LaSeSOM: A Latent Representation Framework for Semantic Soft Object Manipulation

Peng Zhou, *Student Member, IEEE*, Jihong Zhu, *Member, IEEE*, Shengzeng Huo, and David Navarro-Alarcon, *Senior Member, IEEE*

***Abstract*—Soft object manipulation has recently gained popularity within the robotics community due to its potential applications in many economically important areas. Although great progress has been recently achieved in these types of tasks, most state-of-the-art methods are case-specific; They can only be used to perform a single deformation task (e.g. bending), as their shape representation algorithms typically rely on "hard-coded" features. In this paper, we present LaSeSOM, a new feedback latent representation framework for semantic soft object manipulation. Our new method introduces internal latent representation layers between low-level geometric feature extraction and high-level semantic shape analysis; This allows the identification of each compressed semantic function and the formation of a valid shape classifier from different feature extraction levels. The proposed latent framework makes soft object representation more generic (independent from the object's geometry and its mechanical properties) and scalable (it can work with 1D/2D/3D tasks). Its high-level semantic layer enables to perform (quasi) shape planning tasks with soft objects, a valuable and underexplored capability in many soft manipulation tasks. To validate this new methodology, we report a detailed experimental study with robotic manipulators.**

***Index Terms*—Robot manipulators, soft objects, shape control, semantic deformation, feature representation.**

I. Introduction

RECENT studies have shown that the manipulation of soft objects is crucial and indispensable to achieve high autonomy in robots [1]. For instance, many applications need to actively shape food materials [2], handle and fold fabrics [3], assemble flexible automotive parts [4], manipulate cables or sutures [5], palpate organs and tissues [6] (see [7] for a comprehensive review). Although great progress has been recently achieved, the *feedback* manipulation of soft objects is still a challenging research question. The implementation of these types of advanced manipulation capabilities is complicated by various issues. Amongst the most important is the difficulty in characterizing the feedback shape of a soft object. Our aim in this work is to develop new data-driven methods that can quantitatively describe deformable shapes.

Hirai [8] first demonstrated how feedback controls could deform a soft object into a desired 2D shape. This early work is a clear example of a *shape representation* based on points (simple but cannot generalize). Other classical methods are based on geometric features e.g. angles, curvatures, catenaries [9]–[12]; Its disadvantage is that they are case-specific, thus, can only be used to perform a single shaping action (hence inapplicable to unstructured scenarios). Some works have addressed this issue by developing generic representations that only require sensory data. For example, [13], [14], and [15] characterize shapes using Fourier series and feature histograms; These methods, however, create very large feature vectors, which may not be the most efficient feedback metric. A more effective solution is to automatically compute generic feedback features (e.g. as in direct visual servoing [16], [17]) and combine them with dimension reduction techniques, as in e.g. [18], [19].

Data-driven based shape analyses [20], [21] have gained in popularity as it offers a useful alternative to model-based approaches. An increasing amount of research have focus on different-level segmentation and shape classifications (see [22], [23], and [24]). However, these methods purely depend on the designed end-to-end pipeline which ignores the semantic meaning of internal features and thus failing to interpret the entire analytical process. Therefore, latest applications started to examine attribute-based approaches, such as binary attributes [25], relative attributes [26], and semantic image color palette editing [27]. Several works [28], [29] further combine shape analysis and semantic attributes for a in-depth deformation analysis. Our approach has the same purpose, with an emphasis on 3D shape deformation, to achieve a comprehensive semantic shape analysis on 3D object deformation.

Latent space approaches have recently achieved many successful results in text mining and image analysis [30], due to its capability to encode high-dimensional data into a meaningful internal representation. By using concise low-dimensional latent variables and highly flexible generators, a latent space allows us to generate new data samples on data space. In this manner, a deformation planning problem of soft objects can be solved in a novel way by constructing a feasible sequence of deformable shapes in latent space. However, many works [31] have adopted a linear interpolation in remapping the latent variables back to data space, which could cause serious distortions on the generated samples for a shape planning scenario. For example, consider a generator g and a latent variable $\mathbf{z}$ with two infinitesimal shifts δ_1 and δ_2, then the distance with Taylor's expansion [32] is formulated by:

$$\|g(\mathbf{z_0}+\delta_1)-g(\mathbf{z_0}+\delta_2)\|^2 = (\Delta_{12})^\top \left(\mathbf{J}_{\mathbf{z_0}}^\top \mathbf{J}_{\mathbf{z_0}}\right)(\Delta_{12}) \qquad (1)$$

for $\mathbf{J}_{\mathbf{z_0}} = \left.\frac{\partial g}{\partial \mathbf{z}}\right|_{\mathbf{z}=\mathbf{z_0}}$ and $\Delta_{12} = \delta_1 - \delta_2$, which indicates that the

