal. The training dynamics of both methods are illustrated in Figures 3.9 and 3.10, respectively. The advantages and limitations of each approach are apparent. Training with fitness function 3.4.0.5 allows the network to converge quickly to a modest solution; however, the potential upside is limited, as the fitness function is effectively evaluated over the entire benchmark, constraining the network. Nevertheless, within a few generations, the median performance rises above an approximate baseline LBO performance (a single offspring was used to determine a performance comparable to the baseline LBO to assess relative improvement).

In contrast, Figure 3.10, which depicts training using landmarks on manifolds with a contrastive loss (Eq. 3.4.0.7), exhibits different behavior. The maximum performance fluctuates substantially, often reaching higher values, while the Interquartile Range (IQR) is narrower. The median performance, however, hovers around the baseline LBO. This suggests that training with landmarks and contrastive loss removes the implicit regularization present with fitness function 3.4.0.5, which was induced by evaluating multiple metrics. While this approach allows higher performance on specific manifolds, generalization becomes more challenging. Furthermore, it indicates that a small set of landmark-labeled manifolds

3. Neuroevolutionary LBO

may suffice to achieve reasonable performance, but additional training data is necessary to improve generalization.

3.7.1 ELBO

For all experiments, SI-HKS and WKS are used as spectral descriptors, and features are encoded using the super-vector representation. We use the official author-provided MATLAB implementation of the NEAT algorithm [46]. Across all experiments, we employ a population size of 64, a stagnation threshold of 10 generations, a selection pressure of 1.8, and a crossover probability of 0.7. Empirically, we found a scaled sigmoid activation function to be the most effective. All remaining parameters were kept at their default values as specified in [46]. The experiments were conducted on a machine equipped with four Intel Xeon 6130 processors (16 cores each, 2.1 GHz) and 768 GB of Random Access Memory (RAM). The evaluation time for a single ELBO offspring ranged from a few seconds using the similar-points approach to several minutes when using the benchmark-based approach.

Method	NN	FT	ST	E	DCG
SV-WKS (LBO)	0.9900	0.8924	0.9629	0.7053	0.9773
SV-WKS (KLBO)	0.9950	0.9656	0.9952	0.7328	0.9938
SV-WKS (ELBO)	0.9950	0.9874	0.9942	0.7400	0.9962
SV-SIHKS (LBO)	0.9638	0.8157	0.9220	0.6714	0.9479
SV-SIHKS (KLBO)	0.988	0.8665	0.9601	0.7052	0.972
SV-SIHKS (ELBO)	1.0	0.9971	1.0	0.7447	0.998

Table 3.2: Results on the SHREC’10 benchmark. ELBO and KLBO refer to the respective weighting schemes. Features were encoded using a super vector and either SI-HKS or WKS descriptors.

We first evaluate ELBO on the SHREC10 non-rigid benchmark [55], which consists of 200 watertight meshes grouped into ten semantic classes. Retrieval performance is reported in Table 3.2.

When using SI-HKS, ELBO yields a substantial improvement in the FT and E metrics, both of which emphasize early precision in the ranked retrieval results. This behaviour is consistent with our hypothesis that the learned mass-matrix weighting

reduces spectral variability across near-isometric shapes, thereby improving the alignment of eigenbases among shapes within the same class.

In contrast, improvements obtained with WKS are more moderate. This can be attributed to the strong baseline performance of WKS, which leaves less headroom for improvement. Moreover, WKS places greater emphasis on higher-frequency components, whereas the learned weighting predominantly influences the lower end of the spectrum, where eigenfunctions are more stable and informative for non-rigid shape similarity.

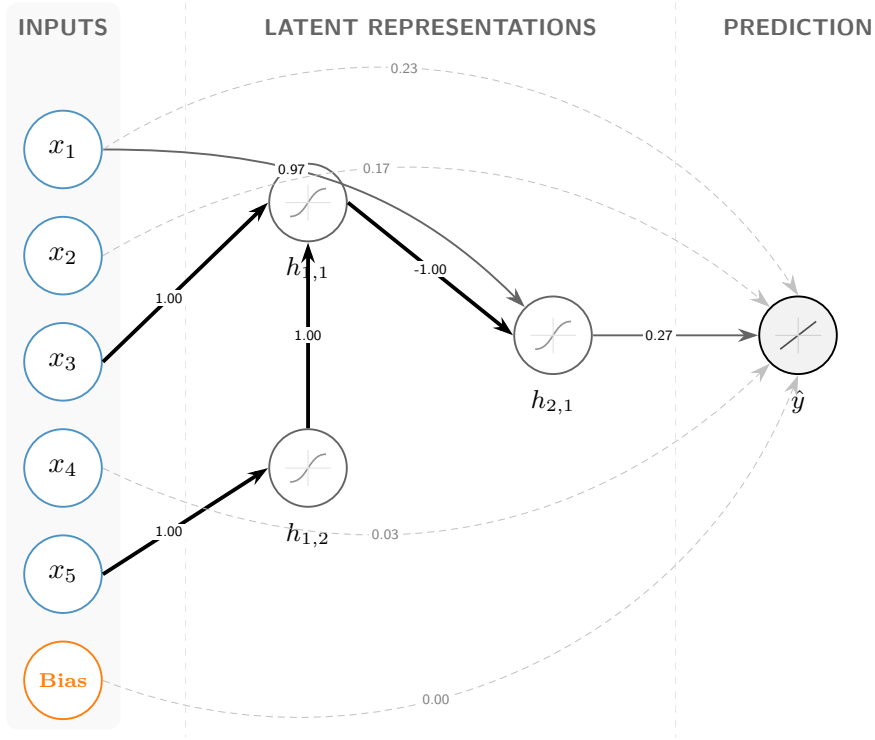


Figure 3.11: Evolved network topology and weights obtained via NEAT on SHREC’10. The network uses a sigmoid activation function, and the output neuron performs regression. Notably, skip connections are extensively employed in the neuroevolutionary design. One could argue that because descriptors are already achieving a good performance on this benchmark, that the topology remains trivial. However, since the descriptors already achieve strong performance on this benchmark, the resulting topology remains relatively simple.

The evolved network responsible for these results is shown in Figure 3.11. NEAT identified the first and fifth input features as the most important, as evidenced by their strong direct connections to the output layer as well as links to a hidden

3. Neuroevolutionary LBO

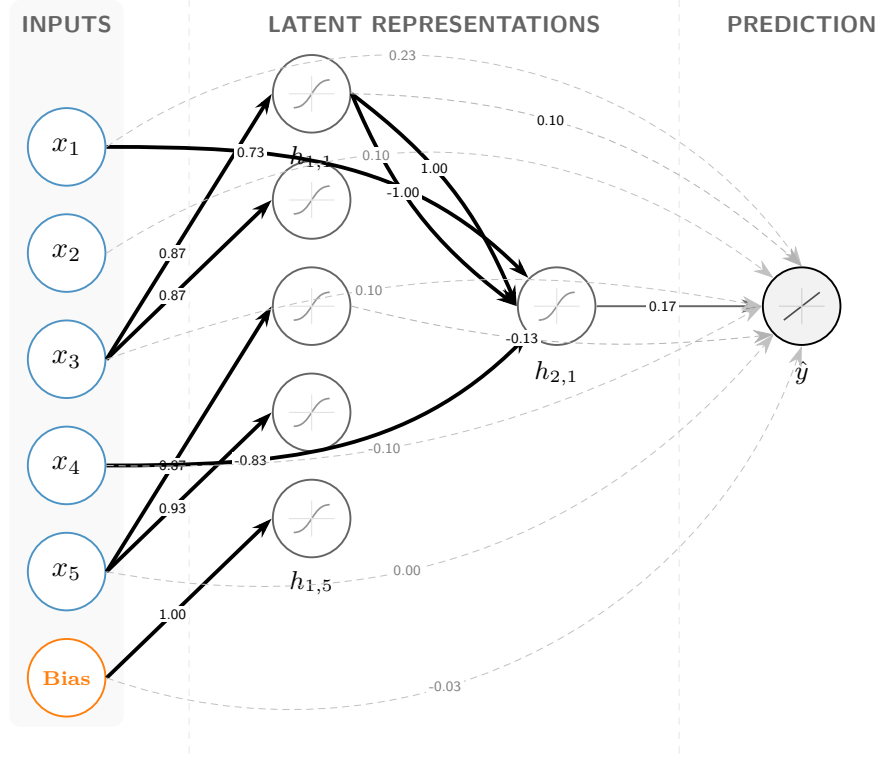


Figure 3.12: Evolved network topology and weights obtained via NEAT on SHREC’14 (real). The network uses a sigmoid activation function, and the output neuron performs regression. Multiple hidden layers and numerous skip connections are observed, reflecting the increased complexity required to achieve strong performance on this benchmark.

layer. An additional noteworthy connection occurs within the first hidden layer, where the neuron receiving input from the fifth feature connects to the neuron receiving input from the third feature. Although the overall architecture is relatively simple, the combination of skip connections and intra-layer links introduces a level of complexity that is difficult to derive manually.

The SHREC14 Non-rigid 3D Human Models benchmark [51] is divided into two subsets: a synthetic and a real dataset. The synthetic set consists of shapes generated using 3D modeling software, whereas the real set is composed of actual 3D scans. For both subsets, the authors provide an independent training set, which we utilize for training ELBO. We choose to train on both the real and synthetic benchmark, as both constitute a more challenging setting: several manifolds are difficult to distinguish even under close visual inspection. The corresponding results are reported in Tables 4.3 and 3.4 respectively. ELBO achieves competitive

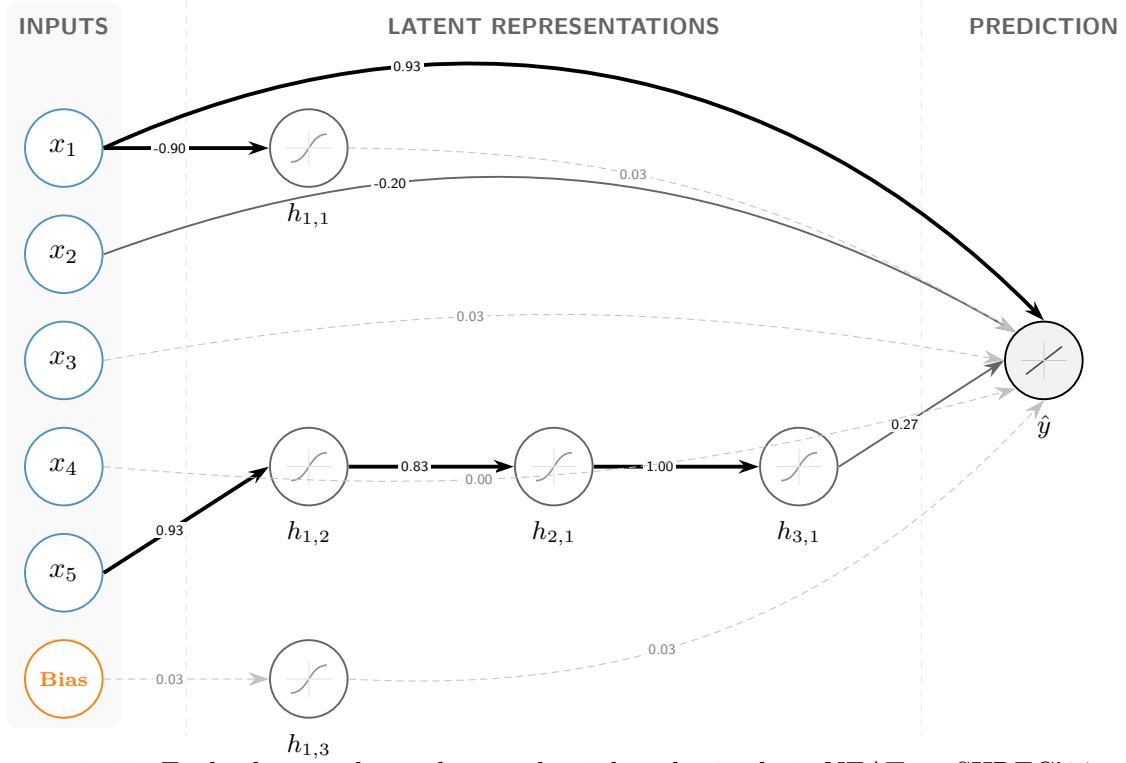


Figure 3.13: Evolved network topology and weights obtained via NEAT on SHREC'14 (synthetic). The network uses a sigmoid activation function, with a regression output neuron. Multiple hidden layers and extensive skip connections are present. Notably, the fifth input feature passes through three activation functions before reaching the output neuron.

Method	NN	FT	ST	E	DCG
3D-DL [56]	0.225	0.193	0.374	0.262	0.504
DeepGM [57]	0.725	0.536	0.827	0.412	0.782
supDLTrain [58]	0.793	0.727	0.914	0.432	0.891
SV-WKS (KLBO)	0.0725	0.0781	0.1596	0.1010	0.3410
SV-WKS (ELBO)	0.1425	0.1072	0.1711	0.1082	0.3614
SV-SIHKS (KLBO)	0.1125	0.1163	0.2242	0.1509	0.383
SV-SIHKS (ELBO)	0.3175	0.2844	0.4481	0.2654	0.5536

Table 3.3: Results on the SHREC'14 (real) benchmark in comparison to SOTA methods. ELBO and KLBO refer to the respective weighting schemes. Features were encoded using a super vector and either SI-HKS or WKS descriptors.

performance and yields a noticeable improvement over KLBO [44]. While these gains are relatively modest on the real benchmark, subsequent experiments in this section demonstrate that the primary performance bottleneck is attributable to architectural limitations rather than the weighting scheme itself. What stands out

3. Neuroevolutionary LBO

is that SI-HKS again exhibits a larger improvement than WKS. This supports the earlier observation that the effects of learning are more pronounced for descriptors dominated by lower-frequency components. Since HKS emphasizes the low end of the spectrum, modifications of the eigenbasis induced by ELBO become more directly visible, whereas descriptors relying more heavily on higher frequencies tend to mask these effects. The results on the synthetic benchmark achieve SOTA performance, with the FT reaching the highest score and outperforming the next best method [58] by a significant margin. This demonstrates that only a small improvement over existing methods is needed to achieve top performance.

The evolved network, including both its topology and weights, is shown in Figures 3.12 and 3.13, respectively. In the first network, two hidden layers were evolved. Notably, the first and fourth input features have direct connections to the second hidden layer in addition to links to the output layer, with the first feature exhibiting higher connection weights. The presence of numerous connections, including skip connections, renders this network relatively complex. In the second network, multiple hidden layers are observed, with the first and fifth features having the greatest impact on the predicted weights. The fifth feature, in particular, appears to be processed multiple times through three separate MLPs.

Overall, the network demonstrates how neuroevolution can discover a compact yet expressive architecture that effectively shapes the eigenbasis while maintaining interpretability of feature importance.

Method	NN	FT	ST	E	DCG
3D-DL [56]	0.923	0.760	0.911	0.641	0.921
DeepGM [57]	0.993	0.814	0.983	0.723	0.967
supDLTrain [58]	0.960	0.887	0.991	0.721	0.975
SV-SIHKS (KLBO)	0.888	0.816	0.951	0.688	0.940
SV-SIHKS (ELBO)	0.8933	0.9039	0.9814	0.7235	0.9613

Table 3.4: Results on the SHREC’14 (synthetic) benchmark in comparison to SOTA methods. ELBO and KLBO refer to the respective weighting schemes. Features were encoded using a super vector and either SI-HKS or WKS descriptors.

When observing the retrieval heatmap in Figure 3.14. ELBO clearly enhances intra-class similarity while preserving inter-class discriminability. In particular,

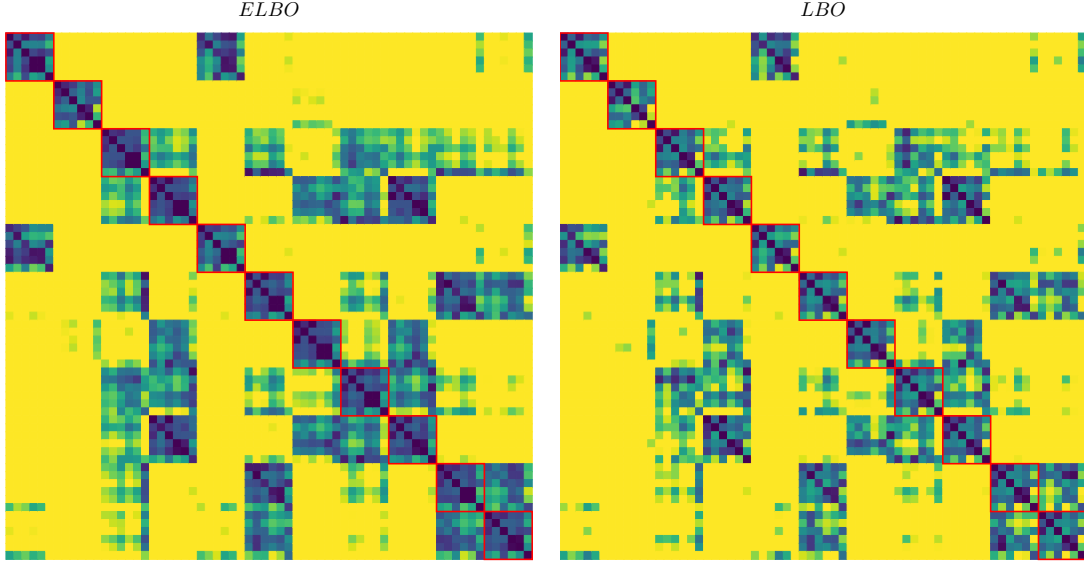


Figure 3.14: Side-by-side comparison of ELBO SI-HKS and standard LBO SI-HKS on the SHREC'14 Real benchmark. ELBO clearly enhances intra-class similarity while preserving inter-class discriminability. Shapes belonging to the same class are highlighted with red borders. To enhance clarity, the heatmap is shown with 11 classes, each containing 6 representative shapes.

the elements in the bottom-right corner of each red-bordered cell are noticeably closer within their respective classes. Notably, ELBO also aligns the off-diagonal elements: although the standard LBO sometimes appears slightly better off-diagonal, this is largely because ELBO enforces more consistent shape alignment. Consequently, if a class is already similar to another, it becomes uniformly similar and vice versa, whereas the standard LBO exhibits a more mixed behavior. This observation supports our initial conjecture that applying weights derived from extrinsic information can potentially compromise the embedding-invariance of the LBO. This further demonstrates that while a descriptor can capture the features of a shape, a robust backend is still necessary to distinguish between shapes, since some are very similar and even difficult for humans to differentiate. Examining the spectral descriptors of a class in Figures 3.16 and 3.15 further highlights this effect. The learned weighting leads to a noticeably stronger alignment of the descriptors, substantially reducing discrepancies in the spectral space. This is particularly evident in the superimposed heatmaps of the plotted shapes.

3. Neuroevolutionary LBO

Under the standard LBO (Figure 3.15), the largest discrepancy is observed between shapes (d) and (f). In contrast, this distance is significantly reduced when using ELBO (Figure 3.16). A closer inspection of this pair reveals that this effect is not coincidental: the two shapes represent mirrored poses and therefore exhibit one of the strongest near-isometric deformations within the class. While this deformation leads to a pronounced separation in the spectral space for the standard LBO, the ELBO successfully mitigates this effect, resulting in a more consistent and aligned descriptor representation.

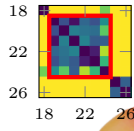
We further investigate this effect by focusing on the two shapes shown in Figure 3.17, where the first ten time frames of the SI-HKS are visualized for both the standard LBO and the ELBO. The remaining time frames exhibit minimal variation and are therefore omitted for clarity.

In regions undergoing near-isometric deformation, most notably around the pectoralis minor and the deltoids of the shapes, the ELBO produces a markedly more stable descriptor. At $t = 1$, the standard LBO clearly reflects the mirrored pose in the heat kernel signature. The raised arm appears more prominent than the opposite arm, with this asymmetry reversed in the mirrored pose. In contrast, the ELBO substantially reduces this discrepancy. A similar effect is observed $t = 6$, where the ELBO again yields a more consistent response across the two shapes.

To quantify this observation, we probe two locations affected by the deformation, namely the deltoid (green marker at $t = 1$) and the triceps brachii (purple marker at $t = 1$). As shown in the lower plots in Figure 3.17, the corresponding heat kernel signatures exhibit a smaller Euclidean distance when using the ELBO, corroborating the qualitative visual evidence. Beyond reducing isolated differences, the ELBO also achieves a tighter temporal alignment of the heat kernel responses across time scales, mitigating even subtle discrepancies. As these improvements accumulate over the entire manifold, their impact on the resulting descriptor becomes substantial.

When plotting the weights produced by the evolved network in Figure 3.12 on a manifold pair in Figure 3.18, we can observe that bent regions of the manifold is heavily downweighted.

LBO (18-26)



(a) First pose.



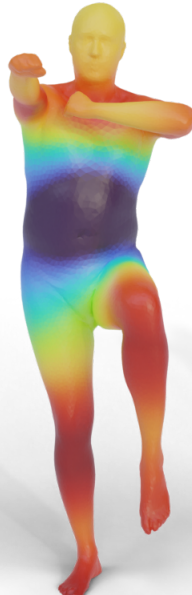
(b) Second pose.



(c) Third pose.



(d) Fourth pose.



(e) Fifth pose.

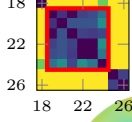


(f) Sixth pose.

Figure 3.15: Standard heat kernel signature at 1st timeframe on two shapes from the DT4D benchmark. The standard heat kernel signature is based on the LBO eigenbasis, which makes it fully intrinsic and retains all LBO eigenbasis benefits.

3. Neuroevolutionary LBO

ELBO (18-26)



(a) First pose.



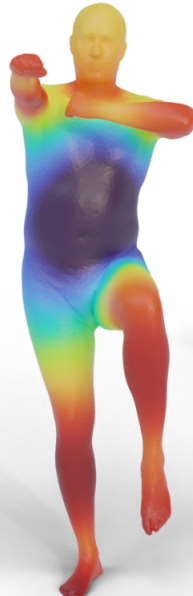
(b) Second pose.



(c) Third pose.



(d) Fourth pose.



(e) Fifth pose.



(f) Sixth pose.

Figure 3.16: ELBO HKS at the first timeframe on submatrix shapes from SHREC'14. Differences in the superimposed submatrix, notably between shapes 4 and 6 and shapes 1 and 6, are reduced compared to the standard HKS, with ELBO partially mitigating discrepancies caused by near-isometric deformations around the shoulders.

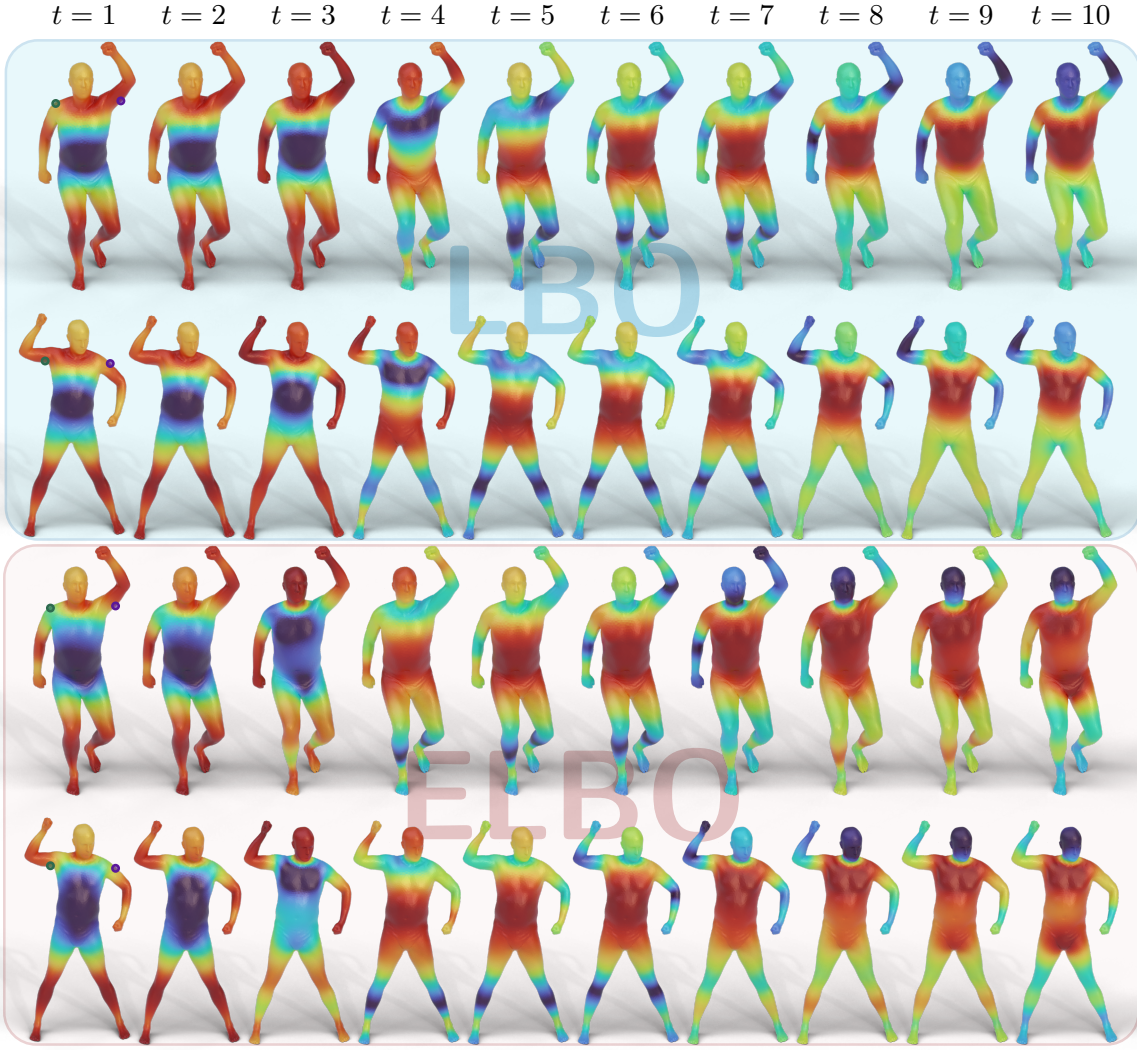
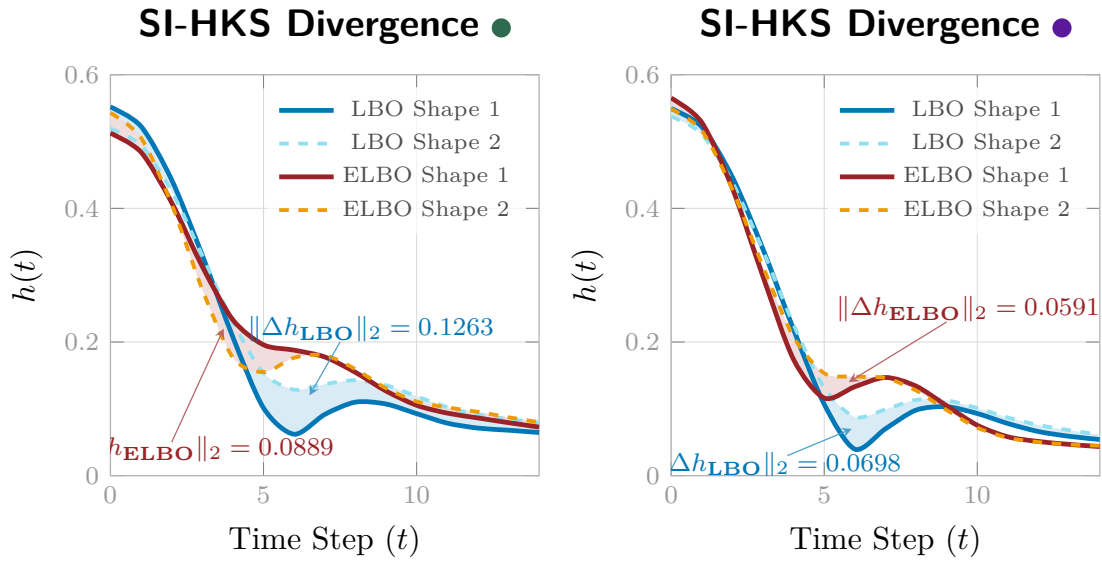


Figure 3.17: Drilldown of the two most discrepant shapes within a class, comparing their standard and ELBO HKS. The ELBO descriptor not only appears visually more stable but also reduces the numerical distance between the shapes, resulting in improved performance.

3. Neuroevolutionary LBO

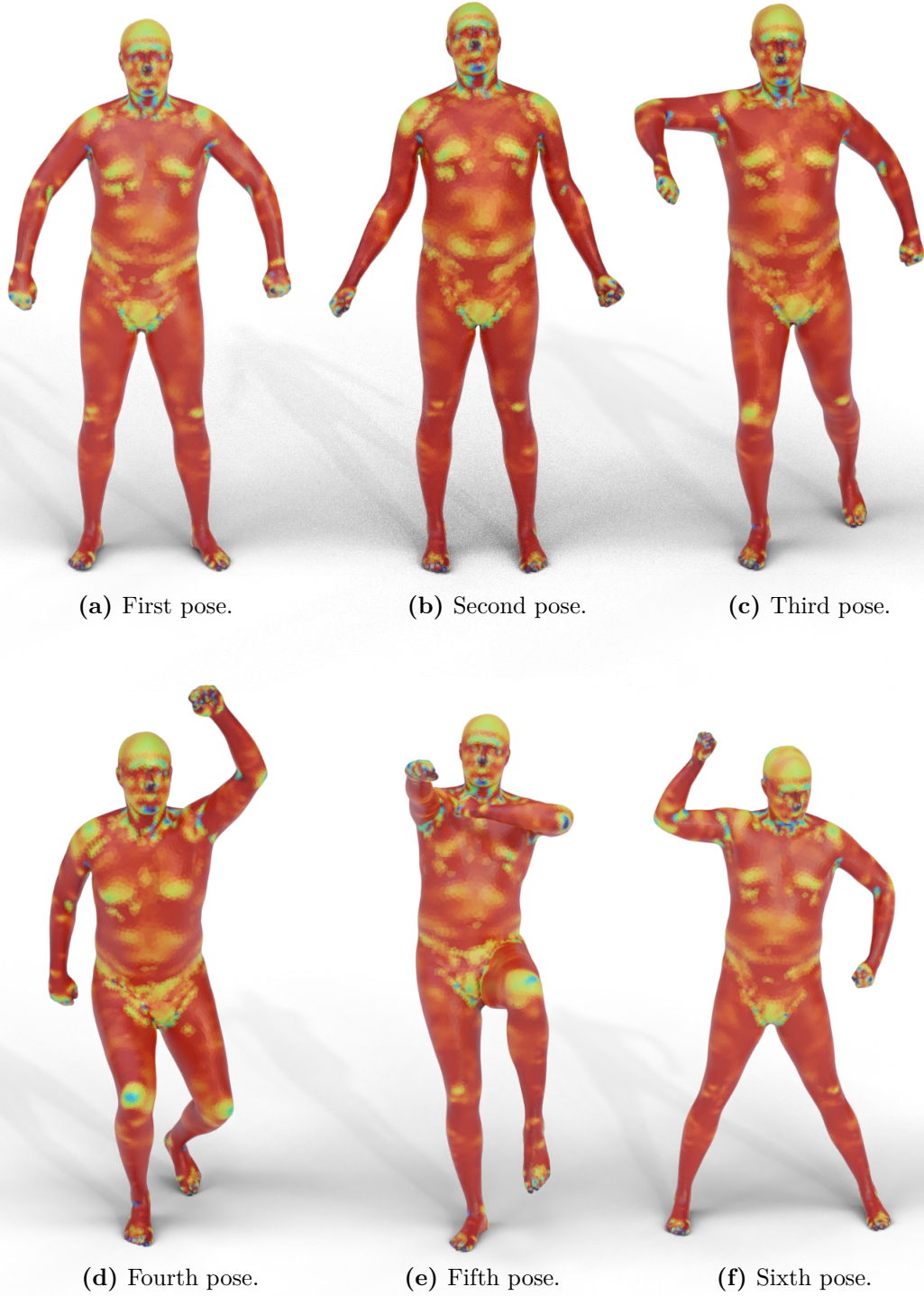


Figure 3.18: Features evolved via ELBO for weighting the mass matrix. Specific areas, notably around the armpits, are weighted to enhance intra-class similarity across shapes.

One can observe the weights in the network, emphasize all features, however, maximum curvature and Gaussian curvature seem to dominate the resulting weighting. While it is difficult to comprehend what these weights do on an underlying level with the LBO eigenbasis, one could interpret that regions, which differ the most from pose to pose, are being reduced. Having identified the bottleneck and observed limited further gains, we conclude the KLBO-based experiments and transition to a more end-to-end learning paradigm (3.4). Even though gradients are never passed fully end-to-end, the pipeline from descriptor to prediction is trained.

3.7.2 ELBO-Net

We introduced two methods of training the ELBO. Firstly, evolving the ELBO by using a benchmark as a training set and using the loss function given in Eq. 3.4.0.5, and secondly by using similar and dissimilar points on manifolds and using the loss function given in Eq. 3.6.0.1. Both approaches share the same ELBO architecture given in 3.4. The latter approach proves itself more powerful, as it requires fewer generations and less computing power to evolve an optimal neural network, when compared with the former. The difficulty lies in choosing the right tradeoff parameter μ as it defines the solution space and thereby the generalization performance. The features in figure 3.15 is specific to a manifold and does not necessarily translate to benchmark performance. Furthermore, it is not always possible to find corresponding points on manifolds. In such cases we need to stick with the former approach.

For our experiments we used a population size of 64, stagnation parameter of 10 generations, selection pressure 1.8, and crossover percentage 0.7. We found by experimentation, that a scaled sigmoid turned out to be as the most effective activation function. The rest of parameters were in default setting as provided in [46].

All following ELBO experiments were conducted on a machine with four Intel Xeon 6130 16-core 2.1 GHz and 768 GB RAM. An evaluation of a single offspring of ELBO ranged from a few seconds in the similar points based approach to a few

3. Neuroevolutionary LBO

minutes in the benchmark based approach. ELBO-Net Training was conducted on a RTX 2070 and Intel i9 9th gen Processor.

3.7.3 SHREC’14

The SHREC’14 Non-rigid 3D Human Models benchmark [51] is divided into two sets: synthetic and real. The synthetic set contains shapes generated using 3D modeling software, whereas the real set consists of actual 3D scans. Additionally, the authors provide an independent training set for each dataset, which we use to train ELBO and ELBO-Net as described.

Real

Method	NN	FT	ST	E	DCG
supDLtrain [58]	79.3	72.7	91.4	43.2	89.1
DeepGM[57]	72.5	53.6	82.7	41.2	78.2
3D-DL[56]	22.5	19.3	37.4	26.2	50.4
KLBO _(SV-SIHKS) [44]	11.25	11.63	22.42	15.09	38.3
ELBO-Net	70.75	46.86	63.56	33.70	73.11

Table 3.5: Results on SHREC’14 Real Dataset (in %)

ELBO Training We hand-labeled corresponding points on manifolds 20, 23, 25, 27, and 29 for similarity, and on manifolds 5, 14, and 46 for dissimilarity. Equation 3.4.0.7 was used as the fitness function. Figure 3.9 illustrates the evolution process of ELBO, and the resulting weighting scheme is shown in Figure 3.18.

ELBO-Net Training We make full use of the supplied training set of 100 manifolds, by creating similar and dissimilar pairs. We randomly select 2000 pairs of all similar and dissimilar pairs for our training.

