Eurographics Symposium on Geometry Processing 2018
T. Ju and A. Vaxman
(Guest Editors)

Volume 37 (2018), Number 5

Modular Latent Spaces for Shape Correspondences

Vignesh Ganapathi-Subramanian[1], Olga Diamanti[1,2], Leonidas J. Guibas[1]

[1] Stanford University, [2] Autodesk Inc.

Abstract

We consider the problem of transporting shape descriptors across shapes in a collection in a modular fashion, in order to establish correspondences between them. A common goal when mapping between multiple shapes is consistency, namely that compositions of maps along a cycle of shapes should be approximately an identity map. Existing attempts to enforce consistency typically require recomputing correspondences whenever a new shape is added to the collection, which can quickly become intractable. Instead, we propose an approach that is fully modular, where the bulk of the computation is done on each shape independently. To achieve this, we use intermediate nonlinear embedding spaces, computed individually on every shape; the embedding functions use ideas from diffusion geometry and capture how different descriptors on the same shape inter-relate. We then establish linear mappings between the different embedding spaces, via a shared latent space. The introduction of nonlinear embeddings allows for more nuanced correspondences, while the modularity of the construction allows for parallelizable calculation and efficient addition of new shapes. We compare the performance of our framework to standard functional correspondence techniques and showcase the use of this framework to simple interpolation and extrapolation tasks.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Picture/Image Generation—Line and curve generation

1. Introduction

Understanding how different geometric shapes relate to each other is a fundamental problem in geometry processing, and can be key in analyzing, representing, learning, reconstructing and modeling shapes in applications. A natural way to determine the relationship between two or more shapes is by first placing them in correspondence, for example by establishing a map between them. Once the correspondence has been established, it is possible to reason about the similarity between a pair of shapes, or the geometric variability among shapes of a given category.

One way to express shape correspondence is in a point-to-point fashion, where each point in one shape is assigned to its matching point on the other shape. Alternatively, correspondences can be established by matching between ‘functions, or descriptors, defined on the shapes. Using descriptors is frequently a more concise and informative way for expressing a correspondence: the fine-grained detail encoded in a point-to-point map is often unnecessary due to inherent shape ambiguities or because only approximate alignment is sought after. In addition, establishing descriptor-based correspondence is often an easier task, especially if linearity assumptions on the mappings are imposed. In any case, as shown in recent literature [OBCS*12], the two kinds of correspondences can be interchangeable.

An issue that frequently arises when computing correspondences within a collection of shapes is that of *consistency*: mapping one shape onto another via a third shape should ideally produce the same result as when the two shapes are directly mapped to each other. Ensuring consistency is generally a difficult task; solving for all pairwise correspondences simultaneously so that global consistency is achieved can be intractable. As a result, recent works have imposed assumptions in order to simplify the problem, such as linearity in the transport or a low-rank condition [HG13]. While these assumptions reduce the underlying complexity, an issue of scalability still remains: the pairwise correspondences need to be recomputed anew every time a new shape is added to the collection. For scalability, a level of *modularity* is desired: adding a new shape should ideally leave the computation on other shapes unaffected.

In this paper, we propose a modular way of establishing function-based correspondences between shapes in a collection. The key idea is to look at how different descriptors functions defined on the same shape interrelate, and use this as a cue to describe the shape via a latent representation. In this way, each shape is mapped to its latent representation completely independently. This not only allows for parallelizability of the computation, but also trivially establishes consistency across the entire network of shapes. Adding a new shape to the collection only requires calculation on that specific shape. The quality of the correspondence depends on selecting an appropriate set of matching functional descriptors used for learning the latent space, which is a standard initial step in many current shape correspondence techniques.

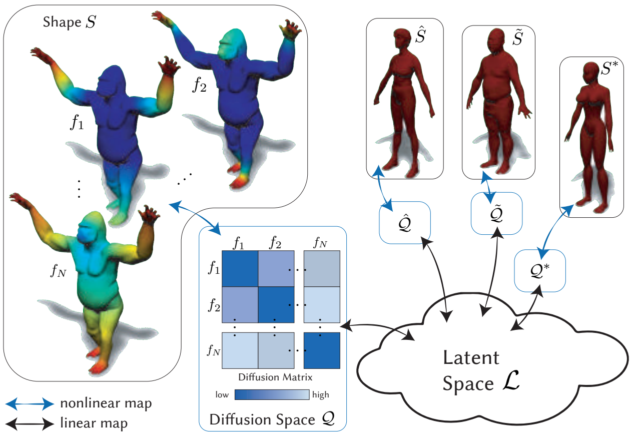

Figure 1: *Assume a shape collection $S,\tilde{S},\hat{S},S^*$, with input descriptors $\{f_1,f_2,\cdots f_N\}$ defined on each shape S; descriptors are in correspondence across shapes. The interdependencies of the descriptors on each individual shape are exploited via diffusion geometry to build nonlinear embedding spaces $\mathcal{Q}$ ($\tilde{\mathcal{Q}},\hat{\mathcal{Q}},\mathcal{Q}^*$) for shape S ($\tilde{S},\hat{S},S^*$ respectively), where the closeness between two features on the same shape is reflected in their embedding in $\mathcal{Q}$. The diffusion embeddings $\mathcal{Q},\tilde{\mathcal{Q}},\hat{\mathcal{Q}},\mathcal{Q}^*$ are related to one another via a linear map, which enables consistent descriptor transport across the network, and thus shape correspondences. The individually-built embeddings ensure modularity of the transport framework, which enables adding and matching new shapes to the network with minimal computational overhead.*

In order to express the relation between different features on the same shape, we use ideas from diffusion geometry; diffusion distances have been previously shown to be powerful in the analysis of shape meshes [RLF09]. We build an intermediate diffusion-based nonlinear embedding for each shape in the collection, and then define maps onto the shared latent space. Compared to many recent, linear functional mapping techniques, the introduction of these nonlinearities can help achieve more nuanced shape correspondences that better respect the input descriptor correspondence. In summary, our contributions are:

- A nonlinear descriptor embedding space, individually computable from pre-computed features on every shape in a collection.
- A technique for connecting these embedding spaces via a shared latent space.
- A straightforward application of this technique for transporting general functions across shapes in the collection. This places the shapes in correspondence in a modular and parallelizable way; since all calculation is done independently, no additional consistency constraint is necessary.
- Qualitative and quantitative validation and comparisons to prior functional transport techniques, as well as applications in segmentation transfer and data interpolation/extrapolation.

2. Related Work

Shape correspondences and models to infer them have been the focus of much prior research in geometry processing. An extensive discussion of all of these methods is beyond the scope of this work, and is covered by existing surveys [VKZHCO11, TCL*13]. Here, we discuss techniques are directly related to our work.

Correspondences between isometric shape pairs, specifically those related by a non-rigid transformation has been explored by the model introduced by [BBK06] and [Mém07], where the core assumption is the invariance of geodesic distances to isometric transformations. This model inspired a number of shape correspondence models such as [OMMG10, RPSS10]; the latter uses feature matching to obtain a point-to-point map between shapes. Other methods use assumptions other than isometry – for example, [LF09,KLF11,APL15] focus on conformal deformation-based correspondences. In both cases, the underlying shape deformation models can be too restrictive for large geometric deformations.

More recently, functional correspondence techniques [OBCS*12, SNB*12, SRGB14] have become prominent, which attempt to obtain soft probabilistic correspondences between shapes by relating real-valued functions on shapes. When the shape deformations are large, these methods generally tend to provide more efficient and less error-prone correspondences. Non-rigid shape matching has also been dealt with using learning techniques, where training data is assumed and a deformation model is learnt and applied onto new shape pairs. Such models have been learnt for both point-to-point correspondences [WHC*16] and functional correspondences [COC14]. More recent work by [GSTOG16] correspondences are shown to be more dependent on the “rank” of the functions being matched (namely, the ordering of the shape points as determined by the function values) rather than the function values themselves. As an extension of this idea, [DMB*17] uses the idea of biclustering to obtain refined region-based correspondences between drastically different shapes. Inspired by these works, we also use function-rank based descriptors as to build the latent shape spaces for descriptor transfer.

Another concept of interest to computer vision and graphics related tasks is that of cyclic consistency. [WHG13] uses cyclic consistency as a tool to co-segment similar objects across an image network, while [HWG14] explores large shape collections using a consistency-based paradigm. A similar idea has also been used very recently to perform unsupervised image translation across image domains, using a cyclic-consistency loss in a Generative Adversarial Network (GAN) [ZPIE17]. In a technique that is very different in spirit to our work, [CRA*17] also aim to solve for consistent maps over an entire shape collection to match partial features through jointly learnt attributes using a sparse modeling based approach. Cyclic consistency ensures the consistent transport of descriptors across intermediate spaces (eg. images, shapes) in all of these domains. With the same goal in mind, we build a cyclically-consistent framework that aids in propagating descriptors across a shape network. Descriptors from one shape are transported to a common latent space, via an intermediate embedding space. This idea is similar in spirit to the multi-level map construction of [HWG14], with the map to the embedding space now being nonlinear. Building consistent nonlinear maps was attempted in [OMPG13], with the goal of breaking symmetries in isometric shape matching. A fairly exhaustive quantization of descriptor space was provided in [HS14], where cycle-consistency is guaranteed over a large shape collection. The major advantage of our technique over [HWG14] and other discussed nonlinear descriptor embedding techniques is its modularity, which makes it possible to add new shapes to a shape collection, as opposed to having to recompute maps between existing shapes every time a new shape is added. This property enables computing maps on shape collections of ever-increasing size such as Shapenet [CFG*15].