The authors are with The Hong Kong Polytechnic University, Kowloon, Hong Kong. Corresponding author e-mail: dna@ieee.org

This work is supported by the Research Grants Council (RGC) of Hong Kong under Grant Number 14203917 and in part by the Key-Area Research and Development Program of Guangdong Province 2020 under project 76.

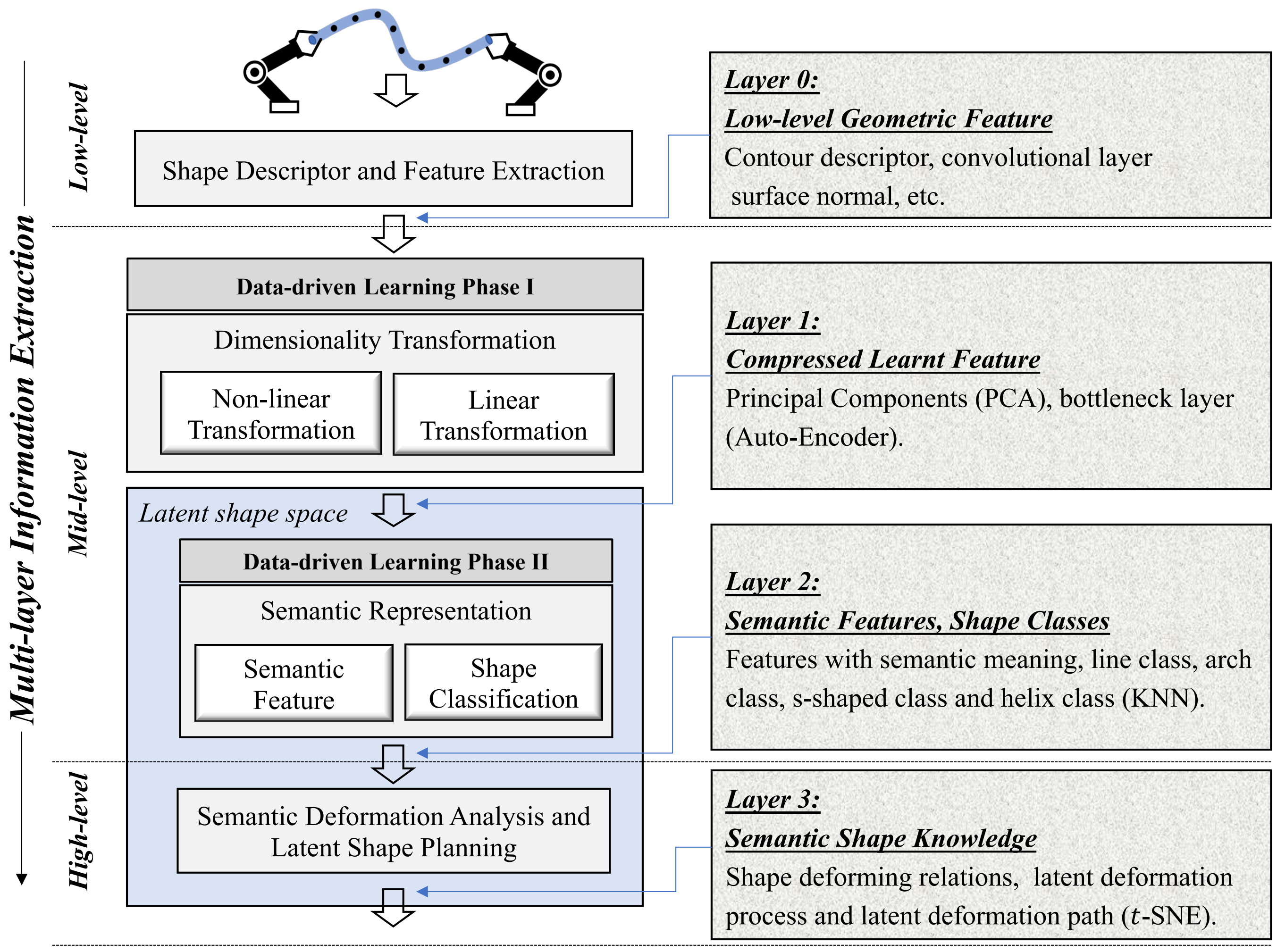

Low-level
Mid-level
High-level
Multi-layer Information Extraction
Shape Descriptor and Feature Extraction
Data-driven Learning Phase I
Dimensionality Transformation
Non-linear Transformation
Linear Transformation
Latent shape space
Data-driven Learning Phase II
Semantic Representation
Semantic Feature
Shape Classification
Semantic Deformation Analysis and Latent Shape Planning
Layer 0: Low-level Geometric Feature
Contour descriptor, convolutional layer surface normal, etc.
Layer 1: Compressed Learnt Feature
Principal Components (PCA), bottleneck layer (Auto-Encoder).
Layer 2: Semantic Features, Shape Classes
Features with semantic meaning, line class, arch class, s-shaped class and helix class (KNN).
Layer 3: Semantic Shape Knowledge
Shape deforming relations, latent deformation process and latent deformation path (t-SNE).

Fig. 1. Conceptual representation of the proposed framework — LaSeSOM that fully describes and represents the soft objects for bimanual manipulation tasks from four layers, namely, the low-level geometric feature layer, compressed learnt feature layer, semantic features and shape classes layer, and semantic shape knowledge layer.

normal distance in $\mathcal{Z}$ space changes locally as it is determined by the local Jacobian. Therefore, the generated data space should not be regarded as a linear Euclidean space, but rather as a curved surface, a manifold. Consequently, seeking the shortest curve along the manifold is a more reasonable way to compute the interpolation and generate undistorted samples.

As a feasible solution to these problems, in this paper we present a general data-driven representation framework for semantic soft object manipulation that is composed of three layers: A low-level soft object geometric shape processing, a mid-level data-driven representation learning, and a high-level semantic shape analysis. Such multilayer semantic framework provides extra flexibility and interpretability in the design of complicated soft object manipulation systems.

The paper's main contributions are summarized as follows:

- An effective representation framework for soft object analysis during manipulation tasks.
- A novel semantic analysis approach for soft object manipulation tasks.
- A solution for shape planning with a geodesic path-based interpolation algorithm in the latent space.