Result The results are given in table 4.3. We obtain decent results and a significant performance gain when compared with KLBO [44]. However, we still underperform against DeepGM [57] and supDLtrain [58]. This could be a result of using a relatively small training set for training ELBO and the disconnected backend.

Synthetic

Method	NN	FT	ST	E	DCG
supDLtrain [58]	96.0	88.7	99.1	72.1	97.5
DeepGM [57]	99.3	81.4	98.3	72.3	96.7
3D-DL[56]	92.3	76.0	91.1	64.1	92.1
KLBO [44]	88.83	81.58	95.12	68.87	94.03
ELBO-Net	98.33	88.05	98.32	72.10	98.17

Table 3.6: Results on the SHREC’14 (synthetic) dataset (in %). ELBO-Net does not reach state-of-the-art performance, primarily due to the disconnected training between the ELBO and its backend network.

ELBO Training We hand-labeled corresponding points on manifold 0, 1, 2, 3, and 4 for similarity and 15, 24, and 36 for dissimilarity. Equation 3.4.0.7 was used as fitness function.

ELBO-Net Training The supplied training set consists of 45 manifolds with 9 different classes. We create similar and dissimilar pairs and select 1000 pairs randomly for training.

Result The results are shown in table 4.4. We achieve superior results on this benchmark competing only with supDLtrain [58], which partially utilizes the test set to tune its dictionary. Training the ELBO over multiple classes of manifolds could bring in some further performance gain.

3.7.4 SHREC’15

As ELBO has achieved almost near-perfect results in 3.2, we train ELBO-Net on a more challenging setting. The SHREC’15 Non-rigid shape retrieval benchmark [59] comprises 1200 shapes, which are divided in 50 classes.

ELBO Training For this benchmark, we use SHREC’10 [55] Dataset to train the ELBO and use benchmark metrics as in eq. 3.4.0.5 to drive the evolution.

3. Neuroevolutionary LBO

ELBO-Net Training As there is no disjoint training set available for SHREC’15, we take out 30% of the benchmark randomly as a test set and evaluate our approach on the remaining 70%.

Result One can observe from the results shown in table 4.5, that our method is on par with DeepGM [57]. The drop in e-measure is due to the training set split.

Method	NN	FT	ST	E	DCG
DeepGM [57]	99.7	94.0	97.2	80.1	98.8
KLBO _(FV-IWKS) [44]	99.92	98.59	99.33	82.69	99.59
ELBO-Net	99.41	94.51	96.78	64.52	98.45

Table 3.7: Results on the SHREC’15 dataset (in %). ELBO-Net underperforms compared to KLBO, suggesting that while the descriptor may be more discriminative, its backend does not achieve the same level of performance.

3.8 Conclusion

From the experiments, it is evident that an effective weighting function $M(\theta)$ can be learned, outperforming heuristic approaches such as [24]. When coupled with a more modern backend, this learned weighting yields competitive performance across benchmarks. A key strength of neuroevolution lies in its ability to jointly evolve both network topology and weights. Architectural features such as skip connections, disabled connections, and recurrent connections typically require extensive manual tuning in gradient-based neural networks, whereas they emerge naturally in a neuroevolutionary framework.

However, our experiments also reveal that the principal bottleneck in the setup of KLBO [24] lies in the backend, where features are aggregated and projected into the task space. In recent years, much of the progress in learning-based shape analysis has shifted toward increasingly powerful feature extractors, with the learning capacity concentrated in the backend. In the ELBO framework, the backend contains relatively few trainable components, which enables the effective learning of the weighting function. Once ELBO is combined with a modern backend that itself

requires training, a fundamental limitation emerges: optimal performance can only be achieved when the frontend and backend are trained jointly.

While offline learning of the frontend followed by separate backend training can be effective, as demonstrated in our experiments, this decoupled setup leads to an inherently suboptimal and cumbersome architecture. The full representational capacity of neither component is fully realized, since the frontend cannot adapt to the precise needs of the backend loss. This observation provides strong motivation for an end-to-end approach, in which the system matrices are learned directly under the supervision of the downstream task. In such a setting, features can be shaped explicitly to complement the backend, rather than optimizing for generic notions such as maximal discriminativity. A backend may already handle certain discriminative cues well, whereas the frontend could instead focus on resolving the ambiguities that remain challenging for the backend.

Transitioning to a supervised, end-to-end formulation introduces its own challenges, most notably scalability. ELBO scales naturally to meshes of arbitrary resolution, since learning is performed exclusively in inference mode via a fitness evaluation, thereby avoiding costly backpropagation through the mesh and the LBO eigensystem. In contrast, a backpropagated approach significantly increases computational cost but enables more expressive designs, including a larger number of parameters and more complex operators. Extending the learning beyond mass matrix weighting to include the stiffness matrix is substantially more involved, as the cotangent Laplacian must remain valid and respect geometric constraints such as the triangle inequality.

Despite these tradeoffs, moving from a neuroevolutionary framework to a gradient-based end-to-end approach is a justified step. It enables the incorporation of neighborhood learning methods such as [6], advanced pooling mechanisms, and additional geometric operators, including modifications of the Riemannian metric or anisotropic formulations. Ultimately, the ELBO framework fulfills its intended purpose: it demonstrates that the LBO possesses significant learning potential, and that a learned weighting can consistently improve upon standard, fixed baselines.

3. Neuroevolutionary LBO

Furthermore, it demonstrates that the baseline LBO can be outperformed when paired with appropriate features and a relatively lean neural network, which in turn informs the design and complexity of the following supervised approach. Since the performance gains are greater when using the SI-HKS compared to the WKS, we conclude that for global tasks such as retrieval and classification, learning the LBO eigenbasis is more advantageous. Note that we will move on to adapting the HKS, even though experiments were carried out with SI-HKS, for some tasks and benchmarks scale matter and we want to exploit the scales of different objects within a set. Since the connection between SI-HKS and HKS was detailed in the background chapter, we expect that the observed effects are similarly valid.

"Die Geometrie wird nur durch die Art und Weise, wie wir den Raum betrachten, bestimmt." [60]

Bernhard Riemann

4

LBONet: Supervised Spectral Descriptors for Shape Analysis

4.1 Introduction

These profound words by Bernhard Riemann (that geometry is only determined by the way we look at space) emphasize that geometry is defined by the way we observe space, a principle closely tied to the Riemannian metric. In this work, we introduce a series of operators, including one that scales the Riemannian metric, which further exemplifies Riemann's insight. Here, the principle is taken even further: through learning, the way we see geometry becomes more flexible, allowing the representation of space itself to adapt and evolve through neural network learning.

The previous section provided valuable insights. Despite the limitations of the neuroevolutionary approach to optimizing the LBO eigenbasis, the results already indicate untapped potential within this direction. Nature-inspired search and optimization methods offer an efficient way to break through conceptual barriers and to assess whether a research direction is worth pursuing before investing substantial time in developing intricate optimization algorithms. Although black-box optimization can be effective for certain problems, it remains computationally demanding, as the fitness function is both expensive to evaluate and non-trivial to

design. Such methods often require access to high-performance computing clusters, and adjusting architectural parameters such as depth, width, or activation functions is far less straightforward than in a fully supervised setup. While the optimization of the LBO eigenbasis has been explored in recent years, several challenges remain unresolved, which this work seeks to address. We refer to this method as LBONet (short for LBO–Network), which conceptualizes the Laplace-Beltrami Operator as a trainable module within a neural network. This framework enables the learning of a spectral descriptor that is optimized toward a specific objective defined by the backend. It is important to emphasize that the backend, regardless of its architecture, actively shapes the spectral descriptor. Consequently, LBONet implicitly learns to compensate for the weaknesses of the backend, tailoring its representations to improve the overall system performance.

When LBONet is trained end-to-end and later decoupled from the backend, the resulting descriptor performs best when reattached to the same backend it was originally trained with. Its performance may decrease when transferred to another backend with pre-trained weights, since the learned representation is closely coupled to the optimization dynamics of the specific network. This observation highlights an important conceptual insight: LBONet is not primarily designed to maximize discriminativeness or descriptiveness in a standalone sense. Instead, it learns to produce the spectral features that are most useful for the backends task. For instance, in a segmentation scenario, this behavior manifests as segments becoming more distinct in the spectral domain, which helps the backend to separate them more easily. That being said, LBONet can also be used directly to produce a spectral descriptor without the need for a full backend. However, a minimal MLP is required to project the prediction space onto the spectral descriptor. Without this projection, LBONet would attempt to learn both the prediction space and the feature space simultaneously, which leads to suboptimal learning. Conversely, if the backend is made too deep, the gradients may vanish, or the backend may absorb most of the learning process, leaving LBONet responsible only for learning trivial features. This, in turn, diminishes the intended objective of the framework.

4. LBONet: Supervised Spectral Descriptors for Shape Analysis

LBONet introduces a sophisticated mechanism that effectively learns to shape the eigenbasis through operations defined over mesh vertices, edges, and faces. An illustration of the implemented operators is provided in Figure 4.1.

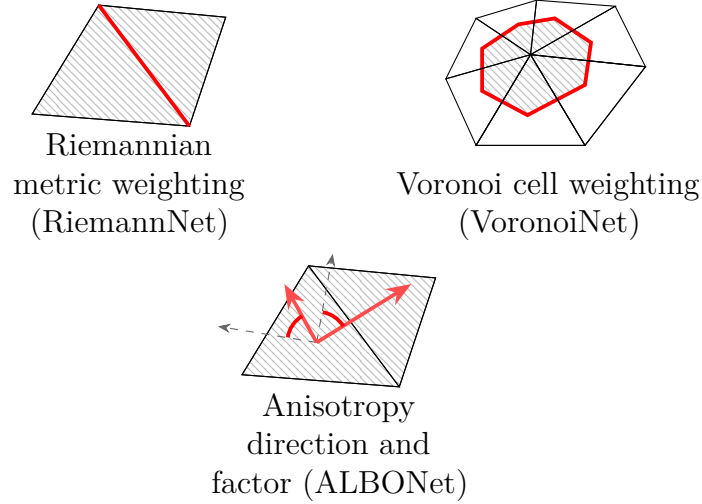


Figure 4.1: LBONet comprises multiple modules introduced in this thesis, which operate directly on the input mesh. RiemannNet weights the Riemannian metric on a manifold, ALBONet introduces anisotropy by rotating and scaling the diffusion speed, and VoronoiNet weights the Voronoi cells. These modules affect the Laplace-Beltrami operator in several different ways, allowing it to adapt to the given data and alleviate performance breakdowns incurred by non-isometric deformations.

At this stage, it is worth referring the reader to HodgeNet [25], which learns Hodge star operators by approximating their gradients. Although HodgeNet presents a general framework for operator learning, its performance remains modest across most benchmarks. Future research could explore potential synergies between LBONet and HodgeNet, particularly in extending the learning process from the metric itself to operators acting across different degrees of forms.

An open question that remains is whether the frontier established by LBONet represents a fundamental limit, or if further potential lies beyond the framework proposed in this work. The findings discussed here, along with several additional insights, will be revisited and summarized at the end of this chapter.

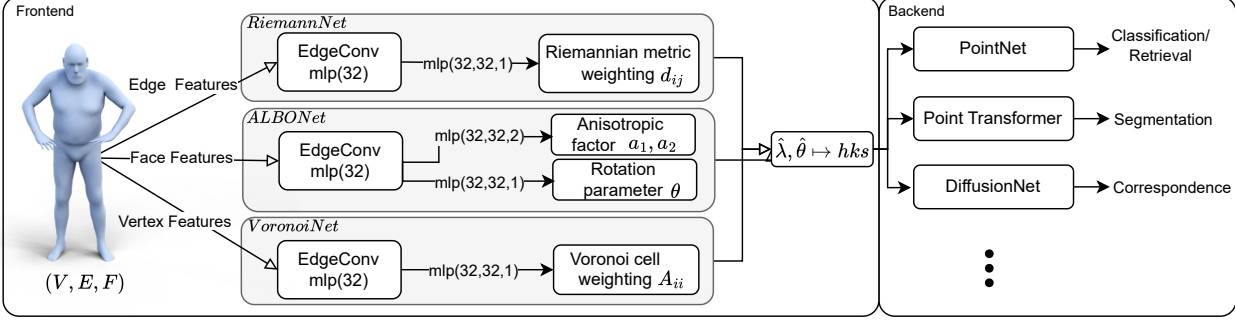


Figure 4.2: LBONet Architecture. LBONet takes features on mesh edges, vertices, and faces as input of its three building blocks, RiemannNet, ALBONet and VoronoiNet. Each block consists of one EdgeConv which is creating a local neighbourhood graph as in [6] and learns an neighbourhood function. Then the features are passed through multiple shared MLPs to predict either the Riemannian metric on an edge, the anisotropy on a face or the Voronoi cell weighting at a vertex. These predictions are applied directly on the discretized LBO, modifying the stiffness and mass matrix, which we detail in Section 4.5 onwards. Its eigenbasis is then used to form a spectral descriptor, which then can be passed into a "backend" for further processing.

4.2 System Overview

This section outlines the high-level architecture of LBONet, establishing a complete reference for its underlying goals, data pathways, end-to-end training framework, and real-time inference execution before detailing the separate discrete operator blocks.

4.2.1 Overall Aim of LBONet

Unlike the decoupled heuristic weighting or evolutionary methods introduced previously, the core aim of LBONet is to implement the Laplace-Beltrami Operator as a fully differentiable, trainable module within an end-to-end supervised deep learning network. By backpropagating task-specific losses directly through the generalized eigenvalue problem, LBONet implicitly adapts the underlying Riemannian metric, anisotropy directions, and Voronoi cells to mitigate performance breakdowns caused by non-isometric shape deformations. The final goal is to output a spectral signature optimized to maximize the performance of a task-specific backend.

4.2.2 Inputs and Outputs

- **Inputs:** The pipeline accepts a raw 3D mesh. From its geometry, we extract robust, rotation-invariant local shape features derived from the first and second

4. LBONet: Supervised Spectral Descriptors for Shape Analysis

fundamental forms (specifically: Gaussian curvature, mean curvature, shape index, and curvature index) on vertices, which are subsequently interpolated across edges and faces.

- **Outputs:** The intermediate output of the framework is a modified, task-specific LBO eigenbasis $(\hat{\lambda}, \hat{\phi})$ derived from solving the adjusted system $W\hat{\phi} = \hat{\lambda}A\hat{\phi}$. The final output of the network is a highly descriptive spectral descriptor tailored to the geometry of the target application.

4.2.3 Training Phase (Supervised End-to-End)

LBONet is trained in an end-to-end supervised fashion using standard gradient-based backpropagation:

- We employ Nelson’s method and its variants to calculate analytic gradients for the eigenvalues and eigenvectors with respect to continuous geometric parameters.
- The task loss (e.g., cross-entropy for segmentation or contrastive loss for shape matching) is propagated backwards from the backend network, directly through the spectral descriptor formulation, through the eigensolver layers, and down into the separate sub-networks.
- This enables simultaneous, joint optimization of the edge-wise metric parameters (RiemannNet), face-wise anisotropy (ALBONet), and vertex-wise Voronoi cells (VoronoiNet).

4.2.4 Evaluation and Inference Pipeline

At inference time, the calculation of the analytical gradients is deactivated, and the network behaves as a clean, highly efficient feed-forward pipeline:

1. Unseen 3D shapes pass through the edge, face, and vertex networks to evaluate the metric adjustments and anisotropy factors.

2. The discretized stiffness matrix W and mass matrix A are calculated in a single pass utilizing the learned operators.
3. The generalized eigenvalue problem is solved directly to extract the adapted eigenbasis $(\hat{\lambda}, \hat{\phi})$.
4. The optimized spectral signature is generated and pushed through a projection MLP or a trained downstream backend to yield instant segmentations, correspondences, or classification predictions.

4.3 Approach

Before delving deeper into the core concepts, we first examine the inherent challenges associated with the LBO eigenbasis in detail. Following this analysis, we introduce a set of operators designed to address these issues. This step-by-step progression enables the reader to follow the underlying reasoning and gain a deeper understanding of the proposed solutions.

4.3.1 Building blocks

As mentioned in the introduction, any non-isometric deformation distorts the LBO eigenbasis drastically. This can be illustrated by using functional maps [22], where the C matrix (Eq. 2.1.7.2), which maps the eigenbases on one manifold to another, deviates from the diagonal matrix with even mild near-isometries. The greater the non-isometric deformation, the further the map deviates from the diagonal, making the map more complex and less sparse and due to this fact the LBO eigenbasis become less ideal as a basis function. This effect is illustrated in figure 4.3, with an intra-class manifold for near-isometry and an inter-class manifold for non-isometry. This sensitivity and instability of the LBO eigenbasis introduces two drastic downsides, firstly, it makes it difficult to transfer features learnt in the spectral space and, secondly, the overlying descriptor needs to span the spectral range a manifold can be embedded in, which could lead to a less precise descriptor.

4. LBONet: Supervised Spectral Descriptors for Shape Analysis

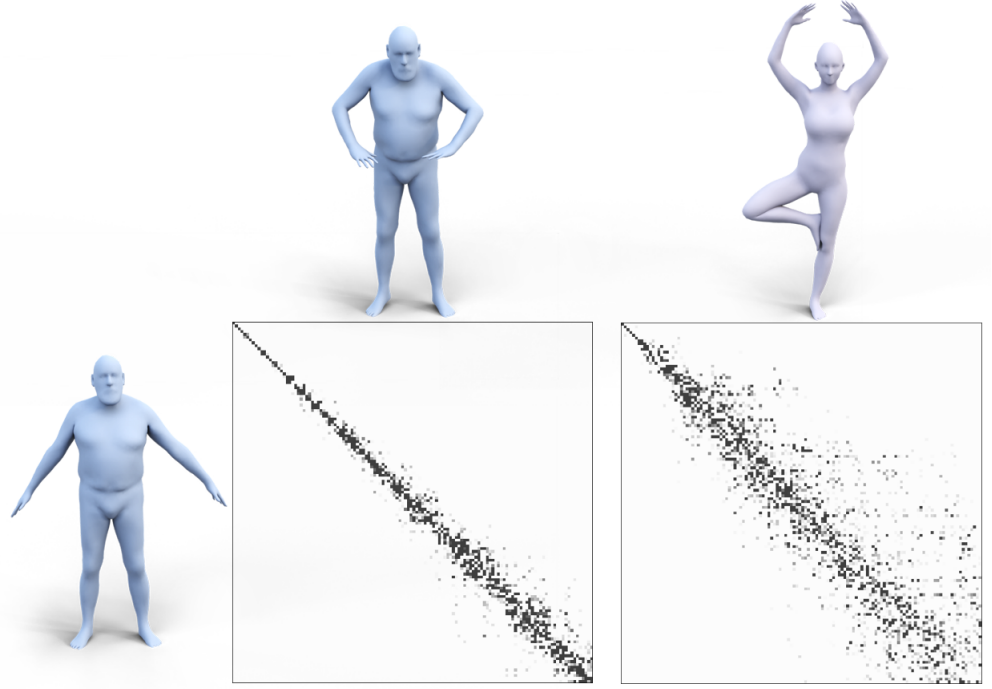


Figure 4.3: Functional matrix between an intra-class manifold pair (left blue and top blue) and a inter-class manifold pair (left blue and top purple). One can observe two things, first, the functional matrix deviates from the diagonal even with mild non-isometries and second, that with greater deformation, the matrix deviates even further.

To alleviate this, learning can be done both implicitly and explicitly. Implicit learning would learn an optimal descriptor based on the LBO eigenbasis and learning thereby only implicitly an ideal LBO eigenbasis, e.g. for a segmentation task, the learning can be done on the heat kernel directly, which then propagates the loss back to the eigenbasis. Whereas explicit learning would learn an LBO eigenbasis directly from the loss function and the loss would be given directly on the LBO eigenbasis, e.g. in a correspondence task, we might learn the eigenbasis directly, rather than going via an overlying descriptor.

The approaches and experiments presented in this chapter focus on implicitly learning the LBO eigenbasis. Explicit eigenbasis learning lies beyond the scope of this chapter and will be examined in detail in Chapter 5. A note to the impatient reader: while it is technically possible to diagonalize the eigenbases of two manifolds, doing so introduces a significant disconnect between the point-to-point map and the functional map, thereby increasing the geodesic error. Consequently, the off-

diagonal elements are both necessary and meaningful. Nonetheless, they serve as a useful illustration of the sensitivity of the LBO eigenbasis and also proof that the proposed technique is powerful enough to achieve this.

We give the gradients below in the context of learning through the HKS, but they can be analogously derived for any other spectral signature such as the WKS [15] or average mixing kernel signature [61]. With the given loss LBONet aims to weight the LBO s.t. the produced eigenbasis and the descriptor is better at the given task. Weighting the LBO is a common practice and as described in the related work section (Sec. 2.2) used quite effectively in various tasks. However, as the backpropagation through the LBO eigenbasis is a challenging task, hyperparameters could not be learned easily and were therefore based on heuristics. We show the weighting based on three different elements, which are depicted in gray in figure 4.2.

Network Sub-Modules

Firstly, by using features on mesh edges to weight the Riemannian metric g_{ij} (RiemannNet), which by default is defined as the edge length of the mesh. Secondly, on faces, by adjusting the anisotropy factors (a_1, a_2) and rotation θ (ALBONet), and lastly using features on vertices to weight the Voronoi cell (VoronoiNet).

We equip each block with EdgeConv which creates a local neighbourhood graph as in [6] and learns a neighbourhood function. This allows the sharing of information in the local neighborhood and to learn more effectively generalizing features. This proved far superior to learning directly from the features only sourcing their information from the local context. As a distance function, Euclidean distance showed good performance, although in some cases constructs neighbourhood erroneously, e.g. when a disjoint body part such as a hand is close to the hip, the hip might exchange information with the hand. We observed some downside effect in the experiments, however, the drawback was negligible, hence we omitted the construction of a geodesic neighborhood.

These features are then passed through multiple shared MLPs to predict the Riemannian metric weighting, which can shorten or expand distances on specific

4. *LBONet: Supervised Spectral Descriptors for Shape Analysis*

regions by scaling the Riemannian metric, steer the anisotropy individually on each face by rotating the orthogonal principal curvature basis and scaling the anisotropic factors, and finally weight the Voronoi cell differently across the manifold. These predictions are applied directly on the discretized LBO, modifying the stiffness W and mass matrix A in various ways, which we detail in Section 4.5 onwards. This modified LBO eigenbasis $(\hat{\lambda}, \hat{\phi})$ is then used to evaluate a spectral descriptor, which then can be either passed into a "backend" (see figure 4.2) for further processing or used directly for various tasks.

4.4 Deep Geometric Processing Layout

Works on learning from unordered point clouds such as in PointNet [5], PointNet++ [8], and DGCNN [6] or spectral methods such as Spectral Transform Block [52] proved useful for shape analysis tasks. More recently point (cloud) transformers [33] [32] and the heat kernel based transformer [34] have further pushed the boundaries in extracting powerful features.

In particular, their performance in non-rigid shape analysis using spectral descriptors rather than point coordinates proves that they are able to utilize the information within spectral descriptors, and furthermore, that there is still unexploited potential in them, as the standard LBO is not invariant under non-isometric deformation. The presented work explores to what extent spectral descriptors can contribute to the descriptiveness of features when using state-of-the-art feature extracting networks, by putting as much learning as possible into the spectral descriptor and the LBO eigenbasis. We highlight the tradeoffs of this level of adaption of the LBO to the given task in the conclusion.

Edge-, Face-, Vertexwise Perceptrons

With the propagated loss through the LBO eigenbasis, we can learn features from mesh edges by RiemannNet, from mesh faces by ALBONet and from mesh vertices by VoronoiNet. For this we use the concept of shared MLPs, which has been used widely in the literature and been referred to as *-wise perceptrons.

4.5. Intrinsic Feature Engineering and Invariance

Furthermore, experiments have shown that EdgeConv introduced by [6] yielded an additional performance gain. For edges and faces, we take the mean of their coordinates to build a coordinate system to find the nearest neighbours and use the asymmetric edge function given by [6],

$$h_{\Theta}(x_i, x_j) = h_{\Theta}(x_i, x_j - x_i). \quad (4.4.0.1)$$

followed by several shared MLP layers.

Finding the right neighbourhood size is a hyperparameter, which needs to be balanced. Its effects can be observed in Figures 4.15, 4.10, 4.11, 4.13, and 4.7, which demonstrate that low neighborhood values isolate high-frequency local variations, while high neighborhood values act as a spatial low-pass filter, smoothing activations across broad, continuous regions of the shape. While calculating these neighborhoods via geodesic distance would offer superior topological accuracy, a Euclidean distance metric is utilized here to avoid prohibitive computational overhead.

4.5 Intrinsic Feature Engineering and Invariance

As input to our pipeline we use extrinsic and intrinsic shape features derived from the second fundamental form such as mean curvature [49], Gaussian curvature, shape index, curvature index [47]. For faces and edges we interpolate the features from their vertices. The intention is to limit the input features to (semi-)intrinsically invariant features to avoid any dependency on the shape orientation and also have robust features, which are mostly independent from the meshing structure and resolution.

Usage of point coordinates as input features could yield an additional benefit, but it raises the risk of shortcut learning instead of real intrinsic learning. Furthermore, intrinsic shape features prove to be significant for non-rigid shape analysis as shown in [62] and are widely used as enhancing input features [8] [6] [7] for non-rigid shape analysis.

By focusing on intrinsic quantities, noting that mean curvature, while not intrinsic, is still invariant under rigid transformations, LBONet is able to learn genuinely intrinsic features of the shape rather than relying on raw coordinates

4. LBONet: Supervised Spectral Descriptors for Shape Analysis

and exploiting shortcuts for prediction. This is particularly relevant for oriented benchmarks commonly used in the literature, such as COSEG, ShapeNet, human segmentation datasets, and FAUST, where raw coordinates already explain much of the data variance. The limitation of coordinate-based learning becomes evident when the training or testing sets contain flipped examples, as illustrated in the rocket example in Figure 1.3. These observations further emphasize the importance of pursuing intrinsic learning, which constitutes a key advantage of LBONet. If all benchmarks were composed of manifolds without a consistent orientation, LBONets performance advantage over extrinsically learning methods would likely be even greater. This shows that intrinsic shape understanding is rather possible when the learning itself remains close to intrinsic measures or extrinsic measures, which do not affect rotation or embedding invariancy.

The predictions of the the modules (RiemannNet, ALBONet, and VoronoiNet) modify the stiffness and mass matrix in different ways, which then can be used to solve the generalized eigenvalue problem,

$$W\hat{\phi} = \hat{\lambda}A\hat{\phi} \quad (4.5.0.1)$$

resulting in a modified task-specific eigenbasis $(\hat{\lambda}, \hat{\phi})$. ALBONet learns W , VoronoiNet learns A , and RiemannNet learns the metric tensor g_{ij} , which affects both $(A$ and $W)$. With these components at hand, the method can learn both geometry and diffusion on the manifold simultaneously. However, there exists a dependency between ALBONet and RiemannNet, since rotating and scaling the anisotropy impacts the metric g_{ij} . Therefore, a new set of gradients must be derived to account for this effect. Another problem arises in this context. For RiemannNet, the embedding of the manifold is irrelevant, since only the metric tensor g_{ij} is scaled. In contrast, ALBONet modifies vector directions and therefore depends on the explicit vertex positions as illustrated in section 2.1.5. However, scaling a triangle is not uniquely defined with respect to its vertex positions. Specifically, if the initial vertex coordinates are used and each edge is scaled sequentially, the resulting vertex positions will vary depending on the order in which the edges are

weighted. Multiple solutions exist for this problem. One possible approach is to average the vertex positions after sequentially scaling each edge, thereby estimating the ground-truth weighted triangle. A more computationally expensive alternative could be to employ the ICP algorithm to position the triangle s.t. it equals to what the gradients expect, however this introduces prohibitively high computational cost for inference. We experimented with the averaging approach; and the gradients were reliable enough to learn robustly. This suggests that, as long as the RiemannNet weights remain within reasonable limits, the computed gradients are sufficiently accurate. We discuss this later in the discussion section, how the gradient approach can be implemented to further stabilize training and increase precision of LBONet.

4.5.1 Sensitivity Analysis of Spectral Descriptors

In chapter 2 subsection 2.1.6 we had introduced Nelsons method [19] and all of its variations, where stiffness, mass and both are used to derive the eigenbasis gradients. With the eigenbasis gradients at hand, we can differentiate any given spectral descriptor, i.e. the HKS,

$$h_t(x, x) = \sum_{k>0} e^{-t\lambda_k} \phi_k(x)^2. \quad (4.5.1.1)$$

wrt. any parameter p on a manifold M ,

$$\frac{\partial h_t}{\partial p_{ij}} = \sum -te^{-\lambda_k t} \phi^2 \frac{\partial \lambda_k}{\partial p_{ij}} + 2e^{-\lambda_k t} \phi \frac{\partial \phi_k}{\partial p_{ij}} \quad (4.5.1.2)$$

which allows us to do an end-to-end analysis, back propagating the loss from the spectral descriptor down to the individual parameter. As the HKS (Eq. 4.5.1.1) is defined on the vertices of the mesh, it is important to emphasize that the following sections will explore how learning on vertices, edges, and faces affects the resulting spectral descriptor. In particular, we analyze the sensitivity of operators acting on these domains and their influence on the vertex-based HKS. This sensitivity analysis produces a gradient in the domain where the operator is defined. For example, when the operator acts on edges, the resulting gradient describes how variations along edges impact the HKS. In this way, the framework captures the

4. *LBONet: Supervised Spectral Descriptors for Shape Analysis*

interaction between different degrees of form, linking vertex-, edge-, and face-based representations. Furthermore, all gradient components are illustrated in diagrams to help the reader understand the gradients introduced. Although the following analysis focuses on a single edge flap, a single face, and a single Voronoi cell for clarity, the implementation itself is fully vectorized. All gradients are computed simultaneously across the entire mesh and for all frequencies of the LBO eigenbasis.

4.5.2 Mass and Stiffness matrix

Before moving on to learning in different discretization forms (vertex-, face-, or edge-based), it is important to recall how the stiffness and mass matrices are constructed. These matrices were introduced in Eqs. 2.1.1.7 and 2.1.1.8. Since they are derived from angles and distances on the manifold, any modification of the underlying metric induces a change in one or both matrices.

Such modifications can be applied at different levels. For instance, one may perform post hoc manipulations, such as reweighting the Voronoi cells, which directly affects the mass matrix. Alternatively, more fundamental changes can be introduced, such as incorporating anisotropy, which alters the stiffness matrix. In all cases, the intended modification and its implications for the stiffness and mass matrices must be explicitly formulated, as these matrices define the generalized eigenvalue problem (GEP) used to compute the LBO eigenbasis.

Consequently, a variety of operators acting on these different forms can be defined. In the following, we introduce several such operators and discuss their effects.

4.5.3 Learning on mesh edges

Having established the groundwork, we move on setting up the parameters we want to learn on a manifold and first approach learning the Riemannian metric g , which by default is defined on the mesh edge as its length d_{ij} . For this we inherit the gradients for the stiffness matrix W from [21], but introduce the gradients for the mass matrix A using Voronoi cells in the following. The following gradients are all based on the equations introduced in Chapter 2 subsection 2.1.3. Therefore,

we will introduce the gradients without much derivation as their derivation is a result of applying basic calculus.

Adjusting any given Riemannian metric inevitably influences the surrounding edge flap, causing changes in the associated angles and distances. Consequently, the gradients for *RiemannNet* are derived from the components illustrated in Figure 4.4. An attentive reader may note that scaling the edge ij would, in principle, also affect the neighbouring triangles sharing edges ik_1 , ik_2 , and so forth, since these are geometrically coupled through the mesh. However, our focus lies on the *intrinsic* metric of the manifold rather than its extrinsic representation. Therefore, we treat each edge as an independent degree of freedom and compute its contribution to the local LBO directly, without propagating changes to adjacent mesh elements. This represents a deliberate design choice: while it would be possible to propagate each metric modification to neighbouring edges to enforce smoother updates, this would reduce the overall expressiveness of the framework.

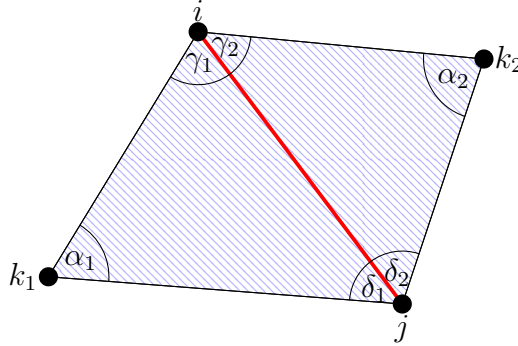


Figure 4.4: Edge flap surrounding edge ij . Adjusting the Riemannian metric, which is highlighted in red, has an impact on all the angles and distances shown and are hence part of the gradients introduced.

4. LBONet: Supervised Spectral Descriptors for Shape Analysis

We introduce the gradients for the mass matrix A for the module RiemannNet:

$$\begin{aligned} \frac{\partial A_{ii}}{\partial d_{ij}} = \frac{1}{8} & \left(\left(-\frac{d_{ij}^3}{\sin^2(\alpha_1)2T_1} + \cot(\alpha_1)2d_{ij} \right) + \frac{d_{jk}^3 \cos(\gamma_1)}{\sin^2(\delta_1)2T_1} \right. \\ & \left. - \left(\frac{d_{ij}^3}{\sin^2(\alpha_2)2T_2} + \cot(\alpha_2)2d_{ij} \right) + \frac{d_{jk}^3 \cos(\gamma_2)}{\sin^2(\delta_2)2T_2} \right) \end{aligned} \quad (4.5.3.1)$$

$$\begin{aligned} \frac{\partial A_{jj}}{\partial d_{ij}} = \frac{1}{8} & \left(\left(-\frac{d_{ij}^3}{\sin^2(\alpha_1)2T_1} + \cot(\alpha_1)2d_{ij} \right) + \frac{d_{jk}^3 \cos(\gamma_1)}{\sin^2(\delta_1)2T_1} \right. \\ & \left. - \left(\frac{d_{ij}^3}{\sin^2(\alpha_2)2T_2} + \cot(\alpha_2)2d_{ij} \right) + \frac{d_{jk}^3 \cos(\gamma_2)}{\sin^2(\delta_2)2T_2} \right) \end{aligned} \quad (4.5.3.2)$$

$$\frac{\partial A_{k_1 k_1}}{\partial d_{ij}} = \frac{1}{8} \left(-\frac{d_{ij}^3 \cos(\gamma_1)}{\sin^2(\delta_1)2T_1} + \frac{d_{ij}^3 \cos(\delta_1)}{\sin^2(\gamma_1)2T_1} \right) \quad (4.5.3.3)$$

$$\frac{\partial A_{k_2 k_2}}{\partial d_{ij}} = \frac{1}{8} \left(-\frac{d_{ij}^3 \cos(\gamma_2)}{\sin^2(\delta_2)2T_2} + \frac{d_{ij}^3 \cos(\delta_2)}{\sin^2(\gamma_2)2T_2} \right) \quad (4.5.3.4)$$

where T is the area of the triangle. As triangles have to obey the triangular inequality to be valid, specific weightings can render a triangle invalid, resulting in a failure to solve for the eigenbasis. To mitigate this, [21] proposed to use Rosen's gradient projection method [63], which projects the gradient into the search direction by taking the triangular inequality into account. As our proposed method takes a different strategy to weight the Riemannian metric, we treat it as a metric nearness problem and solve it in the forward-pass as proposed in [64]. This iterative algorithm first weights the triangle with the given weights, which might result in an invalid triangle. Then for every rotation (ijk, jki, kij) of the triangle, the violation is calculated with $d_{ik} + d_{jk} - d_{ij}$ which then is used to derive correction terms with $\frac{1}{3}(e_{ij} - e_{ik} - e_{jk})$. This corrective term is then subtracted from the violating edge term e_{ij} and is added onto the remaining edge terms (e_{ik}, e_{jk}) . This routine is repeated until no triangular inequality is violated. As collinear vertices in a triangle or triangles very close to having collinear vertices produce exploding gradients (see Figure 4.5),

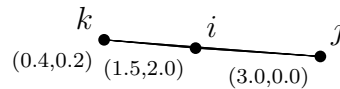


Figure 4.5: A degenerate triangle $\triangle(i, j, k)$.