Simple linear maps between functional spaces on shapes are prone to heavy loss of descriptor information. This is predominantly due to dimensionality: the number of mesh vertices, over which the descriptor functions are typically defined, can be orders of magnitude higher than the dimension of the maps being computed. In the classical functional correspondence techniques, this is addressed by limiting the descriptor space: A Laplace-Beltrami basis representation [Lév06] is used for the descriptors, which works well for transferring descriptors built from the heat-equation or other Laplacian-based differential equations [SOG09, ASC11]. In order to then build linear maps for more general descriptors, it is vital to identify more general dimensionality reduction techniques, e.g. by exploring embedding techniques different than linear projection. Locally linear embeddings [RS00] attempt to partially solve this problem by computing a linear-like embedding on neighborhoods. Other techniques capture different aspects of the source descriptor information, such as classical multi-dimensional scaling (MDS) [Bas99], which aims to preserve distances between descriptors in the source shape space and the target embedding space. Classical Principal Components Analysis (PCA) [Pea01] captures spectral differences between the descriptors in the embedding space. In this paper, we build a novel embedding space inspired by diffusion maps [NLKC06], which also attempts to capture relative relations between shape descriptors. The construction of the diffusion embedding itself is based on inter-descriptor differences - our key assumption is that this enables capturing more nuanced descriptor properties and relations compared to e.g. MDS, while still embedding them into a lower dimensional space. We then build linear maps between the diffusion-based embedding spaces to perform inter-shape transport. We use linear maps because of their ease of use, but employ nonlinear embeddings to minimize information loss. The computationally heavy nonlinear embedding is done within a single shape space, and thus can be performed offline. This modular concept automatically ensures consistency between maps across multiple shapes, and does so without additional time or complexity costs in adding new shapes, which can be an issue with existing works that address consistency [HWG14].

3. Method

3.1. Overview

The key idea in this paper is the indirect construction of a latent space $\mathcal{L}$ which can be used to propagate descriptors across a collection of shapes, thus placing shapes in correspondence. The input is a set of shapes with matching shape descriptors defined on them. The construction proceeds in two stages (Fig. 1). In the first stage, for each shape S *independently*, a nonlinear, diffusion distance-based embedding function e is constructed that maps from the space $\mathcal{F}$ of descriptors on the shape to an intermediate space $\mathcal{Q}$, $e : \mathcal{F} \to \mathcal{Q}$. A mapping along the opposite direction $d : \mathcal{Q} \to \mathcal{F}$ ("de-embedder") is also constructed for each shape. In a second stage, linear full-rank mappings are constructed from each $\mathcal{Q}$ onto the latent space $\mathcal{L}$ – we refer to the linear mapping for shape S via its matrix $\mathbf{E}$.

The functions e and d, as well as the mapping $\mathbf{E}$, are specific to each shape, and can be thought of as components "internal" to the shape. The indirectly constructed latent space $\mathcal{L}$ can be thought of as the "external" component shared across all shapes. Adding a new shape $\tilde{S}$ to the collection only involves computing its internal components. Transferring descriptors between $\tilde{S}$ and any other existing shape S can be achieved via a linear transformation between $\tilde{\mathcal{Q}}$ and $\mathcal{Q}$. In the following sections we detail the construction of all intermediate stages of this construction.

3.2. Diffusion-based embedding.

The input at this stage is a shape S and a set of shape descriptors (on-shape functions $\{f_i\}, f_i \in \mathcal{F}$) defined on it: $f_i : S \to \mathbb{R}$.

The type of information captured by each descriptor, and the variability of each descriptor across different shapes in the collection may vary greatly across descriptors. Thus, each such descriptor can be thought of as a snapshot of the shape from a particular "viewpoint"; the information extracted by one descriptor correlates to that of other descriptors in a shape-dependent way. We propose to understand the variability and relationships between these different views of a shape using diffusion distances; this is similar to an idea explored by [KCLZ10] in a signal processing context. Performing diffusion on descriptors provides us with low-dimensional representation of features, which are dependent only on how they are related to other features.

A brief description of the diffusion kernel is provided below; for a more detailed study see [CLL*05, NLKC06, NLCK06, LT14]. The diffusion kernel as defined in diffusion geometry is a bivariate function $D : P \times P \to \mathbb{R}$ computed on points $p, p' \in P$, where P is an input metric space with norm $\|\cdot\|_P$. This is usually considered to be a "local" Markov kernel, commonly the Gaussian kernel, in which case the function values are uniformly diffused from p across the metric space:

$$K(p,p') = \frac{1}{w(p)} e^{-\|p-p'\|_P{}^2/\varepsilon} \tag{1}$$

Here, $w(p) = \mathbb{E}_{p''}(e^{-\|p-p''\|_P{}^2/\varepsilon})$, where $\mathbb{E}_{p''}$ denotes the expected value over P, which ensures that the kernel remains Markov.

In our discrete surface setup, we use the diffusion process to analyze how shape descriptors on the same shape relate to each other. Specifically, for each shape S we consider diffusion in its space of descriptor functions $\mathcal{F}$. We are given a set of N such descriptors $\{f_i\}_{i=1}^N, f_i \in \mathcal{F}$. Inspired by the diffusion operator above, we define the nonlinear embedding function $e : \mathcal{F} \to \mathcal{Q} \subset \mathbb{R}^N$ as

$$e(f) = \frac{1}{w(f)} \begin{bmatrix} e^{-\|f-f_1\|^2/\varepsilon} & e^{-\|f-f_2\|^2/\varepsilon} & \dots & e^{-\|f-f_N\|^2/\varepsilon} \end{bmatrix}^T \tag{2}$$

where $w(f) = \sum_{j=1}^N e^{-\|f-f_j\|^2/\varepsilon}$ and ε is the average mean-square distance of f from its nearest neighbors in $\mathcal{F}$. The number of nearest neighbors used to compute ε determines the extent of diffusion. This nonlinear mapping transports square-integrable real functions on the shape S (namely, elements in $\mathcal{F}$) to points in $N-$dimensional space. By evaluating the above embedding function on all descriptors in $\{f_i\}$ and concatenating the results as the columns of a matrix, we obtain

$$\mathbf{E} = \begin{bmatrix} e(f_1) & e(f_2) & \dots & e(f_N) \end{bmatrix}. \tag{3}$$

The mapping e and the matrix $\mathbf{E}$ solely depends on the shape S and the relationships between its descriptors: it is agnostic to any other shape in the collection. If multiple shapes are given, the calculation of the matrices are entirely independent and thus also parallelizable.

For completeness, we note that we also define a "de-embedding" function $d : \mathcal{Q} \subset \mathbb{R}^N \to \mathcal{F}$, which maps an embedded descriptor onto an on-shape function. This function emulates the inverse of the embedding function e. In practice, we compute this function numerically; more details will be provided in the next section.

3.3. Latent space and Inter-shape descriptor transfer

We now consider a collection of shapes $\mathcal{S}$. Assume that for any two shapes $S, \tilde{S} \in \mathcal{S}$ we are given a set of *corresponding* pairs of descriptors $\{f_i \in \mathcal{F}\}_{i=1}^N$, $\{\tilde{f}_i \in \tilde{\mathcal{F}}\}_{i=1}^N$; we call these the "training" descriptors. We can then define and evaluate the diffusion-based embedding functions $e : \mathcal{F} \to \mathcal{Q}$, $\tilde{e} : \tilde{\mathcal{F}} \to \tilde{\mathcal{Q}}$ as per Eq. (2) at each shape separately. This produces matrices $\mathbf{E}, \tilde{\mathbf{E}}$ (Eq. (3)) . We assume that the introduction of nonlinearity in e, $\tilde{e}$ is such that a linear mapping can be found between the embedding spaces $\mathcal{Q}$ and $\tilde{\mathcal{Q}}$. Namely, for a pair of corresponding descriptors $(f, \tilde{f})$ on S and $\tilde{S}$ respectively, there exists a matrix $\mathbf{X}_{S\tilde{S}}$ such that

$$\mathbf{X}_{S\tilde{S}} e(f) = \tilde{e}(\tilde{f}). \tag{4}$$

By applying this assumption on the input descriptors $\{f_i\}$ and $\{\tilde{f}_i\}$, we obtain $\mathbf{X}_{S\tilde{S}} e(f_i) = \tilde{e}(\tilde{f}_i), \forall i$, or equivalently $\mathbf{X}_{S\tilde{S}}\mathbf{E} = \tilde{\mathbf{E}}$. Under the further assumption that $\mathbf{E}$ is invertible for every shape S, the unique linear mapping between the spaces $\mathcal{Q}$ and $\tilde{\mathcal{Q}}$ is then

$$\mathbf{X}_{S\tilde{S}} = \tilde{\mathbf{E}}\mathbf{E}^{-1}. \tag{5}$$

Using Eq. (5) and the linearity assumption Eq. (4) on any corresponding descriptor pair $(f, \tilde{f}) \in \mathcal{F} \times \tilde{\mathcal{F}}$, we obtain

$$\mathbf{L}(f) = \mathbf{E}^{-1} e(f) = \tilde{\mathbf{E}}^{-1} \tilde{e}(\tilde{f}) = \tilde{\mathbf{L}}(\tilde{f}) \tag{6}$$

where we defined the operator $\mathbf{L}(\cdot) = \mathbf{E}^{-1} e(\cdot)$ on S, and equivalently for $\tilde{S}$. This indicates that there exist transformations $\mathbf{L}$ and $\tilde{\mathbf{L}}$, computed independently on each shape, that embed the corresponding descriptors f an $\tilde{f}$ onto the same vector $y = \mathbf{L}(f) = \tilde{\mathbf{L}}(\tilde{f})$ in some latent space $\mathcal{L}$. Using the definition Eq. (3) of matrix $\mathbf{E}$, it can be easily verified that the operator $\mathbf{L}$, applied on the input features $\{f_i\}$ yields $\mathbf{L}(f_i) = \mathbb{1}_i$, where $\mathbb{1}_i$ is the indicator function of $\mathbb{R}^N$ in the i^{th} dimension. This is true for any shape. Thus, the operators $\mathbf{L}$, determined entirely by their respective shapes S, map every descriptor $f_i \in \mathcal{F}$ to an independent dimension in the latent space $\mathcal{L}$.

