Eurographics Symposium on Geometry Processing 2017
J. A. Bærentzen and K. Hildebrandt
(Guest Editors)

Volume 36 (2017), Number 5

Spectral Affine-Kernel Embeddings

Max Budninskiy[1] Beibei Liu[1] Yiying Tong[2] Mathieu Desbrun[1]

[1]Caltech [2]Michigan State University

Abstract

In this paper, we propose a controllable embedding method for high- and low-dimensional geometry processing through sparse matrix eigenanalysis. Our approach is equally suitable to perform non-linear dimensionality reduction on big data, or to offer non-linear shape editing of 3D meshes and pointsets. At the core of our approach is the construction of a multi-Laplacian quadratic form that is assembled from local operators whose kernels only contain locally-affine functions. Minimizing this quadratic form provides an embedding that best preserves all relative coordinates of points within their local neighborhoods. We demonstrate the improvements that our approach brings over existing nonlinear dimensionality reduction methods on a number of datasets, and formulate the first eigen-based as-rigid-as-possible shape deformation technique by applying our affine-kernel embedding approach to 3D data augmented with user-imposed constraints on select vertices.

1 Introduction

Computing embeddings of discrete manifolds is one of the most general geometry processing tasks. Surface parameterization, for instance, seeks to embed a three-dimensional triangulated surface into the plane while minimizing some form of distortion; mesh deformation is another example where a different embedding of a surface or volume is sought after through user-specified constraints while also minimizing distortion. While geometry processing has been mostly focusing on 3D datasets, our big data era requires the processing of high-dimensional data as well. Since machine learning algorithms struggle with high dimensions (an issue known as the curse of dimensionality in this context), one typically needs to map these data points from their high-dimensional space into a lower dimensional space without significant distortion. Mapping data (living in $\mathbb{R}^D$ with $D \gg 1$ but sampling a manifold of low intrinsic dimensionality $d \ll D$) into a low-dimensional embedding space can be thought of as a preliminary feature extraction step in machine learning, after which pattern recognition algorithms are applied; yet it simply corresponds to an as-isometric-as-possible *parameterization* of the original manifold. This link between geometry processing and dimensionality reduction has even been exploited for parameterization [ZKK02, CLZW07, SS09] and other graphics applications [ZCO15].

Many *linear* dimensionality reduction methods exist. Principal Component Analysis (PCA) or Singular Value Decomposition (SVD), for instance, use eigenanalysis to find the most significant low-dimensional coordinates in which to express high-dimensional data in the most informative way. However, these linear methods only perform well when the data is close to forming a linear subspace instead of an arbitrary non-flat manifold. Consequently, a large body of work has been dedicated to *nonlinear dimensionality reduction* instead to address this issue. Note that the applicability of such low-dimensional embedding methods is incredibly broad, as it is far from restricted to machine learning algorithms: reduced-dimensional representations of data, also called *manifold learning*, produce a low count of "intrinsic variables" with which markedly faster computations can be performed in the context of physical simulation, rendering, image processing, chemistry, biology, etc.

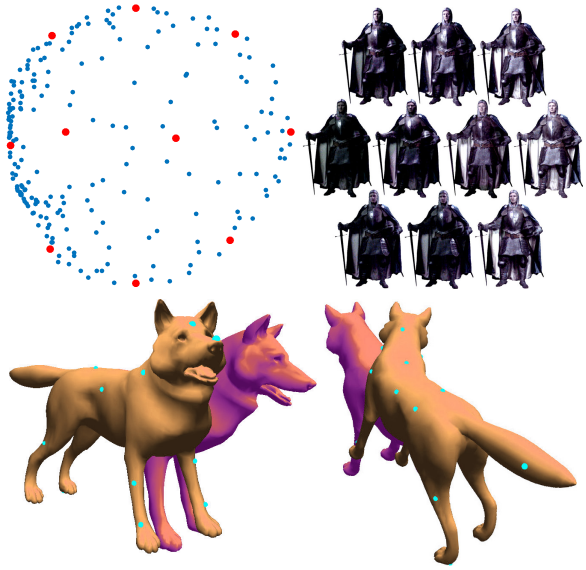


Figure 1: ***SAKE: high- and low-dimensional geometry processing.*** *We introduce a new technique to perform controllable embedding through eigenanalysis. Our approach allows us to find structure in high dimensional datasets: (Top) from 221 RGB images (with 608×456 pixels; 10 are shown on the right) of an actor in a knight costume captured from different lighting directions covering a large sphere of illumination [Lig16], a 2D embedding is computed solely based on local pixel differences (left). Our Spectral Affine-Kernel Embedding method finds a 2D parameterization of the images corresponding to the direction and the intensity of the lighting (the knight images correspond to red dots). (Bottom) The same spectral embedding approach can also be used for user-guided shape editing of 3D meshes and pointsets, where a few handles are moved to precisely control the deformation of an initial object through a simple sparse matrix eigenanalysis.*

In this paper, we introduce a spectral approach to manifold learning that leverages simple and efficient geometry processing techniques. We combine as-isometric-as-possible parameterizations of local patches to generate a global, non-linear map into a low dimension, that preserves as best as possible the points' positions relative to each other in parametrized patches. In stark contrast to previous work (see Fig. 2 for instance), our approach guarantees the recovery of a locally isometric parametrization of the input manifold when such a parametrization exists. We demonstrate on a variety of examples that our Spectral Affine-Kernel Embedding (SAKE) method outperforms existing techniques for both real and synthetic datasets. Finally, we also illustrate how SAKE is useful even in low dimensions by formulating an eigen-based non-linear shape deformation technique for meshes and pointsets that allows the user to deform an input shape as rigidly as possible based on the displacement of a few handles.

1.1 Previous work

We first abstract the problem to its most general form: suppose one is given n points $\{\mathbf{x}_i\}_{i\in[1..n]}$ in $\mathbb{R}^D$ that are supposed to approximately sample a d-dimensional manifold. Nonlinear dimensionality reduction seeks a map of this pointset to $\mathbb{R}^d$ that "unfolds" the manifold with the least amount of intrinsic distortion.

Embeddings via dense matrices. Isomap [TSL00] imposes low distortion on the reduction map by best preserving pairwise geodesic distances between input points. These geodesic distances are approximated through a shortest path (Dijkstra's) algorithm in the proximity graph where each point is linked through an edge to its k closest neighbors in the embedding space. Such intrinsic distances are typically more reliable than straight-line Euclidean distances in $\mathbb{R}^D$ as soon as the manifold is locally curved. From the set of all pairwise geodesic distances, a dense $n\times n$ Gram matrix is explicitly assembled; from its top d eigenvectors are found the optimal coordinates in the new d-dimensional Euclidean space that best preserve all these pairwise distances [BG05]. This reliance on *all* pairwise distances renders it inherently robust to noise, making Isomap one of the most robust dimensionality reduction methods. However, this construction is not without significant drawbacks. First, the pairwise geodesic distances require $\mathcal{O}(n^2\log n)$ operations to compute, and are poorly accurate as they are not even correct for points lying on flat manifolds since only graph paths are considered in the computation of shortest paths. Second, the Gram matrix is dense, which means that finding the final embedding has a worst-case complexity of $\mathcal{O}(n^3)$—although algorithmic improvements have been proposed to reduce this complexity through landmark approximations [DST02]. Finally, Isomap only provides nice maps for manifolds without large-sized holes: geodesic distances would become significantly biased in this case since shortest paths would have to go around hole borders, distorting the local notion of intrinsic distances (see Fig. 3). Any attempt at unfolding a non-simply connected domain (i.e., a non-contractible manifold patch) will fail to be nearly isometric, although algorithms aiming at mitigating this issue have been formulated [RBBK10]. As a consequence, global Isomap-type methods are quite robust, but not efficient or general enough to handle arbitrary inputs.

Embeddings via sparse matrices. Another category of approaches, including Locally Linear Embeddings (LLE [RS00]), Laplacian Eigenmaps [BN01], Local Tangent Space Alignment (LTSA [ZZ04]) and all their variants, infer the global structure of a non-linear manifold by a careful analysis of the interactions between overlapping local neighborhoods. While these approaches differ in how they approach the dimensionality reduction problem, they all encourage nearby points in the original embedding to be mapped to nearby points in the reduced space, ending up with a sparse and symmetric matrix from which the embedding is deduced through eigenanalysis. The sparsity of the resulting matrix makes such local approaches scale relatively well to large data sets compared to Isomap, since extremal eigenvectors of sparse matrices can be typically found in $\mathcal{O}(n^{1.5})$ [SM00]. However, sparsity is also a source of brittleness: the final embedding will preserve the *local* geometry of data as much as possible, but global, large-scale distortions are not penalized. This shortcoming gives rise to very warped embeddings in practice [DST02, VDMPVdH09], an issue that can be partially reduced by reinjecting global distances in the solve [HWX10] at the cost of a higher computational cost. Hessian-based locally linear embedding (Hessian-LLE [DG03], [YZ15]) may provide much improved results through the construction of least-squares approximation of Hessian operators in local tangent spaces, thus avoiding spurious (harmonic) warping that Laplacian-based methods suffer from. However, if noise is present, the approximation of Hessians becomes unreliable, and this technique ends up with even worse warping than LLE (see Fig. 8): automatically "stitching" local flattened neighborhoods into a non-degenerate global unfolding that is as isometric as possible to the original high-dimensional manifold is inherently difficult. Therefore, these local methods to manifold learning are often efficient and general enough to deal with arbitrary inputs, but rarely robust enough to offer a reliable global embedding.

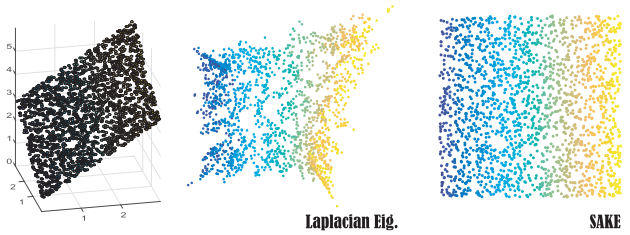


Figure 2: ***Affine Precision.*** *Laplacian Eigenmap (middle) is not linearly precise as it fails to properly capture a uniformly sampled flat patch in 3D (left). SAKE (right) does not suffer from this common limitation.*

Mesh deformation. Conceptually, mesh deformation techniques are embedding methods as well, although they do not map onto a reduced space: given an input simplicial 2- or 3-manifold in $\mathbb{R}^3$ and a few select displacements of handle vertices in space, they compute a deformed embedding that interpolates constrained vertices while preserving the local neighborhoods of the original mesh as closely as possible. In fact, differential coordinates based on the Laplace-Beltrami operator (see a review of existing methods in [Sor05, Sor06]) can be seen as a close relative of Laplacian Eigenmaps, where a particular embedding is found based on the initial Laplacian operator through a linear solve instead of an eigenvalue problem. Non-linear approaches to mesh deformation have been introduced as well to prevent visual artifacts of previous linear methods [ZHS*05, HSL*06, BPGK06, HSTP11].