4.5. Intrinsic Feature Engineering and Invariance

we add a small term ϵ to the corrective term. This results in a much smoother gradient, as one can observe in Figure 4.6 and prevents invalid eigenbases. An interesting extension could be to consider the collapse of vertices in such cases.

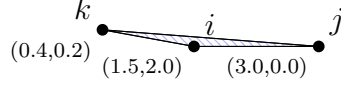


Figure 4.6: A degenerate triangle $\triangle(i, j, k)$.

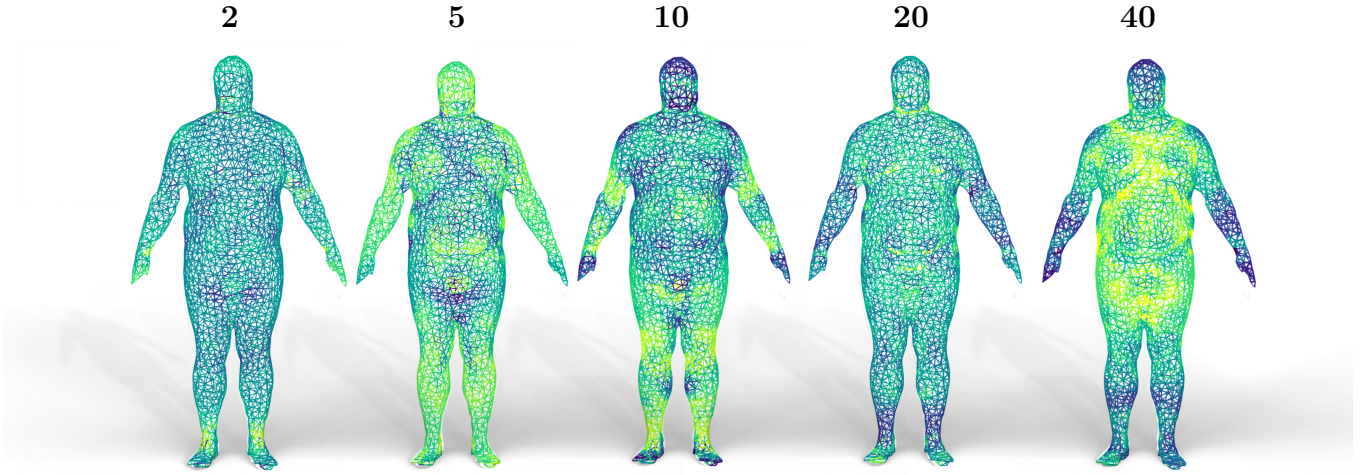


Figure 4.7: Effect of varying the neighborhood size in RiemannNet. The numbers above indicate the specific neighborhood size used for each visualization. Yellow denotes a high degree of stretching, whereas blue indicates the shortening of the corresponding edge.

4.5.4 Learning on mesh faces

ALBO was introduced by [65] and adopted in the wider literature in various neural network applications [29][23][66] proving the potential beyond the standard LBO. However, the increased power comes at the cost of precalculating ALBO for every combination of anisotropic factor α and angle θ . Furthermore, as both parameters are discrete, important discriminativity might lie between two discrete values and might go unused. To avoid confusion with the notation used in the following, we label the anisotropic factors as $a_x \equiv \alpha_x$.

As the presented work allows the propagation of a loss function through the LBO eigenbasis, we take a step further and instead of just using ALBO with its standard parameters, we learn them from given data by face-wise perceptrons. By

4. LBONet: Supervised Spectral Descriptors for Shape Analysis

this we can learn a powerful ALBO which is highly adaptive over the manifold, which would otherwise be very hard to obtain with traditional methods.

We adopt the approach from [65] with absorbing the anisotropic factor a into the principal curvature directions and the extension to the rotation parameter θ introduced later in [23]. Additionally, we extend ALBO to learn into both directions at the same time, which we term as ALBO+. Our ablation study shows that there is a significant benefit in allowing the neural network to learn in both directions.

A more general form of equation 2.1.1.4 can be expressed as,

$$\Delta_D f = \nabla D(\nabla f) \quad (4.5.4.1)$$

where D is referred to as *thermal conductivity* tensor and is defined by a 2×2 matrix and controls the direction and anisotropic level of the diffusion.

Andreux et al. [65] proposed to use a tensor based on the principal curvature directions (κ_m, κ_M) forming an orthonormal basis $V_{ijk} = (v_m, v_M)$ on mesh faces,

$$D_a = \begin{pmatrix} \psi_a^m(\kappa_m, \kappa_M) & 0 \\ 0 & \psi_a^M(\kappa_m, \kappa_M) \end{pmatrix}, \quad (4.5.4.2)$$

where a controls the degree of anisotropy as illustrated in red in figure 4.8.

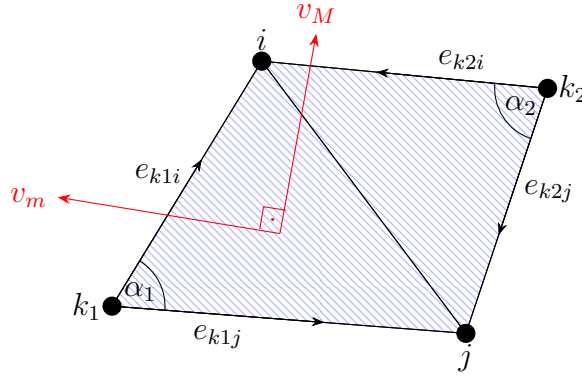


Figure 4.8: Orthonormal reference frame on a face depicted by v_m and v_M based on the principal curvature directions, which is highlighted red. Scaling the directions v_m and v_M by a can change the anisotropy on the given face, which results in scaling dot product, between the corresponding edge and the direction.

Later [23] proposed to include the rotation parameter θ by rotating D_a ,

$$D_{a\theta} = R_\theta D_a(x) R_\theta^T, \quad (4.5.4.3)$$

where R_θ denotes the rotation matrix as illustrated red in figure 4.9.

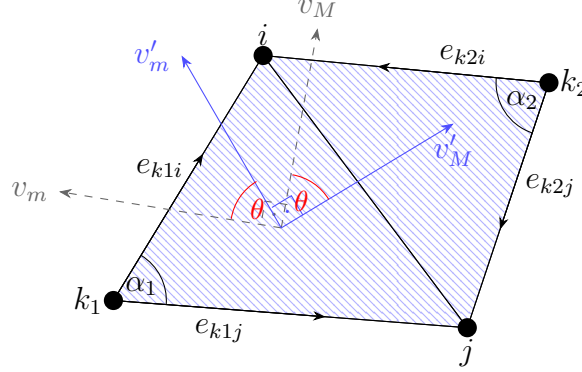


Figure 4.9: Orthonormal reference frame on a face depicted by v_m and v_M based on the principal curvature directions, which is highlighted blue. It is rotated by θ , which is depicted red. Besides, scaling the directions v_m and v_M rotating the directions has a direct impact on the used dot products.

While the authors [23] fix the anisotropic factor in one direction and varying the other direction, we apply ALBO+ in all our experiments as it is more powerful then the standard ALBO.

This results in a change in the stiffness matrix W , which now is defined as:

$$W_D(i, j) = \begin{cases} \frac{\langle e_{k1j}, e_{k1i} \rangle_H}{\sin \alpha_1} + \frac{\langle e_{k2j}, e_{k2i} \rangle_H}{\sin \alpha_2}, & \text{if } (i, j) \in E \\ -\sum_{k \neq i} W(i, k), & \text{if } i = j \\ 0, & \text{otherwise,} \end{cases} \quad (4.5.4.4)$$

where

$$\langle e_{kj}, e_{ki} \rangle_H = e_{kj}^T \underbrace{V_{ijk} D_{\alpha\theta} V_{ijk}^T}_H e_{ki} \quad (4.5.4.5)$$

and e being the respective edge on the triangle depicted in figure 4.9.

As the changes of equation 4.5.4.4 impact only the stiffness matrix W , we adjust Nelson's method given in equation 2.1.6.5 by omitting the gradients for the mass matrix A ,

$$\frac{\partial \lambda_k}{\partial p_{ij}} = \phi_k^T (-\lambda_k \Delta W) \phi_k \quad (4.5.4.6)$$

$$F_i = \phi^T \left(\frac{\partial W_k}{\partial p_{ij}} \right) \phi A \phi - \left(\frac{\partial W_k}{\partial p_{ij}} \right) \phi \quad (4.5.4.7)$$

4. LBONet: Supervised Spectral Descriptors for Shape Analysis

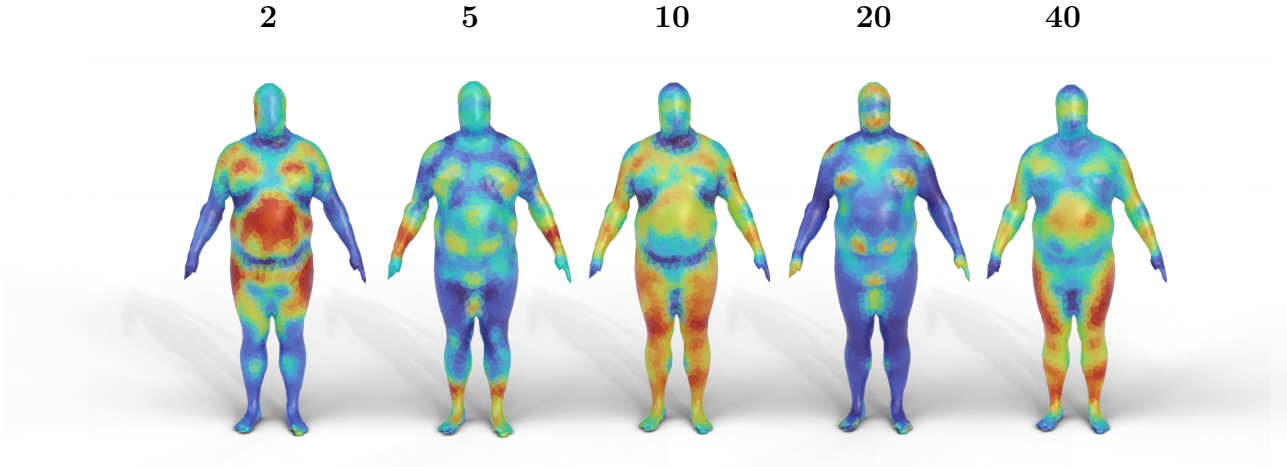


Figure 4.10: Effect of varying neighborhood size in ALBONet Anisotropy factor 1 (part of LBONet). The numbers above indicate the neighborhood size used in each visualization. Red denotes an acceleration of anisotropy, whereas blue indicates a deceleration.

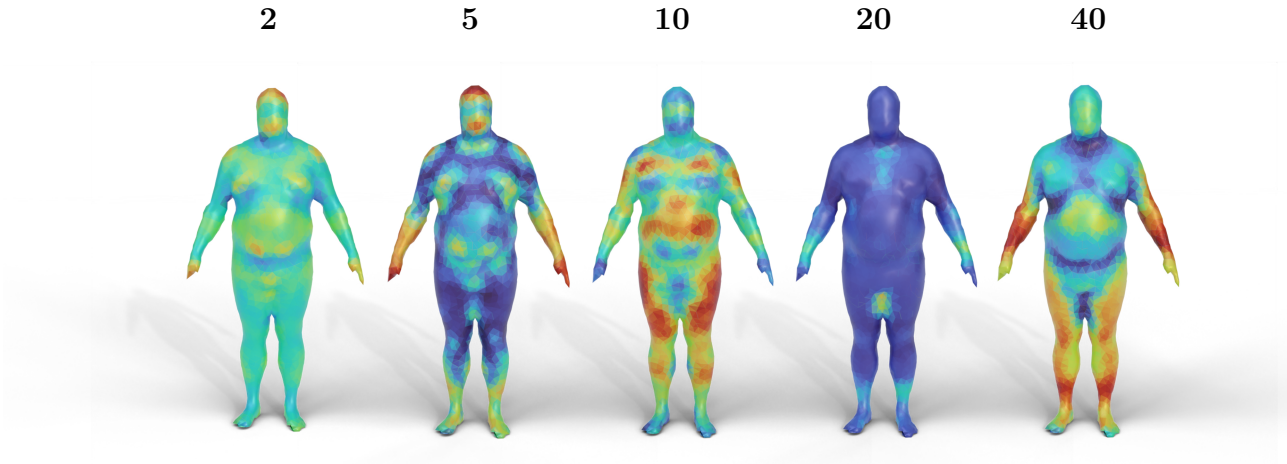


Figure 4.11: Effect of varying neighborhood size in ALBONet Anisotropy factor 2 (part of LBONet). The numbers above indicate the neighborhood size used in each visualization. Red denotes an acceleration of anisotropy, whereas blue indicates a deceleration.

Gradient sign of dot products

The gradient of the dot product in ALBO, i.e. edge e_{k1} and v_m changes its sign, depending on which side e_{k1} is wrt. v_m . To calculate the sign, we use the product of the cross products:

$$\text{sign}(x) = (e_{k1i} \times e_{k1j}) * (e_{k1j} \times v_m). \quad (4.5.4.8)$$

This sign is then applied to all dot products in the ALBO to correct the sign.

Anisotropic factors a_1, a_2

The anisotropic factors can be trivially derived from Eq. 4.5.4.4:

$$\frac{\partial W_{ii}}{\partial a_{1f}} = \frac{1}{2} \langle e_{ki} v_M \rangle^2 (\cot(\alpha_1) + \cot(\alpha_2)), \quad (4.5.4.9)$$

$$\frac{\partial W_{ij}}{\partial a_{1f}} = \frac{1}{2} \frac{\langle e_{ki} v_M \rangle \langle e_{kj} v_M \rangle}{\sin(\alpha_1)} \quad (4.5.4.10)$$

$$\frac{\partial W_{ii}}{\partial a_{2f}} = \frac{1}{2} \langle e_{kj} v_m \rangle^2 (\cot(\alpha_1) + \cot(\alpha_2)) \quad (4.5.4.11)$$

$$\frac{\partial W_{ij}}{\partial a_{2f}} = \frac{1}{2} \frac{\langle e_{ki} v_m \rangle \langle e_{kj} v_m \rangle}{\sin(\alpha_1)} \quad (4.5.4.12)$$

where f is any given face on the mesh, e.g. ijk_1 in figure 4.8.

Rotation parameter θ

One might consider the rotation in 3D at first sight, however any rotation of the orthonormal frame, only occurs on the orthogonal plane formed by the principal directions (see figure 4.9), thereby reducing the problem onto the angle only and then can trivially be calculated with:

$$\begin{aligned} \frac{\partial W_{ii}}{\partial \theta_f} = & -(\cot(\alpha_1) + \cot(\alpha_2))(a_1 \sin \cos^{-1} \langle e_{ki} v_M \rangle \langle e_{kj} v_M \rangle \\ & + a_2 \sin \cos^{-1} \langle e_{ki} v_m \rangle \langle e_{kj} v_m \rangle). \end{aligned} \quad (4.5.4.13)$$

Learning a rotation parameter must account for the nature of the problem. For example, unbounded activation functions such as Rectified Linear Unit (ReLU), when mapped back onto a circle, may either rotate repeatedly without limit or saturate too early. An alternative approach is to use the hyperbolic tangent function, which has a range of $[-1, 1]$, and scale it by π to map it onto the circle. This allows the network to learn rotations for each vector within the range $[-\pi, \pi]$ (depicted in Figure 4.12), enabling full rotation of any vector.

ALBONet and RiemannNet

As the previously introduced RiemannNet gradients assume the LBO to be isotropic, the introduction of ALBONet requires a new set of RiemannNet gradients for the

4. LBONet: Supervised Spectral Descriptors for Shape Analysis

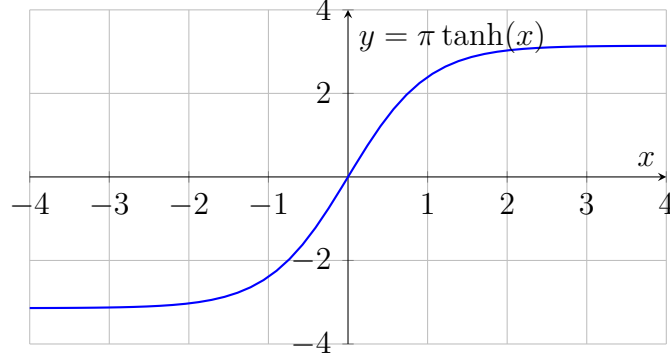


Figure 4.12: Plot of the hyperbolic tangent function scaled by π : $y = \pi \tanh(x)$. This allows the learning of the rotation in both directions effectively and flattens out as it approaches half a revolution.

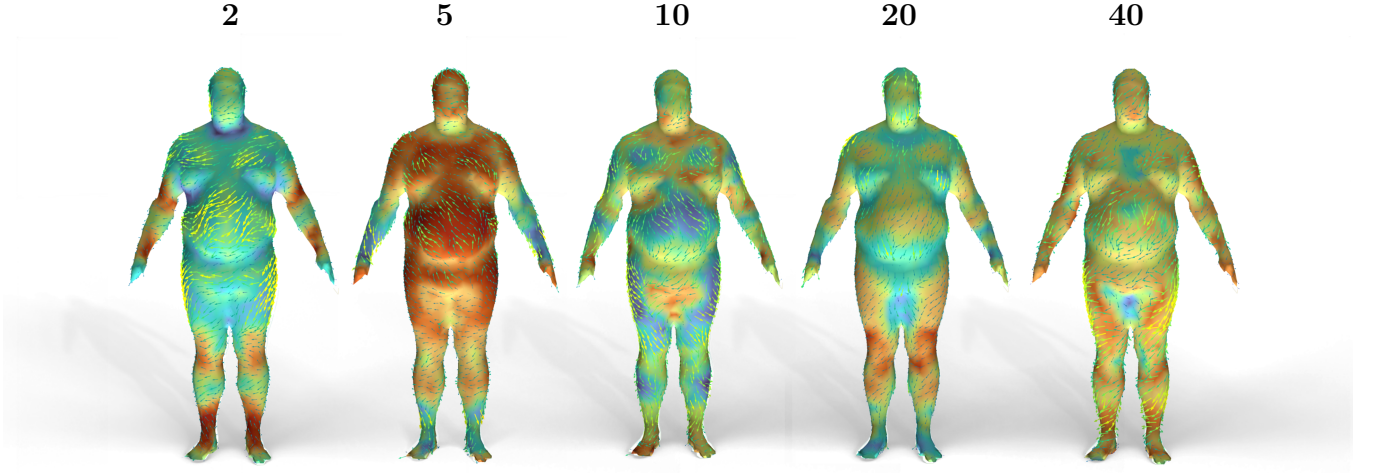


Figure 4.13: Effect of varying neighborhood size in ALBONet Rotation parameter θ (part of LBONet). The numbers above indicate the neighborhood size used in each visualization. Red denotes an acceleration of anisotropy, whereas blue indicates a deceleration. Arrow illustrates the Θ .

stiffness matrix for the anisotropic case,

$$\begin{aligned}
 \frac{\partial W_{ii}}{\partial d_{ij}} &= -\frac{1}{2}(a_1 \langle e_{ki} v_M \rangle^2 + a_2 \langle e_{ki} v_m \rangle^2) \frac{d_{ij}}{2T} \frac{1}{\sin(\delta)^2} \\
 &\quad - \frac{(d_{ik} \cos(\gamma))}{2T} \frac{1}{\sin(\alpha)^2} \\
 &\quad + (a_1 2 \sin \cos^{-1}(\langle e_{ki} v_M \rangle)) \langle e_{ki} v_M \rangle \frac{d_{ik} \cos(\gamma)}{2T} \\
 &\quad + a_2 2 \sin \cos^{-1}(\langle e_{ki} v_m \rangle) \langle e_{ki} v_m \rangle \frac{d_{ik} \cos(\gamma)}{2T} (\cot(\alpha_1) + \cot(\alpha_2)) \quad (4.5.4.14)
 \end{aligned}$$

$$\begin{aligned}
 \frac{\partial W_{jj}}{\partial d_{ij}} &= -\frac{1}{2}(a_1 \langle e_{kj} v_M \rangle^2 + a_2 \langle e_{kj} v_m \rangle^2) \frac{d_{ij}}{2T} \frac{1}{\sin(\delta)^2} \\
 &\quad - \frac{(d_{jk} \cos(\alpha))}{2T} \frac{1}{\sin(\gamma)^2} \\
 &\quad + (a_1 2 \sin(\cos^{-1}(\langle e_{kj} v_M \rangle)) \langle e_{kj} v_M \rangle \frac{d_{ij}}{2T} \\
 &\quad + a_2 2 \sin(\cos^{-1}(\langle e_{kj} v_m \rangle)) \langle e_{kj} v_m \rangle \frac{d_{jk}}{2T} (\cot(\alpha) + \cot(\delta)) \quad (4.5.4.15)
 \end{aligned}$$

$$\begin{aligned}
 \frac{\partial W_{ij}}{\partial d_{ij}} &= -\frac{1}{2} \sin(\gamma_1) ((a_1 \langle e_{kj} v_M \rangle - \sin(\cos^{-1}(\langle e_{kj} v_M \rangle)) \frac{d_{ik} \cos(\delta_1)}{2T} \\
 &\quad + - \sin(\cos^{-2}(\langle e_{ki} v_M \rangle)) \frac{-d_{jk} \cos(\alpha)}{2T} \langle e_{ki} v_M \rangle) \\
 &\quad + (a_2 \langle e_{kj} v_m \rangle - \sin(\cos^{-1}(\langle e_{kj} v_m \rangle)) \frac{d_{ik} \cos(\delta_1)}{2T} \\
 &\quad + - \sin(\cos^{-1}(\langle e_{ki} v_m \rangle)) \frac{-d_{jk} \cos(\alpha)}{2T} \langle e_{ki} v_m \rangle) \\
 &\quad - \frac{1}{2} ((a_1 (\langle e_{ki} v_M \rangle) \langle e_{kj} v_M \rangle) + a_2 (\langle e_{ki} v_m \rangle \langle e_{kj} v_m \rangle)) \frac{-d_{ij} \cos(\delta)}{2T \sin(\delta)^2} \quad (4.5.4.16)
 \end{aligned}$$

$$\begin{aligned}
 \frac{\partial W_{kk}}{\partial d_{ij}} &= -\frac{1}{2} ((a_1 \langle e_{kj} v_M \rangle^2 + a_2 \langle e_{kj} v_m \rangle^2) \frac{d_{jk} \cos(\alpha)}{2T} \frac{1}{\sin(\gamma)^2} \\
 &\quad + (\frac{d_{ik} \cos(\delta)}{2T} \frac{1}{\alpha^2}) + (a_1 2 \sin(\cos^{-1}(\langle e_{kj} v_M \rangle)) \langle e_{kj} v_M \rangle^2 \frac{d_{jk} \cos(\alpha)}{2T} \\
 &\quad + a_2 2 \sin(\cos^{-1}(\langle e_{kj} v_m \rangle)) \langle e_{kj} v_m \rangle^2 \frac{d_{jk} \cos(\alpha)}{2T} (\alpha + \gamma) \quad (4.5.4.17)
 \end{aligned}$$

As the length of these gradients allude, the solution space of the neural network trained becomes increasingly difficult as both geometry and diffusion is being altered simultaneously. Alternating learning can stabilize training. First training with ALBONet only and then continuing with RiemannNet and then jointly again.

4.5.5 Learning on mesh vertices

Weighting the Voronoi cell has been proven to be beneficial by [20][24]. The challenge however remains in the strategy of weighting the Voronoi cell. Similar to [20], we propose to backpropagate the loss through the LBO eigenbasis but learn the Voronoi cell weighting A_{ii} by vertex-wise perceptrons instead.

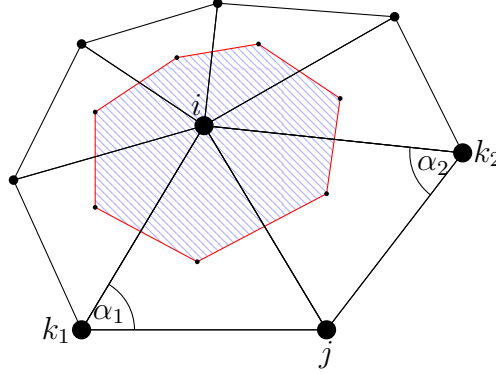


Figure 4.14: Voronoi cell on vertex i , which is outlined in red. The Voronoi cell can be scaled and has only impact on the local Voronoi cell.

The changes for VoronoiNet are limited to the mass matrix A . Therefore we adjust Nelson's method given in equation 2.1.6.5 by omitting the gradients for the stiffness matrix W ,

$$\frac{\partial \lambda_k}{\partial m_{ij}} = \phi_k^T (\Delta A) \phi_k \quad (4.5.5.1)$$

$$F_i = \phi^T \left(-\lambda_i \frac{\partial A_k}{\partial d_{ij}} \right) \phi A \phi - \left(\lambda_i \frac{\partial A_k}{\partial d_{ij}} \right) \phi \quad (4.5.5.2)$$

and introduce the gradients for VoronoiNet as:

$$\frac{\partial A_{ii}}{\partial v_{ii}} = A_{ii}. \quad (4.5.5.3)$$

4.5.6 LBONet Architecture

Our contribution can be plugged into any recent work which we refer to as "backend" (see figure 4.2), however we demonstrate in our ablation study that even with a traditional backend (a single shared MLP), LBONet is able to outperform neural networks with comprehensive backends such as [6] [67] [8]. The performance

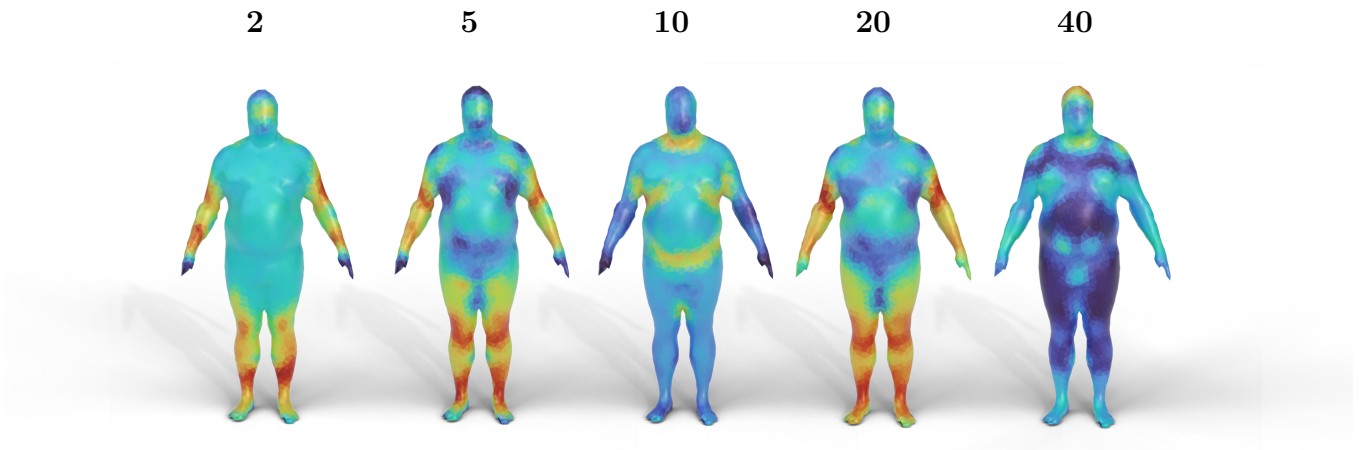


Figure 4.15: Effect of varying neighborhood size in VoronoiNet (part of LBONet). The numbers above indicate the neighborhood size used in each visualization.

comparison to [8] is particularly interesting because our ablation proves that the performance gain obtained by PointNet++, can be similarly obtained by learning with LBONet through the LBO eigenbasis. The upside this work presents however is, that it can still be combined with a more recent backend to gain an even better performance. LBONet outperforms on many benchmarks or achieves competitive results. We show this with one backend derived from PointNet [5] for retrieval and classification. The study of pooling operations in [52] reveals that average pooling is clearly outperforming max pooling; Therefore, we apply average pooling rather than max pooling. For segmentation tasks we derive the backend from point transformer [32], by feeding the resulting spectral descriptor into the transformer and into the dimensionality reduction layer by a skip connection. Finally, for correspondence tasks we observed that the setup of DiffusionNet [7] is a great fit as it is making use of HKS and functional maps, both elements which the presented method is learning. LBONet can be combined with any backend as long as the gradients backpropagated to the spectral descriptors are useful (we observed with deep networks vanishing gradients, which we circumvented with skip connections). Furthermore to speed up learning, we implemented a routine to learn LBONet directly on the given data and then combined it with the backend for fine tuning. While it is possible to run the experiments in a single go, this approach proved to be more efficient with comparable results.

4. *LBONet: Supervised Spectral Descriptors for Shape Analysis*

Furthermore, to stabilize training and accelerate feature learning in LBONet, batch normalization is applied after every shared MLP layer. The default architecture is illustrated in Figure 4.2; however, we also discuss an alternative architecture that exhibits several distinct advantages in the following section. This variant enhances feature learning, particularly in datasets where feature distributions differ significantly across classes, for example, when some classes exhibit fundamentally different curvature characteristics or skewed geometric features. In such cases, LBONet may otherwise struggle to maintain consistent learning behavior, potentially unlearning previously acquired representations or oscillating between incompatible feature scales. Additionally, since the architecture employs batch normalization throughout its layers, further performance gains were observed when conditional batch normalization was introduced.

4.5.7 Learning Stability and Regularization

Adaptive Weighting

When applying operators to the ground-truth metrics of a manifold, whether it is the Riemannian metric, the Voronoi cell, or the anisotropy, the question arises to what degree the network should be allowed to learn. Since these operators directly drive the learning dynamics of the network, it is important to prevent the model from getting stuck in local minima early in training and from learning mesh-specific or class-specific artifacts. However, the default baseline LBO can itself represent a deep local minimum, and if the learnable parameters are too weak to make a meaningful difference, escaping this minimum becomes difficult.

To address this, we introduce global learnable weighting parameters for each layer, initialized with a starting threshold. This threshold can be automatically determined from the input dataset to speed up training. Having these learnable parameters increases the robustness of the framework because the model becomes aware of the relative weighting of each layer. Without this mechanism, the metrics could vary excessively, leading to unstable training dynamics. For VoronoiNet and ALBONet, negative values are not permitted, while for RiemannNet the parameters

are largely unconstrained. This flexibility, if left unregulated, can cause LBONet to diverge or become stuck in poor local minima.

Pooling

Since the shared neural network strongly depends on the quality and distribution of the extracted features, training can become prolonged when multiple classes with highly diverse feature distributions are involved. To stabilize these distributions, we apply average pooling within each learning module. By aggregating features in this manner, variations across classes are reduced, resulting in a more stable and consistent learning signal for the network.

Normalization

Normalization is an essential component throughout our implementation. All input features are normalized, and extreme values are clipped, as previously discussed. Such extremes can lead to ill-conditioned system matrices, which in turn cause downstream numerical errors and hinder effective learning. To further stabilize training, we apply batch normalization in combination with a LeakyReLU activation function. Other activation functions, such as *sigmoid* or *tanh*, result in significantly longer runtimes.

Finally, we perform z -normalization on the output spectral descriptor. This step directs the neural network toward intrinsic learning, since it prevents the model from exploiting magnitude shortcuts to minimize the loss. Instead, the network is compelled to learn genuinely discriminative and descriptive intrinsic features that yield a meaningful spectral representation after normalization.

4.5.8 Implementation

Forward pass We solve the generalized eigenvalue problem on the Central Processing Unit (CPU) using SciPy’s [68] implementation of restarted Arnoldi methods [69]. The neural networks are implemented in PyTorch [70] and are trained on a GPU except for the correspondence task, which due its mesh resolution required to be trained on a data centre GPU (Nvidia H100).

4. LBONet: Supervised Spectral Descriptors for Shape Analysis

Backward pass Standard libraries such as PyTorch or Tensorflow only allow to differentiate the full spectrum of a matrix, creating two problems: firstly the sparsity of the data is not respected, and secondly, as the most important information is contained within the lower frequencies, one cannot target specific eigenfrequencies. HodgeNet [25] takes an alternative approach to mitigate these problems. The Locally Optimal Block Preconditioned Conjugate Gradient (LOBPCG) method in PyTorch does not support backpropagation when both stiffness and mass matrix are provided, moreover it cannot deal with sparse matrices.

Vertices	1024	2048	4096
RiemannNet	0.09	0.51	3.4
ALBONet	0.05	0.4	2.5
ALBONet+	0.08	0.6	3.3
VoronoiNet	0.03	0.2	0.8

Table 4.1: Backpropagation runtimes on varying mesh resolution on a RTX 4090 GPU given in seconds.

LBONet introduces a framework to tackle both these problems by respecting the sparsity of the data and by addressing only a certain range of frequencies (first eigenfrequencies can be skipped as they capture very global features) while differentiating the eigenbasis. We introduce custom functions to deal with sparse data and a subset of the eigenfrequencies. Despite the computational complexity, the presented routine is fully vectorized in both dimensions, edges/faces/vertices and frequencies, calculating the whole gradient of a mesh for the whole eigenbasis in one go resulting in a computation in a reasonable timeframe, see table 4.1 for the running times of the presented modules. In contrast to HodgeNet, the backpropagation procedure of LBONet does not require the computation of additional eigenfrequencies. Conversely, we observed the learning effect to be extending to frequencies, which were not used during the training process. This effect could allude that the features learnt by LBONet could generalize across frequencies as well, which is subject to further research. As the eigenbasis gradient can be very sensitive and take extreme values in certain cases we apply gradient clipping, to mediate

numerical instabilities during the training process. Despite being computationally expensive to train, LBONet itself does not include many parameters, as the neural networks are all shared and relatively small. The features learned are fundamental to the manifold and therefore improvements to the standard LBO can be observed after few epochs of learning and do not require extended amount of training. Due to compactness of the network and the learned features being fundamental, the features learnt do generalize well to unseen data as our evaluation shows. We release our code to reproduce and to be incorporated in further research under <https://github.com/yioguz/LBONet>.

4.6 Evaluation

We evaluate LBONet quantitatively on several experiments in the disciplines of retrieval and classification. As in the retrieval and classification tasks the goal is to learn global descriptors, we show the potency of LBONet in segmentation, and correspondence tasks which rely on descriptors with local accuracy. For every benchmark we give the baseline results, which is achieved by completely omitting LBONet and only relying on the standard HKS. Furthermore, to illustrate the net gain of our contribution we make an ablation study on a segmentation and retrieval benchmarks by comparing standard LBO to LBONet in detail. In all experiments, we train LBONet on the first 32 eigenfrequencies and use EdgeConv with a neighbourhood size of 20 and two layers of shared MLPs (32 neurons each layer) with LeakyReLU activation and batch normalization.