The rest of this paper is organized as follows. Section II describes the proposed representation framework. Section III presents the representation models. Section IV shows the experimental results. Section V gives final conclusions.

II. Framework Overview

Unlike rigid objects, soft objects have infinite degree of freedoms and it is impractical to describe the shape without complex calculation and rich *priori* information. However, a semantic data-driven learning process exhibits a possibility of constructing valid and compact features in a latent space to represent soft objects without *priori* information. We introduce a three-level representation framework partitioned by a four-layer information flow as follows:

Low-Level Processing: This level provides feature extraction as the first step in soft object representation. Various descriptor and 3D representation algorithms have been proposed to date [33] [34] [35], and low-level features (contour descriptor, surface normal, Conv1d, etc.) can be extracted easily from 3D soft object data without compression. Note that this is no unified method for extracting low-level features, please choose appropriate one based on various purposes and different raw data formats. To show the generalization ability of LaSeSOM, two typical data (markers and point clouds) are introduced to describe the deforming shapes. Nevertheless, lacks of representation efficiency and hierarchical structure preclude us from using raw low-level features of soft objects to its fullest extent. Consequently, We should reduce the dimensionality of the low-level extracted shape features.

Mid-Level Representation: As shown in Fig. 1, two modularized data-driven learning phases are applied to produce a set of compressed and semantic representations. In Phase I, we employ both linear and non-linear dimensionality reduction modules to comparatively compress low-level and high-dimensional features. Linear transformation models, such as principal component analysis (PCA), try to provide the principal components for expressing original features on its significant basis. As for the non-linear transformation model, the auto-encoder network is used to construct a bottleneck layer to achieve dimensionality reduction through an unsupervised learning process. With different dimensionality transformation techniques, the entire phase I outputs concise and internal representations for soft objects in a latent space. In Phase II, we design a semantic feature analysis algorithm to detect the function of each reduced feature. Besides, building a classifier in a latent space allows us to efficiently assign each shape into one of the predefined soft object shape categories for different deformable objects. During this process, k-nearest neighbor (kNN) algorithm is selected as the classifier in the generated latent shape space.