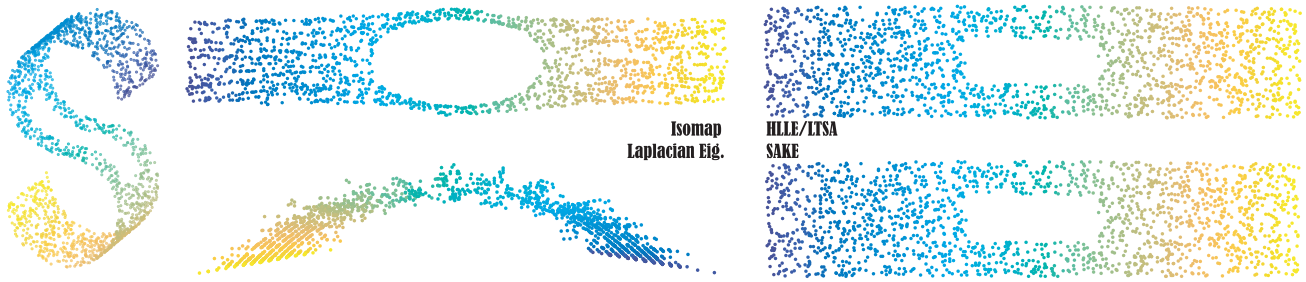


Figure 3: ***Non-simply connected manifold.*** *For points uniformly distributed over a 2-dimensional S-shaped manifold containing a hole (left), Isomap (top middle) overestimates the size of the hole, while Laplacian Eigenmaps result (bottom middle) is severely distorted. However, HLLE with $k=12$ (top right) and Sake with $k=K=12$ (bottom right) recover close to perfect embeddings in this noiseless case.*

One of the simplest non-linear methods is the as-rigid-as-possible (ARAP) modeling approach [SA07], which extends the Laplacian editing paradigm to better enforce surface rigidity—at the price of a slow-converging alternate minimization instead of a simple linear solve. We will show that our spectral approach, when used on low-dimensional data and with a few added terms to accommodate position constraints, provides a non-linear alternative to surface deformation which applies to meshes as well as pointsets and for which off-the-shelf libraries can be used for fast and robust results.

1.2 Overview and contributions

In this paper, we propose a controllable embedding method for high- and low-dimensional geometry processing. Our spectral approach can not only perform non-linear dimensionality reduction on big data, but it also offers a non-linear shape editing tool for 3D meshes and pointsets when applied to 3D data.

We propose a three-step approach to the embedding problem: a) we first parameterize each local neighborhood from $\mathbb{R}^D$ into $\mathbb{R}^d$ as isometrically as possible via Isomap, using an additional geometric correction of geodesics to improve robustness to irregular sampling; b) we then compute an exhaustive set of relative coordinates that captures the position of every point with respect to its local neighborhood; c) lastly, a global embedding is found as the new point positions in $\mathbb{R}^d$ that best preserve all the relative coordinates from all the neighborhoods. Each of these 3 steps is efficiently formulated as a spectral problem: (a) requires independent (thus, trivially parallelizable) partial eigendecompositions of dense, but small matrices; (b) is done via independent partial SVDs; and (c) is achieved via a partial eigendecomposition of a sparse, positive-definite symmetric matrix. Our approach can be understood in geometric terms as constructing the global embedding coordinates that are as linear as possible in the local most-isometric parameterization of each small neighborhood of the original input. Note that this is in marked contrast with most previous methods which fail to properly unfurl even flat or developable manifolds (see Figs. 2 and 3). Finally, we explore applications of manifold learning to 3D geometry processing by formulating a spectral, as-rigid-as-possible deformation technique that computes the most-isometric embedding of an original shape based on user-defined position constraints.

Besides our novel embedding approach, we also provide along the way a series of contributions covering various aspects of manifold learning and shape deformation:

- we introduce a geodesic curvature correction to the distances used in Isomap [TSL00] that adds much improved stability to this staple of nonlinear dimensionality reduction in the case of irregular and sparse sampling, without affecting its computational complexity or adding new parameters;
- we add robustness to Locally-Linear Embeddings [RS00] and Laplacian Eigenmaps [BN01] by using an exhaustive set of relative coordinates to guarantee the absence of spurious harmonic deformation in the final embedding. Our extension is much simpler and more robust to irregular sampling and noise than Hessian-LLE [DG03] which unnecessarily requires quadratic accuracy of their local Hessian operators;
- we introduce a simple approach for user-guided deformation of meshes and pointsets which, unlike previous non-linear editing approaches [BPGK06,SA07], can be efficiently and reliably computed through eigenanalysis of a sparse matrix.

2 Spectral Affine-Kernel Embeddings

We first introduce a procedure to map a pointset $\mathcal{S}$ assumed to sample (possibly with noise) a connected d-manifold from $\mathbb{R}^D$ into a d-dimensional space, where $D>d$. We assume the pointset to be arbitrarily indexed, and we denote its n points as $\{\mathbf{x}_i\}_{i=1..n}$, where each point $\mathbf{x}_i$ is given as a vector of D coordinates. The result of our most-isometric embedding approach will be the corresponding mapped positions $\{\mathbf{z}_i\}_{i=1..n}$ in $\mathbb{R}^d$.

2.1 General algorithm

Our dimensionality reduction proceeds in four distinct steps:

- We first form a proximity graph G by linking nearby input points; we then assemble a local geodesic neighborhood $\mathcal{N}(i)$ of each input point $\mathbf{x}_i$ based on the proximity graph.
- We compute for each $\mathbf{x}_i$ an as-isometric-as-possible embedding of its neighborhood in $\mathbb{R}^d$.
- For each of these resulting embeddings, we assemble a sparse matrix $\mathbf{L}_i$ representing a linear operator ("multi-Laplacian") whose kernel is restricted to constant and linear functions.
- We then assemble a quadratic form $\mathbf{Q}$ derived from the affine-kernel matrices $\mathbf{L}_i$, and find the final embedding $\{\mathbf{z}_i\}_{i=1...n}$ in $\mathbb{R}^d$ by computing the lowest $(d+1)$ eigenvectors of $\mathbf{Q}$.

We now review each step in order to provide both algorithmic details and mathematical justification for our embedding approach.

2.2 Proximity graph and geodesic neighborhoods

We begin by forming a *proximity graph* G by linking every point of $\mathcal{S}$ with its k nearest neighbors based of the Euclidean distance in $\mathbb{R}^D$, found efficiently through a kd-tree, cover tree, or locality sensitive hashing data structure. The value of k must be small to make sure the edges of the graph are short enough to offer reliable approximations of geodesics. Knowing that we will approximate a d-dimensional manifold, we typically choose $k = 4d$, i.e., a valence proportional to d like for a regular grid. We then assign to each point $\mathbf{x}_i$ the set of indices $\mathcal{N}(i)$ corresponding to the K nearest neighbors of the ith point in the proximity graph—hence, defining a *geodesic neighborhood*—for K larger than k. By default, we pick K in the interval $[k; 2^d k]$ to account for the dimensionality of the data and the amount of noise. These K-neighborhoods define n overlapping patches that we will each unfold, then glue together into a global d-dimensional embedding.

Note that our choice of a small value k for the proximity graph and a distinct, larger value K to define neighborhoods prevents the traditional issue of "shortcutting" the manifold: existing methods do not make the distinction between these two values, and pick a neighborhood based on *either* an ε-ball around the point *or* its k-nearest-neighbors in $\mathbb{R}^D$. However, robustness to noise requires a large number of neighbors, which creates graph edges that are far from being geodesics. Instead, our simple alternative allows the use of large neighborhoods (to be robust to noise) while still keeping a sparse edge graph to better approximate geodesic distances. Of course, any additional knowledge on the sampling (such as noise level) can be used to adjust the two parameters k and K. These parameters can even vary from point to point if needed to better deal with varying sampling density for instance, although our strategy can already handle large density variations, see Fig. 10.

2.3 Mapping neighborhoods into reduced space

Each geodesic neighborhood is then mapped as isometrically as possible into dimension d. Given that these neighborhoods are small and contractible, we use Isomap [TSL00] to achieve the mapping reliably: using PCA instead would be significantly less isometric if the input manifold is curved.

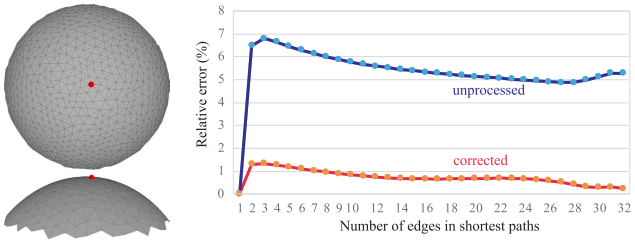


Figure 4: ***Geodesic distance correction.*** *From a triangulated patch of a spherical cap with good aspect ratio elements (left, top and side views), the total relative L^2 error of all pairwise distances computed based on Dijkstra's shortest paths is 29.9; our simple post-processing reduces the error to 6.5. If we bin all the distances based on the number of edges in the edge-based connecting paths, we see that aside from the case of a single edge (where the geodesic distance cannot be improved), our geodesic distance correction reduces errors by a factor five or more.*

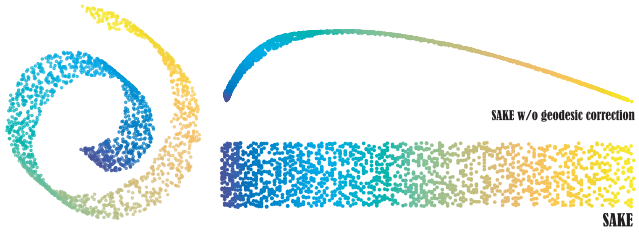
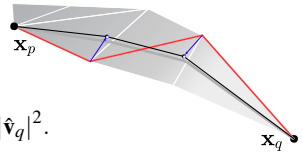


Figure 5: ***Geodesic curvature correction.*** *Our geometric correction improves geodesic approximations for coarse and irregularly sampled data; without it, extreme distortion can happen on imperfectly sampled datasets. We set $k = K = 12$ for this noiseless Swiss Roll example (left).*

Geodesic distances via Dijkstra's algorithm. Each edge of the proximity graph G is considered an intrinsic geodesic curve, thus its Euclidean length in $\mathbb{R}^D$ is an accurate estimate of the intrinsic distance between its end points. For every pair of points in the geodesic neighborhood of point i that are not directly connected by an edge, we compute its approximate geodesic distance by solving the all-pairs-shortest-path problem, which Dijkstra's algorithm [Dij59] achieves optimally in $\mathcal{O}(K^2 \log K)$.