4.6.1 Retrieval

We report our retrieval results with the established performance metrics introduced by [50], which reward different characteristics of the search result. We evaluate our model on the SHREC'11 [71], SHREC'14 [51] and SHREC'15 benchmarks [59]. The SHREC'11 benchmark consists of 600 manifolds divided into 30 classes with 20 shapes for each class. We adapt the approach [7] by training only on 10 shapes. We repeat the experiment five times by drawing randomly different samples to train and

4. LBONet: Supervised Spectral Descriptors for Shape Analysis

average the results. The SHREC’14 benchmark consists of two datasets, which are obtained from real scans and synthetic manifolds. The ‘real’ benchmark especially proves itself as a real challenge in the scientific community, as algorithms fail to achieve a high accuracy on this benchmark. For SHREC’15 benchmark we adapt the approach of [8] and use k-fold cross validation to derive the performance metrics.

We achieve competitive results on SHREC’11 (table 4.2) with perfect results. On both SHREC’14 benchmarks (tables 4.3,4.4), we outperform all other methods. Especially, the results in the ‘real’ benchmark are striking. The jump in the first tier metric demonstrates the capability of LBONet to exploit further potential from the LBO. As the dataset of SHREC’11 is a subset of SHREC’15, we see a similar performance gain on SHREC’15 (table 4.5) with near-perfect results. It is worthy of mention that the addition of manifolds with noise and topological changes has not much of an observable deficit on the results. We elaborate on how LBONet achieves this performance gain of the HKS in the ablation study.

Method	NN	FT	ST	DCG
SD-GDM-meshSIFT [71]	100	97.20	99.01	99.55
FV-IWKS[16]	99.83	95.91	98.60	99.37
KLBO [44]	100	100	100	100
Baseline	99.67	98.04	98.30	99.11
LBONet	100	100	100	100

Table 4.2: Results on the SHREC’11 dataset (in %). Metrics are computed using the evaluation protocol introduced in the Princeton Shape Benchmark [50].

Method	NN	FT	ST	DCG
supDLtrain [58]	79.3	72.7	91.4	89.1
DeepGM[57]	72.5	53.6	82.7	78.2
SOST[52]	85.3	63.2	85.2	85.1
Baseline	85	64.81	86.03	86.41
LBONet	86.5	77.33	91.14	90.69

Table 4.3: Results on SHREC’14 ‘real’ Dataset (in %). Metrics are computed using the evaluation protocol introduced in the Princeton Shape Benchmark [50].

Method	NN	FT	ST	DCG
supDLtrain [58]	96.0	88.7	99.1	97.5
DeepGM [57]	99.3	81.4	98.3	96.7
3D-DL[56]	92.3	76.0	91.1	92 .1
KLBO [44]	88.83	81.58	95.12	94.03
Baseline	97.33	81.40	95.25	95.30
LBONet	99.33	89.21	97.95	97.87

Table 4.4: Results on SHREC’14 ’synthetic’ Dataset (in %). Metrics are computed using the evaluation protocol introduced in the Princeton Shape Benchmark [50].

Method	NN	FT	ST	DCG
HAPT [72]	99.8	96.6	98.2	99.2
DeepGM [57]	99.3	94.0	97.2	98.8
AP-SOST-Net[52]	99.1	93.7	96.5	97.5
KLBO [44]	99.92	98.59	99.33	99.59
Baseline	97.2	96.4	98.5	98.48
LBONet	99.20	98.80	99.70	99.48

Table 4.5: Results on SHREC’15 Dataset (in %). Metrics are computed using the evaluation protocol introduced in the Princeton Shape Benchmark [50].

4.6.2 Classification

In order to compare the presented work to the broader literature, we conduct the experiments on SHREC’11 and SHREC’15 in a classification setting. Therefore we employ the same settings as in the retrieval case, with the exception of using cross entropy loss for learning.

We observe a similar performance gain as in the retrieval case for both benchmarks. The accuracy for SHREC’11 (table 4.6) and on SHREC’15 (table 4.7) is perfect. We observed that after applying LBONet that near-isometries are alleviated to some extent as demonstrated in figure 4.16.

4.6.3 Segmentation

Segmentation has become an important task and challenging benchmarks such as the human segmentation benchmark remain hard. As recent works got better in

4. LBONet: Supervised Spectral Descriptors for Shape Analysis

Method	Accuracy
MeshCNN[67]	91.0
HodgeNet[25]	94.67
MeshWalker[73]	97.1
PD-MeshNet[38]	99.1
FC[74]	99.2
DiffusionNet[7]	99.7
Baseline	98.0
LBONet	100.0

Table 4.6: Results on SHREC’11 Dataset (in %)

Method	Accuracy
DeepGM[57]	93.03
SpiderCNN[75]	95.8
PointNet++[8]	96.09
Baseline	97.8
LBONet	100.0

Table 4.7: Results on SHREC’15 Dataset (in %)

solving them as compared to traditional methods, it is interesting to measure the gain of the presented work, as most spectral based methods have been outperformed recently. To measure the increase of the accuracy of local descriptors and the impact of LBONet, we run experiments on four established segmentation benchmarks. The humans segmentation benchmark [76] has been used widely across the literature and features human shapes in different poses, which we will be referred as humans in the following. The COSEG benchmarks [77] feature vases, chairs, and aliens, which we indicate in the following by their category. As backend we use a modified version of [32], where we use skip connections to allow better gradient flow into LBONet. We evaluate on the humans test set and for the COSEG benchmarks, we adapt the training setup as in [67][38][25] and split the dataset randomly in 85% for training, and 15% for testing. We report our results based on hard labels in table 4.8. We outperform most of the methods, including DiffusionNet and MeshCNN. All other methods were already outperformed by the simple architecture used in our ablation

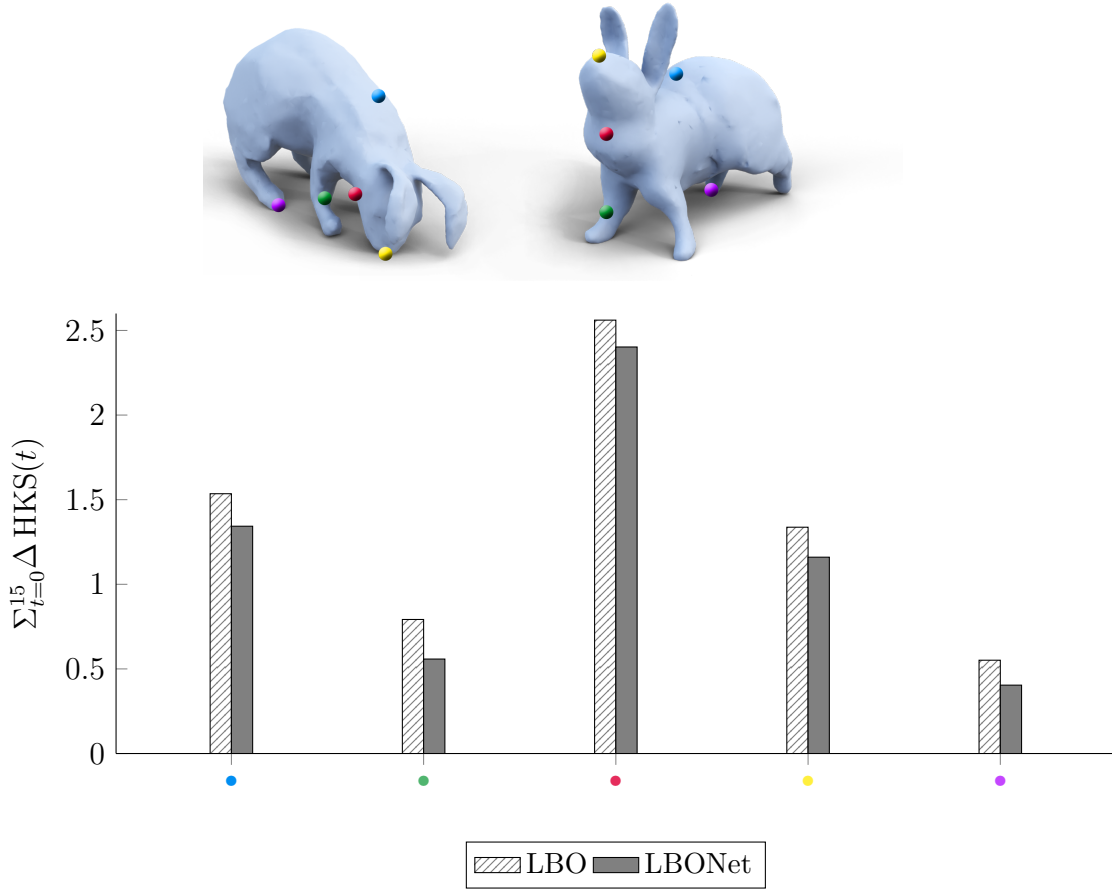


Figure 4.16: HKS at selected points evaluated with the standard LBO and with the LBONet trained on the SHREC11 benchmark. Values are given as the difference of the heat kernel at the highlighted points.

Method	Humans	Vases	Chairs	Aliens
MeshCNN[67]	85.4	92.36	92.99	96.26
HodgeNet[25]	85.0	90.30	95.68	96.03
PD-MeshNet[38]	85.6	95.36	97.23	98.18
DGCNN [6]	89.7	-	-	-
DiffusionNet [7]	91.7	-	-	-
MeshWalker [73]	92.7	-	-	-
FC [74]	92.9	-	-	-
Baseline	91.1	94.7	96.5	97.2
LBONet	92.7	95.6	97.8	98.3

Table 4.8: Segmentation accuracy on humans and COSEG benchmarks (in %)

study (table 4.10) and when LBONet is combined with a recent backend such as a point transformer [32], it becomes even more powerful as the learning can be based

4. LBONet: Supervised Spectral Descriptors for Shape Analysis

on a highly optimized spectral descriptor provided by LBONet. For a fair comparison we also include the performance of our baseline method [32] based on the standard HKS and the net gain of using LBONet becomes apparent. We discuss the effects of LBONet to the LBO eigenbasis in the segmentation tasks in the ablation study, which further reveals how the segments of the manifolds become encoded into the LBO eigenbasis, making it easier for descriptors to separate the segments more easily.

4.6.4 Functional Correspondence

For functional correspondence tasks we adapt the supervised setup from [7] for the FAUST [78] dataset. The FAUST dataset includes both intra- and interclass manifolds of varying degree of near-isometric deformation for training and testing. It serves as a great way to compare the performance gain with the wider literature. We randomly generate all combinations of the 80 manifolds for training and test on the remaining combinations of 20 manifolds. The errors are averaged over all manifolds. We learn the eigenbasis through the HKS directly using triplet loss using corresponding points and generating dissimilar points on the fly. We feed the learned HKS and its eigenbasis into DiffusionNet and achieve a significant reduction in geodesic mean error, proving two facts, first that the learned HKS is more descriptive than the standard used by DiffusionNet and second, that the eigenbasis is better for constructing functional maps after the learning.

KPConv [79]	2.9
HSN [80]	3.3
ACSCNN [81]	2.7
DiffusionNet [†] [7]	2.5
Unsupervised [36]	1.5
LBONet	1.4

Table 4.9: Results on the FAUST [78] benchmark. Values are given in geodesic mean error x 100. Result marked with † serves as the baseline method.

In the texture transfer illustration in figure 4.17, one can observe that LBONet is performing the best in the areas where the stretching occurs (top row: legs,

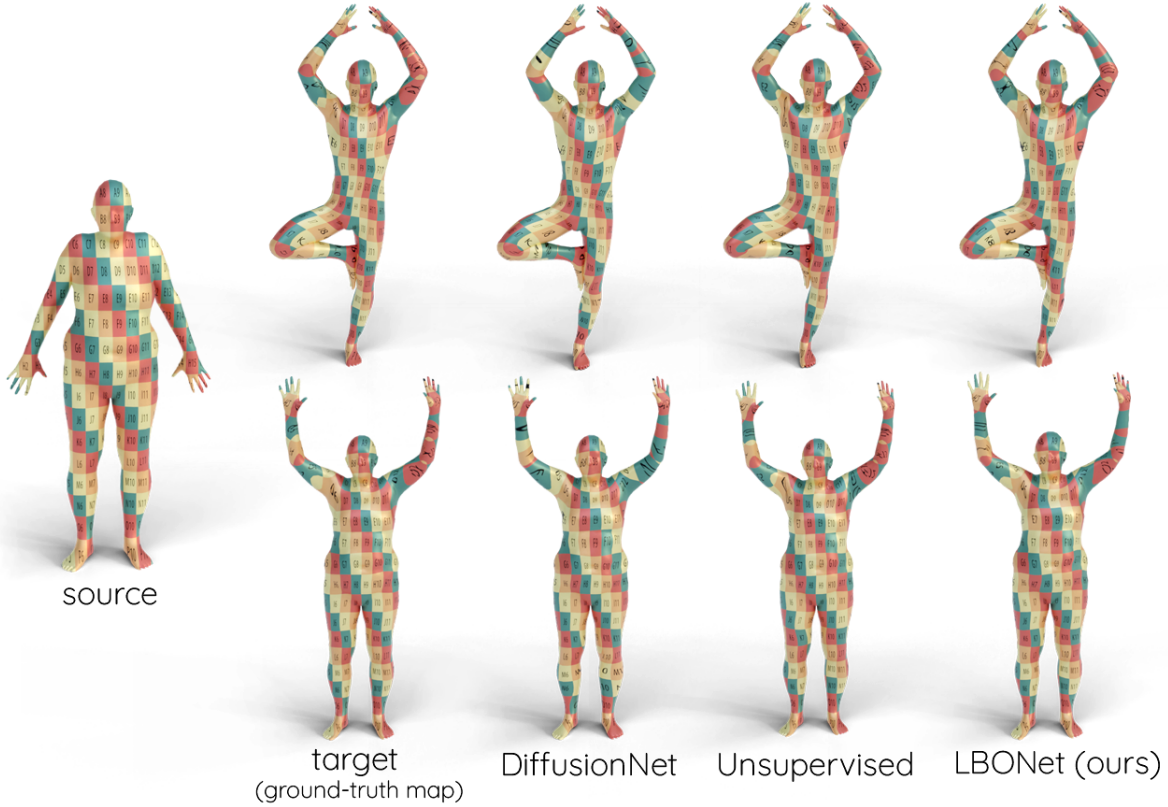


Figure 4.17: Texture transfer experiment on the FAUST test set. Source illustrates the texture applied to the source shape, and target the texture mapped via the ground-truth correspondence to the target shape. Followed by the illustrations of the predictions of DiffusionNet, Unsupervised, and LBONet.

bottom row: armpits), where DiffusionNet and Unsupervised perform significantly worse. Two minor glitches are visible with LBONet. The patches are not as straight as in the ground-truth and the left forearm has some misalignment. Nevertheless, LBONet outperforms Unsupervised and correctly recognizes stretched areas.

4.6.5 Ablation Study

To prove the effectiveness of LBONet, we repeat the human segmentation [76] experiment in a more traditional setting. We conduct the experiment by using the spectral descriptor directly (see figure 4.2 Frontend) with a single shared MLP instead of a Backend. This way it is down to the bare minimum and we can measure LBONets contribution almost in isolation. Table 4.10 shows the results of each layer. Several observations can be made. VoronoiNet on its own does not provide much benefit, however when run in combination of the other networks such as

4. LBONet: Supervised Spectral Descriptors for Shape Analysis

RiemannNet or ALBONet it creates a synergistic effect. RiemannNet and ALBONet seems to have some overlapping effect as they still add some performance, however the performance gain diminishes. The most intriguing effect however is, that when ALBO is allowed to learn the anisotropic factor in both directions (ALBO+), we see a substantial increase in performance, which in the literature and methods using ALBO went mostly unnoticed. ALBO requires much higher weights to achieve the performance gain below, which ALBO+ manages to achieve with much lower weights, allowing it to influence anisotropy, without affecting the generalization. Furthermore, the increased weights with ALBO might be interfering with RiemannNet.

RiemannNet	ALBONet	ALBO+Net	VoronoiNet	Accuracy	Improvement
-	-	-	-	84.1	
-	-	-	✓	84.6	0.6%
-	✓	-	-	85.6	1.78%
✓	-	-	-	86.9	3.33%
-	✓	-	✓	87.6	4.16%
✓	✓	-	-	87.8	4.40%
-	-	✓	-	88.1	4.76%
✓	-	-	✓	88.1	4.76%
✓	✓	-	✓	88.3	4.99%
✓	-	✓	-	89.1	5.95%
-	-	✓	✓	89.9	6.90%
✓	-	✓	✓	90.7	7.85%

Table 4.10: Results on the human segmentation benchmark. The table reports the performance of the proposed modules, both individually and in combination. A ✓ symbol indicates that the corresponding module was used to obtain the reported result.

To further understand how LBONet is learning in a specific task, we visualize the learned features in a segmentation task in figure 4.18. At this point it is important to differentiate that LBONet can be used with any spectral descriptor, but when used with the HKS, the spectrum will be optimized wrt. what the signature picks out from the eigenbasis. Hence, in a correspondence task we can expect the eigenbasis to be closer after learning with LBONet, however in a segmentation task that is not necessarily the case, as LBONet will learn an eigenbasis which allows better separation of segments. As interpretation of learned features within a

neural network is a difficult task and might not always yield to explainable results, some patterns are evident in figure 4.18. This becomes clear when considering the ground-truth segmentation. Firstly, when observing the legs, we see that ALBONet reduces the anisotropic factor into the maximum curvature direction and applies some rotation, which acts as a boundary in the spectral space between the torso and the leg. Secondly, we see a high activity on the head with RiemannNet and VoronoiNet, which both enlarge the head in the spectral space. By enlarging the head in the spectral space, the head is becoming easier to distinguish from the torso and upper arm. Lastly, we see scattered detections of further boundaries, which help to to separate the segments even more. The t-SNE [82] method allows to visualize high-dimensional data in lower dimensions. By this, we can compare the standard HKS and the LBONet HKS. We visualize randomly sampled points of the signature on the test set colored by their ground-truth labels, which can be observed in figure 4.19. Several observations can be made from the distribution of the features and aligns with the hypothesis made earlier. We can observe that the standard LBO misclassifies lots of head parts either as torso or even as upper arms, which LBONet is able to eliminate completely. The overlap between the torso and the leg is reduced by LBONet significantly, only allowing a small part to overlap. Furthermore, we see that each segment boundary has been narrowed down drastically, which allows easy separation of segments in the spectral space, which with the standard LBO seems like a hard task.

To understand the gain in a retrieval task we compare the standard HKS to the LBONet HKS by average pooling. The performance is plotted in figure 4.20 alongside their performance achieved in the experiment 4.3 and 4.4. One can observe that the performance gain in the final experiment was due to LBONet as the HKS with LBONet performs better than the standard HKS and that this performance gain translates into the final result. Similar to the ablation done in the segmentation task, we can observe that the LBONet HKS is far more descriptive.

4. LBONet: Supervised Spectral Descriptors for Shape Analysis

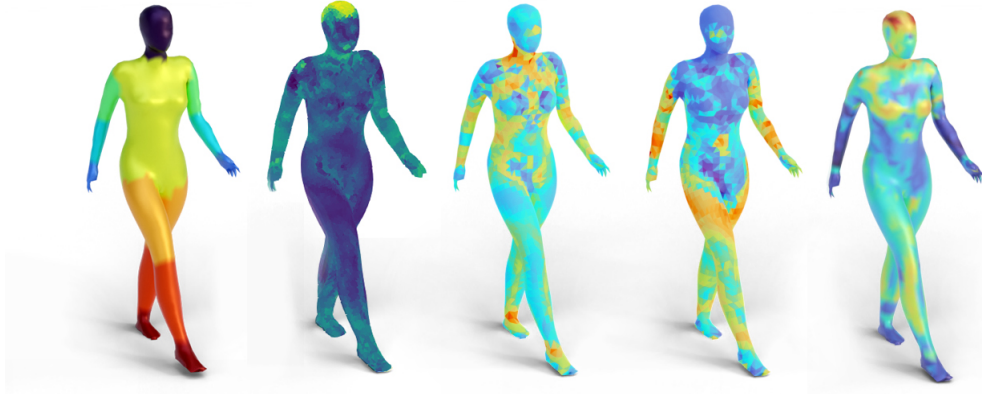


Figure 4.18: Ground-truth segmentation of a manifold and learned features on a test manifold from the human segmentation benchmark by RiemannNet (second), ALBONet (third and fourth), and VoronoiNet (fifth). Interesting patterns emerge along the segmentation boundaries, which demonstrates that individual modules are stronger in certain aspects.

4.7 Conclusion

LBONet serves as a foundational component for neural network learning, enabling the adaptation of spectral descriptors to specific tasks by learning geometric operators directly on the manifold. The comprehensive validation and verification of the method, together with the analysis of common pitfalls, help to overcome major challenges encountered when training on real-world data. This allows subsequent work to focus primarily on data quality and backend optimization to further enhance learning performance.

In this work, we have primarily pursued an implicit learning strategy based on the HKS, thereby avoiding common issues associated with the LBO eigenbasis such as sign ambiguity, eigenvector swapping, and related numerical instabilities. While explicit learning of the LBO eigenbasis is not required for many applications, most contemporary correspondence learning frameworks continue to rely on functional maps, which directly operate on the LBO eigenbasis rather than on a proxy representation such as a spectral descriptor. This observation motivates a dedicated line of inquiry, which is addressed in the following chapter, focusing on explicit LBO eigenbasis learning and the specific challenges arising from it.

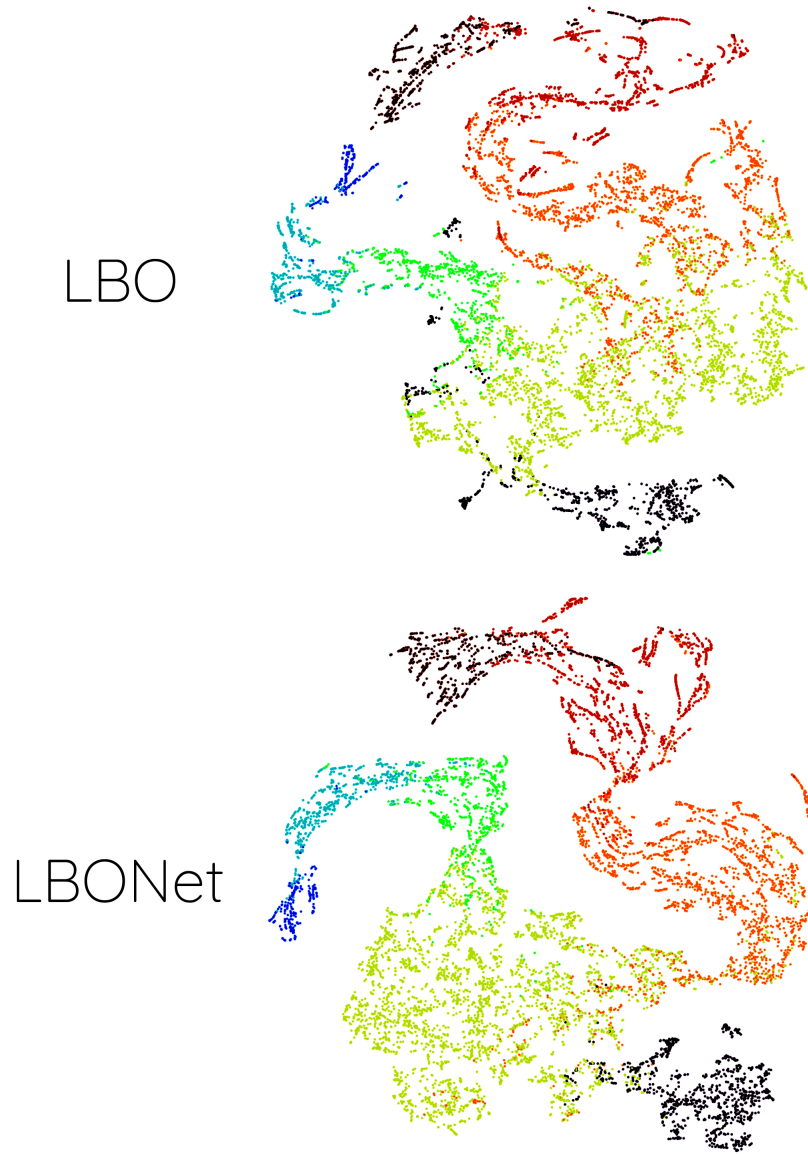


Figure 4.19: HKS features obtained from the ablation study done in chapter 5. Top shows the HKS evaluated with the standard LBO and bottom the HKS evaluated with LBONet. The features were reduced from 16 dimensions to 2D with t-SNE.

4. LBONet: Supervised Spectral Descriptors for Shape Analysis

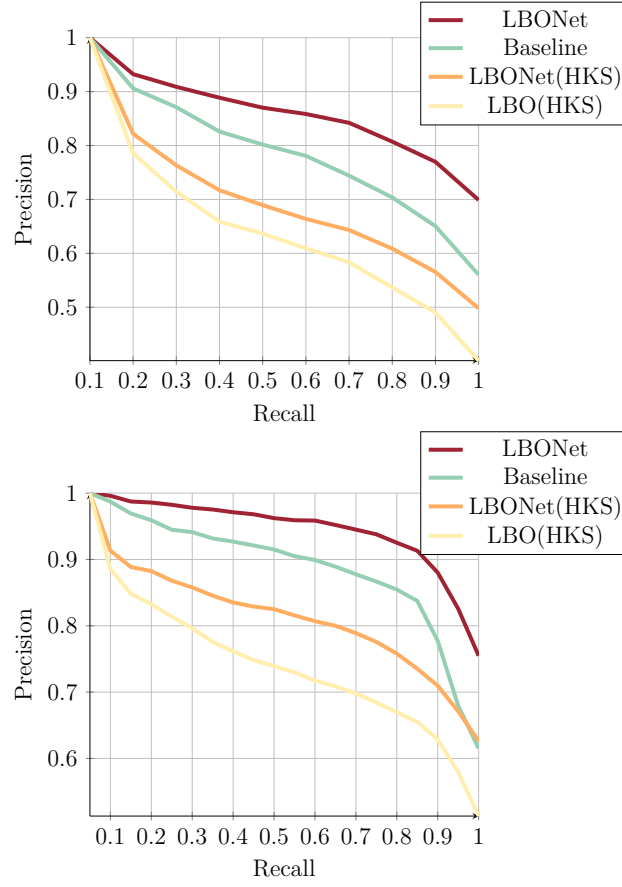


Figure 4.20: Precision/Recall curve of LBONet, Baseline, LBONet HKS, and the LBO HKS evaluated on the SHREC'14 'real' benchmark (top) and SHREC'14 'synthetic' benchmark (bottom). Distances were calculated using Euclidean distance. The performance of the spectral signatures for both LBONet (HKS) and LBO (HKS) were determined by using average pooling.

*"Geometrie ist die Lehre der unter Transformationen
invarianten Eigenschaften." [83]*

Felix Klein

5

IntrinsicFM: Unsupervised Intrinsic Operator Learning for Functional Maps

5.1 Introduction

The introduction of LBONet in the previous chapter established a crucial foundation for enabling intrinsic learning of the LBO eigenbasis. The backpropagated loss was formulated using spectral descriptors. We illustrated this using the HKS, although other descriptors, such as the WKS, could be employed in a similar manner. In the context of shape correspondence, spectral descriptors play only a secondary role, primarily serving to refine correspondence through, for example, a descriptor-based loss term or as input to feature extractors such as DiffusionNet [7], or in landmark-based functional map estimation as in [84]. The central element in shape correspondence is the LBO eigenbasis, which serves as the basis for constructing functional maps, although other bases, such as the Steklov–Dirichlet eigenfunctions [85], have also been explored. Consequently, this chapter focuses on adapting LBONet to the functional maps framework, enabling the explicit learning of the LBO eigenbasis. We introduce an unsupervised approach for training LBONet, along with an additional module designed to improve the accuracy of the unsupervised pipeline. This results in remarkable accuracy gains, demonstrating that a learned

LBO eigenbasis can outperform state-of-the-art methods and that the learning process is stable and generalizable, contrary to the common belief that learning in the spectral domain is limited in terms of generalization.

5.2 System Overview

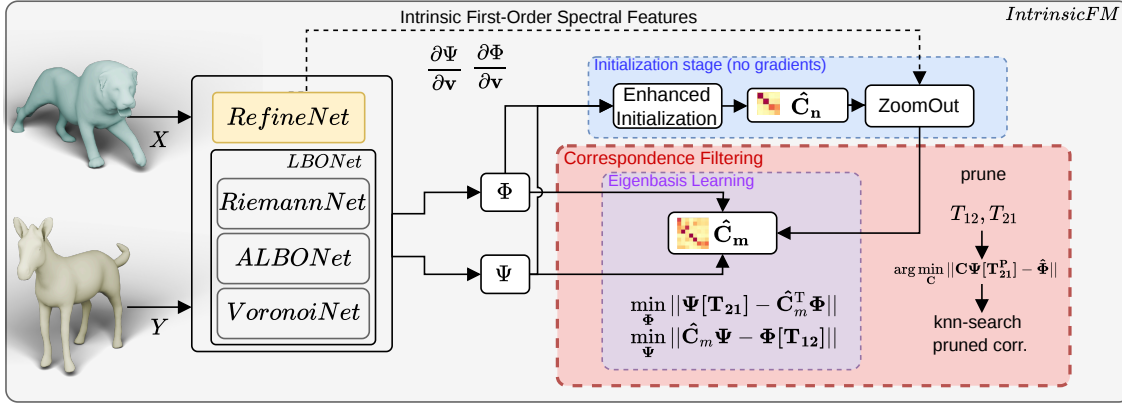


Figure 5.1: IntrinsicFM Overview. Given shapes X and Y , IntrinsicFM learns deformation-aware spectral operators in a fully unsupervised closed-loop framework. **Operator Learning (Left):** LBONet predicts adapted stiffness and mass matrices yielding eigenbases Φ, Ψ ; RefineNet augments these with spectral sensitivities ($\partial\Phi/\partial\mathbf{v}$, $\partial\Psi/\partial\mathbf{v}$) for robust vertex matching. **Initialization (Top):** ZoomOut estimates an initial functional map $\hat{C}_n$. **Eigenbasis Learning (Bottom):** Filtered correspondences construct refined maps (T_{12}, T_{21}) that supervise spectral alignment via bidirectional projection error, forming an iterative cycle that adapts the LBO to the correspondence objective without ground-truth supervision.

This section provides a high-level overview of IntrinsicFM before detailing its underlying mechanics. IntrinsicFM extends deep geometric operator learning into an unsupervised framework tailored explicitly for 3D shape correspondence.

5.2.1 Overall Aim of IntrinsicFM

The primary goal of IntrinsicFM is to explicitly optimize the Laplace-Beltrami Operator (LBO) eigenbasis for the task of functional map-based 3D shape alignment. Rather than relying on fixed downstream descriptors or requiring supervised ground-truth matches, IntrinsicFM optimizes the LBO eigenbasis completely unsupervised

loop. Its purpose is to output an aligned spectral embedding space that minimizes distortions caused by near- and non-isometry.

5.2.2 Inputs and Outputs

- **Inputs:** The system takes a pair of 3D shapes: a source mesh $\mathcal{M}$ and a target mesh $\mathcal{N}$. For both meshes, vertex-wise intrinsic local features are extracted as base signals.
- **Outputs:** The framework outputs a highly accurate functional map matrix $\mathbf{C} \in \mathbb{R}^{k \times k}$ between the two shapes, which is subsequently converted into a point-to-point correspondence map $T : \mathcal{M} \rightarrow \mathcal{N}$ using NN in spectral space.

5.2.3 How It Is Trained (Unsupervised Iterative Loop)

IntrinsicFM uses an unsupervised, iterative closed-loop training process detailed across the subsequent methodology sections:

- **Initialization:** Given a pair of shapes, an initial functional map $\hat{\mathbf{C}}_n$ is estimated using a specialized *enhanced Procrustes* alignment mechanism that we formalize later in this chapter.
- **Supervised Spectral Learning via Proxies:** The spectral map is converted into dense spatial point-to-point correspondences (T_{12}, T_{21}) across the meshes. These initial correspondences act as an unsupervised proxy target. We minimize a bidirectional projection error, backpropagating the alignment loss directly into LBONet to adapt the stiffness and mass matrices (W and A), which ultimately produces eigenbases $(\Phi_{\mathcal{M}}$ and $\Psi_{\mathcal{N}})$, which are more resilient to near- and non-isometry.
- **Conditional Correspondence Filtering:** Once the learned eigenbases have converged to a certain degree of stability, the spatial correspondence filtering module is activated. At this stage, structurally flawed or false correspondences naturally begin to drift across iterations. The system monitors these drifting matches, isolates them, and filters them out.

- **Iterative Refinement:** With the noisy false correspondences successfully pruned out, the network continues to optimize on a refined, highly accurate target set, continuously adapting the geometric operator to the specific shape matching task.

5.2.4 Evaluation and Inference Pipeline

Because IntrinsicFM optimizes its operators dynamically for each unique pair of geometric shapes, the evaluation pipeline on unseen shapes operates identically to the unsupervised iterative training loop detailed above.

When presented with an unseen pair of shapes at test time, the system initializes via the enhanced Procrustes alignment and runs through the exact same iterative optimization and conditional correspondence filtering loop. The framework monitors match drifts on-the-fly to filter out false correspondences as the novel eigenbases adapt. Once this test-time alignment loop converges, the functional map $\mathbf{C}$ is locked, and the final point-to-point correspondence map is extracted using nearest neighbors (NN) in Euclidean space.

5.3 Background

The introduction of functional maps [22] has spawned a plethora of methods that leverage this representation, further bolstering its position in correspondence learning tasks. These methods form the backbone of many recent approaches achieving state-of-the-art results on common benchmarks. When paired with a suitable basis, functional maps offer a compact representation of dense maps between manifolds. Moreover, they have been the subject of research for over a decade, with extensions and creative applications appearing regularly.

A notable approach is DeepFM [35], which learns a descriptor to predict functional maps between manifolds. The quality of these predictions largely depends on the descriptor employed [7], with certain choices yielding highly accurate functional correspondences. This concept was later extended by incorporating

5. *IntrinsicFM: Unsupervised Intrinsic Operator Learning for Functional Maps*

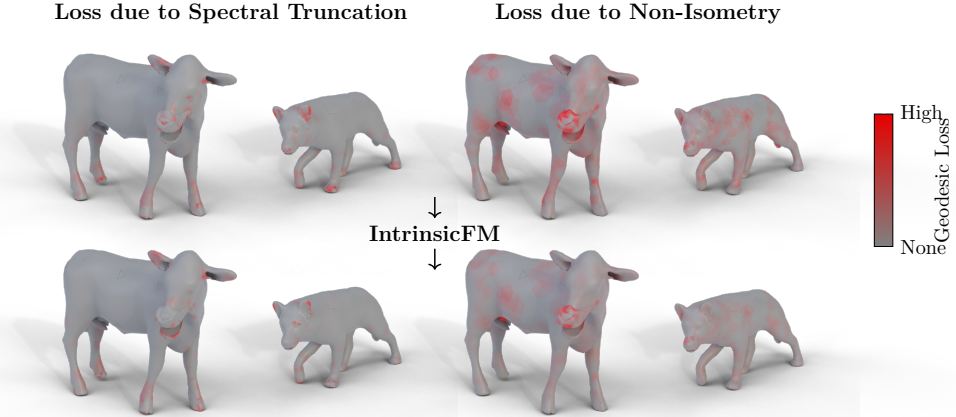


Figure 5.2: Error Analysis: Spectral Truncation vs. Non-Isometry. Top/bottom rows show correspondence error before/after IntrinsicFM. *Spectral Truncation* ($k = 64$): projection error from missing high-frequency components remains spatially structured and moderate. *Non-Isometry*: geometric deviations cause significantly larger spectral misalignment and widespread geodesic error. IntrinsicFM reduces error in both regimes, with pronounced gains under non-isometric deformation where classical LBO assumptions break down. Colorbar: geodesic loss (None $\rightarrow$ High).

orientation awareness [86]. While these methods have shown promise, projection-based approaches can lead to noisy results on the target shape. Ezuz et al. [87] introduced a deblurring technique to smooth the target projection. However, the parameters of this method are highly sensitive, making it challenging to generalize across different cases. The original authors also sought to refine the functional map using ICP [22], where the map is iteratively improved. Additionally, a series of works have focused on aligning the eigenbasis, such as [88], and later on learning arbitrary bases [89][90][91]. Coupling a point-to-point map loss with the predicted functional map [36] enables the correspondence to better adapt to non-isometries of the shape, resulting in improved performance, though at the cost of sacrificing some of the advantages offered by the LBO.