	cat	*dog*	*michael*	*victoria*	*horse*	*gorilla*	*centaur*	*wolf*
SE, ours	**0.005**	**0.005**	**0.007**	**0.013**	**0.281**	**0.076**	**0.121**	**0.005**
SE, FMaps	2.85	3.14	3.39	3.37	2.71	1.67	3.67	3.38
CE, ours	**0.54**	**0.47**	**0.21**	**0.18**	**0.40**	**0.20**	**0.27**	**0.42**
CE, FMaps	3.78	4.06	4.37	4.36	3.41	2.55	4.68	4.27
CE, Scaled	1.05	1.07	1.03	1.04	1.34	0.90	0.99	0.89
CE, Partial	1.53	1.92	1.04	1.28	2.18	-	1.11	2.25

Table 1: *Mean per-category normalized error on self-transfer ($S \to S$, self-error SE), and inter-shape transfer ($S \to \tilde{S}$, cross-error CE) of descriptors in shapes from the TOSCA dataset [BBK08]. We use HKS, WKS and Multi-Scale Mean Curvature descriptors to build diffusion-based latent spaces on each shape, and select from the same pool some features for testing. We also compare against Functional Maps [OBCS*12] (CE, Fmaps). We also compute the cross-error in transporting descriptors from down-sampled version of a shape to the complete shape (CE, Scaled) and from a shape to a partialized version of itself provided by Rodolá et al. [RCB*17] (CE, Partial). The lowest valued entries for our comparison with the Functional Maps technique is presented in bold font, and for all categories in the TOSCA dataset, our method performs better. All errors are in %.*

Using these concepts we can perform feature transfer between any two shapes S and $\tilde{S}$ with matching sets of training descriptors. Assume we are given a feature f on S. We first embed it into $\mathcal{Q}$ using Eq. (2), to obtain $z = e(f)$. Then, using the linearity assumption Eq. (4), we compute a vector in the embedding $\tilde{\mathcal{Q}}$ of the other shape $\tilde{S}$ as $\tilde{z} = \mathbf{X}_{S\tilde{S}} z$. We then use the "de-embedder" $\tilde{d} : \tilde{\mathcal{Q}} \to \tilde{\mathcal{F}}$ to map this vector onto a descriptor function on $\tilde{S}$ – the transported feature on $\tilde{S}$ is then as follows

$$\tilde{f} = \tilde{d}\left(\mathbf{X}_{S\tilde{S}} e(f)\right) \tag{7}$$

An equivalent expression for feature transport can be written using the latent space operator from Eq. (6). Let us define another shape-specific operator $\mathbf{M} : \mathcal{L} \to \mathcal{F}$ that maps from the linear embedding back to the original descriptor space $\mathcal{F}$, for any shape: $\mathbf{M}(\cdot) := d(\mathbf{E}(\cdot))$. Then, we can write the feature transport Eq. (7) as

$$\tilde{f} = (\tilde{\mathbf{M}} \circ \mathbf{L})(f). \tag{8}$$

Eq. (8) provides a modular decomposition of the inter-shape feature transport into intra-shape segments. Each segment $\mathbf{L}(\cdot)$ transports a descriptor to $\mathcal{L}$, while the composition of the individual segments transports from one individual feature space of one shape, to the latent space, then back to the individual feature space of the other shape, similar in spirit to [HG13]. The modularity, by definition, also ensures cyclic consistency of this approach. This is discussed in Appendix A.

Note. In the problematic case where the matrix $\mathbf{E}$ (Eq. (3)) is not invertible for a given shape, we found that it suffices to replace $\mathbf{E}^{-1}$ by the Moore-Penrose pseudo-inverse $\mathbf{E}^{\dagger}$.

3.4. Dictionary-based representation

For a given shape S, the embedding function e in Eq. (2) transports a feature on the shape to a vector in $\mathcal{Q} \subset \mathbb{R}^N$. The feature function, in our discrete setup, is given by its value at each vertex of the mesh. For a mesh of $|V|$ vertices, the dimension of the feature is also $|V|$. Therefore, $d \circ e$ is a function composition that maps through spaces $\mathbb{R}^{|V|} \to \mathbb{R}^N \to \mathbb{R}^{|V|}$. Since in most cases $|V| >> N$, the inversion will not in general be perfect. Thus, not all functions in the function space can be reconstructed perfectly by this composition. Using a dictionary-based descriptor representation, it is possible to reduce the minor reconstruction error for most descriptors.

To this end, we collect all the input descriptors as the columns of a matrix $\mathbf{F} = \begin{bmatrix} f_1 & f_2 & \dots & f_N \end{bmatrix}$ and apply Singular Vector Decomposition (SVD), $\mathbf{F} = \mathbf{U}\Sigma\mathbf{V}^T$. Let $\mathbf{A}^{(r)}$ denote the first r columns and $\mathbf{A}^{(r,r)}$ the top $r \times r$ sub-matrix of a matrix $\mathbf{A}$. By only considering the first r singular vectors of $\mathbf{F}$, we get $\mathbf{U}^{(r)}\Sigma_S^{(r,r)}\mathbf{V}^{(r)^T}$ as an approximation for $\mathbf{F}$. Taking $\mathbf{U}^{(r)}$ as an $r-$dimensional dictionary for $\mathbf{F}$, we obtain $\Sigma^{(r,r)}\mathbf{V}^{(r)^T}$ as the more compact representation for the vectors of $\mathbf{F}$ in the dictionary. Since the dictionary $\mathbf{U}^{(r)}$ is orthonormal, distances between descriptors in the reduced dictionary representation are the same as the Euclidean distance between the original descriptors. We used this reduced representation ($\mathcal{F}_r \subset \mathbb{R}^r$, as opposed to $\mathcal{F} \subset \mathbb{R}^{|V|}$). of the descriptors for all the experiments in the paper Then, the composition function maps through spaces $\mathbb{R}^{|V|} \to \mathbb{R}^r \to \mathbb{R}^N \to \mathbb{R}^r \to \mathbb{R}^{|V|}$ which should allow for correct reconstruction, provided minimal loss is incurred in transfer of descriptor to dictionary basis. The parameter r is set to a value that enables reconstructing the shape space vector from the dictionary effectively, while at the same time ensuring that the loss incurred by the reduced dictionary representation is minimal. For all the experiments in our paper, we set the value of r to $\min(N, 60)$.

3.5. The de-embedding function

We design d to approximate the inverse of the embedder e, namely so that $e \circ d = Id_{\mathcal{F}}$ is ideally satisfied. Given the dictionary-based representation, we only need to build an embedder-de embedder setup e, d so that $e \circ d = Id_{\mathcal{F}_r}$, since then $\mathbf{U}^{(r)} e \circ d \mathbf{U}^{(r)^T} \approx Id_{\mathcal{F}}$. This improvement in efficiency is another motivation for using the SVD-based dictionary. To that end, we solve a nonlinear optimization problem: Given an embedded descriptor $z \in \mathcal{Q}$, we compute its de-embedded value ($d(z) \in \mathcal{F}_r$) by solving:

$$d(z) = \operatorname*{argmin}_{f \in \mathcal{F}} \|e(f) - z\|_2^2 \tag{9}$$

In practice, this is solved using the MATLAB `fminunc` function. Given the non-convex nature of the optimization, a good initialization for the solution is necessary. We choose as an initial solution the input descriptor that minimizes $\|e(f) - z\|_2$. Since $d(.)$ maps a higher-dimensional space ($\mathbb{R}^N$) to a lower-dimensional one ($\mathbb{R}^r$), this map is reliable and lossless. The loss introduced in the transfer is purely dependent on the dictionary-based representation.

3.6. Efficient Transport of Embedded Functions

The transport to the embedding space and the inverse transport to shape space are provided by Eq. (6) and Eq. (8) as $\mathbf{L}(f) = \mathbf{E}^{-1}e(f)$ and $\mathbf{M}(g) = d(\mathbf{E}g)$ respectively. Since $e(\cdot)$ and $d(\cdot)$ are functions, with enough memory, they could be stored as look-up tables for different input vectors. In this case, the main computational overhead comes from multiplying by matrices $\mathbf{E}^{-1}$ and $\mathbf{E}$ respectively; the complexity of computation of each shape space or embedding space vector is then $\mathcal{O}(n^2)$.

It is possible to write $\mathbf{E}$ as $\mathbf{E} = \mathbf{A}\boldsymbol{\Theta}$, where $\boldsymbol{\Theta}$ a diagonal matrix and $\mathbf{A}$ consists of columns which are the columns of $\mathbf{E}$, but normalized. In this manner, $\mathbf{E}^{-1}$ can also be rewritten as $\mathbf{E}^{-1} = \boldsymbol{\Theta}^{-1}\mathbf{A}^{-1}$. In this manner, $\mathbf{L}(f) = \boldsymbol{\Theta}^{-1}(\mathbf{A}^{-1}e(f))$ and $\mathbf{M}(g) = d(\mathbf{A}\boldsymbol{\Theta}g)$. Therefore, if the look-up embedding function is re-defined to encompass $\mathbf{A}^{-1}e(\cdot)$ and the de-embedding function re-defined to encompass $d(\mathbf{A}\cdot)$, then the transport across the latent shape space is just the product $\boldsymbol{\Theta}_M\boldsymbol{\Theta}_L^{-1}$; as the product of two diagonal matrices, this step has computational complexity $\mathcal{O}(n)$. Finally, since the columns of A are unit-norm, each entry of $\boldsymbol{\Theta}_L$ and $\boldsymbol{\Theta}_M$ is representative of how strongly each of the descriptors influences the final transport. Therefore, if there are descriptors which are more desirable than others in their influence of the final transport, this could be achieved by multiplying with an intermediate diagonal weight matrix $\boldsymbol{W}$ to obtain $\boldsymbol{\Theta}_M\boldsymbol{W}\boldsymbol{\Theta}_L^{-1}$.