Geodesic curvature correction. Shortest paths computed on the neighborhood graph suffer from non-zero geodesic curvatures: Dijkstra's algorithm will return shortest paths that are polylines made of graph edges, hence very unlikely to be actual geodesics. To improve geodesic distance estimates, we correct for these graph-induced errors by post-processing each shortest path to remove spurious geodesic curvatures; more precisely, we construct an improved shortest path by shifting its vertices parallel to the neighborhood's local tangent space so as to make its geodesic curvature zero. To this end, we first compute an estimate of the local tangent d-dimensional space of the neighborhood through PCA, which returns a d-dimensional basis of orthonormal vectors $\mathbf{t}_1, \ldots \mathbf{t}_d$ in $\mathbf{R}^D$. Then for each "geodesic" polyline between two points $\mathbf{x}_p$ and $\mathbf{x}_q$ (representing their shortest connecting path in graph G), we displace the intermediate vertices of the polyline parallel to the d-dimensional tangent space and orthogonal to the line $(\mathbf{x}_p \mathbf{x}_q)$ to project out any "zigzagging", thus straightening the geodesic. This is easily achieved numerically: if $\mathbf{x}_r$ is a vertex of the polyline, first project the $\mathbb{R}^D$ vectors $\mathbf{v}_r = \mathbf{x}_r - \mathbf{x_p}$ onto the tangent d-dimensional space to obtain $\hat{\mathbf{v}}_r = \sum_j (\mathbf{t}_j \cdot \mathbf{v}_r)\, \mathbf{t}_j$; from the resulting vectors $\hat{\mathbf{v}}_r$, further extract their projection along the direction $\hat{\mathbf{v}}_q$ through

$$\tilde{\mathbf{v}}_r = \hat{\mathbf{v}}_r - [\hat{\mathbf{v}}_r \cdot \hat{\mathbf{v}}_q]\, \hat{\mathbf{v}}_q / |\hat{\mathbf{v}}_q|^2.$$



These final (tangent) vectors $\tilde{\mathbf{v}}_r$ of the points along the path between $\mathbf{x}_p$ and $\mathbf{x}_q$ are then subtracted from their corresponding original vertices $\mathbf{x}_r$ to determine their improved locations. The length of the resulting polyline (see inset), which has now zero geodesic curvature, is a more accurate geodesic distance l_{pq} between $\mathbf{x}_p$ and $\mathbf{x}_q$. Note that this correction guarantees *exact* distance evaluations when the neighborhood is flat, and provides a robust and consistent estimate of distances in the general case of a curved manifold—even applied

on a well-shaped triangle surface (where the graph is now formed by the edges of the mesh), our simple projection reduces the relative geodesic length errors fivefold as shown in Fig. 4. The only assumption our approach makes is that the PCA-based estimate of the local tangent space on a neighborhood of size K is of reasonable quality. As a result, if the neighborhoods are chosen to be too small relative to the noise present in the data or too large compared to data features, the quality of the correction may partially drop.

From distances to d-dimensional embedding. From all the pairwise geodesic distances l_{pq} (for p and q in $i \cup \mathcal{N}(i)$) of the local neighborhood of point $\mathbf{x}_i$, we apply Multi-Dimensional Scaling [BG05] to obtain a lower dimensional flat embedding of a neighborhood that best preserves the geodesic distances. First, the matrix $\mathbf{D}_i = (l_{pq})$ of pairwise geodesic distances is converted into a Gram matrix $\mathbf{G}_i$ via "double-centering" [TSL00]; then the d top eigenvectors of $\mathbf{G}_i$ are computed. Once scaled by the square root of their respective eigenvalue, they represent the optimal as-isometric-as-possible coordinates $\mathbf{Y}_i$ of a d-dimensional embedding of all the points in the neighborhood of $\mathbf{x}_i$. This MDS procedure is variational, as it amounts to solving a quadratic minimization under rank constraint, since the solution $\mathbf{Y}_i$ minimizes the Frobenius norm $||\mathbf{Y}\mathbf{Y}^t - \mathbf{G}_i||_F^2$ subject to $\text{rank}(\mathbf{Y}) = d$ [VMS16]. Note that if the points are lying on a flat or developable manifold, no distortion occurs in this process—a property that regular Isomap fails to enforce due to the zigzagging of Dijkstra's shortest paths.

Discussion. Pairwise distances on triangle or tetrahedral meshes can of course be computed faster and better through fast marching [ZKK02] or heat-based distances [CWW13]. Diffusion distances have also been proposed to offer approximate geodesic distances through truncation of the graph Laplacian spectrum and tuning of a diffusion time t [CLL*05]. However, our approach applies to pointsets in *arbitrary* dimensions, and its simplicity makes it particularly convenient: improvements over original Dijkstra distances are already significant with only a small computational overhead.

2.4 Relative coordinates

Now that we have unfurled all geodesic neighborhoods in d-dimensions, we assemble for each of these mapped neighborhoods a set of linear equations that represent all possible relative coordinates of a point with respect to its neighbors. Equivalently, we will show that these equations enforce harmonicity in all constant metrics. Since the neighborhood of point $\mathbf{x}_i$ has been nearly-isometrically mapped to points $\mathbf{y}_j \in \mathbb{R}^d$ for each j in the index set $\{i\} \cup \mathcal{N}(i)$, we will denote by $\mathbf{Y}_i$ the $(K+1) \times d$ matrix containing all the coordinates of the neighborhood points, i.e.,

$$\mathbf{Y}_i = \begin{pmatrix} \mathbf{y}_i & \mathbf{y}_{j_1} & \mathbf{y}_{j_2} & \dots & \mathbf{y}_{j_K} \end{pmatrix}^t$$

Affine-precise linear combinations. Define the $d \times K$ matrix $\mathbf{E}_i = \begin{pmatrix} \mathbf{y}_{j_1} - \mathbf{y}_i; & \mathbf{y}_{j_2} - \mathbf{y}_i; & \dots; & \mathbf{y}_{j_K} - \mathbf{y}_i \end{pmatrix}$ containing in the m-th column the d coordinates of the edge vector $\mathbf{y}_{j_m} - \mathbf{y}_i$. This matrix can be thought of as a redundant basis of the d-dimensional tangent space, and its kernel is formed by the space of all linear combinations of edges summing to zero. Because we picked $K > d$ (i.e., more neighbors than the dimensionality), the rank-nullity theorem directly implies that the size of this kernel is $K - d$, as the edges span the entire d dimensions in practice. The basis of this kernel can be constructed efficiently through a Singular Value Decomposition (SVD) of $\mathbf{E}_i = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^t$, where $\mathbf{U}$ is a $d \times d$ orthogonal matrix, $\mathbf{\Lambda}$ is a $d \times K$ rectangular diagonal matrix with the d singular values in decreasing magnitude on the diagonal, and $\mathbf{V}$ is a $K \times K$ orthogonal matrix. We simply select the last $K - d$ right singular vectors $\{\mathbf{w}^p \in \mathbb{R}^K\}_{p=1..K-d}$ of unit length as the basis vectors of the kernel. Now $\mathbf{y}_i$ can be written as a linear combination of its neighbors $\mathbf{y}_j$ for each of these vectors $\mathbf{w}^p$ since, by construction,

$$\forall p \in [1..K-d], \quad \left(\sum_{j \in \mathcal{N}(i)} [\mathbf{w}^p]_j \right) \mathbf{y}_i = \sum_{j \in \mathcal{N}(i)} [\mathbf{w}^p]_j \, \mathbf{y}_j, \qquad (1)$$

where $[\mathbf{w}^p]_j$ denotes the j-th coordinate of $\mathbf{w}^p$. Consequently, the matrix $\mathbf{L}_i$ of size $(K-d) \times (K+1)$ defined as

$$\mathbf{L}_i = \begin{pmatrix} \sum_{j \in \mathcal{N}(i)} [\mathbf{w}^1]_j & \sum_{j \in \mathcal{N}(i)} [\mathbf{w}^2]_j & \dots & \sum_{j \in \mathcal{N}(i)} [\mathbf{w}^{K-d}]_j \\ -\mathbf{w}^1 & -\mathbf{w}^2 & \dots & -\mathbf{w}^{K-d} \end{pmatrix}^t$$

satisfies: $\mathbf{L}_i \, \mathbf{Y}_i = \mathbf{0}$. Note that the set of weights $\mathbf{w}^p$ can be seen as a linearly-independent basis of all relative coordinates, describing the position of $\mathbf{y}_i$ in terms of its neighbors $\mathbf{y}_j$ for $j \in \mathcal{N}(i)$. Using the *whole space of relative coordinates* instead of picking just one results in a more complete encoding of the data, allowing us to capture the geometry of a patch in a reliable way.

Affine kernel. Note that matrix $\mathbf{L}_i$ has an important property: its kernel consists of all (discrete) affine functions. Indeed, consider a scalar function $f : \mathbb{R}^d \to \mathbb{R}$ and call $\mathbf{f}$ the $(K+1)$-dimensional column vector representing the sampling of f in the neighborhood, i.e. $f_j = f(\mathbf{y}_j)$ for $j \in \{i\} \cup \mathcal{N}(i)$. Due to our construction of $\mathbf{L}_i$, any constant function will satisfy $\mathbf{L}_i \mathbf{f} = \mathbf{0}$. The same property holds for linear functions as well because the weights form a basis of the kernel of $\mathbf{E}_i$ (Eq. (1)). Thus, Ker $\mathbf{L}_i$ contains *all* sampled affine functions. Furthermore, the rows of $\mathbf{L}_i$ are $(K-d)$ independent vectors (by construction via the SVD), so by the rank-nullity theorem the kernel of $\mathbf{L}_i$ is of dimension $(K+1) - (K-d) = d+1$. However, the space of affine functions $f(\mathbf{y}) = \mathbf{a}^t \mathbf{y} + b$ ($a \in \mathbb{R}^d, b \in \mathbb{R}$) also has dimension $d+1$. This implies that sampled affine functions are the *only* elements of Ker $\mathbf{L}_i$.

Multi-Laplacian interpretation. Note that our construction can be understood as an extension of Laplacian eigenmaps [BN01]: while their approach uses a single linear equation per neighborhood corresponding to a local condition of harmonicity, we have instead a whole set of linear equations. Each row of our local operator $\mathbf{L}_i$ can be interpreted as a discrete harmonicity condition in a different metric—hence our use of the term "multi-Laplacian." Indeed, the action of a row of $\mathbf{L}_i$ on a discrete function $\mathbf{f}$ is of the form $d \star d \mathbf{f}$ in the DEC notation [DKT08], and equating it to zero corresponds to a harmonic condition in a possibly non-Euclidean metric. The matrix $\mathbf{L}_i$, in fact, encodes harmonicity conditions for *all constant metrics* σ over the unfolded neighborhood: the anisotropic Laplacian operator $\Delta_\sigma f = \nabla \cdot (\sigma \nabla f)$ returns zero for affine functions as discussed in [dGAOD13, dGLB*14] since constant metrics σ satisfy $\nabla \cdot (\sigma \nabla f) = \text{Tr}(\sigma \, \text{Hess} \, f)$. (Linear accuracy is actually valid for all divergence-free metrics, but the restricted case of constant metrics is sufficient for our purpose.) Therefore, our construction can be thought of as identifying the intersection of the spaces of σ-harmonic functions for all constant σ: these "multi-harmonicity conditions" restrict the kernel to be sampled affine functions only—while the normal Laplacian is blind to any non-zero off-diagonal term of the Hessian. Note that the *multi-Laplacian* $\mathbf{L}_i$ we end up

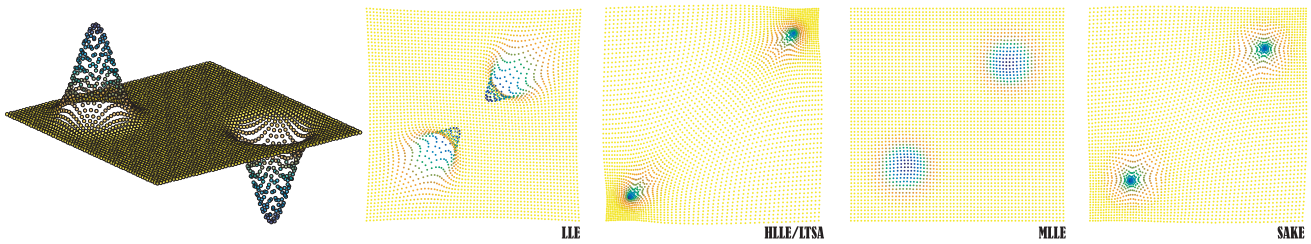


Figure 6: ***Bumps example.*** *For a regularly-sampled "two-bump" surface (left), LLE creates foldovers; HLLE/LTSA both significantly distort areas around the bumps in a non-isometric way; MLLE recovers an almost regular grid, completely ignoring the local curvature of the original data; $k=24$ neighbors were used for all the methods. Instead, SAKE ($k=8, K=24$) finds a most-isometric embedding.*

with is less difficult to assemble than a Hessian operator on which Hessian-LLE is based, since it requires no specific behavior when applied to sampled quadratic functions; yet the two operators have the exact same kernel, bringing robustness to the process as the authors of [DG03] argued.