Another notable extension is ZoomOut [37], which recovers an accurate functional map from a low-dimensional coarse map and efficiently upsamples it in spectral space. While effective in generating reasonable maps, it initially considered only point-to-point correspondences between shapes, leaving some potential untapped. Later, Consistent ZoomOut [92] enhanced the method by enforcing point-to-point correspondences on both shapes simultaneously, thereby improving the accuracy of the resulting map. However, this approach has limitations: topological changes or

unstable bases can cause convergence to inaccurate maps. Despite these challenges, the ability to recover a map from an initial 4×4 coarse map is groundbreaking. That said, the initial map must be reasonably close to the ground truth, and without additional training (as in the case of DeepFM), the process relies on accurately predicting this map. Symmetries on the manifolds can require a larger k to resolve them.

In their example, the authors initialize the matrix C as the identity matrix for a non-isometric pair of shapes (cat and wolf), using it as the initialization matrix for ZoomOut, but also employ the initialization method from [84], which constructs a functional map using the LBO eigenbasis given descriptors on both meshes, with a descriptors preservation term, commutativity with descriptors term, commutativity with Laplacian term, and orientation-preserving/self-symmetric term. We demonstrate later that it is possible to derive an accurate functional map by focusing solely on the spectrum and mesh topology of the shapes.

The first eigenvectors are highly global, capturing robust modes on the manifold. Consequently, the corresponding functional maps remain largely diagonal even under moderate non-isometric deformations. As a result, one can estimate a coarse map and then spectrally upscale it. We leverage this approach to make our pipeline fully unsupervised. Our validation shows that the estimated map is very close to the ground truth and can be used effectively for learning tasks. Note that this framework can be made supervised by using the ground-truth correspondence instead of the estimated functional map from ZoomOut. Doing so, however, introduces a significant problem, as the ground-truth correspondence is often difficult to obtain or establish, especially for non-isometric pairs where labeled data is scarce. The geodesic error introduced by using ZoomOut instead of ground-truth labeled data still yields usable gradients. Even a rough correspondence, for example with a residual mean geodesic error of 0.05 on a pair, provides sufficient accuracy to learn useful features and gradually improve performance toward the ground truth. Note that even when using ground-truth correspondence, a geodesic error remains due to truncating the eigenbasis, for example at $\lambda = 64$. Ground-truth data itself

5. *IntrinsicFM: Unsupervised Intrinsic Operator Learning for Functional Maps*

can also be noisy due to mesh discretization or artifacts, forcing an algorithm to pick up this noise. Considering these factors, ZoomOut serves not only to refine a functional map but also to smooth it across pairs of shapes, acting as a natural regularizer. Given this correspondence, LBONet can be trained to optimize a given loss, in the case of correspondence reducing the geodesic loss. To illustrate these effects and better understand what LBONet is learning, we implement both supervised and unsupervised methods and demonstrate each of their strengths and weaknesses later in this chapter.

Our method adapts the eigenbasis using LBONet, but instead of backpropagating gradients through the HKS as in the previous chapter, we directly backpropagate gradients to the eigenbasis. While this helps refine the eigenbasis, we employ additional techniques to mitigate issues such as sign flipping and eigenvector swapping, which could otherwise hinder effective learning. The specific methods for addressing these issues will be discussed later in this chapter. With this end-to-end approach, we learn generalized intrinsic operators that refine the LBO eigenbasis to be more discriminative, thereby improving the point-to-point map implicitly while retaining all the advantages of the LBO eigenbasis. We coin the term *intrinsic functional maps* to describe our method, as we extend the learning process while keeping it fully intrinsic and preserving all the benefits of the original LBO eigenbasis.

The proposed method shares similarities with [36] in the sense of learning a basis for functional maps. However, our approach specifically adapts the LBO eigenbasis by actively modifying the stiffness and mass matrices to account for deformation and noise, enhancing the accuracy of point-to-point map predictions. By maintaining the original properties of the LBO, our learned basis provides greater robustness to these confounding factors. We show that our approach can deliver a better point-to-point map by learning an optimal LBO eigenbasis and outperforming other methods, including [36], while using fewer eigenfrequencies and remaining fully unsupervised.

Functional maps are constrained by two key factors:

- Precision: Functional maps are an approximation and can suffer losses when projecting one eigenbasis onto another. These losses stem from various sources, which we elaborate on later.
- Symmetry: Ideally, the functional map should provide comparable quality in both directions. However, as shown in [93], the loss during projection is asymmetrical, meaning that projections in one direction are not equivalent when reversed. Although most evaluations in the literature focus on uni-directional projections, some works have explored bi-directional maps [84] [94] [95]. Enforcing symmetry can act as a regularizer, preventing overfitting to specific examples.

Functional maps typically rely on the LBO eigenbasis. However, the quality of functional maps depends not only on the number of eigenfrequencies used (with higher frequencies providing finer details) but also on how precisely the eigenvectors from different bases align. If the eigenvectors differ significantly, errors increase, necessitating more eigenfrequencies to achieve an accurate map. The presented method improves upon this by explicitly learning the eigenbasis, resulting in better-aligned eigenvectors. Methods using a learned LBO eigenbasis, rather than the standard LBO eigenbasis, can yield more precise maps and achieve higher accuracy. The significant gains observed on unseen manifolds, as shown in Figures 4.17, 5.4, 5.13, and 5.17, and even across benchmarks, demonstrate a drastic reduction in mean geodesic error. These results show that it is possible to learn from spectral space and generalize across shape classes and benchmarks. Our contribution is an alternative formulation for coupling the point-to-point map with the functional map, as discussed in [36]. A key difference in our method is that we do not learn any arbitrary point-to-point map to reduce a loss; instead, we improve the LBO eigenbasis, which can then be used to infer a better point-to-point map. Hence, our contribution operates on a more fundamental level.

Unlike existing methods, our approach learns directly on the input manifolds by dynamically adapting the operators. By extracting these relevant geometric features,

5. *IntrinsicFM: Unsupervised Intrinsic Operator Learning for Functional Maps*

we construct a functional map that becomes increasingly precise throughout the learning process. The primary contributions of this method are highlighted below.

Contributions. Our work offers a novel contribution to the concept of LBO and functional maps. We demonstrate that functional maps, when used with an LBO eigenbasis, can be learned to better suit specific tasks.

- Backpropagating a loss based on ZoomOut, making ZoomOut an end-to-end learning process (although not within the upscaling process) instead of operating on fixed eigenbases.
- An additional module to LBONet named *RefineNet*, which is able to supply $\frac{\partial \theta}{\partial \mathbf{v}}$ to guide knnsearch in ZoomOut, improving point-to-point map accuracy.
- A framework to backpropagate loss onto the eigenbasis, enabling the explicit unsupervised learning of the LBO eigenbasis.
- An enhanced Procrustes-based functional map using spectral consistency and topological coherence to recover low-frequency functional maps.
- State-of-the-art performance on common benchmarks, demonstrating that the method can be integrated into existing techniques as a base learning layer.

5.4 Related Work

The LBO has proven highly effective in various tasks, and despite deep learning methods catching up with its performance, it remains dominant in certain areas, such as correspondence. As a result, research has continued to focus on addressing the LBO’s significant downsides while also exploring alternative solutions. The term *Isospectralization* was coined by Cosmo et al. [96], enabling the alignment of eigenbases across manifolds via an iterative eigenvalue optimization process. This process improved correspondence quality and was later applied with Autoencoders [97] to generate meshes based on the eigenspectrum. Although high isometry leads to a diagonal functional map, there is no guarantee that the point-to-point map will

be highly accurate. Point-wise correspondence remains challenging, as even when the spectra are similar, correspondence is determined by spatial features. Approaches like CMOS [20] and RMOS [21] have optimized and weighted the Riemannian metric on manifolds, successfully applying them to intrinsic brain mapping. HodgeNet [25] learned the Hodge Star operator on manifolds and approximated eigenbasis gradients with some success, but it was soon surpassed by more recent methods. More recently, the Finsler LBO [98] was introduced, enabling the learning of an asymmetric LBO by removing the quadratic constraint. LBONet [99] further enabled the learning of different operators on mesh edges, faces, and vertices via shared MLPs, backpropagating loss through the eigenbasis and achieving competitive results on established benchmarks. This approach also mitigates near-isometry challenges to some degree. However, LBONet implicitly learns the eigenbasis through the HKS, which has drawbacks due to the loss passing through a low-pass filter, limiting the learning potential and restricting it to supervised learning only.

This chapter extends LBONet to explicitly learn the eigenbasis and transforms ZoomOut into a learnable backend, enabling the LBO eigenbasis to converge toward the ground truth. While functional maps typically face challenges due to the limitations of the LBO, our approach mitigates these issues, resulting in more accurate functional maps, as demonstrated in our experiments.

5.5 Background

Functional maps were introduced by [22] to establish correspondences between manifolds using a compact, efficient basis. Specifically, functional maps map the eigenbasis of manifold X to manifold Y via a linear operator $T : L^2(X) \rightarrow L^2(Y)$. In this setting, functions on one manifold are mapped to functions on the other manifold. This correspondence can be represented using a matrix $\mathbf{C}$, where its coefficients are given by $c_{ji} = \langle \phi_j, T\psi_i \rangle_Y$, and ϕ, ψ are orthogonal bases on each manifold. This leads to the following expansion,

$$Tf = \sum_{ij>1} \langle \phi_i, f \rangle_X c_{ji} \psi_j. \quad (5.5.0.1)$$

5. *IntrinsicFM: Unsupervised Intrinsic Operator Learning for Functional Maps*

In the context of functional maps, the LBO eigenbasis is an ideal choice, as it allows for truncating the eigenbasis at frequency λ_k , leading to a compact representation. The LBO eigenbasis works well for isometric shapes and provides good correspondence quality. However, in the presence of non-isometric deformations, the LBO eigenbasis can become less effective and, in extreme cases, unusable. As the LBO is intrinsic to the manifold, a manifold’s embedding does not interfere with the map. Most shape correspondence benchmarks (FAUST, SCAPE, DT4D, SMAL) are oriented such that the z-axis of the shapes points upward, so this limitation often goes unnoticed, but the LBO remains invariant to such orientation. Despite these challenges, functional maps have gained significant attention, especially with the introduction of the DeepFM framework, which enables learning of features on manifolds. When these features are projected onto the eigenbasis, functional maps can be predicted with high precision. Over time, feature extractors like SHOT [100] have evolved, and more recently, DiffusionNet has established itself as an effective method for feature extraction using HKS as input descriptors, forming the backbone of many other methods (LBONet [99], Unsupervised [36], etc.). LBONet further improves functional map prediction by learning an optimal HKS in conjunction with DiffusionNet. This is achieved by differentiating the HKS with respect to the correspondence loss using functional maps. The result is highly accurate functional map predictions. However, as noted earlier, the quality of the prediction still depends on the quality of the underlying eigenbasis used in the functional map.

5.6 Near-Isometry and Non-Isometry

A significant amount of recent research has focused on how functional maps can be initialized. Methods such as [101] and [84] have introduced elaborate schemes to initialize them based on rich descriptors, landmarks, and additional shape quantities. [37] demonstrated the effectiveness of these methods and noted that in many cases, 415 initialization frequencies were sufficient to upscale the map nearly to the upper limit, with almost no loss incurred. More frequencies were primarily required when symmetry was the underlying issue. However, with increasing non-isometry,

these initialization methods reveal their limitations. [36] addressed this problem by establishing a learnable softmax-based point-to-point map, which then serves as a learning target for the learned arbitrary basis map. This method is effective because it combines learning both the point-to-point map and the functional map, which the aforementioned methods attempted to set up with other elaborate techniques and post-processing. However, as we demonstrated in the previous chapter, the LBO eigenbasis can be learned in a stable manner. Furthermore, learning is generalizable, and non-isometries are less severe in a learned eigenbasis.

5.7 Approach

Our method shares similarities with the approach in [36] but can be seen as an alternative formulation of the coupled loss. Instead of enforcing a point-to-point map through the coupled loss, we apply cross-entropy learning, which is backpropagated to the operators learned by LBONet, thereby increasing the discriminativity of the basis. Once an increase in accuracy is achieved, the functional map is further refined. LBONet introduced in chapter 4 implicitly learn an eigenbasis via the HKS, which avoids issues such as eigenvector swaps and sign flips. Our approach, however, focuses on learning the eigenbasis explicitly. While these issues make direct eigenbasis learning difficult, we overcome this by utilizing the functional maps paradigm to establish a robust, differentiable loss function. We incorporate all modules introduced by LBONet (RiemannNet, ALBO+Net, and VoronoiNet), which learn features at the edge, face, and vertex levels. The quality of a map is primarily influenced by two factors: the truncation of the eigenbasis at frequency λ_k and the amount of near-isometric or non-isometric deformation present. When truncating the eigenbasis at the k th eigenfrequency, higher frequencies, although carrying some additional information, are omitted. Most information is captured within the first k frequencies, but excluding higher frequencies still results in some loss of detail. The second factor, the amount of deformation (near-isometric or non-isometric), introduces challenges in projecting one eigenbasis onto another with high precision. This problem is visually illustrated in Figure 4.17, which shows the

impact of non-isometries on the LBO eigenbasis and correspondence quality. Our method addresses both issues. We propose a mechanism that enables information typically represented by higher-frequency eigenfunctions to be captured within a lower-frequency basis, and we mitigate the challenges posed by non-isometry and near-isometry by adjusting the manifold weighting. While it is theoretically possible to make the eigenbasis fully invariant to these deformations, doing so would decouple the eigenbasis from the spatial domain, making it challenging to learn meaningful spatial features. Figure 5.4 shows that for every eigenfrequency in our learned LBO eigenbasis, more than two standard LBO eigenfrequencies are required to match the same performance, indicating that our learned eigenbasis is not only more compact but also more robust to deformations. Considering that our framework never sees ground-truth information during the learning process, this is particularly noteworthy. This leads to a more compact and precise eigenbasis and functional maps, which is particularly valuable for tasks involving correspondence mapping.



Figure 5.3: Geodesic Error. Fox and horse in the IntrinsicFM LBO eigenbasis (left) and the standard LBO eigenbasis (right). Geodesic error is computed as the geodesic distance between the predicted vertex correspondence and the ground-truth correspondence.

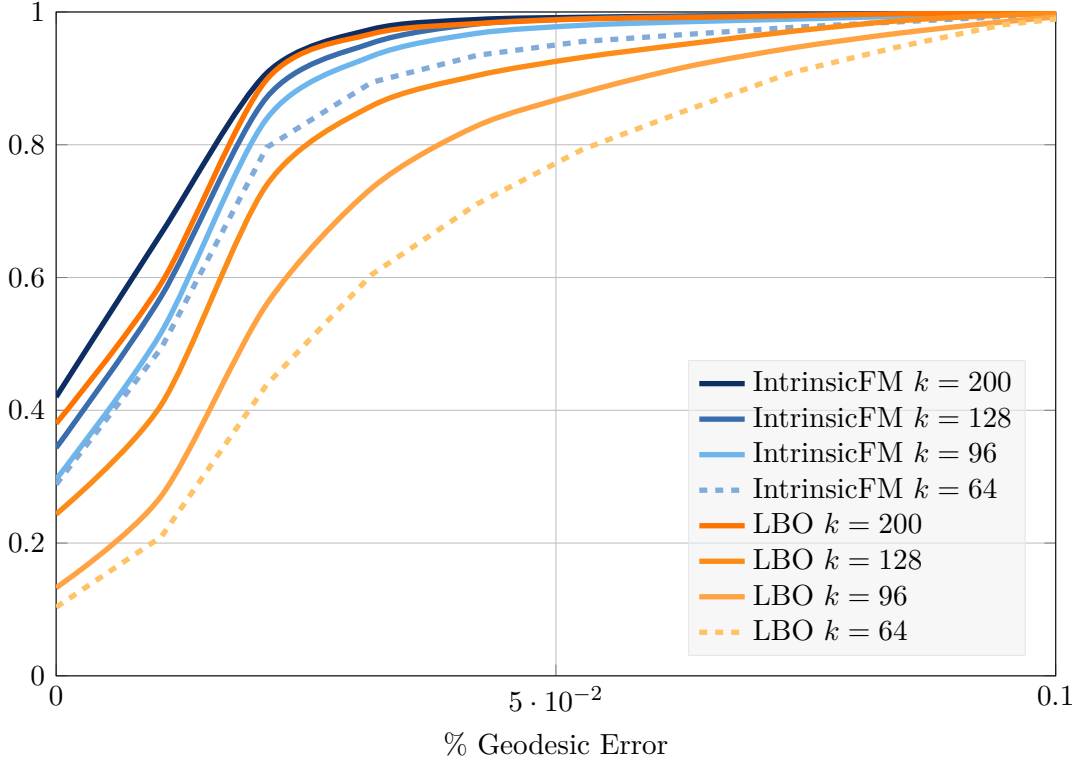


Figure 5.4: Geodesic error between two manifolds with varying eigenfrequency. The left column shows the prediction of an unseen manifold using our approach, while the right column shows the standard LBO eigenbasis. The color scale ranges from blue (indicating smaller error) to red (indicating greater error). Both manifolds share the same color scale for precise comparison. The diagram below illustrates performance based on the number of eigenfrequencies used. Our approach significantly outperforms the baseline, achieving better results with only 64 eigenfrequencies, compared to the baseline LBO, which uses 128 eigenfrequencies. Furthermore, it is evident that learning transfers across frequencies: as more frequencies are used, the accuracy improves. Even with just 64 eigenfrequencies, the majority of the relevant information is already captured.

5.7.1 Building Blocks

IntrinsicFM is built upon three core components that collectively enable a fully unsupervised learning pipeline. The first component, introduced in the previous chapter, is LBONet. Serving as the foundation of IntrinsicFM, LBONet drives the primary learning process to improve the accuracy of the eigenbasis by directly learning the underlying geometry. Within the IntrinsicFM framework, LBONet is trained using estimated functional maps rather than ground-truth correspondences, as depicted by the purple block in Figure 5.1.

To enhance this architecture, we extended LBONet with a compact module

5. *IntrinsicFM: Unsupervised Intrinsic Operator Learning for Functional Maps*

named *RefineNet*. This module provides additional signals to improve the accuracy of ZoomOut, addressing the limitation where relying solely on the eigenbasis for kNN search can yield weak results, particularly in regions requiring high precision. RefineNet computes the gradient of how a vertex position change impacts the eigenbasis. Because similar points across both shapes are expected to yield comparable gradient responses, this addition enables more precise mapping, especially at higher frequencies. LBONet is trained using a triplet loss driven by an estimated functional map, which is generated by the initialization component detailed below.

This initialization component is crucial for establishing a rough initial correspondence, which guides the learning process and allows the pipeline to converge toward the ground truth. Invoked at the start for any given shape pair, this stage leverages an enhanced Procrustes approach 3 capable of recovering low-rank functional maps with a high degree of fidelity. Later, we demonstrate this on a non-trivial, non-isometric pair and detail the specific techniques used. Ultimately, this component produces a functional map sufficiently accurate to drive learning toward the ground truth. Indicated within the blue box titled "Initialization stage" in Figure 5.1, this initial map is subsequently used to train LBONet; despite its initial inaccuracies, its rough correspondence provides enough structure for LBONet to begin learning features, thereby enabling the pipeline to converge toward the ground truth.

The final component is activated only after a sufficiently accurate basis has been learned. We utilize the loss value as a hyperparameter to determine when to trigger this refinement stage, as continuing further would risk LBONet overfitting to false correspondences or noise. This stage is enclosed by a red boundary in Figure 5.1 and is titled "Correspondence Filtering". In addition to steering the eigenvectors toward the ground truth, this filtering process refines the initial map estimated by ZoomOut, which was previously computed on a weaker, unlearned eigenbasis, resulting in further gains in overall accuracy.

5.7.2 Estimating Functional Maps

As previously mentioned, near-isometric cases can be trivially computed using a Procrustes approach, as in the following equation:

$$C_0 = \Phi^T M_2 \Psi \quad (5.7.2.1)$$

This holds in near-isometric cases. However, as reported by [37], symmetries can be difficult to differentiate, and even in near-isometric cases, many frequencies may be required to disentangle them, even when using operators specifically designed to break symmetries, such as [84]. Non-isometric cases present a significantly greater challenge, as even initialization methods like 5.7.2.1 fail under stronger non-isometry. Over the years, many techniques have been introduced to address the issues surrounding estimation and refinement of functional maps. The sign ambiguity of eigenvectors and symmetrical parts of objects are especially challenging, as spectral symmetries behave similarly, and both signs can be valid, further exacerbating the problem. More recently, [85] introduced landmark-adapted bases for non-isometric shape correspondence using the Steklov–Dirichlet eigenfunctions, raising concerns that, due to the global support of the Laplace–Beltrami basis, it might not extend well to non-isometric cases. Prior methods used landmarks in combination with descriptor preservation losses, such as [101], which used manual landmarks, or [102], which even employed automated registration methods to establish correspondences based on landmarks. Landmarks are especially effective because they can break symmetry when they are informative enough in symmetric regions of the given shapes.

While we draw inspiration from previously introduced methods, we focus on two basic ingredients that stabilize low-frequency functional map recovery. We observed that combining spectral consistency with connectivity constraints, such as enforcing local topological isomorphism of the induced subgraph, significantly improves robustness. These anchors are then used to resolve ambiguities present in low-frequency functional maps, providing a reliable initialization for ZoomOut.

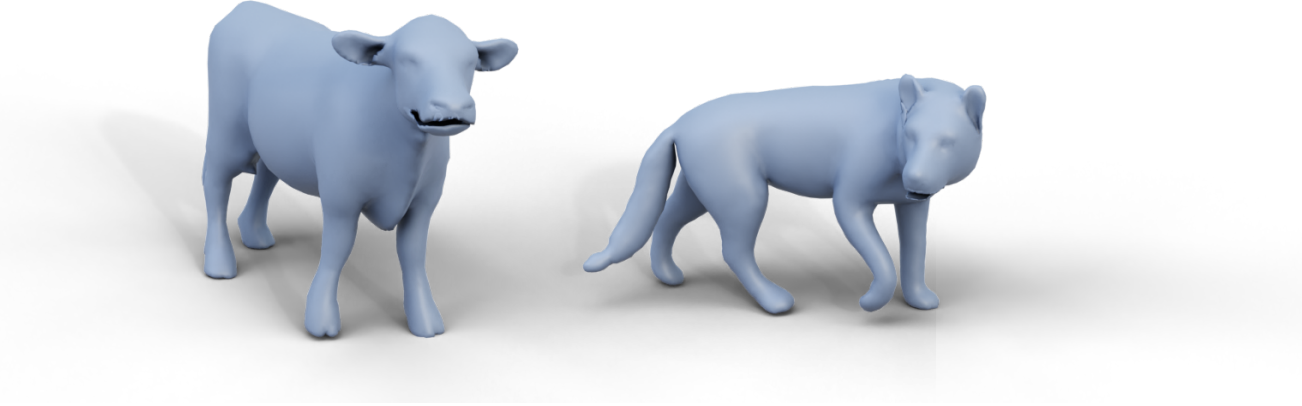


Figure 5.5: Non-isometric Shape pair used in the plotted functional maps.

Spectral Consistency

We define candidate anchors based on their stability during the ZoomOut process. A point is considered stable if both the candidate point and its corresponding point remain within an n -ring over m iterations of ZoomOut. This ensures that these anchors are supported both spectrally and topologically. Enforcing this across shapes results in intrinsic spectral stability, meaning that the eigenvector embeddings are consistent. Furthermore, it leads to topological consistency, as the points only move within a defined neighborhood and are agreed upon on both shapes. By using an m -window, where m is set strictly, we exclude points that drift during this refinement stage.

Topological Constraints

While spectral consistency alone is sufficient to infer low-frequency functional maps, it does little to break symmetries present in the shapes. Even when anchors on both shapes remain locally consistent, they might reside in symmetric regions. As shown in Figure 5.6, multiple symmetries exist in the low-rank functional map,

Algorithm 3 Enhanced Procrustes Estimation

Require: Shapes S_1, S_2 , Laplace–Beltrami bases B_1, B_2 **Ensure:** Estimated Procrustes (or random) map between B_1 and B_2 : C_{init}

```

1:  $C_S^{(0)} \leftarrow C_{\text{init}}$ 
2: for  $k = 1, \dots, |\lambda_n|$  do
3:   for  $i = 1, \dots, |m|$  do
4:      $\hat{C}_{1,i}^{(k)}, T_{12}^{\text{hist}} \leftarrow \text{ZoomOut}(B_1, B_2, C_{S,i}^{(k)})$ 
5:      $\hat{C}_{2,i}^{(k)} \leftarrow (\hat{C}_{1,i}^{(k)})^\top$ 
6:     Identify stable correspondences
7:      $T_{12}^S \leftarrow \text{stable}(T_{12}) \wedge \text{mutual}(T_{12})$ 
8:      $T_{21}^S \leftarrow \text{reverse}(T_{12}^S)$ 
9:      $C_{S,i}^{(k)} = \arg \min_C \|CB_1[T_{12}^S] - B_2\|$ 
9:   end for
10: end for
11: return  $C_m^n$ 

```

and a naive approach fails to derive the correct map due to multiple incorrect signs. These incorrect signs have a detrimental effect on the final correspondence quality and cannot be used for learning. We restrict candidate anchors to be connected by the same n-ring relationships across both shapes. If a point on shape A is part of a neighborhood structure but does not belong to the same structure on shape B, we can conclude that this point does not correspond to the correct point on the other manifold.

In the following, we demonstrate the enhanced Procrustes approach on a 4×4 functional map with a symmetry. We first show a naive initialization in Figure 5.6 and compare it to the ground truth. Next, we illustrate the introduced enhanced Procrustes approach in Figure 5.7. In Figure 5.8, we "unfold" the map to 11 frequencies and compare it to the ground truth. Finally, in Figure 5.9, we run ZoomOut to 16 frequencies and compare it to the baseline. Additional examples from the SMAL and DT4D datasets are provided in Figures 5.7.2 and 5.7.2.

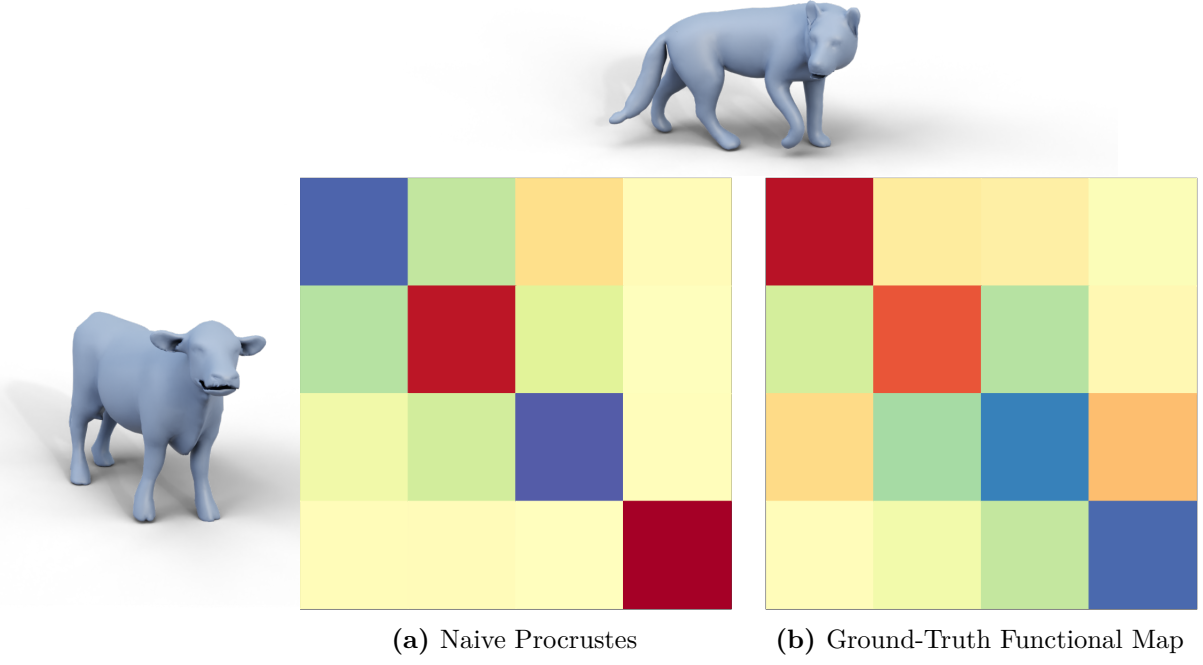


Figure 5.6: Side-by-side comparison of naive Procrustes functional map estimation versus ground truth for a non-isometric pair.

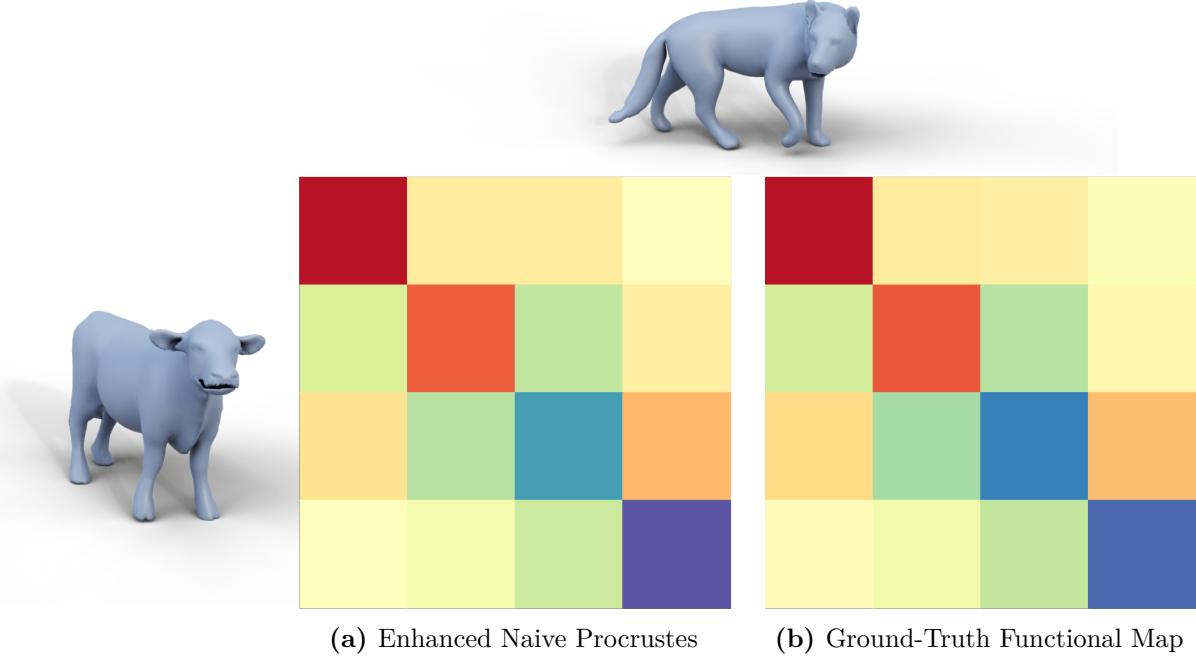


Figure 5.7: Side-by-side comparison of enhanced naive procrustes functional map estimation versus ground-truth of a non-isometric pair.

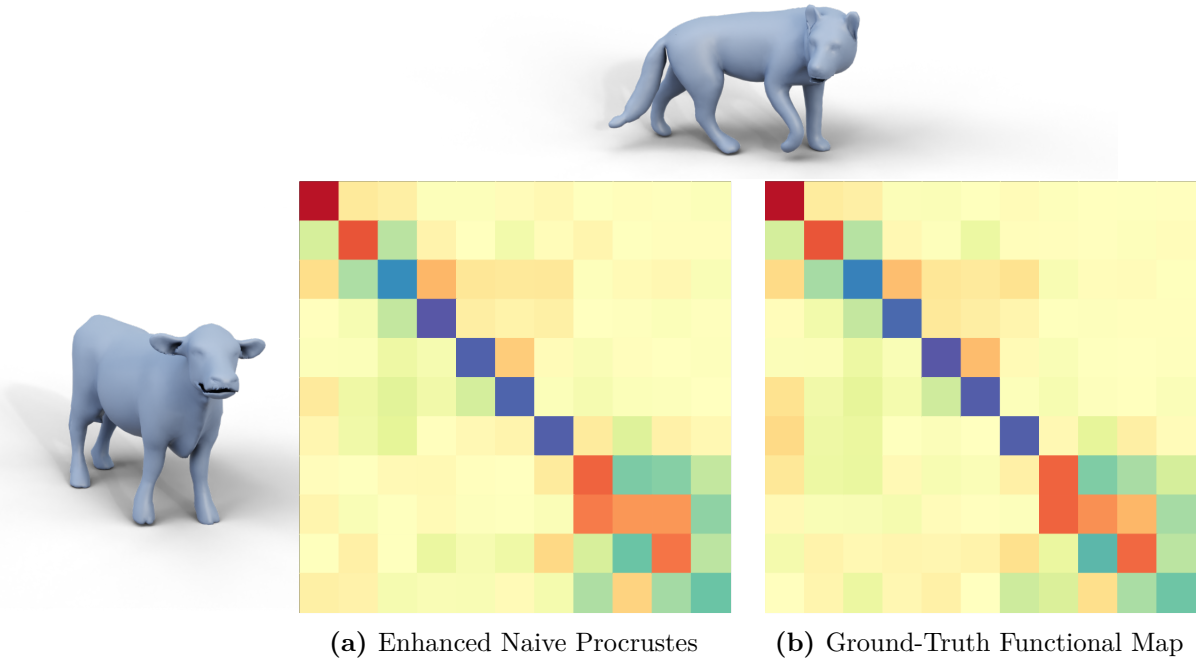


Figure 5.8: Side-by-side comparison of enhanced naive procrustes functional map estimation versus ground-truth of a non-isometric pair.