	CE	SE		CE	SE		CE	SE
human	16.44	15.87	*chair*	18.97	16.79	*plier*	15.63	15.26
octopus	15.51	14.92	*fish*	16.12	15.80	*bearing*	16.21	12.15
glass	18.26	15.26	*table*	15.71	13.78	*bird*	17.69	16.11
airplane	16.41	16.22	*teddy*	17.12	16.88	*four-leg*	16.46	16.19
ant	15.43	15.01	*hand*	16.38	16.14	*bust*	18.95	17.90
cup	17.04	15.30	*armadillo*	16.86	16.69	*mech*	14.96	13.68
			vase	16.97	15.17			

Table 2: *Mean per-category normalized error on transfer of HKS / WKS descriptors, using latent spaces built on contextual descriptors of Kalogerakis et al. [KHS10]. Evaluation is done on the Princeton Segmentation Benchmark [CGF09]. Self-error (SE) and cross-error (CE) are defined as in Table 1. All errors are in %.*

4. Results

In this section, we validate the method discussed in Section 3, discuss applications and evaluate them both qualitatively and quantitatively. We assume that the input training shape descriptors are all in the $[0, 1]$ range. While this is not generally the case, it has been argued [GSTOG16] that shape correspondences benefit less from matching the exact descriptor values at the shape vertices and more from matching descriptor *ranks* – namely, the ordering of shape vertices if we were to sort them by their descriptor values. Since normalizing a function does not affect the ordering of vertices in terms of that function, we normalize all input descriptors in preprocessing.

4.1. Validation

For validation, we typically measure the normalized error of transferring shape descriptors between pairs of shapes, or from one shape to itself. We refer to the former type of error as *cross-error*, and to the latter as *self-error*. Namely, given any transport technique $T_{S\tilde{S}}$ between two shapes S and $\tilde{S}$, and a set of *corresponding* descriptor pairs $(f_k, \tilde{f}_k)$ $k = 1, 2, \dots, N$ on both shapes, the cross-error is measured via the sum $\frac{1}{N}\sum_{k=1}^{N} \frac{\|T_{S\tilde{S}}(f_k) - \tilde{f}_k\|_2}{\|\tilde{f}_k\|_2}$. The self-error is defined equivalently, where now S and $\tilde{S}$ coincide, and the transport technique corresponds to the $d \circ e$. We typically also average these normalized errors across all shape pairs of a given shape category, depending on the type of dataset used.

Embedding techniques. In Fig. 2, we compare the diffusion-based embedding against other embedding techniques, used for transporting shape descriptors from one shape to another. We use shapes from the "horse" category of TOSCA dataset, and evaluate

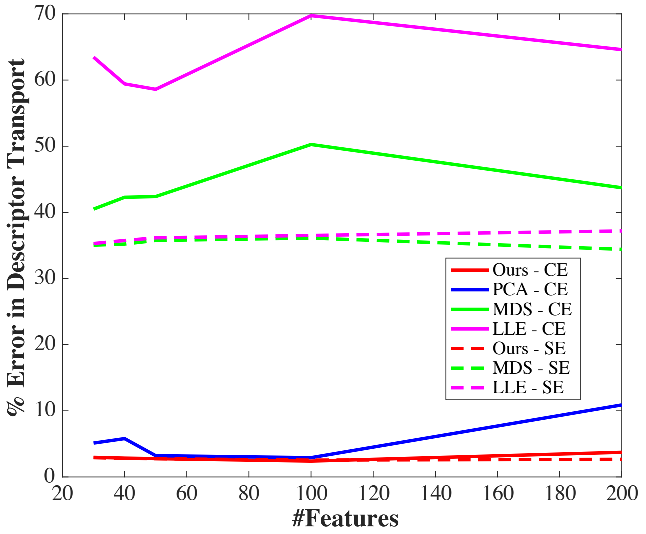

Figure 2: *Comparison between different embedding techniques for the task of descriptor transport. Each curve shows the mean normalized transfer error (on unseen HKS descriptors) for respective methods, using embedding spaces built with varying numbers of HKS descriptors (x-axis) for "horse" shapes in the TOSCA dataset [BBK08]. Bold lines demonstrate self-errors (SE), while dashed-lines demonstrate cross-errors (CE), both defined as in Table 1. Our diffusion-based technique outperforms all others (PCA, MDS, LLE [RS00, Bas99, Pea01]) in cross-error. It also outperforms MDS and LLE in self-error. Note that PCA has zero self-error due to itself being a full rank representation.*

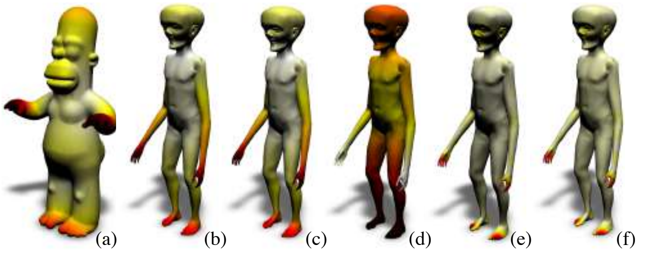

Figure 3: *Transport of WKS descriptor from homer to alien. (a) Input descriptor on homer (b) Ground-truth corresponding descriptor on alien. The corresponding output descriptors are shown after using the following embedding techniques: (c) our diffusion-based embedding (d) PCA [Pea01] (e) MDS [Bas99] (f) LLE [RS00]*

#Shapes	2	4	8	10	12	14	18	20
[HG13]	1.52	4.76	18.60	37.50	43.32	60.98	104.76	142.20
Ours	0.018	0.040	0.256	0.500	0.840	1.344	2.844	3.860
Ours, Eff.	**0.018**	**0.020**	**0.039**	**0.058**	**0.060**	**0.082**	**0.090**	**0.089**

Table 3: *Timing comparisons. The first row shows times shows times (in s) for computing maps across multiple shapes using [HG13]. The last two rows refer to our method, its vanilla version and the efficient modification discussed in Section 3.6. The modularity of our technique is demonstrated by the lower computation time. The lowest time taken for map computation is presented in bold font.*

the transport error across embedding spaces built on different numbers of HKS descriptors. The relative (%) error of transporting new descriptors across different shapes, as well as the self-transport error (computed by cyclically transporting a descriptor on a single shape to the intermediate latent space and back to the same shape) is very high for Locally Linear Embedding (LLE) [RS00] and Incremental Multidimensional Scaling [Bas99]. The percentage self-transport error for Principal Components Analysis (IPCA) [Pea01] is trivially zero, since IPCA essentially utilizes a full rank matrix of principal component vectors to project the input descriptor into the embedding space. However, the diffusion-based embedding is seen to always outperform the PCA-based embedding when transporting across different shapes. It can also be seen that the self-transport and cross-transport errors of the diffusion-based embedding are very similar in magnitude, indicating the generalizability of this approach. We also provide a qualitative comparison in Fig. 3. In this example, we transport an unseen new WKS descriptor between two shapes, via embedding spaces built using different WKS descriptors. Our diffusion-based embedding space, with a 3% descriptor reconstruction error, outperforms other methods (PCA - 15%, MDS- 43%, and LLE - 44% respectively).

Self transport. The embedding/de-embedding functions e, d are computed by means of the diffusion-based embedding; the former directly (Eq. (2)) and the latter via optimization (Eq. (9)). Given the dimensionality issues affecting the invertibility of e, discussed in Section 3.3, and the nonlinear nature of the optimization, this inversion might in general be lossy. Moreover, the use of a reduced dictionary basis to represent the input descriptor features can contribute an additional lossy component. We validate the error introduced by the inversion process and the introduction of the dictionary in the first two rows of Table 1, by evaluating the deviation of the composition function $d \circ e$ from identity on a shape. Namely, we select a feature on S, transport it back to S via Eq. (7) and measure the normalized reconstruction error, referred to as the self-error. Shapes are taken from the TOSCA dataset, and the latent space is built using a set of 20 Wave Kernel Signature (WKS) [ASC11] descriptors on the shape. We select the test features being transported from the same training set of input descriptors used to build the latent space. According to the table, the average error, normalized by descriptor magnitude and averaged within shape category, is fairly low; which indicates that the embedding/de-embedding process has almost complete reconstruction capability. We also compare to classical functional maps [OBCS*12].

Modularity. An important contribution of this work is its modularity in computing maps, which speeds up map computation across many shapes, as can be seen in Table 3. For maps between multiple shapes, mean time for map computation goes up in quadratic fashion for the non-modular approach of Huang and Guibas [HG13], since all $\binom{N}{2}$ maps need to be computed together. Instead, modularity provides a way to compute these maps in parallel, speedens our technique, especially in the presence of the modifications discussed in Section 3.6. A quadratic increase is observed in timing of Huang and Guibas when number of shapes increase, while our method manages to parallelize this process and drastically cut down on computation time.