2.5 Assembling the global quadratic form Q

In order to find the final embedding $\{\mathbf{z}_i\}_{i=1..n}$ in $\mathbf{R}^d$ of the pointset $\mathcal{S}$, we assemble a sparse, symmetric, and positive-definite $n\times n$ matrix $\mathbf{Q}$. Let $\mathbf{Z}=(\mathbf{z}_1\ \mathbf{z}_2\ \dots\ \mathbf{z}_n)^t$, and $\mathbf{S}_i$ be the $(K+1)\times n$ selection matrix of neighborhood i, i.e., the sparse matrix such that each component (p,q) is 1 if $q\in\mathcal{N}(p)$, and 0 otherwise—so that $\mathbf{S}_i\mathbf{Z}$ is the mapped neighborhood of i in the final d-dimensional embedding. We then define our global quadratic form $\mathbf{Q}$ as:

$$\mathbf{Q}=\sum_{i=1}^{n}\mathbf{S}_i^t\,\mathbf{L}_i^t\,\mathbf{L}_i\,\mathbf{S}_i.$$

Given our interpretation of the matrices $\mathbf{L}_i$ as storing harmonicity conditions for all constant metrics, one can understand the global quadratic form as the sum of local Dirichlet energies computed in all possible locally constant metrics. The resulting quadratic form thus penalizes any non σ-harmonic functions—that is, any non-affine functions— in any given neighborhood: it is therefore much more "discerning" than a simple Laplacian.

2.6 Extracting the final embedding

From the sparse SPD matrix $\mathbf{Q}$, we extract the final, global embedding of the input pointset as a set of positions $\{\mathbf{z}_i\}_{i=1..n}$ in $\mathbb{R}^d$ by computing the first $(d+1)$ smallest eigenvectors $\mathbf{q}_m$ of $\mathbf{Q}$ (satisfying $\mathbf{Q}\mathbf{q}_m=\lambda_m\mathbf{q}_m$). Note that the first eigenvector is constant with an associated zero eigenvalue by construction. The d coordinates of points $\mathbf{z}_i$ simply correspond to the second, third, ..., and $(d+1)$st smallest d eigenvectors:

$$\mathbf{z}_i=\begin{pmatrix}[\mathbf{q}_2]_i & [\mathbf{q}_3]_i & \dots & [\mathbf{q}_{d+1}]_i\end{pmatrix}^t.$$

Indeed, these are the positions that make every local neighborhood satisfy the multi-harmonic conditions as closely as possible, in a least squares sense, since the matrix $\mathbf{Z}=(\mathbf{z}_1\ \mathbf{z}_2\ \dots\ \mathbf{z}_n)^t$ is the solution of the following minimization:

$$\underset{\mathbf{Z}}{\operatorname{argmin}}\ \tfrac{1}{2}\operatorname{Tr}\left[\mathbf{Z}^t\mathbf{Q}\mathbf{Z}\right]\quad\text{s.t. }\mathbf{Z}^t\mathbf{Z}=\mathbf{Id},$$

where the constraint is used to prevent degenerate solutions (similar to the use of Fiedler vectors in [MTAD08]). The final embedding thus preserves the local relative coordinates as best as possible since, by construction of $\mathbf{w}^p$, we have $\mathbf{L}_i\mathbf{Y}_i=0$ over all the isometrically parametrized patches of the original manifold. Notice that if the input pointset was finely sampling a developable manifold, we exactly recover, up to an affine transformation, the intrinsic discretization of the manifold isometrically unfolded in $\mathbb{R}^d$ (and the first $(d+1)$ eigenvalues of $\mathbf{Q}$ are zero in that case). If the original manifold is not developable, this extraction picks the unfolding which is as affine as possible in each locally isometric coordinates—and the eigenvalues of $\mathbf{Q}$ inform us on how non-developable the initial manifold was, and how much *metric distortion* we can expect from our optimal low-dimensional embedding.

2.7 Computational complexity

Once the initial proximity graph is computed (a task that is common to all manifold learning methods), our approach involves four distinct stages. First, we compute an all-pair-shortest-path algorithm per neighborhood so that each approximate geodesic distances between point pairs can be evaluated; this step has a complexity of $\mathcal{O}(K^2\log K)$ for each of the n neighborhoods. Second, we compute an as-isometric-as-possible parameterization of each neighborhood, which requires the $\mathcal{O}(K^3)$ eigenanalysis of a dense $(K+1)\times(K+1)$ matrix of distances. Third, we compute a SVD in $\mathcal{O}(K^3)$ to find the affine-kernel matrix $\mathbf{L}_i$ in each neighborhood.

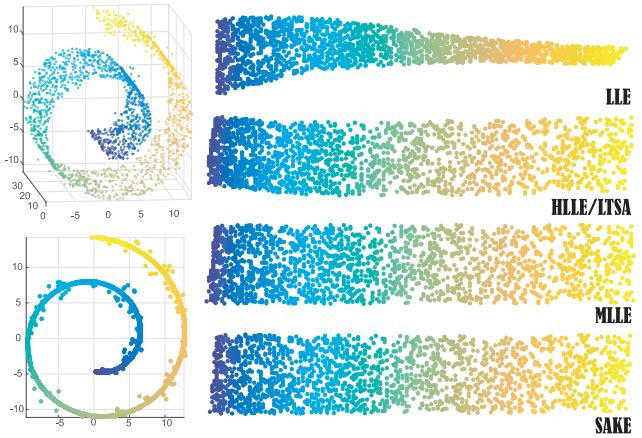


Figure 7: ***Swiss roll with sparse noise.*** *Flexibility in the size of geodesic neighborhoods renders SAKE stable to noise. We use points sampled from an intrinsically 2D Swiss roll embedded in 3D with sparse, uniformly-distributed noise in the normal direction (left, 2 views). Results of non-linear 2D embeddings (right column, from top to bottom): LLE ($k=15$), HLLE/LTSA ($k=15$), MLLE ($k=15$) and SAKE ($k=15, K=30$). Results of LTSA and HLLE are visually indistinguishable on this example.*

Note that these three first steps can be done in a massively parallel way as they proceed independently on each neighborhood. Finally, finding the $(d+1)$ bottom eigenvectors of the sparse matrix $\mathbf{Q}$ requires an expected $\mathcal{O}(n^{1.5})$ number of operations (as already experimentally found in [SM00] and confirmed in our tests). Assuming that K is small compared to the number of input points, the SAKE embedding algorithm scales much better than Isomap, and does not suffer from any restriction on the geometric nature of the manifold.

3 Analysis

We provide an extensive analysis to clearly identify the properties of SAKE, before presenting numerical tests to confirm our claims. We used the SciKit Learn [Sci16] manifold learning implementation of Isomap, LLE, HLLE, LTSA and Laplacian Eigenmaps.

3.1 Comparisons to prior art in manifold learning

Before delving into numerical comparisons, we first detail how our approach markedly differs from previous work. In particular, we explicitly describe the key differences between SAKE and the two most closely related approaches, HLLE and MLLE. In our discussion of the computational complexity of the various existing methods, we will denote by k the average number of neighbors for each input point $\mathbf{x}_i$ to be consistent with our notation.

SAKE vs. Isomap. Besides its issues of computational complexity and stringent limitation to simply connected patches, Isomap is not robust to low sampling: shortest-path geodesic approximations can become quite poor. Despite the existence of proofs of convergence to the proper geodesic distances [DST02], the local *differences* between local geodesic distances are often very inaccurate. Our improvement through geodesic curvature correction allows for much more robust results, see Figs. 4 & 5. While we use this distance correction only for local neighborhoods in the context of SAKE, the same simple geometric insight can also enhance the approximation of geodesic distances used in Isomap algorithm by performing this correction over long geodesics through local corrections. Proper testing of this local/global correction of Dijkstra's shortest paths in Isomap is left as future work.

SAKE vs. Laplacian Eigenmaps. The approach of Belkin and Niyogi [BN01] rely on an eigenanalysis of the Laplacian operator. However, the null space of this operator may contain much more than linear functions: it includes all harmonic functions such as $f(x,y)=xy$ for instance. The final embedding is thus often polluted by harmonic deformations, as observed in practice in Fig. 3. SAKE, instead, reduces the kernel of its quadratic form by enforcing more than a single linear equation per vertex, in order to guarantee a better isometry of the final embedding. Moreover, the approximation of the Laplacian in high dimension by a matrix $\mathbf{L}$ with entries L_{ij} proportional to $\exp(-\|\mathbf{x}_i-\mathbf{x}_j\|^2)$ as proposed in Laplacian Eigenmaps does not satisfy linear precision. Consequently, a flat manifold would not even be properly mapped without distortion (Fig. 2), while our approach returns the exact result in this case.

SAKE vs. Locally Linear Embedding. While LLE [RS00] does not rely on the Laplacian operator, its foundations are quite similar to Laplacian Eigenmaps. The only difference in practice is that the linear equation assembled per point $\mathbf{x}_i$ is not derived from a local Laplace estimate, but determined by solving a constrained least squares problem to best capture the local linear structure: it computes the linear combination of neighborhood points $\{\mathbf{x}_j\}_{j\in\mathcal{N}(i)}$ that best reconstructs $\mathbf{x}_i$. The first advantage of this modification is that in the case of a flat manifold, this linear equation is exactly satisfied by the input, so the result will be perfect. However, as soon as the input manifold is curved, the lack of isometric unfolding of each neighborhood and the reliance on a single linear equation per point to find the final embedding renders the approach extremely brittle: different samplings of a same manifold may result in dramatically different embeddings, see Figs. 7 and 8. A follow-up work [GR08] proposes to first project the neighborhood in d-dimensions through PCA. While this tends to reduce the sensitivity to noise since regularization is no longer needed, this approach remains quite brittle due to the reliance on a single linear equation per point.