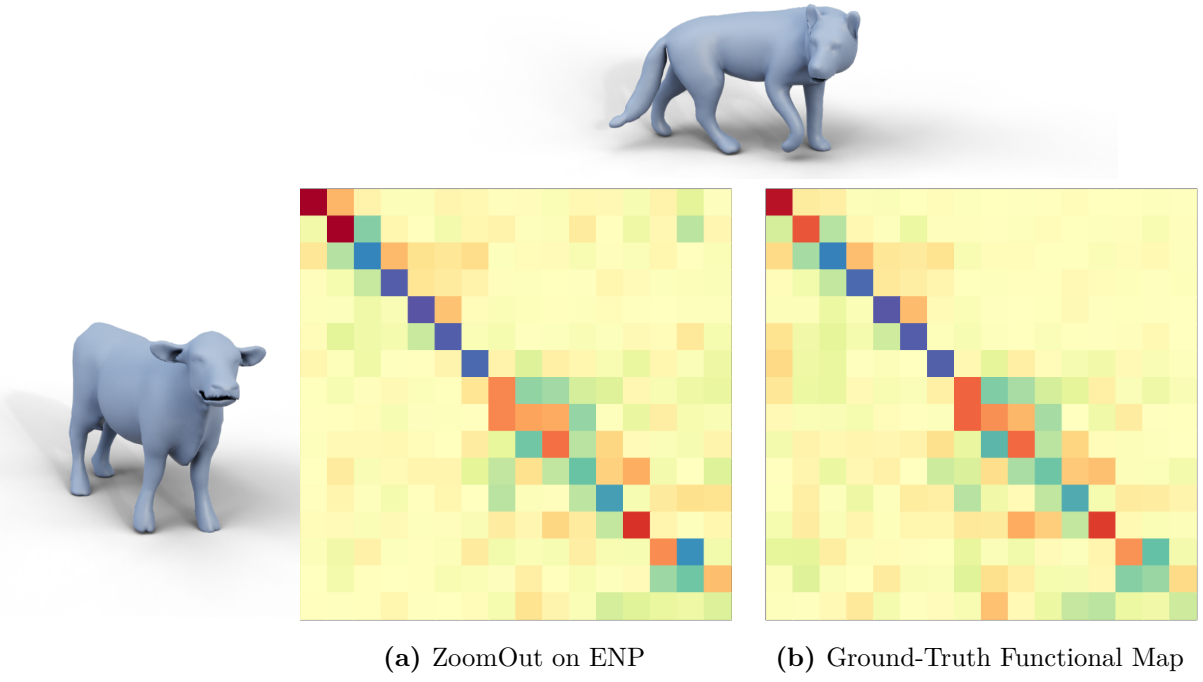


Figure 5.9: Side-by-side comparison of enhanced naive procrustes functional map estimation versus ground-truth of a non-isometric pair.

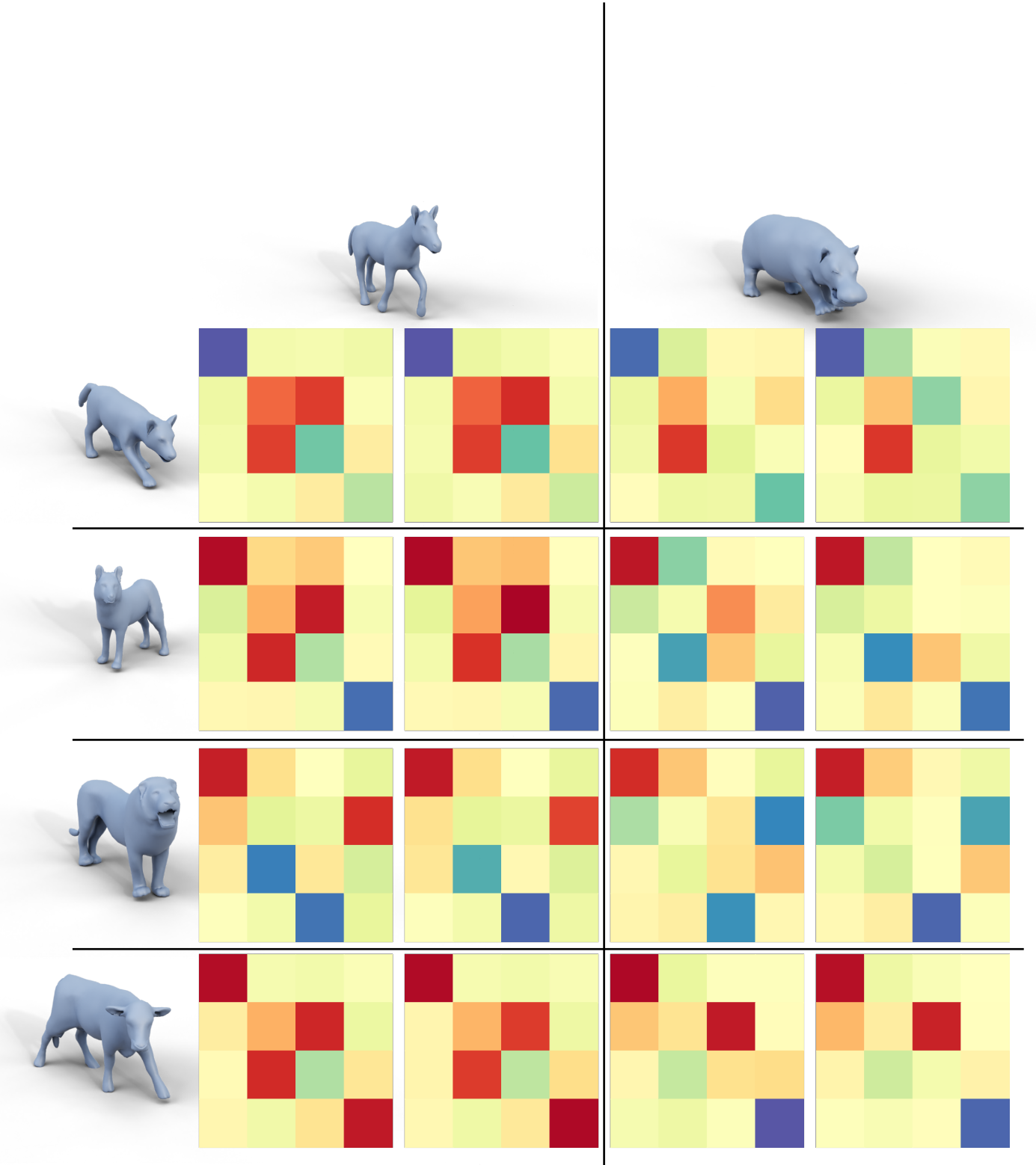


Figure 5.10: Further non-isometric functional map estimations on SMAL.

5. *IntrinsicFM: Unsupervised Intrinsic Operator Learning for Functional Maps*

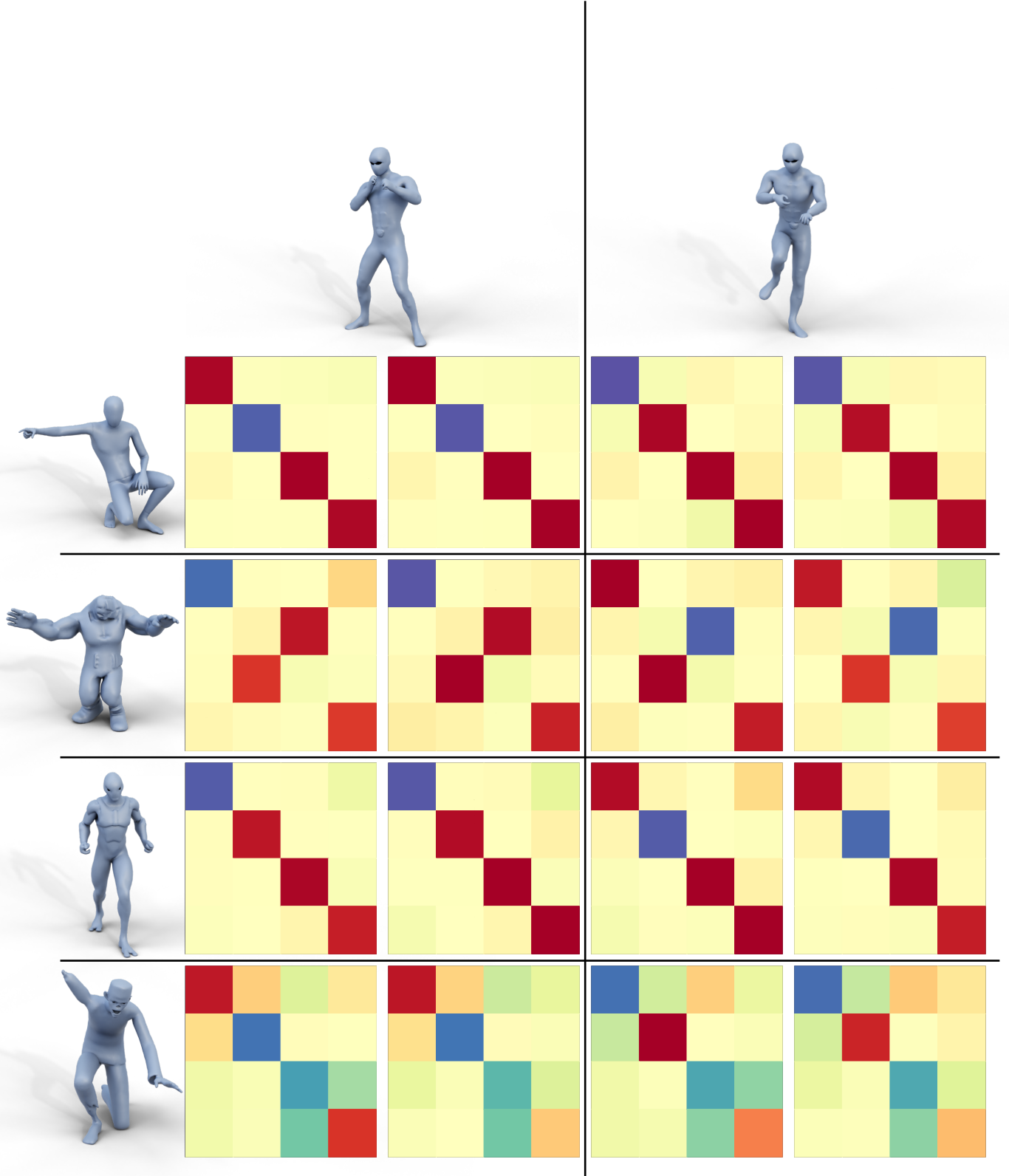


Figure 5.11: Further non-isometric functional map estimations on DT4D.

From Figures 5.7 to 5.7.2, it is evident that the introduced approach is robust enough to recover a low-rank functional map, which can be used to set up the learning pipeline. Some discrepancies in the coefficients of eigenvectors remain; however, sign ambiguities are all reliably resolved, which is the primary issue this approach aims to address.

5.7.3 Intrinsic Learning for Functional Maps

The concept of functional maps has been further extended into deep learning [35], enabling descriptors that predict the functional map between two manifolds. The functional map, $\mathbf{C}$, is solved by the following optimization problem:

$$\mathbf{C} = \arg \min_{\mathbf{C}} |\mathbf{C}\hat{\mathbf{F}} - \hat{\mathbf{G}}| \quad (5.7.3.1)$$

where $\hat{\mathbf{F}}$ and $\hat{\mathbf{G}}$ are features F and G projected onto the eigenbasis of the corresponding shapes. The dense point-to-point map is recovered using a nearest-neighbor approach. However, even when the ground-truth map $\mathbf{C}_{gt}$ is available, there is no guarantee that the nearest neighbor corresponds to the actual point, or to points in the 1-ring, 2-ring, or 3-ring spatial neighborhood. This issue is further exacerbated by near-isometric or non-isometric deformations between the compared manifolds. As a result, functional maps and the LBO eigenbasis are largely limited to applications with near-isometric deformations. Once more significant deformations occur, the functional map becomes decoupled from the point-to-point correspondence. To address this, we extend the learning to the LBO eigenbasis. By reducing the inherent loss caused by near-isometric and non-isometric deformations, we enforce a stronger linkage between the functional map and the point-to-point map through unsupervised learning from estimated corresponding points. We estimate a small functional map $\hat{\mathbf{C}}_{4 \times 4}$ using the previously introduced enhanced Procrustes approach, which we then feed into ZoomOut to obtain a full $\hat{\mathbf{C}}_{\mathbf{n} \times \mathbf{n}}$ along with the point-to-point maps $\mathbf{T}_{12}$ and $\mathbf{T}_{21}$. We then minimize the following:

$$\min_{\Psi} |\hat{\mathbf{C}}\Psi - \Phi[\mathbf{T}_{12}]|. \quad (5.7.3.2)$$

5. IntrinsicFM: Unsupervised Intrinsic Operator Learning for Functional Maps

To enforce symmetric performance improvement, we also minimize:

$$\min_{\Phi} |\Psi[\mathbf{T}_{21}] - \hat{\mathbf{C}}^T \Phi|. \quad (5.7.3.3)$$

This setup allows us to learn from the correspondences between the Ψ and Φ operators, which helps adjust the LBO for both near-isometric and non-isometric deformations. By using the point-to-point maps to index the corresponding eigenvectors, we remove the requirement for the shapes to have the same number of vertices, making this approach more flexible.

5.7.4 Accurate Functional Maps

To improve the accuracy of the functional map, we treat the eigenbases Φ and Ψ , whose columns consist of eigenvectors ϕ_i, ψ_i in ascending order, as pointwise descriptors. Learning an eigenbasis explicitly can be challenging due to issues such as sign flipping or eigenvector swaps. To address this, we use the estimated functional map $\hat{\mathbf{C}}_{\text{nxn}}$ to project the eigenbasis of one manifold onto the eigenbasis of the other. This produces a pointwise descriptor on a common basis, allowing us to optimize the descriptor through a loss function. This approach enables the learning process to make corresponding points more similar by backpropagating the pointwise loss to the eigenbasis. As a result, the eigenbasis adapts so that the correspondences via the functional map become more precise, bringing the point-to-point map and the functional map closer together. It is important to note that this learning does not necessarily make the manifolds more similar; rather, it makes their bases more suitable for the functional maps paradigm by accounting for non-isometries through the learned operators, which differs from the concept of *Isospectralization*. For the learning process, we employ triplet loss, which provides strong training signals. The triplet loss is formulated as:

$$\mathcal{L}(\Phi(x), \hat{\mathbf{C}}\Psi(y^+), \hat{\mathbf{C}}\Psi(y^-)) = \max(|\Phi(x) - \hat{\mathbf{C}}\Psi(y^+)|^2 - |\Phi(x) - \hat{\mathbf{C}}\Psi(y^-)|^2 + \alpha, 0) \quad (5.7.4.1)$$

where x represents points on the manifold with eigenbasis Φ , and y represents points on the manifold with eigenbasis Ψ . The symbols $+$ and $-$ denote similar

and dissimilar points, respectively, and α is the margin. To ensure symmetrical learning, we also optimize the reverse by transposing $\hat{\mathbf{C}}$:

$$\mathcal{L}(\Psi(x), \hat{\mathbf{C}}^T \Phi(y^+), \hat{\mathbf{C}}^T \Phi(y^-)) = \max(|\Psi(x) - \hat{\mathbf{C}}^T \Phi(y^+)|^2 - |\Psi(x) - \hat{\mathbf{C}}^T \Phi(y^-)|^2 + \alpha, 0), \quad (5.7.4.2)$$

Selecting negative samples for triplet loss can be challenging. We propose a hybrid approach that smooths the learning process by introducing rules based on the prediction quality at each vertex. Strong signals are crucial for triplet loss, but selecting negative points that are too close, or even identical in feature space, may cause the network to learn peculiar rather than generalizable patterns. To address this, we sample dissimilar points based on their proximity in spectral space and the quality of the prediction at each vertex. These rules promote smooth learning by delivering strong signals that progressively refine predictions toward the ground truth while preserving the accuracy of existing predictions. By aligning both spatial and spectral spaces, we ensure that the chosen dissimilar vertices are meaningful and relevant to the learning process.

5.7.5 Refining Functional Maps

The previously introduced learning mechanism allows the network to learn the geometry representation itself and better encode intrinsic distances, resulting in improved geodesic accuracy. However, this improvement stems from the eigenbasis quality rather than the map itself, as the ground-truth functional map retains its own residual error. The authors of [85] used the Steklov–Dirichlet approach to mitigate this type of error. The error is minimized through the formulation presented in the previous section. As the eigenbasis improves, we also refine the point-to-point map upscaled by ZoomOut after a certain amount of learning, allowing it to converge toward the newly better-aligned ground truth. During upscaling, the poor quality of the eigenbasis can lead to many false correspondences, especially at higher frequencies. We address this issue by pruning unstable points that lose their rank in the spectral space, even as the overall spectra of the shapes become closer. Points that become more dissimilar spectrally, despite effective learning, do not represent

5. *IntrinsicFM: Unsupervised Intrinsic Operator Learning for Functional Maps*

valid correspondences. Pruning these points allows the map to better explore ground-truth alignments during the learning process. Figure 5.12 illustrates how this pruning strategy can be applied effectively, resulting in a more accurate and refined point-to-point map. This effect has several consequences, as the estimated point-to-point

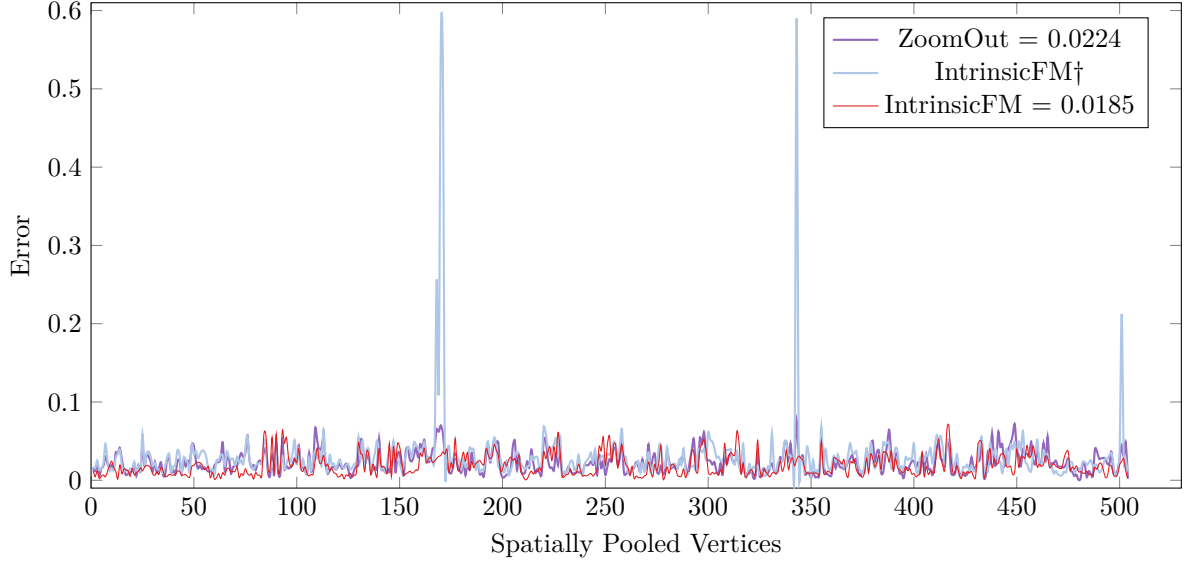


Figure 5.12: Pruning effects on the point-to-point map. To iteratively refine the correspondence and exploit improvements in the LBO eigenbasis, we identify drifting points, i.e., points whose spectral distance increases despite the overall enhancement of the eigenbasis. This apparent contradiction provides a useful signal: such points can be selectively pruned from the point-to-point map and reassigned to alternative matching candidates, enabling a more stable and accurate correspondence. The standard ZoomOut performance is shown in purple. The refined correspondence produced by IntrinsicFM is indicated by the red curve. The blue curve ($\dagger$) represents the spectrum induced by the initial ZoomOut point-to-point map under the updated LBO eigenbasis. It can be clearly observed that this prediction is no longer compatible with the modified eigenbasis, exposing false correspondences.

map gradually converges to the ground-truth over time. To reliably detect drifting points, we consider all points within a geodesic radius and monitor their behavior, since individual points can drift for reasons unrelated to correspondence errors. Additionally, if the map globally deteriorates or the underlying spectral coordinate system degrades, we disregard these drifting points, as their motion is then caused by overall worsening conditions rather than misalignment.

5.7.6 LBONet

We directly learn features derived from both intrinsic and extrinsic manifold properties, such as mean curvature, Gaussian curvature, shape index, and the HKS, using the three modules (RiemannNet, ALBO+Net, and VoronoiNet) introduced by LBONet in chapter 4. These modules significantly enhance the manifold’s representation by modifying the stiffness and mass matrices. Each module learns from mesh edges, faces, and vertices, respectively, with an ablation study demonstrating the performance gains achieved by each component.

- **RiemannNet** scales the Riemannian metric, allowing diffusion to slow down or speed up between vertices.
- **ALBO+Net** learns the anisotropic diffusion factor along both the *maximum* and *minimum principal curvature*, including their rotation.
- **VoronoiNet** weighs the Voronoi cells to emphasize or suppress specific regions of the manifold.

The learning capacity of LBONet is based on a combination of DGCNN’s neighborhood learning function [6] and PointNet’s [5] shared MLP architecture. This enables the learning of localized features using neighboring points, which is critical for correspondence learning, particularly in noisy regions around the 2-ring and 3-ring neighborhoods (see Figure 4.17).

5.7.7 RefineNet

ZoomOut’s performance depends on the success of its k-nearest neighbor (KNN) search on the other manifold. To achieve accurate results, the spectra of both manifolds need to be aligned. This is partly addressed by LBONet. However, to further improve the search, we leverage the LBONet eigenbasis gradient framework and introduce a specialized module, RefineNet, that computes the change in eigenbases with respect to vertex positions: $\frac{\partial \Psi}{\partial \mathbf{v}}$ and $\frac{\partial \Phi}{\partial \mathbf{v}}$. This additional information provides powerful guidance, as the core assumption of spectral correspondence

5. *IntrinsicFM: Unsupervised Intrinsic Operator Learning for Functional Maps*

requires that matched vertices exhibit symmetric responses to changes in their respective eigenbases. For example, geometrically prominent features, such as a nose tip, should excite analogous eigenvectors on both shapes, producing commensurate positional gradients. In chapter 4 detailed gradients are derived for the eigenbasis, including stiffness, mass, and combined stiffness and mass matrices were introduced. Therefore, we continue here directly with the gradients wrt. coordinates and skip the intermediate steps:

$$\frac{\partial A_i}{\partial v_i} = \frac{1}{8} \sum_{j \in N(i)} \left[2(v_i - v_j)(\cot \alpha_{ij} + \cot \beta_{ij}) + \|v_i - v_j\|^2 \left(\frac{\partial \cot \alpha_{ij}}{\partial v_i} + \frac{\partial \cot \beta_{ij}}{\partial v_i} \right) \right] \in \mathbb{R}^3 \quad (5.7.7.1)$$

with $N(i)$ the 1-ring neighbors of vertex i . For the stiffness matrix, the entries are calculated as

$$W_{ij} = \begin{cases} -\frac{1}{2} \sum_{T \ni i,j} (\cot \alpha_{ij} + \cot \beta_{ij}), & i \neq j \text{ and } (i, j) \text{ connected,} \\ -\sum_{k \neq i} W_{ik}, & i = j, \\ 0, & \text{otherwise.} \end{cases} \quad (5.7.7.2)$$

Using the vector-area formulation:

$$\frac{\partial W_{ij}}{\partial v_i} = \frac{1}{2} \sum_{T \ni i,j} \left(\frac{\partial \cot \alpha_{ij}}{\partial v_i} + \frac{\partial \cot \beta_{ij}}{\partial v_i} \right), \quad (5.7.7.3)$$

The diagonal entries follow from the row-sum property:

$$\frac{\partial W_{ii}}{\partial v_i} = \sum_{j \neq i} \frac{\partial W_{ij}}{\partial v_i}. \quad (5.7.7.4)$$

5.8 Evaluation

We evaluate our approach on established near-isometric correspondence benchmarks, including FAUST [78] and SCAPE [103]. Cross-dataset evaluation is performed by training the network on FAUST and evaluating on SCAPE, and vice versa, highlighting the generalization capabilities of our method across similar mesh structures. Compared to baseline LBO eigenbases, IntrinsicFM produces more consistent functional maps with reduced geodesic errors, as illustrated in Figure

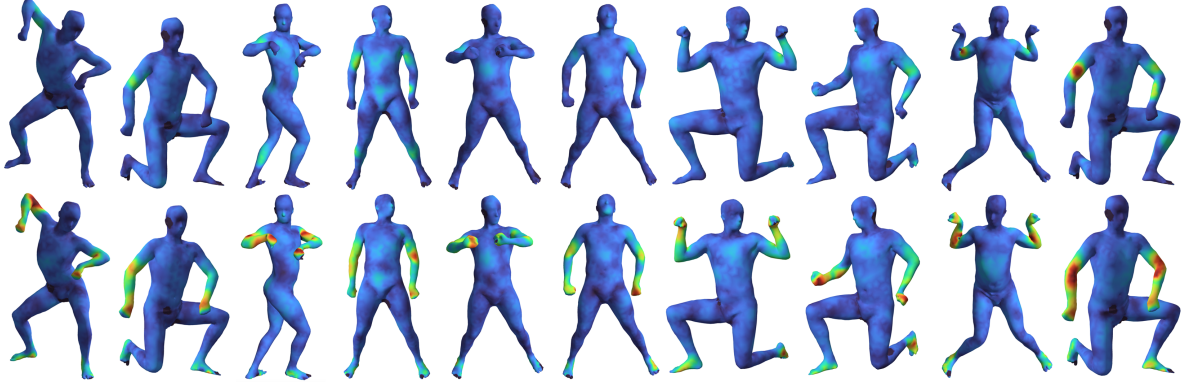


Figure 5.13: Near-Isometric Qualitative Results. IntrinsicFM trained on the SCAPE train set and evaluated on the SCAPE test set. The top row shows the IntrinsicFM Eigenbasis prediction, while the bottom row shows the standard LBO Eigenbasis. Pairs are rendered side by side for comparison. IntrinsicFM consistently improves correspondence accuracy across poses and manifolds. Topological changes, such as intersections, remain challenging (see the second figure from the right), likely due to reliance on spatial proximity during training, which can introduce false positives in intersecting meshes. Preprocessing the training data could help mitigate this limitation.

5.13. Beyond near-isometries, our method is tested in highly non-isometric settings on the SMAL [104] dataset and the DT4D-H (inter) [86] benchmark. These datasets include significant non-isometries across quadruped species, such as rhinos, cows, and dogs, as well as between more diverse creatures. Even in these challenging scenarios, our learned eigenbasis adapts effectively, improving functional map accuracy and point-to-point correspondence relative to standard LBO-based approaches. These results demonstrate that our framework is robust and generalizable across a wide range of shape variations.

Implementation Details For all experiments, we truncate the eigenbasis at λ_{64} and apply batch normalization. Although correspondence learning typically uses small batch sizes, which can make batch statistics less representative, we observe significant improvements in performance and generalization when using batch normalization with IntrinsicFM. This effect likely arises from the smoothing of the loss landscape. To mitigate the limitations of small batch sizes, we feed features from 30 randomly selected manifolds into LBONet but only use the features of the actual pair for training. This preserves the benefits of batch normalization while

5. *IntrinsicFM: Unsupervised Intrinsic Operator Learning for Functional Maps*

adding minimal memory overhead. We employ triplet loss with a margin $\alpha = 1.0$ and optimize the network using AdamW [105] with a learning rate of 0.0001.

To ensure transparency regarding IntrinsicFM’s contributions, we report performance in two ways:

1. **Eigenbasis Prediction:** We evaluate the eigenbasis predicted by IntrinsicFM using ground-truth correspondences. This measures the reduction in loss achieved by IntrinsicFM compared to the standard LBO. For reference, we also report the standard LBO’s performance. These results appear below the dashed lines in the tables.
2. **Functional Map Prediction:** We evaluate the predicted functional map using the full IntrinsicFM architecture shown in Figure 5.1, denoted simply as IntrinsicFM.

5.8.1 Near-Isometric Shape Correspondence

Table 5.1 presents our results, where IntrinsicFM outperforms existing approaches by a small margin. Specifically, we achieve a slight improvement over the standard LBO, which translates directly into better functional map predictions. Interestingly, although [36] jointly learns both the basis and feature extraction, it does not surpass the standard ground-truth LBO evaluated with 64 eigenfrequencies on the FAUST benchmark. Our method, however, achieves better performance than the standard LBO. We also observe strong generalization across datasets, as demonstrated by evaluating FAUST-trained models on SCAPE and vice versa. Contrary to common belief, spectral-domain weights transfer effectively, with both eigenbasis and functional map predictions maintaining high accuracy across datasets. This highlights a significant advantage of our spectral learning approach.

5.8.2 Non-Isometric Shape Correspondence

Our method achieves remarkable results in non-isometric settings. On the SMAL benchmark (Table 5.2), we observe a substantial reduction in geodesic mean error

5.9. Supervised vs. Unsupervised Learning

Method	FAUST	SCAPE	FonS	SonF
ACSCNN [66]	2.7	3.2	8.4	6.0
DOA-FM[86]	2.5	3.0	2.6	2.7
DiffusionNet [7]	1.9	2.4	2.4	1.9
Deep Shells [106]	1.7	2.5	2.7	5.4
Unsupervised [36]	1.6	1.9	2.2	1.6
IntrinsicFM	1.4	1.8	1.9	1.5
--- IntrinsicFM ---	1.2	1.2	1.3	1.3
LBO	1.5	1.5	N/A	N/A

Table 5.1: Geodesic Mean Error x 100 on the FAUST and SCAPE benchmark. All benchmarks were run on the remeshed versions of the meshes with ca. 5k vertices. Results below the dashed line show the ground-truth correspondence error with the given basis.

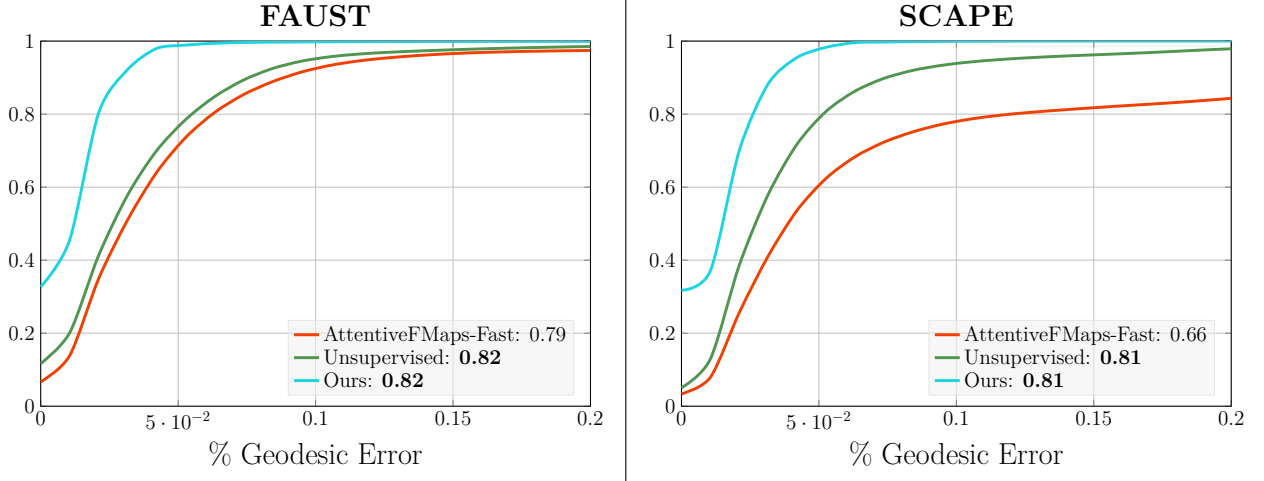


Figure 5.14: Near-isometric matching on (Left) SCAPE and (Right) FAUST. Our method substantially outperforms existing state-of-the-art methods.

compared to the standard LBO, nearly halving the ground-truth error. This improvement directly enhances functional map prediction, allowing our approach to outperform competing methods by a considerable margin. Similar gains are observed on the DT4D-H (inter) benchmark, demonstrating the robustness of IntrinsicFM to large deformations and diverse shape classes.

5.9 Supervised vs. Unsupervised Learning

We compare supervised learning, where ground-truth correspondences are available, with unsupervised learning, where no ground-truth is provided. Evaluations are

5. *IntrinsicFM: Unsupervised Intrinsic Operator Learning for Functional Maps*

Method	SMAL DT4D-H (inter)	
WSupFMNet [66]	7.6	22.6
GeomFMaps[86]	8.4	4.1
AttentiveFMaps [7]	5.4	11.6
Deep Shells [106]	29.3	31.1
Unsupervised [36]	3.9	4.1
IntrinsicFM	2.0	2.2
<hr style="border-top: 1px dashed black;"/>		
IntrinsicFM	1.3	1.6
LBO	2.5	2.6

Table 5.2: Geodesic Mean Error x 100 on SMAL and DT4D-H. Results below the dashed line show the ground-truth correspondence error with the given basis.

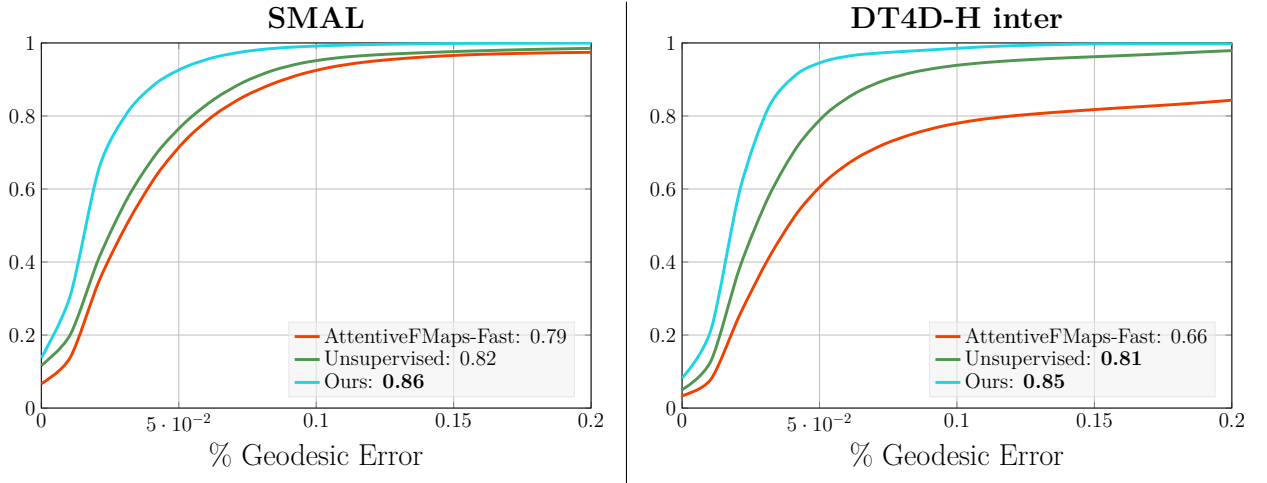


Figure 5.15: Non-isometric matching on (Left) SMAL and (Right) DT4D-H. Our method substantially outperforms existing state-of-the-art methods.

performed on both near-isometric and non-isometric pairs. We specifically selected challenging examples to highlight how IntrinsicFM operates at a fundamental level, effectively learning correspondences even without explicit supervision. The results of both supervised and unsupervised learning, as illustrated in Figures 5.18-5.27, may appear superficially similar. However, a closer examination of the derived weightings reveals where and how the increased accuracy is achieved. We first consider the near-isometric case and compare the supervised signal with the unsupervised signal. In Figure 5.18, the learned Voronoi weighting is shown. Both signals share many similarities, yet the supervised signal in panels (a) and (b) appears more balanced and nuanced. For instance, finer regions such as the feet



Figure 5.16: Spectral sensitivity computed by RefineNet. Although this is a high-dimensional feature, it provides valuable insight into the richness and complexity of the representation. Spectral sensitivity addresses the question of how infinitesimal changes in vertex positions affect the LBO eigenbasis. This information can be leveraged to refine functional maps, assist in their upscaling, and improve initial functional map estimation. While many regions on a manifold may appear spectrally similar, their influence, meaning the gradient of the eigenbasis with respect to vertex perturbations, can differ significantly. Capturing this sensitivity is therefore crucial for disambiguating such regions and enabling more precise correspondence localization.

and hands are more pronounced compared to the unsupervised signal in panels (c) and (d). This likely results from the ground-truth signal providing more precise feedback in regions with higher loss, such as small body parts like fingers and toes. In contrast, the unsupervised signal is present but weaker. In Figure 5.19, when learning the first anisotropic factor, a larger difference between the signals emerges. Although the regions are largely consistent, the intensity varies significantly and is partially reversed (e.g., the head in (a) and (b) versus (c) and (d)). This may be attributed to differences in learning dynamics. Nonetheless, the consistent boundaries and regions indicate that the underlying learning signal is similar, albeit slightly muted in the unsupervised setting.