Descriptor transfer. Here, we validate the quality of descriptor transfer between different shapes, using the transport equation of Eq. (8). We build the latent space using 50 Heat Kernel Signature (HKS) [SOG09], 50 Wave Kernel Signature (WKS) and 50 multi-scale mean curvature descriptors [MDSB03], then use the same pool of descriptors for transport and testing. We also compare the map provided by our technique to classical functional maps [OBCS*12], which also aims at transporting functions. We present results on selected shapes of the TOSCA dataset in rows 3 and 4 of Table 1, in terms of the inter-shape transport cross-error. As is evident, introducing the nonlinearity in the first embedding step of the mapping improves the performance of the transport function significantly.

Transport to downsampled/partial meshes. The effect of using our embedding-based technique to transport between differently sampled versions of meshes in the TOSCA dataset is shown in Table 1(row 5). We report the mean-normalized percentage cross error in descriptor transfer from a downsampled mesh to the original full-resolution mesh. While there is some loss of descriptor accuracy due to scaling up of these meshes, the descriptors are mostly transported accurately. We also perform transport from full onto partial meshes of the TOSCA dataset in Table 1(row 6). We report the mean-normalized percentage cross error in descriptor transfer from the complete TOSCA shape to all partializations of the shape provided in the dataset of Rodolá et al. [RCB*17]. The experimental setup for both these experiments is as in the previous paragraph.

Training/Testing on different feature types. In Table 2, we validate the quality of feature transport when the set of test features being transported is different from the training set used for building the latent space. Here, we build latent spaces across every shape category of the Princeton Segmentation Benchmark [CGF09], using the features of [KHS10] – 628 features in total. We then transport 10 HKS and 10 WKS features between shapes of each category, and measure the mean percentage error between the ground-truth and transported descriptors. High self-error on certain categories indicates a failure of the embedder / de-embedder framework, which was trained on the contextual features of [KHS10], to reconstruct the more classical HKS/WKS features. Naturally, failure of the embedder / de-embedder framework also results in the inter-shape transporting mechanism failing – which explains why the cross-error is strictly greater than the self-error in all categories. Note also that in order to accurately compute the HKS/WKS features used for transport testing, the meshes need to be manifold, which is not the case in several shape categories. This could affect the quality of transport, since the training features do not suffer from such inaccuracies. In the table, while the cross-error is always more than self-error, the two errors are always similar in magnitude. This shows the generalizability of diffusion-embedding based descriptor transport, as long as the embedding spaces are built with descriptors in correspondence.

Dependence on feature type. Fig. 4 and Fig. 5 qualitatively highlight the effect of transporting features of a different kind than the training features used for building the latent space. Fig. 4 illustrates transport of a WKS feature across multiple shape categories using latent spaces trained on the features from [KHS10]. Despite some inaccuracies in the feature transport, prominent aspects of the feature are preserved – this includes feature extrema, e.g. on the handle and central parts of *vase*, the wings, body and tail of *airplane* and the nose, fins and hump of *fish*. This result indicates generalizability of our technique: it is possible to train on certain kinds of features and transport features of different kinds.

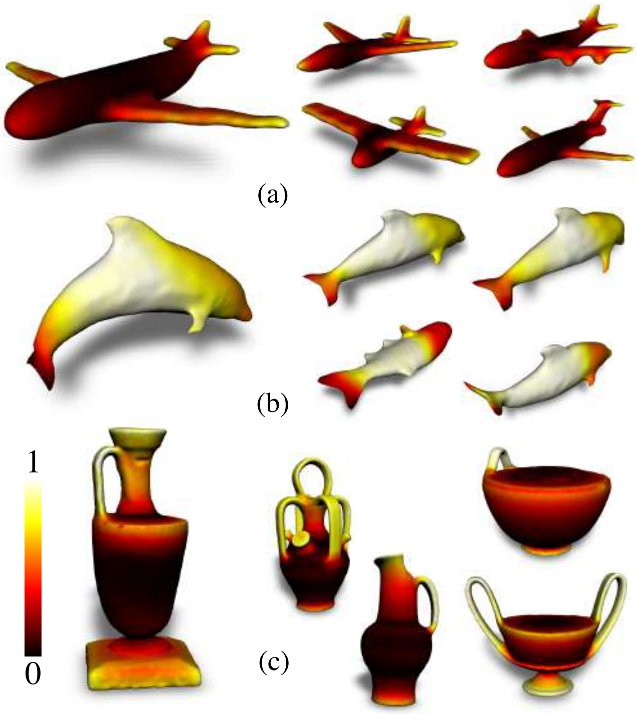

Figure 4: *Qualitative results of transferring HKS / WKS descriptors across shapes, using latent spaces built on a different kind of descriptors [KHS10]. The image on the left shows the input descriptor on a source shape. Images of the right show the same descriptor transported onto different shapes in the same shape category – shown are (a)* ***airplane****, (b)* ***fish****, and (c)* ***vase*** *categories. In all three categories, the input image has certain high-valued and low-valued regions as denoted by red and white colors respectively, and this is reflected in the transported descriptors on other shapes as well.*

When transport is done using the same kind of geometric features both for training and testing, we observe good feature reconstruction, as illustrated in Fig. 5. Note that the features being transported are different than the ones used for training, although they are both of the same HKS/WKS type. In the *cat / dog* experiment, the latent spaces were built using 50 WKS descriptors, while the feature transported is a HKS descriptor. In the *michael / victoria* experiment, the latent spaces were built using 50 HKS descriptors, and a WKS descriptor is being transported. In both cases, almost perfect reconstruction is achieved.

Cyclic Consistency. Fig. 6 shows transport of a part label feature on a shape S back to S via multiple intermediate shapes in a network of shapes. Notice that features are generally transported consistently, even when on the intermediate shapes the signal might

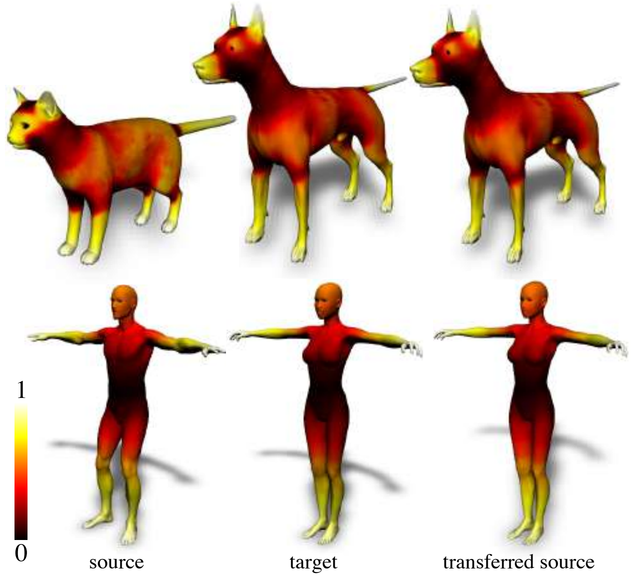

Figure 5: *Feature transfer across non-isometric shapes from the TOSCA dataset, using the same kind of feature for training as for testing. In the first row, the latent spaces were built on WKS descriptors, but the feature being transferred is the HKS descriptor at time 0. In the second row, the latent spaces were built on HKS descriptors, but the feature being transferred is the WKS descriptor at time 0. Left to right: the source descriptor on the source mesh (HKS/WKS for first/second row respectively), the matching ground-truth HKS/WKS descriptor on the target mesh, and the source descriptor transferred onto the target shape.*

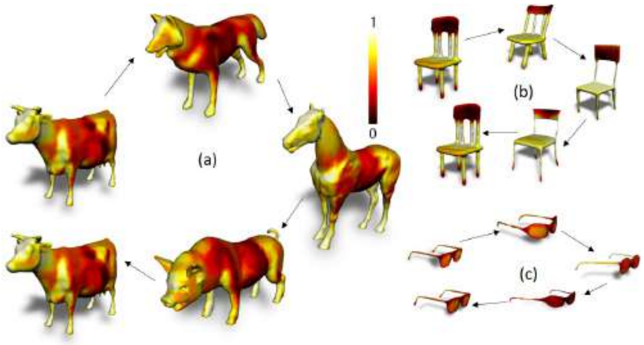

Figure 6: *Transport of soft part descriptors across a shape network back to the source shape. (a)* ***four-leg*** *- torso, (b)* ***glasses*** *- lens, (c)* ***chair*** *- backrest.*

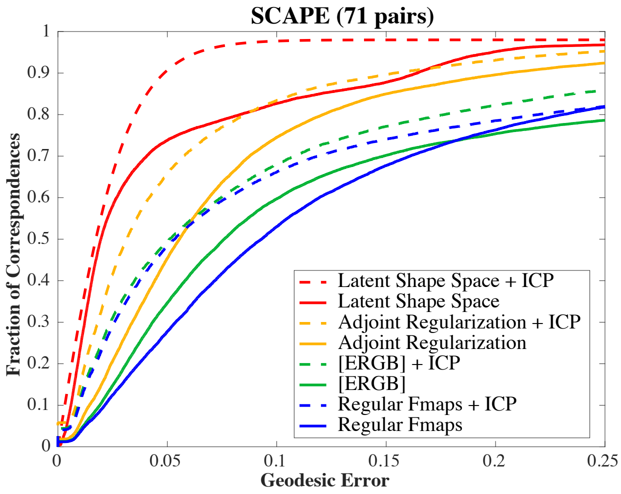

Figure 7: *Map estimation accuracy (RC curves) on pairs of shapes using our latent shape space framework, compared against the adjoint-regularized [HO17], coupled [ERGB16], and classical Functional Maps [OBCS* 12] for shapes in SCAPE [ASK*05].*

become corrupted. The minor deviation from identity between the the final output on S and the original input is not due to lack of theoretical consistency in the framework but to losses caused during the embedding / de-embedding process.