SAKE vs. LTSA. Local Tangent Space Alignment [ZZ04] proceeds by first constructing an approximation for the tangent space at each data point, before aligning these tangent spaces to form global coordinates. In essence, this approach is similar to ours; but they instead directly enforce that transitions between neighborhood charts be as affine as possible, and the initial construction of the tangent space relies on PCA, which does not handle curved manifolds well. As a consequence, LTSA is quite successful at unfolding nearly flat manifolds (Fig. 6) because it also relies on more than a single linear equation per neighborhood (in fact, in many cases, LTSA and HLLE have identical results); but LTSA systematically fail on more challenging examples as demonstrated in Fig. 8.

SAKE vs. Multiple-weights LLE. A variant of the original LLE method, named Multiple-weights LLE [ZW07], is worth discussing further: the authors were the first to notice that the brittleness of LLE-type methods is mostly due to their reliance on the enforcement of a single linear equation per neighborhood. Consequently, they added to the regularized set of weights that LLE uses all the sets of weights corresponding to singular values close enough to zero (requiring a user-specified threshold), which offered much improved stability. However, just like LLE, the authors do not use a local, as-isometric-as-possible embedding, so the kernel of their operator is "polluted" by the way the manifold is embedded in higher dimension. Consequently, the result of their dimensionality reduction is far from being isometric, even if the input is nearly developable. Additionally, consider the example in Fig. 6: while the result of MLLE may at first glance look best, close inspection reveals

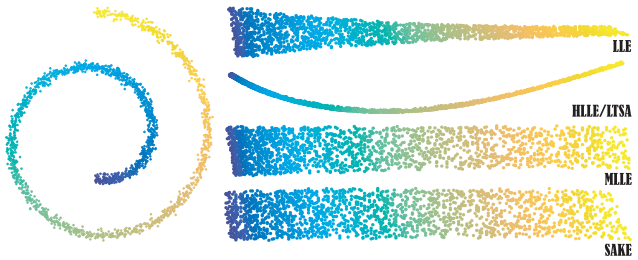


Figure 8: ***Swiss roll with Gaussian noise.*** *SAKE can handle strong noise in the input. Here, points are sampled from a 2D Swiss roll embedded in 3D with added Gaussian noise along the normal (left, profile view). Results of 2D embeddings (right, top to bottom): LLE ($k=15$), HLLE/LTSA ($k=15$), MLLE ($k=15$) and SAKE ($k=15, K=75$). Again, LTSA and HLLE return visually identical results, significantly deformed due to the instability of PCA-based tangent plane approximation in presence of strong noise.*

that the mapped points are simply ignoring the curvature of the initial manifold. This non-isometric behavior thus obscures the nature of the input data, and is hardly useful in concrete applications.

SAKE vs. Hessian-LLE. Hessian-LLE [DG03], shares also a close relationship to our approach: the authors recognized the value of reproducing affine functions on each chart of the manifold to find a reliable embedding. Consequently, they propose to construct local Hessian operators since their theoretical null space is restricted to affine functions. They proceed by first unfolding each neighborhood via PCA, which fails to be robust as soon as the input data is a bit noisy, see Fig. 8. From the neighborhood projected onto a low-dimensional tangent embedding, they construct a discrete basis of constant, linear, and quadratic functions (this last subspace is constructed through pointwise product of linear functions). They finally orthonormalize these bases through Gram-Schmidt. Thus, their method requires $\mathcal{O}(ds^2)$ more operations [GVL96], where $s=d(d+1)/2$ is the dimensionality of the space of quadratic functions. Additionally, we note that HLLE uses tangent planes at the center of their local PCA as noticed in [XDW16]; however this means that, for very irregular sampling, two vertices may lead to nearly identical Hessian estimates, thus not providing added information. In sharp contrast, our approach does *not* try to construct an approximate Hessian, but an operator whose kernel contains only affine functions—which can be done faster and is significantly less sensitive to noise in the data, see Figs. 7 and 8 for example.

3.2 Synthetic 3D datasets

For the past decades, manifold learning approaches have used a number of synthetic examples to demonstrate their results. We provide our results on a number of classical examples for comparison purposes, with and without noise, for regular and irregular sampling, and for low and high-dimensional datasets.

3D-to-2D datasets. Fig. 5 shows that SAKE can handle the usual "Swiss roll" example that all other methods use in their tests, even with highly irregular sampling. Note that our geodesic curvature correction makes a big difference on such an example: if the regular, non-corrected Isomap is used to unfold every neighborhood, trying to make the final coordinates affine in these distorted local coordinates induces a large global distortion, confirming that our correction is crucial in our approach. Fig. 3 shows the noise-free 'S-shape', with a uniform sampling and a hole in the middle. As expected, Isomap cannot unfold this non-simply connected domain without distortion. The Laplacian eigenmap is surprisingly deformed, most likely due to the presence of harmonic functions. In sharp contrast, HLLE and LTSA return very similar (and correct) results, and SAKE matches these results. Fig. 6 shows as-isometric-as-possible flattenings of a noise-free two-bump height function sampled on a regular grid. On this example, LLE creates foldovers on the bumps. HLLE and LTSA are once again visually indistinguishable, but exhibit shrinkage on the two corners near the bumps. MLLE keeps the symmetry of the domain perfectly; but the curvature of the bumps is totally ignored, amounting to a orthogonal projection onto the support plane—hence creating significant metric distortion between the original manifold and its 2D parameterization. SAKE captures the domain and its symmetries, with the expected metric-preserving parameterization of the bumps. Fig. 2 exhibits how even an irregularly sampled simple plane fails to be correctly captured by a Laplacian eigenmap, while SAKE guarantees perfect projection. Note that Isomap would also fail to keep the original sampling since shortest distances are computed on a graph, which creates zigzagging (thus, inaccurate) geodesics.

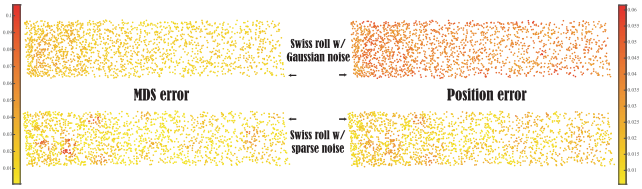


Figure 9: ***Embedding errors.*** *For the SAKE results shown in Fig. 8 (top) and Fig. 7 (bottom), we show the local reconstruction errors in pairwise distances (MDS errors, left) and in relative l^2 position errors (right).*

Noisy datasets. Testing robustness to noise is also informative. We tried the well-known 'Swiss roll' with noise "peppered" around: only 10% of the points are being displaced with a uniform noise distribution along the surface normal of the roll to simulate inaccurate samples, see Fig. 7. LLE already suffers from this noise (other values of k result in worse deformation), and so are HLLE and LTSA to a lesser extent. MLLE and SAKE return very similar and visually plausible results. If we now try the Swiss roll with Gaussian noise (standard deviation of 0.28) as shown in Fig. 8, HLLE and LTSA fail entirely, while LLE is still quite deformed. MLLE appears relatively good, although the right end of the strip is squeezed in a way similar to LLE, but with a less pronounced effect. SAKE remains best among all the methods. We analyze two types of error, arising from the MDS and the global embedding construction steps of SAKE respectively: one corresponds to local distance preservation computed pointwise as the relative 2-norm of the rank d approximation of the Gram matrix $\mathbf{G}_i$, while the other measures the quality of the local relative positions as the 2-norm of the changes in relative positions $||\mathbf{L}_i\mathbf{S}_i\mathbf{Z}||_2$. Looking at Fig. 9, we see that Gaussian noise results in small, uniform reconstruction and distance errors, while sparse outliers do not disrupt the non-noisy parts of the domain, proving robustness to various types of noise.

Handling large density variations. We also tried significant variations in sampling density (in addition to noise) to test the robustness of various manifold learning approaches in Fig. 10. Here again, SAKE matches or surpasses other results, which prove to be extremely dependent on the number of neighbors used. Note that we did not try to adapt the value of k and K based on density (which would improve accuracy) to offer a fair comparison.

Handling large noise and outliers. While previous methods rarely discuss the issue of outliers and large amounts of noise, SAKE offers a number of opportunities to derive more robust strategies to noise and outliers. As we discussed in Sec. 2.2, one can tweak our use of k closest neighbors to remove obvious outliers and noise: for instance, the authors of [PZZK04] propose a weighted local linear smoothing and one-ring minimum spanning trees (to detect significantly large edge size in the neighborhood, a tell-tale for outliers) for noise reduction and outlier handling. The choice of K geodesic neighbors may also be made adaptive by using iterative robust PCA [XDW16] instead, to select the number of neighbors based on a local noise level estimate. Finally, our curvature corrected approach to intrinsic distances can also be extended to "smooth out"

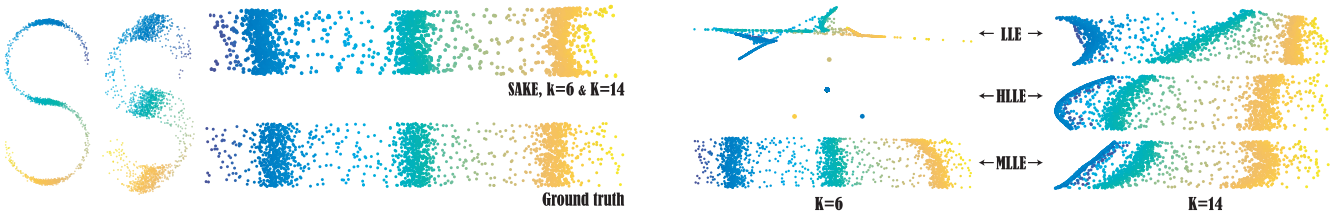


Figure 10: ***S-shaped manifold with variable density and Gaussian noise.*** *SAKE is also robust to variable density in the input. A non-uniformly sampled 'S' shape is embedded in 3D with Gaussian noise added (left, profile views). We compare our results with other existing approaches, for various parameters.*

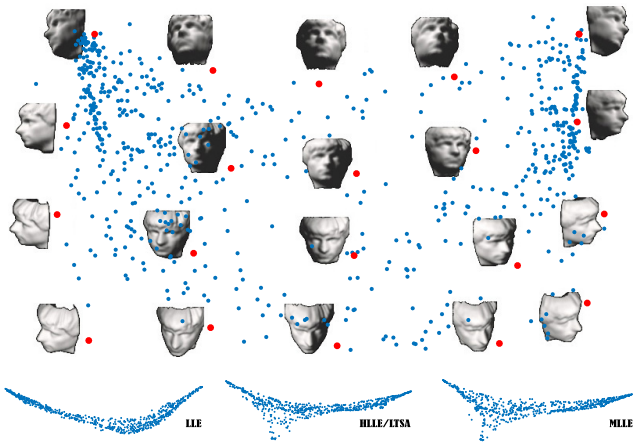


Figure 11: ***Faces dataset.*** *From* 698 *images representing the same 3D face from different viewpoints (each image is a point in* $\mathbb{R}^{4096}$*, see [TSL00]), we compute the 2D SAKE* ($k=6, K=36$) *embedding purely based on local pixel-per-pixel distances of the images. The result parameterizes the camera angle quite accurately, with no a-priori knowledge. Examples of the original face images are given for the 20 red dots (image backgrounds removed for clarity). Other methods give markedly more distorted results (bottom).*

(through a curve straightening flow) the geodesic curves obtained by our process if noise is significant. These add-ons are trivial to incorporate, but a full assessment of how they fare on real defect-ladden data is out of the scope of this paper. We favored instead a parameter-free SAKE, which is already more noise-resilient than the existing methods we tested.