High-Level Analysis: Soft object deformation knowledge plays an important role in manipulation tasks. This knowledge comprises two parts, namely, semantic shape relations and shape deformation planning. The morphing relations between different shapes can be discovered by observing the connectivity of a deformation trace in latent shape space. In addition, the application of a geodesic path-based interpolation technique in the latent shape space shows a valid method for a shape planning in externally observed data manifold for soft object manipulations. More importantly, the generated shapes of soft objects lies on the input data manifold, which means that the planed shapes are not only equally and softly changed but also free of distortions.

III. Methods

In this section, we first introduce basic techniques (shape features, dimensionality transformations, etc.) used in different layers, and then illustrate the connections between the latent space and original data space. More importantly, with this latent space, we design several semantic analysis algorithms to help describe the soft object deformation process. Lastly, a geodesic path generation algorithm with Riemannian metrics is proposed to solve the deformation planning problem.

A. Shape Feature

In order to apply this framework into various soft object manipulation tasks, two typical data formats are selected to depict the soft object shape. One is the ordered marker point data in the format of a set of ordered 3D points that is widely used in the motion tracking system, and the other is a popular point cloud data to represent a geometric shape surface via a set of large quantities of unordered 3D points in a Euclidean space. Formally, m soft object shapes compose of the entire deformation process, and each soft object shape S_i can be denoted as an ordered marker point set $\mathcal{M}_i$ represented by n marker points. Consequently, the entire deformation can represented as a shape matrix $\mathbf{X}$ as below:

$$\boldsymbol{X} = \begin{pmatrix} x_1^{(1)} & y_1^{(1)} & z_1^{(1)} & \dots & x_n^{(1)} & y_n^{(1)} & z_n^{(1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ x_1^{(m)} & y_1^{(m)} & z_1^{(m)} & \dots & x_n^{(m)} & y_n^{(m)} & z_n^{(m)} \end{pmatrix}$$

where the coordinates of the markers have been fatten so each row with n markers has $3n$ features and the number of total shapes during this deformation is denoted by m. To approximate the contour composed of 3D marker points, Fourier approximation [36] is selected since this shape descriptor can depict the shape with arbitrary precision. However, this descriptor is usually used for 2D shape modeling problems. Consequently, in the low-level representation, we expand this descriptor into a 3D configuration as below:

$$\begin{aligned} x(l) &= a_0 + \sum_{N}^{n=1}(a_n cos(wnl) + b_n sin(wnl)) \\ y(l) &= c_0 + \sum_{N}^{n=1}(c_n cos(wnl) + d_n sin(wnl)) \\ z(l) &= e_0 + \sum_{N}^{n=1}(e_n cos(wnl) + f_n sin(wnl)) \end{aligned} \quad (2)$$

where a_0, c_0, and e_0 are the bias components of the Fourier descriptor with a frequency of 0, and l is a same length that periodically circles along the entire length of soft object denoted by L, and the coefficients of the n-th harmonic are represented by a_n, b_n, c_n, d_n, e_n and f_n which can be solved with expressions of [36] to constitute the description of the shape.

A deformable shape S_i can also be represented as a point cloud data $\mathcal{P}_i$. With farthest point sampling algorithm used in PointNet++ [37], the raw point cloud can be sampled into $\mathcal{P}_i^{'}$ with a fixed input size $3N$, where N is the resolution of the resampled point cloud, which means is the total number of points in this point cloud. Thus, given a point cloud $\mathcal{P}^{'}$, the input shape matrix can be represented as $\mathbf{X}_{in} \in \mathbb{R}^{N \times 3}$. The feature extraction process follows the design principle of PointNet [38]: increasing the features with convolutional 1D layers (By doing so, each point in $\mathcal{P}^{'}$ can be encoded independently); After the convolutions is connected a "symmetric" and permutation-invariant function (e.g. a max pooling) to generate a joint feature representation in a size of $1 \times N$. In this paper, we select the Chamfer(pseudo)-distance (CD) as the permutation-invariant metric for comparing unordered point sets. Given two point cloud set $\mathcal{P}_i$ and $\mathcal{P}_j$, this metric measures the squared distance between corresponding nearest neighbors in different sets:

$$d_{CD}\left(\mathcal{P}_i, \mathcal{P}_j\right) = \sum_{x \in \mathcal{P}_i} \min_{y \in \mathcal{P}_j} \|x - y\|_2^2 + \sum_{y \in \mathcal{P}_j} \min_{x \in \mathcal{P}_i} \|x - y\|_2^2 \quad (3)$$