4.2. Applications

Segmentation Transfer. Our method can be used to transport semantically meaningful descriptors across shapes; we can apply it for transporting segmentation labels. Given a segmentation of an input shape, we can use Eq. (8) to transport it onto a different shape by transporting the indicator functions corresponding to individual shape parts. Fig. 8 shows qualitative results of this process on a variety of shape categories. The training features $\{f_i\}, f_i \in \mathcal{F}$ used to build the latent spaces for the shapes were produced using [KHS10], for a total of 628 features per shape. Note, however, that our method is more applicable to "soft" descriptors, not discrete part-indicator functions. In general, transporting an indicator function from S using Eq. (8) will not produce another indicator function on $\tilde{S}$. Instead, it will be a soft segmentation: a smoothly varying function on $\tilde{S}$, with higher values concentrated on the vertices of $\tilde{S}$ that are more likely to correspond to the respective input semantic part that is being transported. We convert this soft labeling into a hard segmentation via post-processing, described below. Using this approach, we are able to get viable segmentation results, though not of equivalent quality to more sophisticated approaches tailored to segmentation specifically (eg. Huang et al. [HKG11]).

Given a "softly" transported segmentation labeling, we convert it to a discrete labeling function in a post-processing step. To this end, we utilize the known layout of parts in the input S and aim to preserve the adjacencies between parts in $\tilde{S}$: e.g., if $S, \tilde{S}$ are human shapes, a mesh segment corresponding to a leg cannot be adjacent to a segment corresponding to the head. We encode these adjacencies in a graph involving the connected components of the part labeling in S and $\tilde{S}$. Post-processing proceeds as follows

- Transfer all part labels from S to $\tilde{S}$. For m parts, this results in m functions on $\tilde{S}$, each indicating the likelihood for a specific part at each vertex of $\tilde{S}$.
- Assign to each vertex on $\tilde{S}$ the label corresponding to the function (among the m) that has the highest value. This provides us an initial labeling of the output mesh.
- Extract connected components on $\tilde{S}$ using this initial labeling.

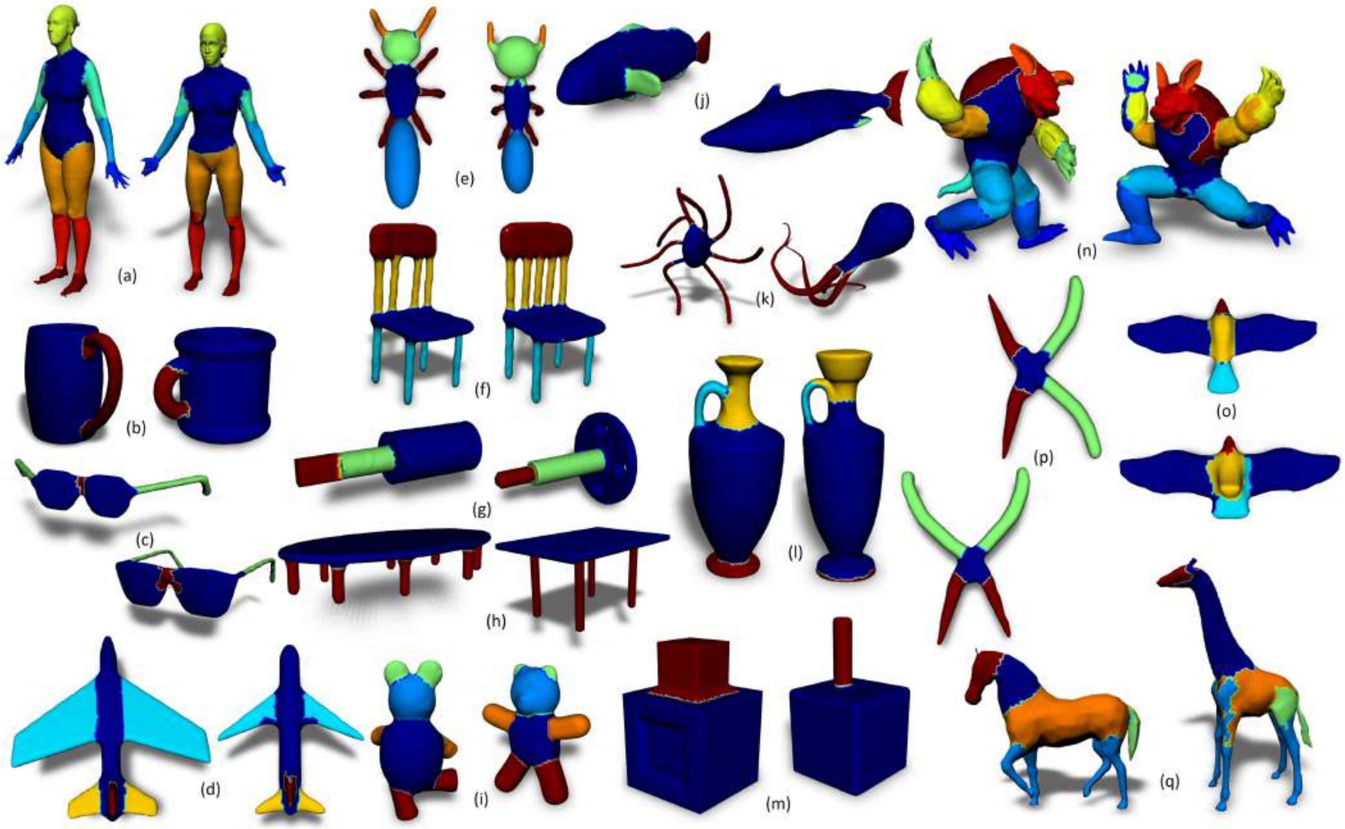

Figure 8: *Segmentation transfer between non-isometric shapes. In each of the shape pairs, the first image shows the ground-truth part labeling on the first shape, while the second image shows the same labeling transferred onto the second shape via our latent space. The shape categories are taken from the Princeton Segmentation Benchmark [CGF09]: (a)* ***human****, (b)* ***cup****, (c)* ***glasses****, (d)* ***airplane****, (e)* ***ant****, (f)* ***chair****, (g)* ***bearing****, (h)* ***table****, (i)* ***teddy****, (j)* ***fish****, (k)* ***octopus****, (l)* ***vase****, (m)* ***mech****, (n)* ***armadillo****, (o)* ***bird****, (p)* ***plier****, (q)* ***four-leg****.*

- Mark any components whose number of vertices is less than 5% of the number of vertices in $\tilde{S}$ as un-labeled.
- Fix the labels for all large components (larger than 15% of $\tilde{S}$).
- For the remaining components, if their labels violate expected part adjacencies, as determined by the input graph on S, relabel them by propagating labels iteratively: for any unlabeled component neighboring two or more labeled components, mark it with the label of the largest of these neighboring components.

Note that while we aim to preserve adjacencies between semantic parts, we do not preserve the exact structure of the component graph, since the number of connected components that belong to the same semantic part may vary across shapes - see e.g. Fig. 8(g) where the two chairs have different numbers of slats on their back, or Fig. 8(i) where the tables have different numbers of legs. Additionally, whenever a part label does not appear at all in the initial transferred segmentation, the part adjacencies cannot be enforced.

Shape Correspondence. Most shape correspondence techniques aim at obtaining either point-to-point maps or "soft" probabilistic maps, which can then be converted into point-to-point maps using various techniques. Since our technique is based on descriptor transfer, it also produces a soft map, which can then be used to indirectly place shapes in correspondence, in a spirit similar to other approaches that focus on functional transport, e.g. as in the classical functional map case [HO17, ERGB16, OBCS*12]. Note that in contrast to these approaches, our method does not provide an explicit linear map. In Fig. 7 we perform quantitative comparisons to other such linear descriptor mapping techniques. For these experiments, we use our latent space-based feature transfer mechanism vs. the linear mapping technique to transport the Laplace-Beltrami eigenfunctions between two shapes; as shown in [OBCS*12], this transport can be used to extract point-to-point maps. As in [HO17], we use 71 pairs of shapes from the SCAPE dataset, with 20 WKS descriptors [ASC11]. Our selected dictionary uses 60 dimensions ($r = 60$), to ensure a fair comparison to the other techniques. Contrary to some of the methods we compare against, we do not post-process the results using ICP, since obtaining a point-to-point map is not the end goal of our approach; even so, our method performs comparably to these state-of-the-art correspondence techniques. We also show the transport of WKS descriptors between shape pairs and compare it to the performance of classical functional maps in Table 1. We build the embedding space and functional map based on 50 WKS descriptors in correspondence and then test it on 50 new WKS descriptors.