3.3 High-dimensional datasets

We also ran our SAKE algorithm on the classical 'Faces' datasets, where grayscale pictures with 64×64 pixels capture the same face from different viewpoints [TSL00]. From this set of 698 images, each represented as a point in $\mathbb{R}^{4096}$ (one dimension per pixel), we can ask SAKE to reduce the dataset to a 2D projection purely based on pixel-per-pixel *differences* between pairs of images. Fig. 11 shows the result, where the points are clearly embedded in a position dependent on the left-right, up-down angle of view. For comparison purposes, we also show the results of a few other methods for this dataset on the same figure.

We also tried to our approach on a dataset of reflectance fields captured using the Light Stage apparatus [Lig16]. A static character (in a knight costume) was captured under 221 individual lighting directions covering a large sphere of illumination. Once again, we use all 608×456 RGB images (given in random order) stored as points in $\mathbb{R}^{831744}$ and try to learn a flat 2D manifold that best fits this high dimensional dataset. The result, shown in Fig. 1(top), recovers positions related to light angles without any knowledge of the setup, while other approaches lead to (sometimes severely) distorted embeddings, see Fig. 12. Note that the black background of each image was removed for clarity of the figure.

4 Mesh and Pointset SAKE Deformation

While we detailed how high dimensional datasets can be most-isometrically mapped into low dimensions, our embedding approach has a simple, yet highly-relevant application to geometry processing: it allows for efficient non-linear shape deformation.

4.1 Non-linear deformation as an embedding problem

Mesh deformation has a long history in geometry processing as it is one of the most important tools in practical graphics applications [Sor05]. One can understand mesh editing methods as also being embedding methods, but for $D=d=3$ (the term reduction becomes hardly appropriate in this case) and with added embedding constraints in the form of user-specified handles to control how to deform the shape—see, e.g., [SZT*07,HTZ*11,HSTP11]. In fact, a simple non-linear 3D modeling approach known as ARAP [SA07] (for as-rigid-as-possible) is precisely a special case of Local Tangent Space Alignment (LTSA) restricted to special orthogonal transformations: both the local alignment from the original shape to the final embedding through SVD and the alternating optimization between fixed rotations and positions were already spelled out in the Appendix of [ZZ04]. The only differences in the mesh case of ARAP reside in the choice of local neighborhoods (mesh one-rings for ARAP vs. K nearest neighbors in LTSA) and the additional constraints controlling the coordinates of the few handles. Based on these similarities, it is clear that we can leverage our new SAKE embedding approach to provide an alternative mesh (or pointset) deformation technique, where a standard off-the-shelf eigensolver can be used to solve the resulting non-linear optimization.

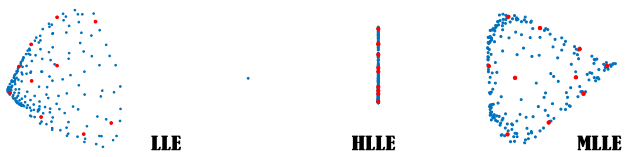
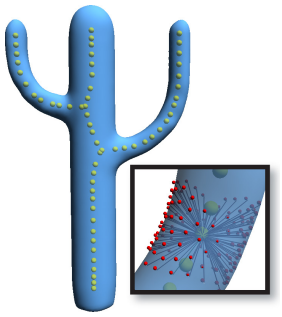


Figure 12: ***Knights.*** *Examples of results for the knights images (see Fig. 1 for comparison). Other methods give markedly more distorted results; HLLE, in particular, fails to return a valid parameterization.*

4.2 Geometry deformation through eigenanalysis

We propose to use the SAKE framework to achieve mesh deformation with handle position constraints. Notably, we aim at formulating a non-linear deformation approach that only requires a sparse matrix eigenanalysis—instead of relying on customized non-linear solvers with often slow convergence properties. This is trivially achieved (and without the need for as-isometric-as-possible unfolding of neighborhoods in this pure 3D case) by leveraging and extending our multi-Laplacian operator and its affine kernel: we accumulate constraints for the 3D points to stay close to their original positions with respect to their neighbors, and treat user-provided absolute handle positions by systematically replacing them by positions *relative* to other handle positions. We thus bypass the use of absolute position constraints or local rotations, and can thus invoke a regular eigensolver instead. This allows us to efficiently accommodate deformation properties of the 3D volume defined by the input geometry that are typically desirable in surface editing tools such as low local stretching and low volume dilation.

Surface and volume neighborhoods. In order to make our explanations valid for meshes and pointsets alike, we will use the term "point" to refer either to a point within a pointset, or to a vertex of an input mesh. For each point $\mathbf{x}_i$ of the input, we fix a set of K neighbors $\mathbf{x}_j$ for indices j in $\mathcal{N}(i)$ just like in Sec. 2.2—if we are dealing with an input mesh, the neighbors can also be defined via a fixed number of rings around the vertex instead. We also add a few internal points that are coarsely sampling the skeleton (also called medial axis) of the input geometry; this is easily achieved by, e.g., computing a Voronoi tessellation of the input points and extracting a subset of the *inside poles* [AB99] that form a discrete approximation of the skeleton. Each of these sparsely inserted skeletal points are then given a neighborhood which consists of their immediate skeletal neighbors and the few closest input points (i.e., on the medial ball [AB99]) along with their associated neighborhoods: these neighborhood can be understood as forming an internal set of radial trusses (see inset) for which we will *also* impose isometry preservation during the deformation—thus avoiding local collapses of the initial volume as much as possible. This is very much in the spirit of the volumetric graph Laplacian approach in [ZHS*05] where a sparse sampling of the interior of a mesh is also used to prevent local volume loss. Finally, the user picks a subset of input points $\{\mathbf{x}_h\}$ as "handles" for which she assigns target positions $\bar{\mathbf{x}}_h$ to guide the deformation. For clarity thereafter, we denote by $\mathcal{X}$ the set of indices of the input points that are not handles, $\mathcal{V}$ the set of indices of the added inner skeletal points, and $\mathcal{H}$ the set of indices of the handles.



Multi-Laplacian assembly. For every point $\mathbf{x}_i$ (be it a handle, regular, or skeletal point) and its associated set $\{\mathbf{x}_j\}_{j\in\mathcal{N}(i)}$ of neighbors, we assemble the multi-Laplacian matrix $\mathbf{L}_i$ as described in Sec. 2.4. Moreover, we also assemble an additional multi-Laplacian, denoted $\mathbf{H}_h$, per handle point as follows: we assign for each handle $\mathbf{x}_h (h\in\mathcal{H})$ the set of indices $\mathcal{N}(h)=\mathcal{H}\setminus\{h\}$ containing all the other handles (or only the K nearest handles when the number of handles is large). The matrix $\mathbf{H}_h$ is then defined as a regular multi-Laplacian, but for this virtual neighborhood made of handles only, and for the user-prescribed target positions $\bar{\mathbf{x}}_h$ instead of the original positions $\mathbf{x}_h$. The linear equations encoded by this matrix will force the handle positions to be placed, *relative to each other*, the way the user asked for.

Quadratic form to enforce constraints. We can now formulate our geometry deformation approach as a special instance of SAKE where we look for the vertices $\mathbf{z}$ satisfying

$$\mathbf{z} = \operatorname*{argmin}_{\mathbf{y}\ \text{s.t.}\ \mathbf{y}^t\mathbf{y}=\mathbf{Id}} \operatorname{Tr}\left[\mathbf{y}^t\mathbf{Q}\,\mathbf{y}\right],$$

where the quadratic form $\mathbf{Q}$ is defined through:

$$\begin{aligned}\mathbf{Q} = &\sum_{i\in\mathcal{X}} \mathbf{S}_i^t\,\mathbf{L}_i^t\,\mathbf{L}_i\,\mathbf{S}_i \\ &+\sum_{h\in\mathcal{H}} \mathbf{S}_h^t\,\left(w_{\mathcal{H}}\mathbf{L}_h^t\,\mathbf{L}_h + w_C\mathbf{H}_h^t\,\mathbf{H}_h\right)\,\mathbf{S}_h \\ &+w_{\mathcal{V}}\sum_{i\in\mathcal{V}} \mathbf{S}_i^t\,\mathbf{L}_i^t\,\mathbf{L}_i\,\mathbf{S}_i.\end{aligned} \tag{2}$$

The form $\mathbf{Q}$ thus enforces the linear constraints of not only the surface neighborhoods (first term and part of the second term, where handle neighborhoods are scaled by $w_{\mathcal{H}}$ to strengthen the local rigidity around handles), but also of the volume neighborhoods (third term, with a specific volume control coefficient $w_{\mathcal{V}}$), and of the handles with respect to each other (final part of the second term, using a strength w_C for these constraints).

Final shape extraction. Finding the final deformed shape is achieved by extracting the first four eigenvectors $\mathbf{q}_m$ corresponding to the lowest eigenvalues of the sparse SPD matrix $\mathbf{Q}$ (the first one being constant and corresponding to the zero eigenvalue, see Sec. 2.6). Because the eigensolver returns the positions up to a rigid transform and up to scale, we must compute a global affine transform to map these resulting positions where they need to be. Denoting the resulting handle positions $\mathbf{p}_h = (1,(\mathbf{q}_2)_h,(\mathbf{q}_3)_h,(\mathbf{q}_4)_h)^t$ for $h\in\mathcal{H}$, we find through a least squares solve the affine transformation encoded by a 3×4 matrix $\mathbf{A}$ such that $\mathbf{A}\mathbf{p}_h$ best fits the user-prescribed $\bar{\mathbf{x}}_h$. Then the final, deformed geometric positions are simply set via

$$\mathbf{z}_i = \mathbf{A}\begin{pmatrix}1 & (\mathbf{q}_2)_i & (\mathbf{q}_3)_i & (\mathbf{q}_4)_i\end{pmatrix}^t.$$

4.3 Implementation details

Our approach is rather simple as it only requires the assembly a large, sparse, and symmetric matrix $\mathbf{Q}$ and the use of sparse matrix eigenanalysis tool—we use the *Spectra* library [QGN15] in our implementation. Aside from the modeling parameters to influence the way a shape deforms (we use $w_{\mathcal{H}}=5$, and $w_C=w_{\mathcal{V}}=1$ in all 3D examples in this paper, but other choices including geometry-dependent coefficients can be used to achieve desired effects), there is no need for a customized numerical technique to solve for our non-linear deformation. Note that if the total number of handles is 4 or less, we cannot express their target 3D locations as linear combinations of the others as we did in Eq. (1) since their edge matrices $\mathbf{E}_h$ will have trivial kernels. We thus create weaker constraints by enforcing that only one or two of the handle coordinates are linear combinations of the other handles' coordinates.