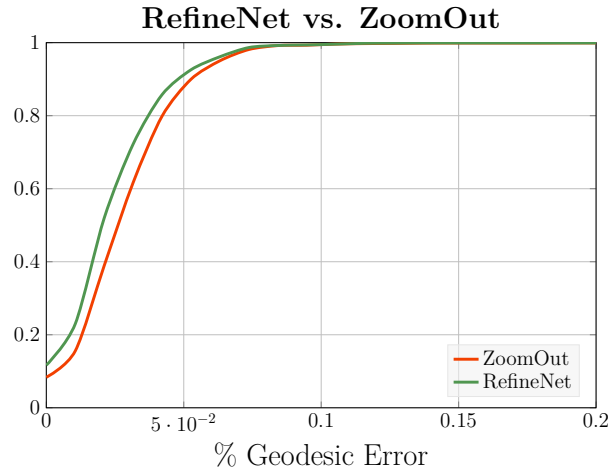
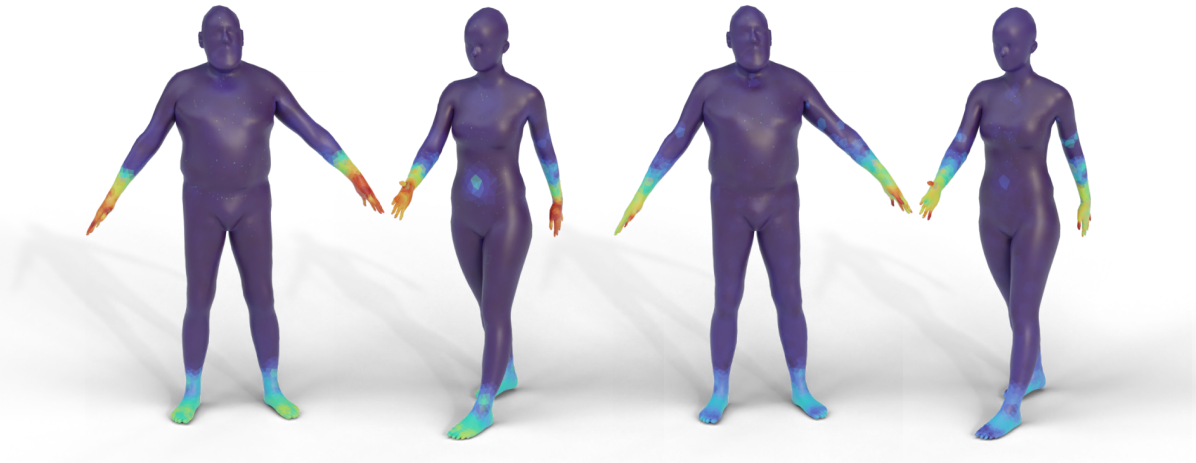


Figure 5.17: RefineNet vs. ZoomOut. When RefineNet is employed to further enhance k-nearest neighbor search, it consistently outperforms the standard ZoomOut method.

Voronoi Weighting



(a) Supervised A (b) Supervised B (c) Unsupervised A (d) Unsupervised B

Figure 5.18: Comparison of supervised and unsupervised Voronoi results.

Figure 5.20 further emphasizes the similarity between the supervised and unsupervised signals. The unsupervised signal in panels (c) and (d) appears smoother or slightly blunted compared to panels (a) and (b). For example, the armpit regions in (a) and (b) show improved detail, reflecting areas where the functional map struggles to align points accurately. In the unsupervised case, this detail is less pronounced, indicating a loss of fine-grained information. Nevertheless,

Anisotropic factor α_1



Low Medium High



(a) Supervised A (b) Supervised B (c) Unsupervised A (d) Unsupervised B

Figure 5.19: Comparison of supervised and unsupervised Anisotropy factor α_1 results.

this smoothing in the unsupervised signal may act as a form of regularization, potentially aiding generalization.

The rotation parameter shown in Figure 5.21 highlights additional differences between the supervised and unsupervised approaches. Although the overall pattern appears similar, regional discrepancies are evident. In the unsupervised signal, almost no rotation is applied to the stomach and shoulders, whereas in the supervised signal, these regions retain more rotation. This further demonstrates that the unsupervised signal is slightly less accurate in capturing fine-grained details.

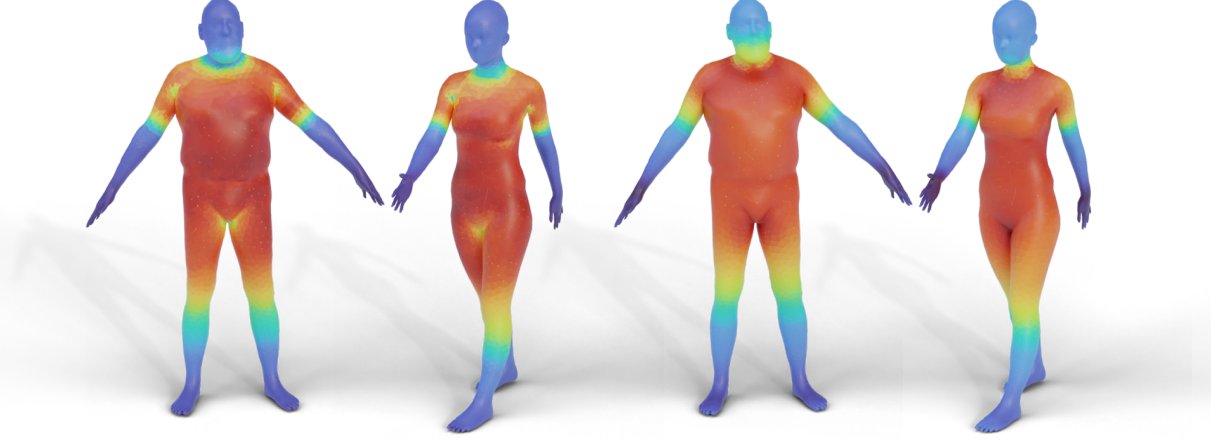
The final figure 5.22 for the near-isometric case illustrates the overall effect of the learned weightings. While it is difficult to precisely explain why certain weightings improve correspondence quality especially in the LBO eigenbasis, where the effects of Voronoi, anisotropy, or Riemannian metric weightings are not directly interpretable some trends can be observed. The figure highlights regions where geodesic accuracy was improved. Both the supervised and unsupervised signals target the same geodesic errors, demonstrating that the estimated functional map had sufficient

5. *IntrinsicFM: Unsupervised Intrinsic Operator Learning for Functional Maps*

Anisotropic factor α_2



Low Medium High



(a) Supervised A (b) Supervised B (c) Unsupervised A (d) Unsupervised B

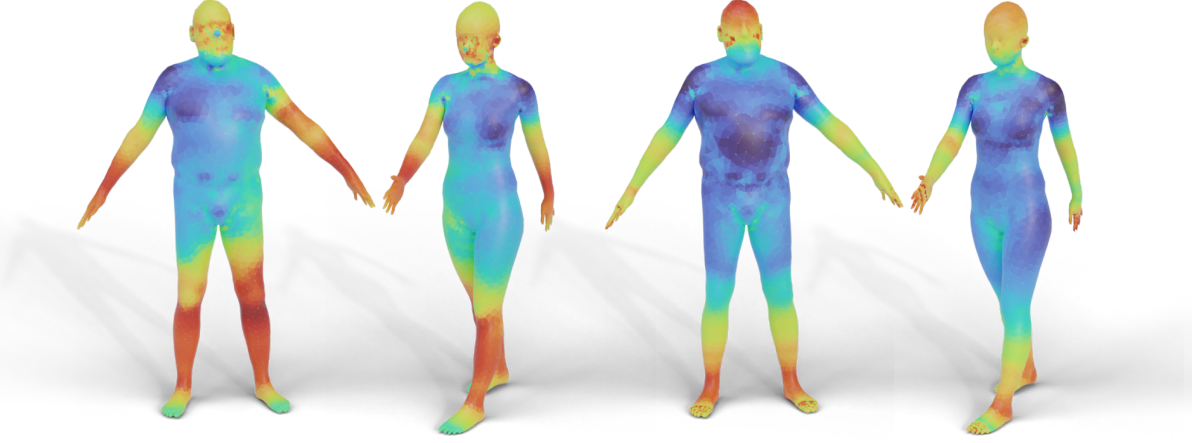
Figure 5.20: Comparison of supervised and unsupervised Anisotropy factor α_2 results.

fidelity to reveal these regions. However, the unsupervised signal achieves smaller gains, as the improvements are not as strong as those seen in (a) and (b).

From Figure 5.21 (a), one can deduce that some of the improvements observed in Figure 5.22 (a) are attributable to specific local effects. The three distinct dots in the lower torso correspond closely to the observed correspondence gains. Additionally, the activity in the faces, shown in Figures 5.21 (a) to (d), suggests that the rotation parameter may have contributed to the geodesic accuracy improvements in these regions. Consequently, all modules, RiemannNet, AnisotropyNet, and VoronoiNet, appear to learn a stable basis, with the rotation parameter providing the final adjustment necessary to fully realize the accuracy gains. The weightings remain surprisingly similar in the non-isometric case for VoronoiNet, as shown in Figure 5.23. The feet of both animals are highlighted, but the effect is stronger in the supervised case, illustrated in (a) and (b). In (c), there is more activity on the cow’s head, whereas in (d), the activity is reduced compared to (b). Compared with the near-isometric case in Figure 5.18, it is noteworthy that the head shows much more activity in the non-isometric case. This can be attributed to the larger

Anisotropic Rotation θ 

Low Medium High



(a) Supervised A (b) Supervised B (c) Unsupervised A (d) Unsupervised B

Figure 5.21: Comparison of supervised and unsupervised Anisotropy Rotation θ results.

deformations and greater errors caused by the substantial differences in the size and shape of a cow and wolf head compared to the difference between male and female human heads. We can conclude that VoronoiNet increases weights in regions where geodesic error occurs. Furthermore, the unsupervised signal remains stable, although, as in the near-isometric case, it appears blunted and smoothed.

The anisotropic factor α_1 in Figure 5.22 aligns closely with the observations from Figure 5.20. Both the supervised and unsupervised signals cover the same regions, but the unsupervised signal is noticeably smoother and exhibits less variation within these areas. It is worth noting that the learned anisotropy highlights the same regions for humans as for animals. This demonstrates that the network consistently interprets legs, arms, and heads in a similar way, regardless of the subject, suggesting a fundamental learning pattern, as briefly discussed in the introduction to this thesis.

The anisotropic factor α_2 generally aligns with the patterns observed in earlier anisotropy illustrations. Interestingly, there is less activity in (a), which is also apparent in Figure 5.19 for the unsupervised case. This suggests that the reduced

5. *IntrinsicFM: Unsupervised Intrinsic Operator Learning for Functional Maps*

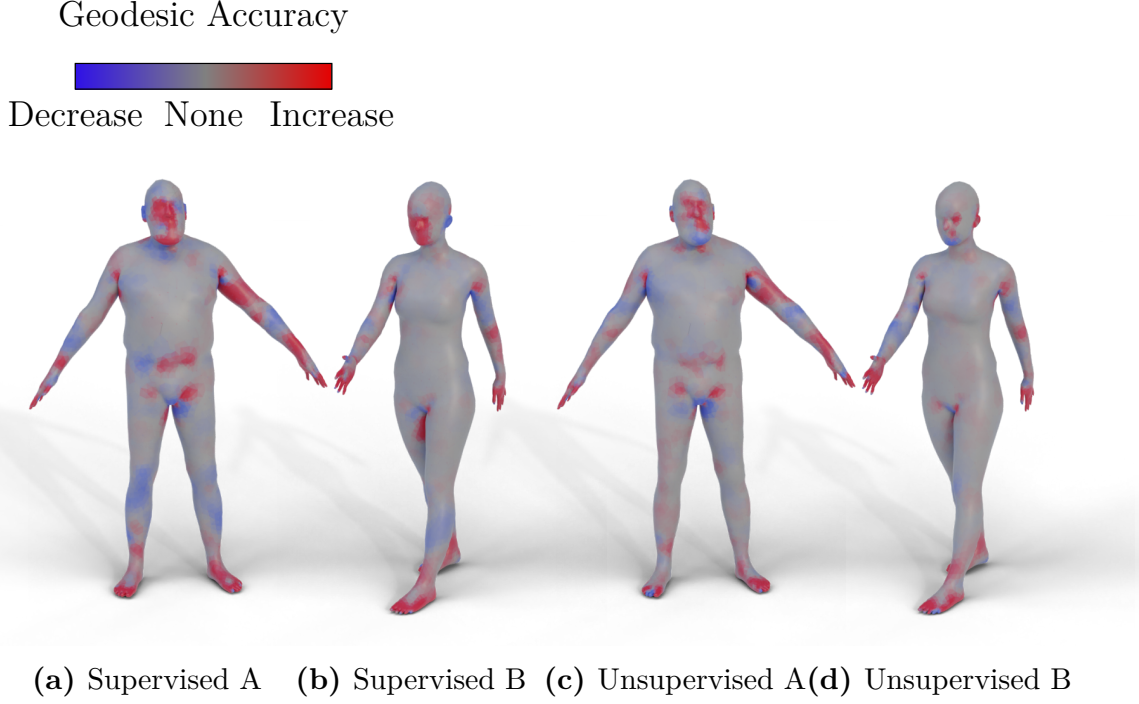


Figure 5.22: Comparison of supervised and unsupervised geodesic error.

weightings may result from the learning dynamics and the complex interactions between the operators.

The largest discrepancy in the signal appears in Figure 5.26. Here, the unsupervised signal seems trapped in a local optimum, as it mostly contains saturated values, either full rotation or almost no rotation, with very few intermediate values. This phenomenon is difficult to explain definitively, but it may result from problematic gradients during learning, which caused a large update that the network could not recover from.

In the last figure 5.27, the effects of the weightings are visible, revealing two surprising observations. First, the unsupervised signal reduces a significant amount of geodesic error on the cow’s nose (c) compared to (a), whereas the supervised signal actually loses some accuracy in this region. Second, the chest of the cow shows decreased accuracy in the unsupervised method, while it is improved in the supervised signal. In other words, the chest and mouth/nose regions show opposite patterns of improvement. This finding could have multiple explanations. One possibility is that the network balances errors, so improving the chest may

Voronoi Weighting

 Low Medium High



(a) Supervised A (b) Supervised B (c) Unsupervised A (d) Unsupervised B

Figure 5.23: Comparison of supervised and unsupervised Voronoi weighting results

cause the mouth/nose accuracy to decrease, and vice versa. Another possibility is that noise in the ground truth prevents the supervised network from reducing the error on the mouth/nose, whereas the unsupervised signal smooths out the noise and achieves this gain. Interestingly, the upper legs of the wolf in (d) also appear to improve compared to the supervised wolf (b), highlighting a more nuanced difference between the two signals.

The near-isometric and non-isometric cases, learned both supervised and unsupervised, revealed many hidden behaviors of LBONet, functional map estimation, and ZoomOut. While the supervised signal appears stronger, the geodesic accuracy gains show a more nuanced picture. Both the cow and the wolf have regions where the unsupervised signal improves performance, whereas the supervised signal sometimes decreases it. With supervision, the network focuses on aligning the signal to the ground-truth vertices. This can be beneficial but also has downsides. Ground-truth data may contain artifacts, noise, or other defects, which the network may attempt to learn forcefully. This issue is particularly evident in the non-isometric case. In contrast, the unsupervised method produces a smoother signal,

Anisotropic factor α_1



Low Medium High



(a) Supervised A (b) Supervised B (c) Unsupervised A (d) Unsupervised B

Figure 5.24: Comparison of supervised and unsupervised Anisotropic factor α_1 results.

resulting in a more robust prediction even when the superficial performance of both methods is similar. Moreover, because the unsupervised method requires no labels, its results are arguably more valuable and suggest that the unsupervised signal is the superior approach.

5.10 Conclusion and Limitations

We have shown that the LBO basis can be effectively extended to both near-isometric and non-isometric scenarios, contrary to common belief. This enables the creation of a differentiable learning pipeline using an estimated functional map and ZoomOut. It is worth to mention that no descriptors were used at any time in the full pipeline. The accuracy gains achieved by an estimated map, which improves over time due to two reasons.

- **Improved spectral coordinate system:** The system accounts for non- and near-isometric deformations of the shapes. This is particularly evident when the ground-truth map is available, as shown in Table 5.2, where the

Anisotropic factor α_2 

Low Medium High



(a) Supervised A (b) Supervised B (c) Unsupervised A (d) Unsupervised B

Figure 5.25: Comparison of supervised and unsupervised Anisotropy factor α_2 results.

error is nearly halved. This demonstrates that the standard LBO eigenbasis is far from optimal and that IntrinsicFM can significantly improve upon it. While these numbers reflect the current technical limitations of IntrinsicFM, performance could likely be enhanced by extending its backend LBONet with additional modules tailored to correspondence tasks. This technical ceiling, imposed by IntrinsicFM, can be further raised if the following aspects are improved.

- **Interaction between functional and point-to-point maps:** IntrinsicFM ensures that the point-to-point map becomes aware of its underlying spectral basis by monitoring drifting correspondences, even as the basis improves. Such drifts indicate false correspondences, which can then be discarded and replaced with better alternatives. However, the point-to-point map still lags behind the true potential of the eigenbasis because these drifts must be observed over several iterations before they can be fully corrected from the initial estimation.

Consequently, any method seeking to improve shape correspondence must address

Anisotropic Rotation θ



Low Medium High



(a) Supervised A (b) Supervised B (c) Unsupervised A (d) Unsupervised B

Figure 5.26: Comparison of supervised and unsupervised Anisotropy Rotation θ results.

both of these challenges, as demonstrated by [36] using arbitrary bases alongside a highly accurate point-to-point map. The approach presented in this work provides a competitive alternative by leveraging the LBO and benefiting from its inherent advantages, while simultaneously mitigating its limitations. Both aspects offer avenues for further research and potential improvement.

Despite these advances, partial shape correspondence remains an open challenge when used with LBO. Our experiments indicate that the eigenbasis of partial shapes is highly unstable, which hinders effective training and yields minimal accuracy gains. Methods like [107] might pave the future for these use cases.

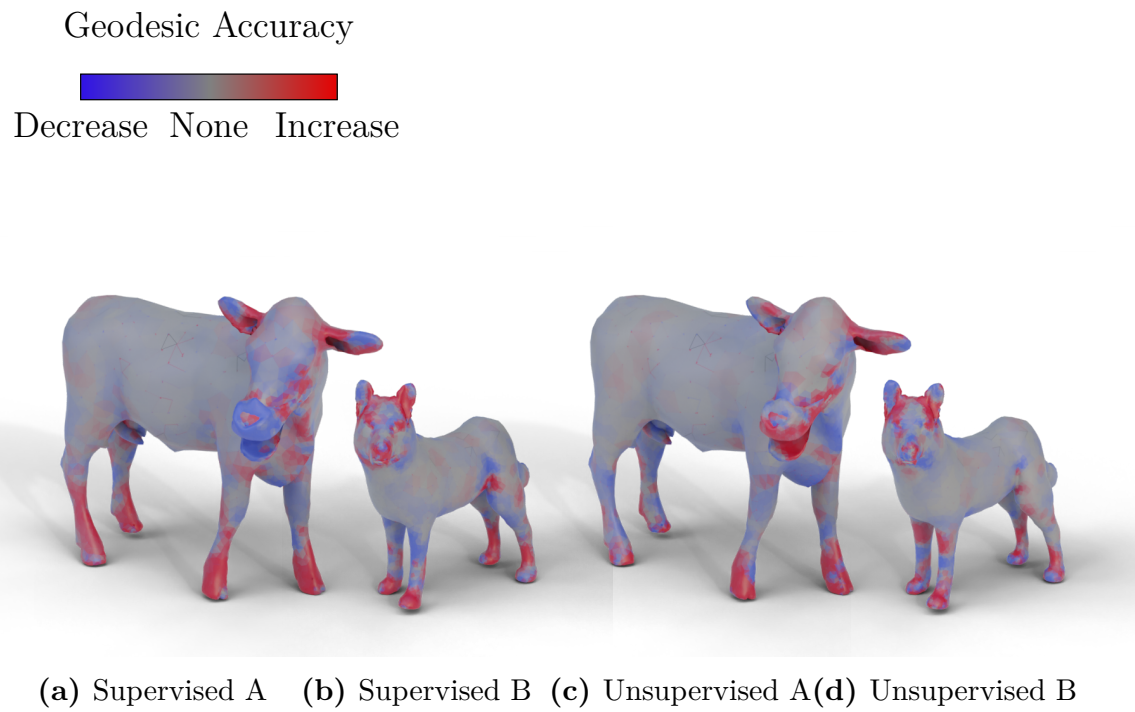


Figure 5.27: Comparison of supervised and unsupervised geodesic error.

"Bekanntlich setzt die Geometrie sowohl den Begriff des Raumes, als die ersten Grundbegriffe für die Constructionen im Raume als etwas Gegebenes voraus." [60]

Bernhard Riemann

6

Conclusions

6.1 Intrinsic and Extrinsic Features for Spectral Learning

In this section, we analyze the distribution and stability of input features, as these characteristics directly affect the efficiency and convergence of learning. While a neural network is capable of adapting to features across multiple scales, highly variable or noisy features can significantly slow down training and make convergence more difficult. Given the computational cost of backpropagation, it is therefore crucial to provide features that are as clean and well-behaved as possible, ensuring that gradient signals propagate effectively through the network. Beyond feature scaling and distributional considerations, the design of the loss function plays a crucial role in encouraging the learning of an improved eigenbasis. In particular, the loss function must be carefully constructed to prevent LBONet from exploiting shortcut solutions or converging to superficially good but uninformative optima. For instance, when learning to enhance the descriptiveness of the HKS, proper normalization is essential to ensure genuine descriptor discriminativity. Without normalization, LBONet could exploit feature magnitudes by arbitrarily scaling the manifold weights. For example, the Riemannian metric could be trivially

increased by a constant amount at each edge, allowing a loss function that enforces descriptor similarity, but lacks sufficient regularization, to be satisfied without learning meaningful geometric structure. Although this issue could alternatively be addressed by using the scale-invariant SI-HKS, empirically normalizing the HKS proves to be equally effective.

6.2 Optimization in Non-Smooth Spectral Landscapes

In Chapter 3, we analyzed the non-smooth search space associated with the learned LBO eigenbasis. Effectively navigating this landscape requires using a sufficiently large learning rate to escape local minima. The largest of these local minima corresponds to the default LBO, making it essential to initialize the network with appropriately large weights to prevent the optimization from being dominated by this trivial solution. Once the network converges to a favorable LBO eigenbasis, it demonstrates strong generalization to other shapes and benchmarks, provided that the feature distributions are similar and extrinsic features are avoided, while additional data exposure yields diminishing improvements. This is partly due to the relatively simple neural network used throughout all experiments for training and to the tendency of the LBO operator to converge rapidly into a basin. Throughout all experiments, batch normalization was applied to accelerate training, as it can provide an order-of-magnitude speedup. However, tasks such as correspondence learning are typically performed using only a single pair of shapes. Even when learning with shared neural networks on vertices, faces, or edges, batch normalization led to unstable training and proved ineffective. To address this, multiple pairs of shapes are processed in each batch, but backpropagation is performed only through the first pair. This strategy allows the network to benefit from the stabilizing effects of batch normalization while ensuring that learning occurs exclusively through the primary shape pair.

6.3 Contributions

The contributions of this work aim to improve our intrinsic understanding of shapes. Achieving this required overcoming multiple challenges and introducing novel techniques for learning and adapting intrinsic geometry.

A central contribution of this thesis is the demonstration that the Laplace–Beltrami operator and its eigenbasis can be optimized for a given task rather than treated as fixed geometric quantities. This is explored through both neuroevolutionary and gradient-based optimization, showing that learned weighting schemes can improve the resulting spectral descriptors while remaining applicable to high-resolution meshes.

A second contribution is the development of a framework for learning modifications of the underlying manifold geometry. By learning weightings of the Riemannian metric, anisotropic diffusion properties, and Voronoi mass matrix, the geometry itself can be adapted to the data. Critical geometric constraints ensure that these learned modifications preserve a valid mesh structure while improving the quality of the resulting eigenbasis.

A third contribution is the development of an unsupervised functional map framework. By leveraging stable low-rank functional map estimation and topological constraints, correspondence information can be estimated without relying on ground-truth correspondences. This estimated correspondence serves as a learning signal, enabling the optimization of intrinsic geometric operators in an unsupervised setting.

Finally, this work demonstrates that learning an adapted intrinsic geometry improves both the quality of the spectral coordinate system and the resulting shape correspondences. The learned operators are more robust to non-isometric deformations, allowing functional maps to better capture meaningful geometric structure while improving correspondence accuracy.

These contributions are integrated into a unified end-to-end pipeline, where the proposed techniques jointly learn an optimized Laplace–Beltrami eigenbasis

and refine functional maps. The resulting framework demonstrates that adapting intrinsic geometry through learning can substantially improve both shape representation and correspondence.

6.4 Results and Insight

Contrary to some prevailing understanding that the LBO eigenbasis may be inadequate for non-isometric shape correspondence, our work shows that its global nature is not a fundamental limitation. Other methods such as [95] or [107] were introduced to combat the global nature thereof. By introducing a suite of learning-based modules, we make the LBO eigenbasis fully adaptable, enabling the derivation of task-specific spectral descriptors and even directly refining the LBO eigenvectors themselves. This demonstrates that the LBO eigenbasis can be elevated to a level of accuracy previously thought unattainable.

Beyond correspondence, part segmentation is also enhanced. Prior works focused on spectral descriptors for their smoothness, discriminativity, or stability, but LBONet introduces separability as an emergent property. By learning the LBO eigenbasis to produce more linearly separable features, downstream networks can segment parts more effectively.

Finally, for global tasks such as classification and retrieval, LBONet aligns the LBO eigenbasis within classes and amplifies discriminative power where the backend requires it most. Unlike standard approaches such as PointNet++ [8], whose performance is limited by the descriptors provided, LBONet adapts the spectral descriptors precisely to the needs of the backend. This transforms the LBO eigenbasis from a general-purpose tool into a finely tuned instrument, demonstrating that learning the underlying operators can yield measurable improvements across multiple shape analysis tasks.

6.5 Limitations

There are several clear limitations that should be explicitly addressed, as they form key areas for future research in LBO methods. First, partial or corrupted meshes (e.g., those with cuts or holes) present a significant challenge for spectral analysis. Although geometric operators such as anisotropic factors generally have a local impact, the LBO is inherently a global operator (as it is a result of the GEP): changes at a single face, such as at one end of an object, can influence the eigenbasis across the entire mesh. This global behavior means that missing regions (whether small holes or large portions of the mesh) disrupt the entire LBO eigenbasis, resulting in significant spectral changes.

While the presented work narrows the gap in non-isometric cases, partiality and missing data present a much more formidable challenge. To align the spectrum of two meshes with missing data, LBONet must learn to compensate for the missing region, a task that can work in correspondence learning when the missing regions are consistent across the training set. However, generalizing this alignment to meshes with different missing regions becomes much more complex, as the learned spectrum would need to account for entirely different geometric features depending on the location and scale of the missing parts. This variability in missing data makes it difficult to develop a robust, generalized solution.

6.6 Future Work

While LBONet focuses on improving the LBO by learning task-adaptive geometric operators, this framework could be further generalized by incorporating the Hodge star operators used in Discrete Exterior Calculus, as done in HodgeNet, as an additional learning stage.

In such a setting, LBONet would first learn how the underlying geometry should be modified in order to produce a task-adaptive LaplaceBeltrami eigenbasis. Since LBONet maintains geometric validity throughout training, the learned metric can be used to construct consistent DEC operators, including Hodge stars and Hodge

Laplacians. A HodgeNet-style architecture could then be built on top of these operators to process signals defined on different geometric entities.

Although LBONet makes use of vertices, edges, and faces when constructing its operators, it does not model interactions between signals living on different form degrees. While such interactions could, in principle, be introduced through additional internal neural network components, a cleaner and more principled approach is to integrate LBONet with a DEC-based framework, such as HodgeNet, which explicitly supports cross-form interactions. This integration has the potential to enhance the overall framework more effectively than simply adding modules, which may eventually lead to diminishing returns, while simultaneously improving performance across all experiments and benchmarks.

Appendices



Gradient Verification

A.1 Numerical-Analytical Gradient Verification

To validate the correctness of the proposed end-to-end approach, we perform an *analytical-numerical gradient verification*, in which the previously derived analytical gradients are compared against their numerical counterparts. This procedure ensures that the underlying *toolkit* forming the backbone of *LBONet* is both robust and rigorous, thereby simplifying downstream debugging by establishing trust in the core gradient computations. The numerical gradients are computed using the standard finite-difference formulation:

$$g_{\text{num}} = \frac{f(x + \epsilon) - f(x - \epsilon)}{2\epsilon}. \quad (\text{A.1.0.1})$$

Where $f(x)$ can be any layer and therefore the verification can be performed at any intermediate stage of the computational pipeline. For example, we could take Eq. (4.5.1.2) and put it as $f(x)$ and test the gradients of the spectral descriptor. All gradients presented in this work have been meticulously derived and independently verified step by step to ensure the correctness of each component. In the following, we illustrate a representative subset of these checks, one for each stage of the computation, while noting that substantial effort was dedicated to addressing edge cases, inconsistencies in triangle or vector orderings, and other numerical subtleties

A.1. Numerical-Analytical Gradient Verification

during development. All benchmark meshes employed throughout this work have been subjected to these validation procedures; nevertheless, it is acknowledged that real-world meshes may still exhibit complexities beyond those fully resolved by the presented framework. When working with previously unseen data, it is advisable to perform gradient checking to ensure the robustness of the learning pipeline. This step helps identify issues such as broken or inconsistent meshes, which could otherwise interfere with the learning dynamics and, in the worst case, prevent a sensitive learning method such as the one presented here from converging at all. It will demonstrate that the spectrum of the given meshes are useful and their spectrum can be used robustly for backpropagation. However, at this stage it is essential to distinguish between inaccuracies arising from gradient computation and those stemming from unfavorable mesh characteristics. For instance, a pronouncedly sharp edge can distort both the local feature distribution and the resulting spectral descriptor if it originates from a mesh-related artifact. Consequently, passing gradient checks does not guarantee successful learning, but it acts as a prerequisite and ensure that the given mesh does not exhibit bigger defects such as disjoint parts or holes. For all gradient checks presented in the following, we enable 64-bit computation, ensuring that all calculations are carried out with higher numerical precision. Furthermore, we enforce the intermediate calculations to be carried out in 64 bit (PyTorch otherwise might default to 32 bit in intermediate steps to save resources):

```
1 torch.backends.cudnn.allow_tf32 = False
2 torch.backends.cuda.matmul.allow_tf32 = False
```

Listing A.1: PyTorch setting to carry out intermediate calculations with 64 bit precision

Additionally, we disable gradient clipping, which can smooth the learning dynamics but is detrimental for accurate gradient verification.

A.1.1 Operators Check

In the first verification step, we focus on the geometric operators and their associated gradients. These operators were derived and verified in a bottom-up manner,

A. Gradient Verification

beginning with the smallest building blocks and progressively extending to the full formulation. This end-to-end validation ensures that all gradients, from the initial expression in Eq. 2.1.4.4 to the final gradients introduced in Section 4.3, are computed correctly. For simplicity, we perform separate gradient checks for the stiffness and mass matrices.

Mass matrix

```
1 from torch.autograd import gradcheck
2
3 x.requires_grad = True
4 a1.requires_grad = True
5 a2.requires_grad = True
6 v1.requires_grad = True
7 Theta.requires_grad = True
8 spectre = Geo.apply
9
10 # Prepare input tuple for gradcheck
11 input = (
12     self.v.unsqueeze(0),
13     self.edges.unsqueeze(0),
14     self.f.unsqueeze(0),
15     x.unsqueeze(0).unsqueeze(0),
16     el.unsqueeze(0),
17     ts1,
18     corners1,
19     minCurvature,
20     maxCurvature,
21     torch.tensor(rn.transpose(1,0)).unsqueeze(0),
22     a1.unsqueeze(0),
23     a2.unsqueeze(0),
24     Theta.unsqueeze(0),
25     v1.unsqueeze(0).unsqueeze(0)
26 )
27
28 # Run gradcheck
29 test = gradcheck(spectre, input, eps=1e-5, atol=1e-4, rtol=1e-3,
30                 fast_mode=True)
31 print("PyTorch Gradcheck Geometric Operators Test: ", test)
```

Listing A.2: Gradcheck for Geometric operators in PyTorch

Note that the 2nd, 3rd, and 4th gradients correspond to ALBONet. Since the

A.1. Numerical-Analytical Gradient Verification

ALBONet operator does not modify the mass matrix, these gradients remain zero.

$$\text{Layer: } [Riemann, ALBO\alpha_1, ALBO\alpha_2, ALBO\phi, Voronoi] \quad (\text{A.1.1.1})$$

$$\text{Analytical gradient: } [0.0011, 0.0, 0.0, 0.0, 3.2622 \times 10^{-5}] \quad (\text{A.1.1.2})$$

$$\text{Numerical gradient: } [0.0011, 0.0, 0.0, 0.0, 3.2549 \times 10^{-5}] \quad (\text{A.1.1.3})$$

$$\text{Relative error: } [0.0, 0.0, 0.0, 0.0, 1.12 \times 10^{-3}] \quad (\text{A.1.1.4})$$

The small added perturbation is indicated by the red font in the subsequent equations. The first perturbation is added to the exponent ^{.2} of the diagonal term of the mass matrix equation of RiemannNet.

$$\frac{\partial A_{ii}}{\partial d_{ij}} = \frac{1}{8} \left(\left(-\frac{d_{ij}^{3.2}}{\sin^2(\alpha_1)2T_1} + \cot(\alpha_1)2d_{ij} \right) + \left(\frac{d_{jk}^3 \cos(\gamma_1)}{\sin^2(\delta_1)2T_1} \right) \right) \quad (\text{A.1.1.5})$$

$$- \left(\frac{d_{ij}^3}{\sin^2(\alpha_2)2T_2} + \cot(\alpha_2)2d_{ij} \right) + \left(\frac{d_{jk}^3 \cos(\gamma_2)}{\sin^2(\delta_2)2T_2} \right) \quad (\text{A.1.1.6})$$

It results in a drastic deviation of the RiemannNet gradient. The relative error increases from 0.0 in A.1.1.1 to 8.33×10^{-2} in A.1.1.7, which validates the correctness of the presented equation:

$$\text{Analytical gradient: } [0.0013, 0.0, 0.0, 0.0, 3.2622 \times 10^{-5}] \quad (\text{A.1.1.7})$$

$$\text{Numerical gradient: } [0.0011, 0.0, 0.0, 0.0, 3.2549 \times 10^{-5}] \quad (\text{A.1.1.8})$$

$$\text{Relative error: } [8.33 \times 10^{-2}, 0.0, 0.0, 0.0, 1.12 \times 10^{-3}] \quad (\text{A.1.1.9})$$

Note that the Voronoi gradient is not affected by this perturbation. It however illustrates the noise incurred by the cotangent weighting scheme, which is prone to small rounding errors.