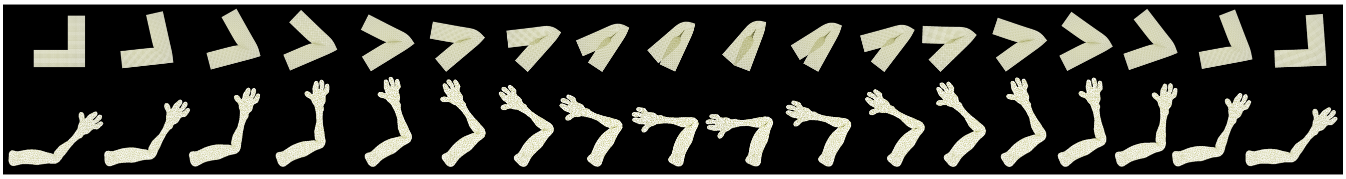

Figure 9: *Multiple uniformly sampled frames from two corresponding video sequences (boxes - top row, and arm - bottom row) generated by rendering simple animated 2D meshes. Frames extracted from these sequences are used as training data to build a latent space, and then tested on interpolation / extrapolation tasks, where the boxes sequence serves as the source animation and the arm sequence is being synthetically generated. See text for more details.*

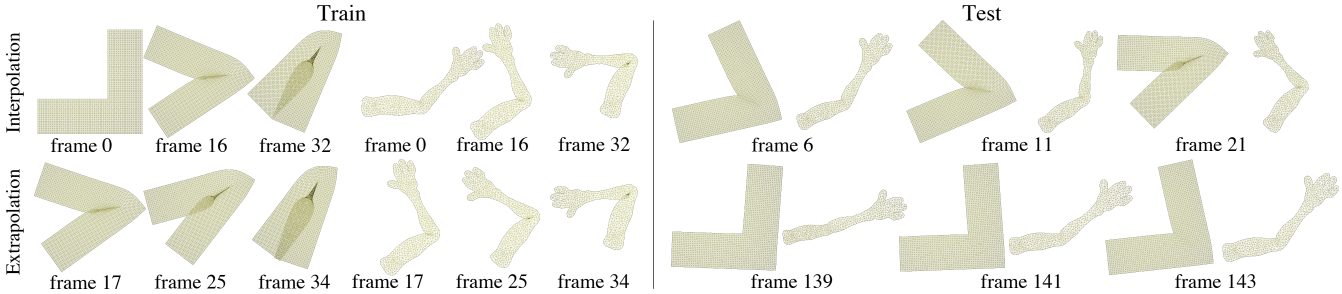

Figure 10: *Top row: Sequence interpolation. On the left, we show three sample boxes and arm mesh sequence frames (extracted from Fig. 9) used to train for the interpolation task. On the right, we show three boxes frames used for testing, and the corresponding arm frames synthesized by transporting the boxes frames using our diffusion-based latent space method. The test frames are taken to be intermediate frames in the video sequence. Bottom row: Sequence extrapolation. Similarly to the top row, we use parts of both sequences to train a latent space, and test by transporting from the boxes sequence to the arm sequence. However, now the training frames form a contiguous subsequence of the original ground truth video (Fig. 9) and we test on the remaining subsequence.*

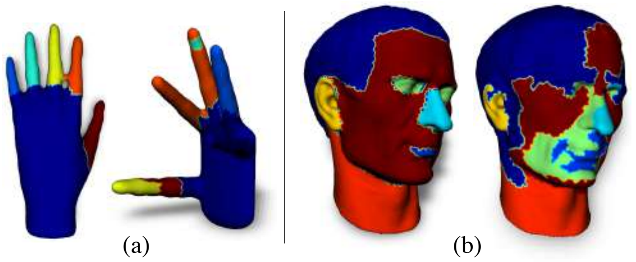

Figure 11: *Failure cases. We show failure cases in part segmentation transfer using our method on (a)* ***finger*** *(b)* ***head****, from the Princeton Segmentation Benchmark dataset [CGF09]*

Video Interpolation/Extrapolation. As an additional proof of concept and future application beyond shape correspondence, we demonstrate interpolation and extrapolation of simple video data. This application illustrates the generative potential of the diffusion-based latent spaces for generating data. The input and ground truth is a pair of corresponding video sequences, each with 200 frames, produced by rendering two similar 2D meshes – a mesh representing a human arm, and a mesh consisting of two rectangular boxes attached at one end, which is a rough geometric proxy for a human arm shape. Both meshes are animated (by modifying rig parameters) in a synchronized way to perform similar-looking actions, corresponding to an arm flexing; since the animation is synchronized, the two videos correspond in a frame-by-frame basis, samples of which are illustrated in Fig. 9. We can then think of the video frames as "descriptors" ; using a subset of the compatible video frames as training data to build a latent space, we can interpolate or extrapolate frames for one of the meshes by using frames from the other mesh as test data and transporting them. Thus we can use the *boxes* sequence to synthetically generate the *arm* sequence, and compare against the ground truth.

We show such experiments in Fig. 10. In the first experiment (interpolation - Fig. 10(a)), we train the shape space on the extremal frames of the sequence (first row), together with other frames sampled from the two sequences every 20 frames. For the remainder of the sequence, we use the *boxes* sequence as test data and transport it to generate the intermediate frames for the *arm* mesh. As expected, the quality of the interpolation improves with more training frames. In the second experiment (extrapolation - Fig. 10(b)), we build the latent space based on a contiguous part of the two sequences corresponding to 120 frames of the respective videos, roughly corresponding to the "flexing" part of the sequence. The remainder of the sequence (the "opening" of the arms) is then extrapolated from the *boxes* to the *arm* mesh. We observe that the stretching-out of the arm is generated fairly reliably. While this is a simple example, it indicates that our technique could potentially be applied to more complex parameter-based generative scenarios, as long as the parameters can be used as descriptors and can be interpolated/extrapolated.

5. Conclusion and Future Work

Modular processing of shapes is key to analyzing ever-growing networks of shapes. In this paper, we seek modularity in the problem of placing shapes in correspondence. In our framework, matching shape descriptors are utilized to build nonlinear embedding spaces, which can further be related through a latent space. The embedding spaces are built independently for each shape based on the interdependencies of the on-shape descriptors, computed exclusively on each individual shape. This makes for a modular construction, which can be used to perform descriptor transport across shapes, establish shape part correspondences and interpolate/extrapolate data. The modular nature of our framework makes it possible to incrementally add shapes to the shape network and place them in correspondence with the ones already present, without further recomputation for the existing shapes.

A limitation of our method is the dependency on the input descriptors; transporting a descriptor of a much different type than the ones used to build the latent space might lead to suboptimal transport. This can be seen when segmentations, that are essentially concatenated part indicator functions, are transported across shapes, whose embedding spaces have been built using soft descriptors. An example of this is seen in Fig. 11(a), where most soft descriptors do not distinguish between the different fingers of a hand, and that shows up as a problem when a segmentation of one hand is attempted to be transferred to another. In Fig. 11(b), most parts of the face are islands within other parts of the face, and this also causes a similar problem, where soft descriptors, which are used to build the embedding spaces have more gradual means of changing value from the eyes or nose to the rest of the face, while this is not seen with segmentations and therefore lead to poor part transfer. As a future direction, we aim to build embedding spaces that are not dependent on descriptors, but instead can be built directly from shapes represented in different modalities, eg. meshes, scanned point clouds etc. An example would be to apply learning-based techniques, especially given the recent surge in deep learning techniques, which has produced deep networks able to process many different modalities of 3D data. Automatically learning latent spaces that can understand the essence of a shape, preserve modularity, and build consistent correspondences across different shapes would be an exciting avenue for extending this work.

Acknowledgements: This work was supported by NSF grants IIS-1528025, DMS-1546206, as well as gifts from Amazon AWS and Autodesk.

References

[APL15] Aigerman N., Poranne R., Lipman Y.: Seamless surface mappings. ACM Transactions on Graphics (TOG) 34, 4 (2015), 72. 2

[ASC11] Aubry M., Schlickewei U., Cremers D.: The wave kernel signature: A quantum mechanical approach to shape analysis. In Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on (2011), IEEE, pp. 1626–1633. 3, 6, 9

[ASK*05] Anguelov D., Srinivasan P., Koller D., Thrun S., Rodgers J., Davis J.: Scape: shape completion and animation of people. In ACM transactions on graphics (TOG) (2005), vol. 24, ACM, pp. 408–416. 8

[Bas99] Basalaj W.: Incremental multidimensional scaling method for database visualization. In Visual Data Exploration and Analysis VI (1999), vol. 3643, International Society for Optics and Photonics, pp. 149–159. 3, 6

[BBK06] Bronstein A. M., Bronstein M. M., Kimmel R.: Generalized multidimensional scaling: a framework for isometry-invariant partial surface matching. Proceedings of the National Academy of Sciences 103, 5 (2006), 1168–1172. 2

[BBK08] Bronstein A. M., Bronstein M. M., Kimmel R.: Numerical geometry of non-rigid shapes. Springer Science & Business Media, 2008. 4, 6

[CFG*15] Chang A. X., Funkhouser T., Guibas L., Hanrahan P., Huang Q., Li Z., Savarese S., Savva M., Song S., Su H., et al.: Shapenet: An information-rich 3d model repository. arXiv preprint arXiv:1512.03012 (2015). 3

[CGF09] Chen X., Golovinskiy A., Funkhouser T.: A benchmark for 3D mesh segmentation. ACM Transactions on Graphics (Proc. SIGGRAPH) 28, 3 (Aug. 2009). 5, 7, 9, 10

[CLL*05] Coifman R. R., Lafon S., Lee A. B., Maggioni M., Nadler B., Warner F., Zucker S. W.: Geometric diffusions as a tool for harmonic analysis and structure definition of data: Multiscale methods. Proceedings of the National Academy of Sciences of the United States of America 102, 21 (2005), 7432–7437. 3

[COC14] Corman É., Ovsjanikov M., Chambolle A.: Supervised descriptor learning for non-rigid shape matching. In European Conference on Computer Vision (2014), Springer, pp. 283–298. 2

[CRA*17] Cosmo L., Rodolà E., Albarelli A., Mémoli F., Cremers D.: Consistent partial matching of shape collections via sparse modeling. In Computer Graphics Forum (2017), vol. 36, Wiley Online Library, pp. 209–221. 2

[DMB*17] Denitto M., Melzi S., Bicego M., Castellani U., Farinelli A., Figueiredo M. A. T., Kleiman Y., Ovsjanikov M.: Region-based correspondence between 3d shapes via spatially smooth biclustering. In The IEEE International Conference on Computer Vision (ICCV) (Oct 2017). 2