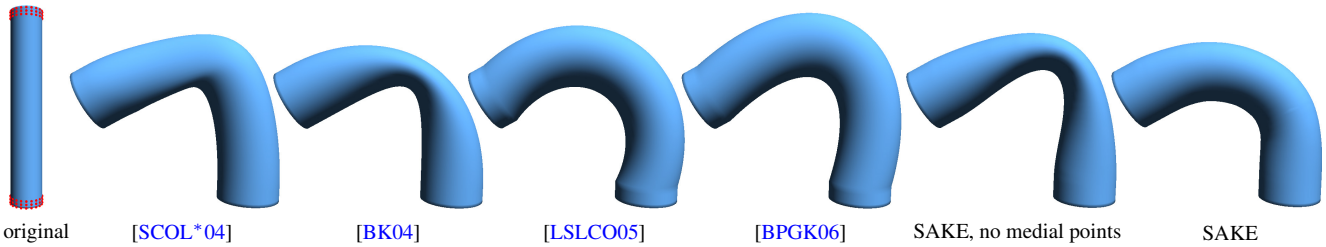


Figure 13: ***Bending bars.*** *We compare our approach (with and without added medial points for clarity) with a number of existing deformation methods on the* 120° *bend example with handles (marked in red on the original mesh) taken from [BS08]. Note the absence of artifacts near the pinned extremities for SAKE.*

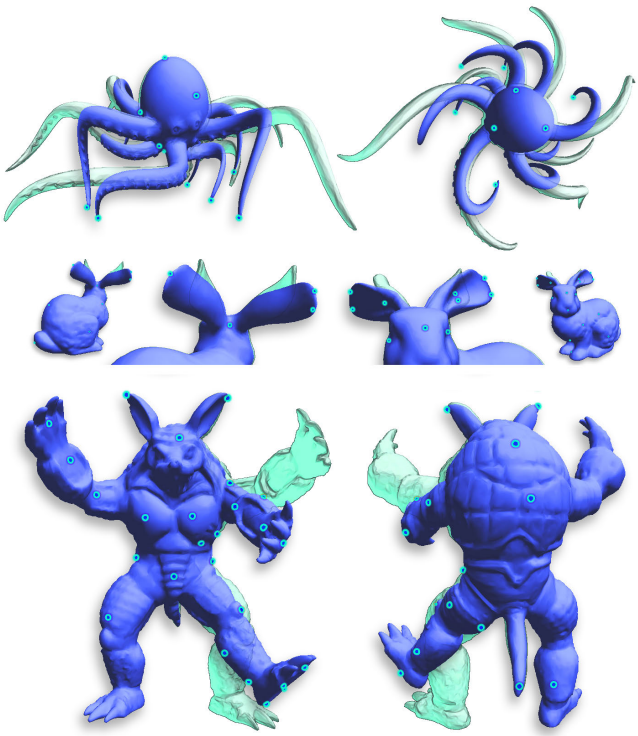


Figure 14: ***Zoo.*** *We use our SAKE editing approach on a few basic 3D models. Light blue markers indicate handle vertices (held fixed or displaced). All models use the same editing parameters; a subsampled medial axis was only added to the octopus model (20 added points) to increase the volume rigidity of the long and thin tentacles.*

Possible variants. More specific deformation behavior of the input geometry can be also easily added; for instance, a shell-like behavior can be incorporated by adding offset points along the positive and/or negative normal direction to the surface to mimic the way PRIMO [BPGK06] was adding bending rigidity. While we described how to apply SAKE to surfaces, discrete volumes (given as meshes or pointsets) are handled in exactly the same way. Additionally, notice that our multi-Laplacian based quadratic form can also be used to achieve realtime deformation using handle constraints as in [WJBK15]: substituting the modified natural boundary Laplacian used in their approach by our multi-Laplacian will add rigidity as our multiple relative coordinates capture the original geometry more tightly. However, requiring every handle position to be exactly satisfied can inherently lead to a loss of local smoothness (see, for instance, artifacts near pinned extremities [WJBK15] compared to our results in Fig. 13), an issue that our spectral approach avoids—without significant computational overhead. Finally, we note that our treatment of handles through *relative coordinates* can also turn their work into a spectral approach.

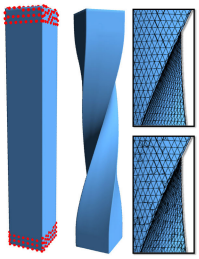
Computational efficiency. Once an input geometry is given, one can compute the local multi-Laplacian operator for each neighborhood as a preprocessing step. When the handles have been selected and positioned as desired, the small matrices $\mathbf{H}_h$ can be computed on the fly and the corresponding terms added to the quadratic form according to Eq. (2). Assembling $\mathbf{Q}$ is done in $O(n)$ as in any other mesh deformation method. An eigensolver can then be used on $\mathbf{Q}$ to find its lowest eigenvectors, and the undeformed positions can be used as a good initialization. On a Intel® dual-core Core i7 powered laptop, creating and Choleski-factorizing the quadratic form takes 5 seconds on a 10kV mesh like the bunny with a straightforward non-optimized code, while the eigensolver returns the proper embedding in 0.2 seconds. Note that the result of the deformation could be updated quite efficiently as the handles are moved if realtime editing is needed: low-rank update of the partial eigendecomposition of our symmetric matrix $\mathbf{Q}$ can be leveraged [BNS78, Bra06] since only the matrices $\mathbf{H}_h$ are changing—although we did not need to implement this variant given the current timings. Finally, note that control over rigidity could have been achieved by changing the locality of the neighborhoods: the larger they are, the more rigid the local volumes become. Our addition of a few points along the skeleton was preferred to offer rigidity control by only marginally increasing the size of the matrix, but without impacting its sparsity. The user can easily add more constraints by inserting extra rows and columns in $\mathbf{Q}$.

Numerics. Our approach uses an eigensolve to handle fully non-linear mesh deformation. Compared to other non-linear methods, solving an eigenvalue problem is simple and efficient: many off-the-shelf libraries exist as well, offering efficient eigensolvers on multi-core architectures with convergence guarantees (often lacking both in theory and in practice for other non-linear methods) even for significant deformations.

4.4 Results

We present examples of deformation in Fig. 14 on the bunny, armadillo, and octopus meshes. In each example, the same default values for the weights in Eq. 2 are used. We only used inside poles on the octopus (20 virtual points) to add volume rigidity to the tentacles. We also provide comparisons with previous deformation techniques on a simple cylinder mesh (Fig. 13) based on handle constraints provided in [BS08]; without inside poles, the

surface deformation is isometric but the resulting shape exhibits local volume change (it behaves like an inextensible shell). With just 10 inside poles added as virtual points, the bar keeps its round section through the whole cylinder length. Contrary to the results of [LSLCO05, BPGK06, WJBK15] in Fig. 13, our approach does not exhibit smoothness artifacts near the pinned extremities of the tube. We also tested twisting the bar from [BS08] without poles, see inset; note that the resulting shape does not depend on the quality of the triangulation: both the mesh and a refined version with nearly degenerate elements (closeups) lead to the same twisted shape. With our eigensolver, user-defined handle positions were always matched within 10^{-3} by the reconstruction for normalized bounding boxes, and further reduced to 10^{-5} if the handle weight $w_{\mathcal{H}}$ is raised to 50. Finally, we note that one can think of our SAKE deforming tool for 3D shapes as a ***multi-cage approach***: every point is considered as part of a local cage as it is expressed as a linear combination of neighbors like in a conventional cage-based deformation method. The only difference is that the final embedding is found via a non-linear solve involving a global, sparse eigenvalue problem to satisfy *all* cage constraints as well as possible.



5 Conclusions

In this paper, we introduced an approach for finding a low dimensional embedding of a d-manifold originally embedded in a high dimension. A global embedding in $\mathbb{R}^d$ is found such that its coordinates are as affine as possible within the local isometric parameterization of each small neighborhood, enforcing a most-isometric map. In the process, we revisited a variety of well-established manifold learning approaches, providing geometric improvements such as local linear precision to these nonlinear reduction methods. We formulated a spectral affine-kernel embedding framework, based on local eigenanalysis of each neighborhood followed by a global, sparse spectral solve to find the best low-dimension embedding, which is significantly more robust to irregular sampling and to reasonable amounts of noise than previous methods. We also proposed a spectral as-rigid-as-possible deformation tool for 3D data which leverages our SAKE approach to offer non-linear shape editing without having to resort to dedicated non-linear solvers.

With the increasing interest in data-driven computations, further extending current geometry processing tools to analyze, encode, and edit high dimensional datasets as we started in this paper is an interesting direction of future research. Improved robustness to noise and outliers could be studied by leveraging the various "Robust PCA" methods within each step of our approach—but a thorough evaluation of its adequacy for various contexts is an endeavor in itself. Targeting other parameterization properties (such as conformality, measure preservation, etc) may also improve generality. Directly guessing the dimensionality of the input data (see Fig. 15) has been studied in many contexts [PBJD79, CH04, LB04], but an approach using local scale selection like in [Bra03] or [GCSA13] could also prove valuable. Spectral clustering [NJW01], in which an embedding is used as an intermediate step, may also benefit from SAKE. Finally, out of sample extensions [BPV*03] and consolidation of high-dimensional datasets (where high dimensional points are denoised, resampled, and even inserted based on the voids in the reduced embedding as is commonly used for 3D pointsets [ABCO*03]) may be worth exploring as well.

Acknowledgments. This work was partially funded by the National Science Foundation (CCF-1655306, CCF-1655422, and III-1302285). MD gratefully acknowledges the Inria International Chair program, and wishes to thank all the members of the TITANE team for their kindness and support during his visits. Thanks to Gloria Yin as well for finding typos.