Stiffness matrix

Next, we verify that any modification affecting the stiffness matrix is correctly handled. As discussed earlier, both ALBONet and RiemannNet influence the stiffness matrix, and these layers interact with each other, potentially producing

A. Gradient Verification

coupled effects that must be accounted for.

$$\text{Analytical gradient: } [-0.0580, 0.0012, 0.0001, -2.6411 \times 10^{-7}, 0.0] \quad (\text{A.1.1.10})$$

$$\text{Numerical gradient: } [-0.0580, 0.0012, 0.0001, -2.6411 \times 10^{-7}, 0.0] \quad (\text{A.1.1.11})$$

$$\text{Relative error: } [0.0, 0.0, 0.0, 0.0, 0.0] \quad (\text{A.1.1.12})$$

In this step, we modify the following equation by introducing a small perturbation $\frac{1}{500}$ in the Anisotropy direction a_1 :

$$\frac{\partial W_{ii}}{\partial a_{1f}} = \left(\frac{1}{2} + \frac{1}{500} \right) \langle e_{ki} v_M \rangle^2 (\cot(\alpha_1) + \cot(\alpha_2)) \quad (\text{A.1.1.13})$$

which leads to a failing gradient check for the $ALBO\alpha_1$ component, which increased from 0.0 to 7.69×10^{-2} :

$$\text{Analytical gradient: } [-0.0580, 0.0014, 0.0001, -2.6411 \times 10^{-7}, 0.0] \quad (\text{A.1.1.14})$$

$$\text{Numerical gradient: } [-0.0580, 0.0012, 0.0001, -2.6411 \times 10^{-7}, 0.0] \quad (\text{A.1.1.15})$$

$$\text{Relative error: } [0.0, 7.69 \times 10^{-2}, 0.0, 0.0, 0.0] \quad (\text{A.1.1.16})$$

A.1.2 Eigenbasis Check (Nelsons Method)

Secondly, we evaluate the implementation of Nelsons [19] multi-step approach. Testing this method is particularly important for two reasons. First, it relies on the gradients verified previously. Second, the eigenbasis is far more sensitive than the geometric operators themselves. While geometric operators typically exhibit exploding gradients only in extreme cases, such as degenerate triangles or dot products approaching 0 or 1, the eigenbasis requires the system matrices to be well-conditioned. Ill-conditioned matrices are undesirable because they can render the system highly unstable, where even minimal perturbations in the input may produce disproportionately large variations in the output, effectively making the system hypersensitive. Such instability can arise from poor mesh quality, geometric artifacts such as sharp edges or creases, or from the weighting scheme employed in LBONet. Consequently, the gradient check may also fail under these conditions. For clarity, we separate the checks for eigenvalues and eigenvectors.

Eigenvalue Check

```

1 from torch.autograd import gradcheck
2
3 x.requires_grad = True
4 a1.requires_grad = True
5 a2.requires_grad = True
6 v1.requires_grad = True
7 Theta.requires_grad = True
8 spectre = Nelson.apply
9
10 # Prepare input tuple for gradcheck
11 input = (
12     self.v.unsqueeze(0),
13     self.edges.unsqueeze(0),
14     self.f.unsqueeze(0),
15     x.unsqueeze(0).unsqueeze(0),
16     el.unsqueeze(0),
17     ts1,
18     corners1,
19     minCurvature,
20     maxCurvature,
21     torch.tensor(rn.transpose(1,0)).unsqueeze(0),
22     a1.unsqueeze(0),
23     a2.unsqueeze(0),
24     Theta.unsqueeze(0),
25     v1.unsqueeze(0).unsqueeze(0)
26 )
27
28 # Run gradcheck
29 test = gradcheck(spectre, input, eps=1e-5, atol=1e-4, rtol=1e-3,
30                 fast_mode=True)
31 print("PyTorch Gradcheck Nelson Layer Test: ", test)

```

Listing A.3: Gradcheck for Nelson method in PyTorch

Analytical gradient: $[-118.8079, 1.7586, 2.2051, -3.2702 \times 10^{-5}, -5.6003]$

Numerical gradient: $[-118.8085, 1.7586, 2.2051, -3.2701 \times 10^{-5}, -5.6003]$

Relative error: $[2.53 \times 10^{-6}, 0.0, 0.0, 1.53 \times 10^{-5}, 0.0]$

To further verify the correctness of our implementation of Nelsons method, we introduce a small perturbation to the gradient by adding a minor scalar adjustment

A. Gradient Verification

$\frac{1}{1000}$ to all layers (RiemannNet, ALBONet, VoronoiNet):

$$F_i = \phi^T \left(\frac{\partial W_k}{\partial p_{ij}} + \frac{1}{1000} \right) \phi A \phi - \left(\frac{\partial W_k}{\partial p_{ij}} + \frac{1}{1000} \right) \phi \quad (\text{A.1.2.1})$$

$$F_i = \phi^T \left(-\lambda_i \frac{\partial A_k}{\partial d_{ij}} + \frac{1}{1000} \right) \phi A \phi - \left(\lambda_i \frac{\partial A_k}{\partial d_{ij}} + \frac{1}{1000} \right) \phi \quad (\text{A.1.2.2})$$

$$F_i = \phi^T \left(\left(\frac{\partial W_k}{\partial p_{ij}} - \lambda_i \frac{\partial A_k}{\partial d_{ij}} \right) + \frac{1}{1000} \right) \phi A \phi - \left(\left(\frac{\partial W_k}{\partial p_{ij}} - \lambda_i \frac{\partial A_k}{\partial d_{ij}} \right) + \frac{1}{1000} \right) \phi \quad (\text{A.1.2.3})$$

This leads as expected to drastic discrepancies, most notably on the *ALBO* components. One can observe that RiemannNet and VoronoiNet seem to be more robust to tiny changes, which is in line with the expectation of what each layer modifies in the system.

Analytical gradient: $[-117.5809, 2.3327, 2.7943, 0.5871, -5.5508]$

Numerical gradient: $[-118.8085, 1.7586, 2.2051, -3.2701 \times 10^{-5}, -5.6003]$

Relative error: $[5.19 \times 10^{-3}, 1.40 \times 10^{-1}, 1.18 \times 10^{-1}, 1.0, 4.44 \times 10^{-3}]$

Eigenvector Check

Verifying eigenvectors is more challenging than verifying eigenvalues due to several inherent issues. Eigenvectors are defined only up to a sign, and eigenvectors corresponding to closely spaced eigenvalues may swap. To ensure the correctness of our implementation, we record the signs of the initial eigenbasis and consistently apply them in subsequent computations. This approach is justified because the perturbations applied are infinitesimal, making eigenvector swaps unlikely. Any sign flips that still occur, even when the mesh remains unchanged, can be corrected by aligning with the signs of the initial eigenbasis. In this way, both the reference and perturbed eigenbases remain oriented consistently. It is important to note that forcing all vector components to be positive would violate the eigenvector property and therefore break the algorithm.

Analytical gradient: $[0.3658, 0.0184, 0.0084, -2.6903 \times 10^{-5}, 0.0406]$

Numerical gradient: $[0.3655, 0.0183, 0.0084, -2.6894 \times 10^{-5}, 0.0406]$

Relative error: $[4.10 \times 10^{-4}, 2.72 \times 10^{-3}, 0.0, 1.67 \times 10^{-5}, 0.0]$

A.1. Numerical-Analytical Gradient Verification

Now we repeat the same experiment from Eq. A.1.2.1 on eigenvectors instead of eigenvalues:

Analytical gradient: [0.3909, 0.0174, 0.0196, 0.0102, 0.0405]

Numerical gradient: [0.3655, 0.0183, 0.0084, -2.6894×10^{-5} , 0.0406]

Relative error: [3.36×10^{-2} , 2.52×10^{-2} , 4.00×10^{-1} , 1.0, 1.23×10^{-3}]

One can observe a similar order of magnitude in the errors for the eigenvectors. RiemannNet and VoronoiNet illustrate robustness repeatedly.

A.1.3 Spectral Descriptor Check

In the final verification, we assess the resulting spectral descriptor, specifically the HKS, under perturbations of the geometric operator. This procedure allows us to examine how modifications to the underlying Riemannian metric g_{ij} influence the computed spectral descriptor HKS. Notably, even a localized change in g_{ij} produces a global effect on the eigenbasis, which subsequently impacts the entire descriptor. This final check ensures coverage of all custom layers introduced in our framework, as any preceding or subsequent components are either standard (off-the-shelf) layers or have been independently validated.

```
1 from torch.autograd import gradcheck
2
3 x.requires_grad = True
4 a1.requires_grad = True
5 a2.requires_grad = True
6 v1.requires_grad = True
7 Theta.requires_grad = True
8 spectre = Spectral.apply
9
10 # Prepare input tuple for gradcheck
11 input = (
12     self.v.unsqueeze(0),
13     self.edges.unsqueeze(0),
14     self.f.unsqueeze(0),
15     x.unsqueeze(0).unsqueeze(0),
16     el.unsqueeze(0),
17     ts1,
18     corners1,
19     minCurvature,
20     maxCurvature,
21     torch.tensor(rn.transpose(1,0)).unsqueeze(0),
22     a1.unsqueeze(0),
```

A. Gradient Verification

```

23     a2.unsqueeze(0),
24     Theta.unsqueeze(0),
25     v1.unsqueeze(0).unsqueeze(0)
26 )
27
28 # Run gradcheck
29 test = gradcheck(spectre, input, eps=1e-5, atol=1e-4, rtol=1e-3,
30                  fast_mode=True)
31 print("PyTorch Gradcheck Spectral Layer Test: ", test)

```

Listing A.4: Gradcheck for the Spectral Layer in PyTorch

Analytical gradient: $[-139.4286, -1.7798, -0.5616, -0.0002, -3.5337]$

Numerical gradient: $[-139.4235, -1.7801, -0.5614, -0.0002, -3.5334]$

Relative error: $[1.83 \times 10^{-5}, 8.43 \times 10^{-5}, 1.78 \times 10^{-4}, 0.0, 4.24 \times 10^{-5}]$

To illustrate the sensitivity of the system, we introduce a small perturbation to one of the subgradients of RiemannNet in Eq. 4.5.4.16 by adding a factor of $\frac{1}{1000}$ and observe how this modification affects the outcome of the gradient check.

$$\begin{aligned}
\frac{\partial W_{ij}}{\partial d_{ij}} = & -\left(\frac{1}{2} + \frac{1}{1000}\right) \sin(\gamma_1) \left((a_1 \langle e_{kj} v_M \rangle - \sin(\cos^{-1}(\langle e_{kj} v_M \rangle))) \frac{d_{ik} \cos(\delta_1)}{2T} \right. \\
& + \left. -\sin(\cos^{-2}(\langle e_{ki} v_M \rangle)) \frac{-d_{jk} \cos(\alpha)}{2T} \langle e_{ki} v_M \rangle \right) \\
& + (a_2 \langle e_{kj} v_m \rangle - \sin(\cos^{-1}(\langle e_{kj} v_m \rangle))) \frac{d_{ik} \cos(\delta_1)}{2T} \\
& + \left. -\sin(\cos^{-1}(\langle e_{ki} v_m \rangle)) \frac{-d_{jk} \cos(\alpha)}{2T} \langle e_{ki} v_m \rangle \right) \\
& - \frac{1}{2} \left((a_1 (\langle e_{ki} v_M \rangle \langle e_{kj} v_M \rangle) + a_2 (\langle e_{ki} v_m \rangle \langle e_{kj} v_m \rangle)) \frac{-d_{ij} \cos(\delta)}{2T \sin(\delta)^2} \right) \quad (\text{A.1.3.1})
\end{aligned}$$

Even a small perturbation added to one of the subgradients in the anisotropic RiemannNet results in a substantial deviation in the gradient, thereby confirming the correctness of the end-to-end gradient computation.

A.1. Numerical-Analytical Gradient Verification

Analytical gradient: $[-154.8009, -1.7798, -0.5616, -0.0002, -3.5337]$

Numerical gradient: $[-139.4235, -1.7801, -0.5614, -0.0002, -3.5334]$

Relative error: $[0.0523, 8.43 \times 10^{-5}, 1.78 \times 10^{-4}, 0.0, 4.24 \times 10^{-5}]$

For the aforementioned gradient checks a model from SHREC15 has been used. Detailed derivation and check has been done on the all benchmarks before training and is strongly suggested as a preparation for successful learning. For the tests conducted in this section, the manifold depicted in Figure A.1 has been used.

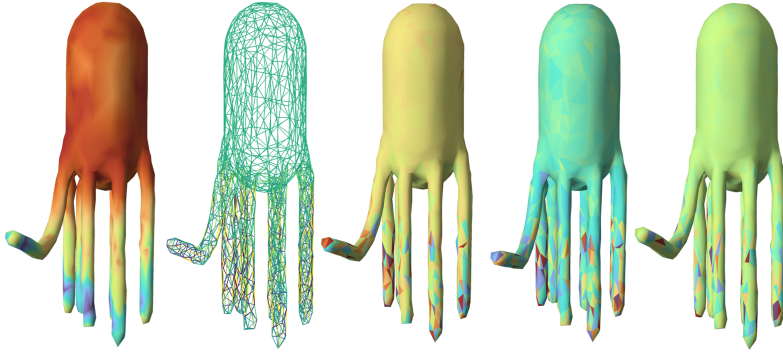


Figure A.1: Sample mesh used to validate the introduced gradients. From left to right: VoronoiNet, RiemannNet, and ALBO+Net. Red indicates higher values while blue indicates lower values. All gradients are computed using an LBO eigenbasis with 32 frequencies to simulate a realistic benchmarking scenario.

Abbreviations

ACSP	Airline Crew Scheduling Problem
ADD	Anisotropic Diffusion Descriptor
CI	Curvature Index
CMOS	Conformal Metric Optimization on Surfaces
CNN	Convolutional Neural Network
CPU	Central Processing Unit
DCG	Discounted Cumulative Gain
DGCNN	Dynamic Graph Convolutional Neural Network
E	E-Measure
FT	First Tier
GEP	Generalized Eigenvalue Problem
GMM	Gaussian Mixture Model
GPS	Global Point Signature
GPU	Graphics Processing Unit
HKS	Heat Kernel Signature
ICP	Iterative Closest Point
IEEE	Institute of Electrical and Electronics Engineers
IQR	Interquartile Range
KLBO	Kinetic Laplace–Beltrami Operator
KNN	k-nearest neighbor

LBO	Laplace-Beltrami Operator
LOBPCG	Locally Optimal Block Preconditioned Conjugate Gradient
LSCNN	Localized Spectral CNN
NEAT	NeuroEvolution of Augmenting Topologies
NN	Nearest Neighbour
OSD	Optimal Spectral Descriptor
PDE	Partial Differential Equation
RAM	Random Access Memory
ReLU	Rectified Linear Unit
RMOS	Riemannian Metric Optimization on Surfaces
RPG	Rocket-Propelled Grenade
SHREC	Shape Retrieval Challenge
SI	Shape Index
SI-HKS	Scale-Invariant Heat Kernel Signature
SOTA	State of the Art
ST	Second Tier
TSP	Travelling Salesman Problem
VGL	Vision, Graphics and Learning Research Group
WFT	Windowed Fourier Transform
WKS	Wave Kernel Signature

Glossary

E	E-measure, a retrieval metric that combines precision and recall into a single value
EdgeConv	Edge Convolution, a convolution operation applied to edges in graph-based neural networks
eigenbasis	A basis formed by the eigenvectors of a linear operator or matrix
extrinsic	Describing quantities or properties that depend on the embedding of a manifold in an ambient space
FT	First Tier, a measure used in retrieval evaluation for ranking performance
intrinsic	Describing quantities or properties that depend only on the intrinsic geometry of a manifold and not on its embedding
MATLAB	MATLAB, a numerical computing environment and programming language

MLP	Multi-Layer Perceptron, a type of feedforward artificial neural network with multiple layers of neurons
near-isometric	Describing deformations that approximately preserve intrinsic distances, up to a bounded distortion
non-rigid	Describing deformations that are not limited to rigid transformations and may involve bending or stretching
non-isometric	Describing deformations that do not preserve intrinsic distances and violate isometry assumptions
PyTorch	Python-based open-source deep learning framework

References

- [1] David Hilbert. *Natureerkennen und Logik*. Address delivered at the Society of German Scientists and Physicians meeting in Königsberg, Germany. 1930.
- [2] Jianfeng Xiang et al. “Structured 3d latents for scalable and versatile 3d generation”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2025, pp. 21469–21480.
- [3] Raphael. *The School of Athens*. Fresco, Stanza della Segnatura, Apostolic Palace, Vatican City. 1511.
- [4] Roger Penrose. *The road to reality*. Random house, 2006.
- [5] Charles R Qi et al. “Pointnet: Deep learning on point sets for 3d classification and segmentation”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 652–660.
- [6] Yue Wang et al. “Dynamic graph cnn for learning on point clouds”. In: *Acm Transactions On Graphics (tog)* 38.5 (2019), pp. 1–12.
- [7] Nicholas Sharp et al. “Diffusionnet: Discretization agnostic learning on surfaces”. In: *ACM Transactions on Graphics (TOG)* 41.3 (2022), pp. 1–16.
- [8] Charles Ruizhongtai Qi et al. “Pointnet++: Deep hierarchical feature learning on point sets in a metric space”. In: *Advances in neural information processing systems*. 2017, pp. 5099–5108.
- [9] Guy Waldo Dunnington. *Carl Friedrich Gauss: Titan of Science*. The Mathematical Association of America, 1955.
- [10] Martin Reuter, Franz-Erich Wolter, and Niklas Peinecke. “Laplace-spectra as fingerprints for shape matching”. In: *Proceedings of the 2005 ACM symposium on Solid and physical modeling*. ACM. 2005, pp. 101–106.
- [11] Mark Meyer et al. “Discrete differential-geometry operators for triangulated 2-manifolds”. In: *Visualization and mathematics III*. Springer, 2003, pp. 35–57.
- [12] Raif M Rustamov et al. “Laplace-Beltrami eigenfunctions for deformation invariant shape representation”. In: *Symposium on geometry processing*. Vol. 257. 2007, pp. 225–233.
- [13] Jian Sun, Maks Ovsjanikov, and Leonidas Guibas. “A concise and provably informative multi-scale signature based on heat diffusion”. In: *Computer graphics forum*. Vol. 28. 5. Wiley Online Library. 2009, pp. 1383–1392.
- [14] Michael M Bronstein and Iasonas Kokkinos. “Scale-invariant heat kernel signatures for non-rigid shape recognition”. In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE. 2010, pp. 1704–1711.

- [15] Mathieu Aubry, Ulrich Schlickewei, and Daniel Cremers. “The wave kernel signature: A quantum mechanical approach to shape analysis”. In: *2011 IEEE international conference on computer vision workshops (ICCV workshops)*. IEEE. 2011, pp. 1626–1633.
- [16] Frederico A Limberger and Richard C Wilson. “Feature Encoding of Spectral Signatures for 3D Non-Rigid Shape Retrieval.” In.
- [17] Seif Eddine Naffouti et al. “Heuristic optimization-based wave kernel descriptor for deformable 3D shape matching and retrieval”. In: *Signal, Image and Video Processing* 12.5 (2018), pp. 915–923.
- [18] Roei Litman and Alexander M Bronstein. “Learning spectral descriptors for deformable shape correspondence”. In: *IEEE transactions on pattern analysis and machine intelligence* 36.1 (2013), pp. 171–180.
- [19] Richard B Nelson. “Simplified calculation of eigenvector derivatives”. In: *AIAA journal* 14.9 (1976), pp. 1201–1205.
- [20] Yonggang Shi et al. “Conformal metric optimization on surface (CMOS) for deformation and mapping in Laplace-Beltrami embedding space”. In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer. 2011, pp. 327–334.
- [21] Jin Kyu Gahm, Yonggang Shi, Alzheimers Disease Neuroimaging Initiative, et al. “Riemannian metric optimization on surfaces (rmos) for intrinsic brain mapping in the laplace-beltrami embedding space”. In: *Medical image analysis* 46 (2018), pp. 189–201.
- [22] Maks Ovsjanikov et al. “Functional maps: a flexible representation of maps between shapes”. In: *ACM Transactions on Graphics (ToG)* 31.4 (2012), pp. 1–11.
- [23] Davide Boscaini et al. “Anisotropic diffusion descriptors”. In: *Computer Graphics Forum*. Vol. 35. 2. Wiley Online Library. 2016, pp. 431–441.
- [24] Frederico A Limberger and Richard C Wilson. “Curvature-based spectral signatures for non-rigid shape retrieval”. In: *Computer Vision and Image Understanding* 172 (2018), pp. 1–11.
- [25] Dmitriy Smirnov and Justin Solomon. “HodgeNet: Learning spectral geometry on triangle meshes”. In: *ACM Transactions on Graphics (TOG)* 40.4 (2021), pp. 1–11.
- [26] Iasonas Kokkinos et al. “Intrinsic shape context descriptors for deformable shapes”. In: *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2012, pp. 159–166.
- [27] Davide Boscaini et al. “Learning class-specific descriptors for deformable shapes using localized spectral convolutional networks”. In: *Computer Graphics Forum*. Vol. 34. 5. Wiley Online Library. 2015, pp. 13–23.
- [28] Jonathan Masci et al. *Shapenet: Convolutional neural networks on non-euclidean manifolds*. Tech. rep. 2015.
- [29] Davide Boscaini et al. “Learning shape correspondence with anisotropic convolutional neural networks”. In: *Advances in neural information processing systems* 29 (2016).

References

- [30] Li Yi et al. “Syncspecnn: Synchronized spectral cnn for 3d shape segmentation”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 2282–2290.
- [31] Federico Monti et al. “Geometric deep learning on graphs and manifolds using mixture model cnns”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 5115–5124.
- [32] Hengshuang Zhao et al. “Point transformer”. In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2021, pp. 16259–16268.
- [33] Meng-Hao Guo et al. “Pct: Point cloud transformer”. In: *Computational Visual Media* 7 (2021), pp. 187–199.
- [34] Chi-Chong Wong. “Heat Diffusion Based Multi-Scale and Geometric Structure-Aware Transformer for Mesh Segmentation”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 4413–4422.
- [35] Or Litany et al. “Deep functional maps: Structured prediction for dense shape correspondence”. In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 5659–5667.
- [36] Dongliang Cao, Paul Roetzer, and Florian Bernard. “Unsupervised learning of robust spectral shape matching”. In: *arXiv preprint arXiv:2304.14419* (2023).
- [37] Simone Melzi et al. “Zoomout: Spectral upsampling for efficient shape correspondence”. In: *arXiv preprint arXiv:1904.07865* (2019).
- [38] Francesco Milano et al. “Primal-dual mesh convolutional neural networks”. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 952–963.
- [39] Proclus. *A Commentary on the First Book of Euclid’s Elements*. Trans. by Glenn R. Morrow. Princeton University Press, 1992.
- [40] Gregory Hornby et al. “Automated antenna design with evolutionary algorithms”. In: *Space 2006*. 2006, p. 7242.
- [41] Leonard Andreevi Rastrigin. “Systems of extremal control”. In: *Nauka* (1974).
- [42] Günter Rudolph. “Globale optimierung mit parallelen evolutionsstrategien”. PhD thesis. Diplomarbeit, Universit at Dortmund, Fachbereich Informatik, 1990.
- [43] H. H. Rosenbrock. “An Automatic Method for Finding the Greatest or Least Value of a Function”. In: *The Computer Journal* 3.3 (Jan. 1960), pp. 175–184. DOI: 10.1093/comjnl/3.3.175. eprint: <https://academic.oup.com/comjnl/article-pdf/3/3/175/988633/030175.pdf>. URL: <https://doi.org/10.1093/comjnl/3.3.175>.
- [44] Frederico Artur Limberger. “Spectral signatures for non-rigid 3D shape retrieval”. PhD thesis. University of York, 2017.
- [45] Yoni Choukroun et al. “Hamiltonian operator for spectral shape analysis”. In: *IEEE transactions on visualization and computer graphics* 26.2 (2018), pp. 1320–1331.
- [46] Kenneth O Stanley and Risto Miikkulainen. “Evolving neural networks through augmenting topologies”. In: *Evolutionary computation* 10.2 (2002), pp. 99–127.

- [47] Paul Heider et al. “Local shape descriptors, a survey and evaluation”. In: *Proceedings of the 4th Eurographics conference on 3D Object Retrieval*. 2011, pp. 49–56.
- [48] Bo Li, Afzal Godil, and Henry Johan. “Hybrid shape descriptor and meta similarity generation for non-rigid and partial 3D model retrieval”. In: *Multimedia tools and applications* 72.2 (2014), pp. 1531–1560.
- [49] Szymon Rusinkiewicz. “Estimating curvatures and their derivatives on triangle meshes”. In: *Proceedings. 2nd International Symposium on 3D Data Processing, Visualization and Transmission, 2004. 3DPVT 2004*. IEEE. 2004, pp. 486–493.
- [50] Philip Shilane et al. “The princeton shape benchmark”. In: *Proceedings Shape Modeling Applications, 2004*. IEEE. 2004, pp. 167–178.
- [51] David Pickup et al. “SHREC14 track: Shape retrieval of non-rigid 3D human models”. In: *Proceedings of the 7th Eurographics workshop on 3D Object Retrieval*. Vol. 1. 2. Eurographics Association. 2014, p. 6.
- [52] Ruixuan Yu, Jian Sun, and Huibin Li. “Second-order spectral transform block for 3D shape classification and retrieval”. In: *IEEE Transactions on Image Processing* 29 (2020), pp. 4530–4543.
- [53] Raia Hadsell, Sumit Chopra, and Yann LeCun. “Dimensionality reduction by learning an invariant mapping”. In: *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*. Vol. 2. IEEE. 2006, pp. 1735–1742.
- [54] Xi Zhou et al. “Image classification using super-vector coding of local image descriptors”. In: *European conference on computer vision*. Springer. 2010, pp. 141–154.
- [55] Zhouhui Lian et al. “SHREC’10 Track: Non-rigid 3D Shape Retrieval.” In: *3DOR 10* (2010), pp. 101–108.
- [56] Shuhui Bu et al. “Learning high-level feature by deep belief networks for 3-D model retrieval and recognition”. In: *IEEE Transactions on Multimedia* 16.8 (2014), pp. 2154–2167.
- [57] Lorenzo Luciano and A Ben Hamza. “A global geometric framework for 3D shape retrieval using deep learning”. In: *Computers & Graphics* 79 (2019), pp. 14–23.
- [58] Roe Litman et al. “Supervised learning of bag-of-features shape descriptors using sparse coding”. In: *Computer Graphics Forum*. Vol. 33. 5. Wiley Online Library. 2014, pp. 127–136.
- [59] Z. Lian et al. “Non-Rigid 3D Shape Retrieval”. In: *Proceedings of the 2015 Eurographics Workshop on 3D Object Retrieval*. 3DOR ’15. Zurich, Switzerland: Eurographics Association, 2015, pp. 107–120.
- [60] Bernhard Riemann. “Über die Hypothesen, welche der Geometrie zu Grunde liegen”. In: *Abhandlungen der Königlich Gesellschaft der Wissenschaften zu Göttingen*. Habilitationsvortrag 1854. 1868.
- [61] Luca Cosmo et al. “3D shape analysis through a quantum lens: the average mixing kernel signature”. In: *International Journal of Computer Vision* 130.6 (2022), pp. 1474–1493.

References

- [62] Sarah Tang and Afzal Godil. “An evaluation of local shape descriptors for 3D shape retrieval”. In: *Three-Dimensional Image Processing (3DIP) and Applications II*. Vol. 8290. International Society for Optics and Photonics. 2012, 82900N.
- [63] Jo Bo Rosen. “The gradient projection method for nonlinear programming. Part I. Linear constraints”. In: *Journal of the society for industrial and applied mathematics* 8.1 (1960), pp. 181–217.
- [64] Inderjit S. Dhillon, Suvrit Sra, and Joel A. Tropp. “Triangle Fixing Algorithms for the Metric Nearness Problem”. In: *NIPS*. 2004, pp. 361–368. URL: <http://papers.nips.cc/paper/2598-triangle-fixing-algorithms-for-the-metric-nearness-problem>.
- [65] Mathieu Andreux et al. “Anisotropic Laplace-Beltrami operators for shape analysis”. In: *Computer Vision-ECCV 2014 Workshops: Zurich, Switzerland, September 6-7 and 12, 2014, Proceedings, Part IV 13*. Springer. 2015, pp. 299–312.
- [66] Qinsong Li et al. “Shape correspondence using anisotropic Chebyshev spectral CNNs”. In: *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*. 2020, pp. 14658–14667.
- [67] Rana Hanocka et al. “Meshcnn: a network with an edge”. In: *ACM Transactions on Graphics (TOG)* 38.4 (2019), pp. 1–12.
- [68] Pauli Virtanen et al. “SciPy 1.0: fundamental algorithms for scientific computing in Python”. In: *Nature Methods* 17.3 (Mar. 2020), pp. 261–272. DOI: 10.1038/s41592-019-0686-2. URL: <https://doi.org/10.1038/s41592-019-0686-2>.
- [69] Richard B Lehoucq, Danny C Sorensen, and Chao Yang. *ARPACK users’ guide: solution of large-scale eigenvalue problems with implicitly restarted Arnoldi methods*. SIAM, 1998.
- [70] Adam Paszke et al. “Pytorch: An imperative style, high-performance deep learning library”. In: *Advances in neural information processing systems* 32 (2019).
- [71] Z Lian et al. “Shape retrieval on non-rigid 3D watertight meshes”. In: *Eurographics workshop on 3d object retrieval (3DOR)*. Citeseer. 2011.
- [72] Andrea Giachetti and Christian Lovato. “Radial symmetry detection and shape characterization with the multiscale area projection transform”. In: *Computer graphics forum*. Vol. 31. 5. Wiley Online Library. 2012, pp. 1669–1678.
- [73] Alon Lahav and Ayellet Tal. “Meshwalker: Deep mesh understanding by random walks”. In: *ACM Transactions on Graphics (TOG)* 39.6 (2020), pp. 1–13.
- [74] Thomas W Mitchel, Vladimir G Kim, and Michael Kazhdan. “Field convolutions for surface CNNs”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 10001–10011.
- [75] Yifan Xu et al. “Spidercnn: Deep learning on point sets with parameterized convolutional filters”. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 87–102.
- [76] Haggai Maron et al. “Convolutional neural networks on surfaces via seamless toric covers.” In: *ACM Trans. Graph.* 36.4 (2017), pp. 71–1.
- [77] Yunhai Wang et al. “Active co-analysis of a set of shapes”. In: *ACM Transactions on Graphics (TOG)* 31.6 (2012), pp. 1–10.

- [78] Federica Bogo et al. “FAUST: Dataset and evaluation for 3D mesh registration”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014, pp. 3794–3801.
- [79] Hugues Thomas et al. “Kpconv: Flexible and deformable convolution for point clouds”. In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2019, pp. 6411–6420.
- [80] Klaus Hildebrandt Ruben Wiersma Elmar Eisemann. “CNNs on Surfaces using Rotation-Equivariant Features”. In: *Transactions on Graphics* 39.4 (July 2020). DOI: 10.1145/3386569.3392437.
- [81] Arianna Rampini et al. “Correspondence-free region localization for partial shape similarity via hamiltonian spectrum alignment”. In: *2019 International Conference on 3D Vision (3DV)*. IEEE. 2019, pp. 37–46.
- [82] Laurens Van der Maaten and Geoffrey Hinton. “Visualizing data using t-SNE.” In: *Journal of machine learning research* 9.11 (2008).
- [83] Felix Klein. “Vergleichende Betrachtungen über neuere geometrische Forschungen”. In: *Mathematische Annalen*. Erlangen Program (Original programmatic text). 1872.
- [84] Jing Ren et al. “Continuous and orientation-preserving correspondences via functional maps”. In: *ACM Transactions on Graphics (ToG)* 37.6 (2018), pp. 1–16.
- [85] Mikhail Panine, Maxime Kirgo, and Maks Ovsjanikov. “Non-Isometric Shape Matching via Functional Maps on Landmark-Adapted Bases”. In: *Computer graphics forum*. Vol. 41. 6. Wiley Online Library. 2022, pp. 394–417.
- [86] Nicolas Donati, Etienne Corman, and Maks Ovsjanikov. “Deep orientation-aware functional maps: Tackling symmetry issues in shape matching”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 742–751.
- [87] Danielle Ezuz and Mirela Ben-Chen. “Deblurring and denoising of maps between shapes”. In: *Computer Graphics Forum*. Vol. 36. 5. Wiley Online Library. 2017, pp. 165–174.
- [88] Amit Bracha, Oshri Halim, and Ron Kimmel. “Shape correspondence by aligning scale-invariant LBO eigenfunctions”. In: (2020).
- [89] Riccardo Marin et al. “Correspondence learning via linearly-invariant embedding”. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 1608–1620.
- [90] Sharik Siddiqi and Zorah Löhner. “A network analysis for correspondence learning via linearly-embedded functions”. In: *DAGM German Conference on Pattern Recognition*. Springer. 2023, pp. 100–114.
- [91] Giulio Viganò and Simone Melzi. “Bijective upsampling and learned embedding for point clouds correspondences”. In: *Computers & Graphics* 122 (2024), p. 103985.
- [92] Ruqi Huang et al. “Consistent zoomout: Efficient spectral map synchronization”. In: *Computer Graphics Forum*. Vol. 39. 5. Wiley Online Library. 2020, pp. 265–278.
- [93] Davide Eynard et al. “Coupled functional maps”. In: *2016 Fourth International Conference on 3D Vision (3DV)*. IEEE. 2016, pp. 399–407.

References

- [94] Jing Ren et al. “Discrete optimization for shape matching”. In: *Computer Graphics Forum*. Vol. 40. 5. Wiley Online Library. 2021, pp. 81–96.
- [95] Robin Magnet et al. “Smooth non-rigid shape matching via effective dirichlet energy optimization”. In: *2022 International Conference on 3D Vision (3DV)*. IEEE. 2022, pp. 495–504.
- [96] Luca Cosmo et al. “Isospectralization, or how to hear shape, style, and correspondence”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 7529–7538.
- [97] Riccardo Marin et al. “Spectral shape recovery and analysis via data-driven connections”. In: *International journal of computer vision* 129 (2021), pp. 2745–2760.
- [98] Simon Weber et al. “Finsler-Laplace-Beltrami Operators with Application to Shape Analysis”. In: *arXiv preprint arXiv:2404.03999* (2024).
- [99] Oguzhan Yigit and Richard C Wilson. “LBONet: Supervised Spectral Descriptors for Shape Analysis”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2025).
- [100] Samuele Salti, Federico Tombari, and Luigi Di Stefano. “SHOT: Unique signatures of histograms for surface and texture description”. In: *Computer Vision and Image Understanding* 125 (2014), pp. 251–264.
- [101] Dorian Nogneng and Maks Ovsjanikov. “Informative descriptor preservation via commutativity for shape matching”. In: *Computer Graphics Forum*. Vol. 36. 2. Wiley Online Library. 2017, pp. 259–267.
- [102] Riccardo Marin et al. “Farm: Functional automatic registration method for 3d human bodies”. In: *Computer Graphics Forum*. Vol. 39. 1. Wiley Online Library. 2020, pp. 160–173.
- [103] Dragomir Anguelov et al. “Scape: shape completion and animation of people”. In: *ACM SIGGRAPH 2005 Papers*. 2005, pp. 408–416.
- [104] Silvia Zuffi et al. “3D menagerie: Modeling the 3D shape and pose of animals”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 6365–6373.
- [105] Ilya Loshchilov and Frank Hutter. “Decoupled weight decay regularization”. In: *arXiv preprint arXiv:1711.05101* (2017).
- [106] Marvin Eisenberger et al. “Deep shells: Unsupervised shape correspondence with optimal transport”. In: *Advances in Neural information processing systems* 33 (2020), pp. 10491–10502.
- [107] Simone Melzi et al. “Localized manifold harmonics for spectral shape analysis”. In: *Computer Graphics Forum*. Vol. 37. 6. Wiley Online Library. 2018, pp. 20–34.