[ERGB16] Eynard D., Rodola E., Glashoff K., Bronstein M. M.: Coupled functional maps. In 3D Vision (3DV), 2016 Fourth International Conference on (2016), IEEE, pp. 399–407. 8, 9

[GSTOG16] Ganapathi-Subramanian V., Thibert B., Ovsjanikov M., Guibas L.: Stable region correspondences between non-isometric shapes. In Computer Graphics Forum (2016), vol. 35, Wiley Online Library, pp. 121–133. 2, 5

[HG13] Huang Q.-X., Guibas L.: Consistent shape maps via semidefinite programming. In Computer Graphics Forum (2013), vol. 32, Wiley Online Library, pp. 177–186. 1, 4, 6

[HKG11] Huang Q., Koltun V., Guibas L.: Joint shape segmentation with linear programming. In ACM transactions on graphics (TOG) (2011), vol. 30, ACM, p. 125. 8

[HO17] Huang R., Ovsjanikov M.: Adjoint map representation for shape analysis and matching. Computer Graphics Forum 36, 5 (2017), 151–163. URL: http://dx.doi.org/10.1111/cgf.13253, doi:10.1111/cgf.13253. 8, 9

[HS14] Havlena M., Schindler K.: Vocmatch: Efficient multiview correspondence for structure from motion. In European Conference on Computer Vision (2014), Springer, pp. 46–60. 2

[HWG14] Huang Q., Wang F., Guibas L.: Functional map networks for analyzing and exploring large shape collections. ACM Transactions on Graphics (TOG) 33, 4 (2014), 36. 2, 3

[KCLZ10] Keller Y., Coifman R. R., Lafon S., Zucker S. W.: Audio-visual group recognition using diffusion maps. IEEE Transactions on Signal Processing 58, 1 (2010), 403–413. 3

[KHS10] Kalogerakis E., Hertzmann A., Singh K.: Learning 3d mesh segmentation and labeling. In ACM Transactions on Graphics (TOG) (2010), vol. 29, ACM, p. 102. 5, 7, 8

[KLF11] Kim V. G., Lipman Y., Funkhouser T.: Blended intrinsic maps. In ACM Transactions on Graphics (TOG) (2011), vol. 30, ACM, p. 79. 2

[Lév06] Lévy B.: Laplace-beltrami eigenfunctions towards an algorithm that" understands" geometry. In Shape Modeling and Applications, 2006. SMI 2006. IEEE International Conference on (2006), IEEE, pp. 13–13. 3

[LF09] Lipman Y., Funkhouser T.: Möbius voting for surface correspondence. In ACM Transactions on Graphics (TOG) (2009), vol. 28, ACM, p. 72. 2

[LT14] Lederman R. R., Talmon R.: Common manifold learning using alternating-diffusion. 3

[MDSB03] Meyer M., Desbrun M., Schröder P., Barr A. H.: Discrete differential-geometry operators for triangulated 2-manifolds. In Visualization and mathematics III. Springer, 2003, pp. 35–57. 7

[Mém07] Mémoli F.: On the use of gromov-hausdorff distances for shape comparison. 2

[NLCK06] Nadler B., Lafon S., Coifman R. R., Kevrekidis I. G.: Diffusion maps, spectral clustering and reaction coordinates of dynamical systems. Applied and Computational Harmonic Analysis 21, 1 (2006), 113–127. 3

[NLKC06] Nadler B., Lafon S., Kevrekidis I., Coifman R. R.: Diffusion maps, spectral clustering and eigenfunctions of fokker-planck operators. In Advances in neural information processing systems (2006), pp. 955–962. 3

[OBCS*12] Ovsjanikov M., Ben-Chen M., Solomon J., Butscher A., Guibas L.: Functional maps: a flexible representation of maps between shapes. ACM Transactions on Graphics (TOG) 31, 4 (2012), 30. 1, 2, 4, 6, 7, 8, 9

[OMMG10] Ovsjanikov M., Mérigot Q., Mémoli F., Guibas L.: One point isometric matching with the heat kernel. In Computer Graphics Forum (2010), vol. 29, Wiley Online Library, pp. 1555–1564. 2

[OMPG13] Ovsjanikov M., Mérigot Q., Pătrăucean V., Guibas L.: Shape matching via quotient spaces. In Computer Graphics Forum (2013), vol. 32, Wiley Online Library, pp. 1–11. 2

[Pea01] Pearson K.: Liii. on lines and planes of closest fit to systems of points in space. The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science 2, 11 (1901), 559–572. 3, 6

[RCB*17] Rodolà E., Cosmo L., Bronstein M. M., Torsello A., Cremers D.: Partial functional correspondence. In Computer Graphics Forum (2017), vol. 36, Wiley Online Library, pp. 222–236. 4, 7

[RLF09] Rustamov R. M., Lipman Y., Funkhouser T.: Interior distance using barycentric coordinates. In Computer Graphics Forum (2009), vol. 28, Wiley Online Library, pp. 1279–1288. 2

[RPSS10] Ruggeri M. R., Patanè G., Spagnuolo M., Saupe D.: Spectral-driven isometry-invariant matching of 3d shapes. International Journal of Computer Vision 89, 2 (2010), 248–265. 2

[RS00] Roweis S. T., Saul L. K.: Nonlinear dimensionality reduction by locally linear embedding. science 290, 5500 (2000), 2323–2326. 3, 6

[SNB*12] Solomon J., Nguyen A., Butscher A., Ben-Chen M., Guibas L.: Soft maps between surfaces. In Computer Graphics Forum (2012), vol. 31, Wiley Online Library, pp. 1617–1626. 2

[SOG09] Sun J., Ovsjanikov M., Guibas L.: A concise and provably informative multi-scale signature based on heat diffusion. In Computer graphics forum (2009), vol. 28, Wiley Online Library, pp. 1383–1392. 3, 7

[SRGB14] Solomon J., Rustamov R., Guibas L., Butscher A.: Earth mover's distances on discrete surfaces. ACM Transactions on Graphics (TOG) 33, 4 (2014), 67. 2

[TCL*13] Tam G. K., Cheng Z.-Q., Lai Y.-K., Langbein F. C., Liu Y., Marshall D., Martin R. R., Sun X.-F., Rosin P. L.: Registration of 3d point clouds and meshes: a survey from rigid to nonrigid. IEEE transactions on visualization and computer graphics 19, 7 (2013), 1199–1217. 2

[VKZHCO11] Van Kaick O., Zhang H., Hamarneh G., Cohen-Or D.: A survey on shape correspondence. In Computer Graphics Forum (2011), vol. 30, Wiley Online Library, pp. 1681–1707. 2

[WHC*16] Wei L., Huang Q., Ceylan D., Vouga E., Li H.: Dense human body correspondences using convolutional networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2016), pp. 1544–1553. 2

[WHG13] Wang F., Huang Q., Guibas L. J.: Image co-segmentation via consistent functional maps. In Proceedings of the IEEE International Conference on Computer Vision (2013), pp. 849–856. 2

[ZPIE17] Zhu J.-Y., Park T., Isola P., Efros A. A.: Unpaired image-to-image translation using cycle-consistent adversarial networks. arXiv preprint arXiv:1703.10593 (2017). 2

Appendix A: Consistency of the Modular Framework

Theorem 1 Assume a network of shapes, with embedding and deembedding functions $e : \mathcal{F} \to \mathcal{Q}, d : \mathcal{Q} \to \mathcal{F}$ such that $e \circ d = Id_{\mathcal{Q}}$. Then, the descriptor-transporting function across any walk on the shape network that begins on shape S and ends on shape $\tilde{S}$ is $\tilde{\mathbf{M}} \circ \mathbf{L}$, regardless of the sequence of intermediate shapes visited.

Proof First, note that if $e \circ d = Id_{\mathcal{Q}}$, then we can write $\mathbf{L} \circ \mathbf{M} = (\mathbf{E}^{-1}e(\cdot)) \circ (d(\mathbf{E}(\cdot))) = \mathbf{E}^{-1}(e \circ d(\mathbf{E}(\cdot))) = Id_{\mathcal{Q}}$. The theorem can be proven by induction on the size of the walk. Consider the walk of size one, namely with zero intermediate shapes, where the descriptor transport is directly from S to $\tilde{S}$. Then, under the assumption of invertibility, the descriptor transport function is exactly given by Eq. (8); thus the theorem holds in this case. Now let us assume the theorem holds for all walks of size $k-1$. A walk involving k shapes can be decomposed into a walk of size $k-1$ followed by a walk of size one. Assuming the walk of size $k-1$ begins at shape S and ends at shape $\hat{S}$, and the walk of size one connects $\hat{S}$ with $\tilde{S}$, the individual transport functions for the two walks are given by $\hat{\mathbf{M}} \circ \mathbf{L}$ (using the induction hypothesis) and $\tilde{\mathbf{M}} \circ \hat{\mathbf{L}}$ respectively. Then the descriptor transport function for the entire walk will be $(\tilde{\mathbf{M}} \circ \hat{\mathbf{L}}) \circ (\hat{\mathbf{M}} \circ \mathbf{L}) = \tilde{\mathbf{M}} \circ \mathbf{L}$. □

Corollary 1 In a network of shapes as above, where d and e are perfectly invertible, the descriptor transport is cyclically consistent.

Proof The descriptor transport is cyclically consistent if the transport across a loop of shapes is the identity. A loop is a network walk where the start and end shapes are the same shape S. Using Theorem 1, the transport around the loop is then given by $\mathbf{M} \circ \mathbf{L}$. Invertibility implies $d \circ e = Id_{\mathcal{F}}$, hence $\mathbf{M} \circ \mathbf{L} = Id_{\mathcal{F}}$. □