References

[AB99] AMENTA N., BERN M.: Surface reconstruction by Voronoi filtering. *Discrete Comput. Geom. 22*, 4 (1999), 81–504. 10

[ABCO*03] ALEXA M., BEHR J., COHEN-OR D., FLEISHMAN S., LEVIN D., T. SILVA C.: Computing and rendering point set surfaces. *IEEE Trans. Vis. Comput. Graphics 9*, 1 (2003), 3–15. 12

[BG05] BORG I., GROENEN P. J. F.: *Modern multidimensional scaling: Theory and applications*, 3rd ed. Springer, 2005. 2, 5

[BK04] BOTSCH M., KOBBELT L.: An intuitive framework for real-time freeform modeling. *ACM Trans. Graph. 23*, 3 (2004), 630–634. 11

[BN01] BELKIN M., NIYOGI P.: Laplacian eigenmaps and spectral techniques for embedding and clustering. In *NIPS* (2001), pp. 585–591. 2, 3, 5, 7

[BNS78] BUNCH J. R., NIELSEN C. P., SORENSEN D. C.: Rank-one modification of the symmetric eigenproblem. *Numerische Mathematik 31*, 1 (1978), 31–48. 11

[BPGK06] BOTSCH M., PAULY M., GROSS M., KOBBELT L.: PriMo: Coupled prisms for intuitive surface modeling. In *Symp. Geom. Proc.* (2006), pp. 11–20. 2, 3, 11, 12

[BPV*03] BENGIO Y., PAIEMENT J.-F., VINCENT P., DELALLEAU O., ROUX N. L., OUIMET M.: Out-of-sample extensions for LLE, Isomap, MDS, Eigenmaps, and Spectral Clustering. In *Neural Information Processing Systems* (2003), pp. 177–184. 12

[Bra03] BRAND M.: Charting a manifold. In *Neural Information Processing Systems* (2003), pp. 985–992. 12

[Bra06] BRAND M.: Fast low-rank modifications of the thin Singular Value Decomposition. *Linear Algebra and its Applications 415*, 1 (2006), 20–30. 11

[BS08] BOTSCH M., SORKINE O.: On linear variational surface deformation methods. *IEEE Trans. Vis. Comp. Graph. 14*, 1 (2008), 213–230. 11, 12

[CH04] COSTA J. A., HERO A. O.: Geodesic entropic graphs for dimension and entropy estimation in manifold learning. *IEEE Trans. Signal Process. 52*, 8 (2004), 2210–2221. 12

[CLL*05] COIFMAN R. R., LAFON S., LEE A. B., MAGGIONI M., WARNER F. F. J., ZUCKER S. W.: Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps. In *National Academy of Sciences* (2005), vol. 102(21), pp. 7426–7431. 5

[CLZW07] CHEN Z., LIU L., ZHANG Z., WANG G.: Surface parameterization via aligning optimal local flattening. In *ACM Symposium on Solid and Physical Modeling* (2007), pp. 291–296. 1

[CWW13] CRANE K., WEISCHEDEL C., WARDETZKY M.: Geodesics in heat: A new approach to computing distance based on heat flow. *ACM Trans. Graph. 32*, 5 (Oct. 2013), Art. 152. 5

[DG03] DONOHO D. L., GRIMES C.: Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. *Proceedings of the National Academy of Sciences 100*, 10 (2003), 5591–5596. 2, 3, 6, 8

[dGAOD13] DE GOES F., ALLIEZ P., OWHADI H., DESBRUN M.: On the equilibrium of simplicial masonry structures. *ACM Trans. Graph. 32*, 4 (2013), Art. 93. 5

[dGLB*14] DE GOES F., LIU B., BUDNINSKIY M., TONG Y., DESBRUN M.: Discrete 2-tensor fields on triangulations. *Comput. Graph. Forum 33*, 5 (2014), 13–24. 5

[Dij59] DIJKSTRA E. W.: A note on two problems in connexion with graphs. *Numerische Mathematik 1* (1959), 269–271. 4

[DKT08] DESBRUN M., KANSO E., TONG Y.: Discrete differential forms for computational modeling. In *Discrete Differential Geometry*, A. Bobenko et al., (Ed.). Birkhäuser Basel, 2008, pp. 287–324. 5

[DST02] DE SILVA V., TENENBAUM J. B.: Global vs. local methods in nonlinear dimensionality reduction. In *NIPS* (2002), pp. 705–712. 2, 7

[GCSA13] GIRAUDOT S., COHEN-STEINER D., ALLIEZ P.: Noise-adaptive shape reconstruction from raw point sets. *Comput. Graph. Forum 32*, 5 (2013), 229–238. 12

[GR08] GOLDBERG Y., RITOV Y.: LDR-LLE: LLE with low-dimensional neighborhood representation. In *Symposium on Visual Computing* (2008), pp. 43–54. 7

[GVL96] GOLUB G. H., VAN LOAN C. F.: *Matrix Computations (3rd ed.)*. Johns Hopkins University Press, 1996. 8

[HSL*06] HUANG J., SHI X., LIU X., ZHOU K., WEI L.-Y., TENG S.-H., BAO H., GUO B., SHUM H.-Y.: Subspace gradient domain mesh deformation. *ACM Trans. Graph. 25*, 3 (2006), 1126–1134. 2

[HSTP11] HILDEBRANDT K., SCHULZ C., TYCOWICZ C. V., POLTHIER K.: Interactive surface modeling using modal analysis. *ACM Trans. Graph. 30*, 5 (2011), Art. 119. 2, 9

[HTZ*11] HUANG J., TONG Y., ZHOU K., BAO H., DESBRUN M.: Interactive shape interpolation through controllable dynamic deformation. *IEEE Trans. Vis. Comput. Graphics 17*, 7 (2011), 983–992. 9

[HWX10] HUI K., WANG C., XIAO B.: Globally-preserving based locally linear embedding. In *International Conference on Pattern Recognition* (2010), pp. 531–534. 2

[LB04] LEVINA E., BICKEL P. J.: Maximum likelihood estimation of intrinsic dimension. In *NIPS* (2004), pp. 777–784. 12

[Lig16] LIGHTSCAPE: Light Stage Data Gallery, University of Southern California's Institute for Creative Technologies; http://gl.ict.usc.edu/Data/LightStage/, 2016. 1, 9

[LSLCO05] LIPMAN Y., SORKINE O., LEVIN D., COHEN-OR D.: Linear rotation-invariant coordinates for meshes. *ACM Trans. Graph. 24*, 3 (2005), 479–487. 11, 12

[MTAD08] MULLEN P., TONG Y., ALLIEZ P., DESBRUN M.: Spectral conformal parameterization. In *Symp. Geom. Proc.* (2008), pp. 1487–1494. 6

[NJW01] NG A. Y., JORDAN M. I., WEISS Y.: On Spectral Clustering: Analysis and an algorithm. In *NIPS* (2001), pp. 849–856. 12

[PBJD79] PETTIS K. W., BAILEY T. A., JAIN A. K., DUBES R. C.: An intrinsic dimensionality estimator from near-neighbor information. *IEEE Trans. Pattern Anal. Mach. Intell 1*, 1 (1979), 25–37. 12

[PZZK04] PARK J., ZHANG Z., ZHA H., KASTURI R.: Local smoothing for manifold learning. In *Computer Vision and Pattern Recognition* (2004), vol. 2, pp. 452–459. 8

[QGN15] QIU Y., GUENNEBAUD G., NIESEN J.: Spectra. http://yixuan.cos.name/spectra, 2015. 10

[RBBK10] ROSMAN G., BRONSTEIN M. M., BRONSTEIN A. M., KIMMEL R.: Nonlinear dimensionality reduction by topologically constrained isometric embedding. *International Journal of Computer Vision 89*, 1 (2010), 56–68. 2

[RS00] ROWEIS S. T., SAUL L. K.: Nonlinear dimensionality reduction by locally linear embedding. *Science 290* (2000), 2323–2326. 2, 3, 7

[SA07] SORKINE O., ALEXA M.: As-rigid-as-possible surface modeling. In *Symp. Geom. Proc.* (2007), pp. 109–116. 3, 9

[Sci16] SCIKIT: Scikit Learn Machine Learning in Python, sklearn.manifold module; http://scikit-learn.org/, 2016. 7

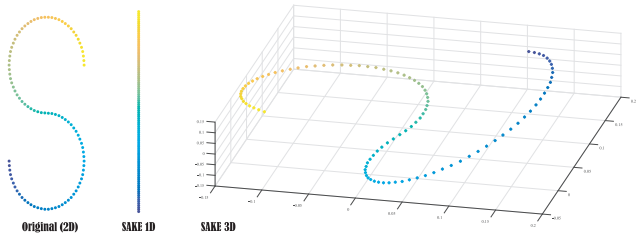


Figure 15: ***Choice of dimensionality.*** *For a 2D S-shaped pointset (left), its reduction to 1D provides the expected parameterization (middle); a 2D embedding would return the exact same input shape as the original; a 3D embedding, instead, exhibits unexpected distortions in the extra dimension (right); and of course, a 0D embedding would lose injectivity.*

[SCOL*04] SORKINE O., COHEN-OR D., LIPMAN Y., ALEXA M., RÖSSL C., SEIDEL H.-P.: Laplacian surface editing. In *Symp. Geom. Proc.* (2004), pp. 175–184. 11

[SM00] SHI J., MALIK J.: Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell 22*, 8 (2000), 888–905. 2, 7

[Sor05] SORKINE O.: Laplacian mesh processing. In *Eurographics State of the Art Reports* (2005). 2, 9

[Sor06] SORKINE O.: Differential representations for mesh processing. *Comput. Graph. Forum 25* (2006), 789–807. 2

[SS09] SCHMIDT R., SINGH K.: *Approximate Conformal Parameterization of Point-Sampled Surfaces*. Report, Technical report CSRG-605, Computer Science Department, University of Toronto, 2009. 1

[SZT*07] SHI X., ZHOU K., TONG Y., DESBRUN M., BAO H., GUO B.: Mesh puppetry: Cascading optimization of mesh deformation with inverse kinematics. *ACM Trans. Graph. 26*, 3 (2007), Art. 81. 9

[TSL00] TENENBAUM J. B., SILVA V. D., LANGFORD J. C.: A global geometric framework for nonlinear dimensionality reduction. *Science 290*, 5500 (2000), 2319–2323. 2, 3, 4, 5, 9

[VDMPVdH09] VAN DER MAATEN L., POSTMA E., VAN DEN HERIK J.: Dimensionality reduction: a comparative review. *J. Mach. Learn. Res. 10* (2009), 66–71. 2

[VMS16] VIDAL R., MA Y., SASTRY S. S.: *Generalized Principal Component Analysis*. Springer, 2016. 5

[WJBK15] WANG Y., JACOBSON A., BARBIČ J., KAVAN L.: Linear subspace design for real-time shape deformation. *ACM Trans. Graph. 34*, 4 (July 2015), Art. 57. 11, 12

[XDW16] XING X., DU S., WANG K.: Robust Hessian locally linear embedding techniques for high-dimensional data. *Algorithms 9*, 2 (2016), 36. 8

[YZ15] YE Q., ZHI W.: Discrete Hessian Eigenmaps method for dimensionality reduction. *Journal of Computational and Applied Mathematics 278* (2015), 197–212. 2

[ZCO15] ZHANG H. R., COHEN-OR D.: Dimensionality reduction. In *A Sampler of Useful Computational Tools for Applied Geometry, Graphics, and Image Processing*. CRC Press, 2015, ch. 9, pp. 131–145. 1

[ZHS*05] ZHOU K., HUANG J., SNYDER J., LIU X., BAO H., GUO B., SHUM H.-Y.: Large mesh deformation using the volumetric graph Laplacian. *ACM Trans. Graph. 24*, 3 (2005), 496–503. 2, 10

[ZKK02] ZIGELMAN G., KIMMEL R., KIRYATI N.: Texture mapping using surface flattening via multidimensional scaling. *IEEE Trans. Vis. Comp. Graph. 8*, 2 (Apr 2002), 198–207. 1, 5

[ZW07] ZHANG Z., WANG J.: MLLE: Modified locally linear embedding using multiple weights. In *NIPS* (2007), pp. 1593–1600. 7

[ZZ04] ZHANG Z., ZHA H.: Principal manifolds and nonlinear dimensionality reduction via tangent space alignment. *SIAM J. Sci. Comput. 26*, 1 (2004), 313–338. 2, 7, 9