B. Linear Transformation: PCA

The dimensionality reduction technique is applied in the layer 1 to discover optimal compressed features for a mid-level representation. Principal components analysis (PCA) [39] aims to provide a sequence of optimal linear transformations for low-level and high-dimensional raw features. To achieve this goal, PCA computes new variables called *principal components* which are obtained as linear combinations of the original variables. Formally, considering a shape feature matrix $\boldsymbol{X}$ with m shapes and n feature dimensions, the goal of PCA is to find a transformation $\boldsymbol{P}$ to linearly convert $\boldsymbol{X}$ to $\boldsymbol{Y}$ and reduce the original n feature dimensions into k dimensions ($k << n$), which can be denoted by $\boldsymbol{Y} = \boldsymbol{P}\boldsymbol{X}$. By re-expressing $\boldsymbol{Y}$ with regards to $\boldsymbol{P}$ and $\boldsymbol{X}$, the covariance matrix $\boldsymbol{\Sigma}_y$ can be obtained as below:

$$\boldsymbol{\Sigma}_y = \frac{1}{n-1}\boldsymbol{Y}\boldsymbol{Y}^T = \frac{1}{n-1}\boldsymbol{P}\boldsymbol{S}\boldsymbol{P}^T \tag{4}$$

where $\boldsymbol{S} = \boldsymbol{X}\boldsymbol{X}^T$ is a symmetric matrix and can be diagonalizable by a matrix which is orthonormal of its eigenvectors, such that,

$$\boldsymbol{S} = \boldsymbol{E}\boldsymbol{D}\boldsymbol{E}^T \tag{5}$$

where $\boldsymbol{D}$ is a matrix that is diagonal contains the eigenvalues of $\boldsymbol{S}$, $\boldsymbol{E}$ is a matrix that is orthonormal whose columns are orthogonal eigenvectors of $\boldsymbol{S}$. By this conversion, rows of $\boldsymbol{P}$ become the eigenvectors of $\boldsymbol{S}$, that is $\boldsymbol{E}^T$ and $\boldsymbol{\Sigma}_y$ becomes:

$$\boldsymbol{\Sigma}_y = \frac{1}{n-1}\boldsymbol{E}^T(\boldsymbol{E}\boldsymbol{D}\boldsymbol{E}^T)\boldsymbol{E} \tag{6}$$

Finally, diagonalizing the covariance matrix $\boldsymbol{\Sigma}_y$ is achieved and the principal components are the eigenvectors of $\boldsymbol{S}$. One efficient solution for the PCA problem is known as the singular value decomposition (SVD) [40].

The following semantic analysis part of LaSeSOM needs to identify the semantic meaning for compressed features. For this reason, the inverse samples reconstructed from the compressed features are needed, which can solved by:

$$\boldsymbol{X}_{rec} = \hat{\boldsymbol{X}}_{rec} + \boldsymbol{\mu} = \boldsymbol{P}^{-1}\boldsymbol{Y} + \boldsymbol{\mu} \tag{7}$$

Besides, to select an appropriate number of components, the explained variance, by calculating the fraction between variance explained by partial principal components and the total variance, is defined as:

$$v_{exp} = \frac{\sum_{i=1}^{K} v_i}{\sum_{i=1}^{N} v_i} \tag{8}$$

C. Nonlinear Transformation: Auto-Encoder

In mid-level representation, the auto-encoder(AE) [41] is used to compress shape features with non-linear transformations. As shown in Fig. 2, AE is a neural network constituted by three parts, namely, a hidden layer representing the input data known as the code or bottleneck layer, an encoder that maps the input into the code, and a decoder to map the code to a reconstruction of the original input.

Formally, an AE takes an n-dimensional soft object shape vector $\boldsymbol{x}$ as its input, which is mapped to its k-dimensional

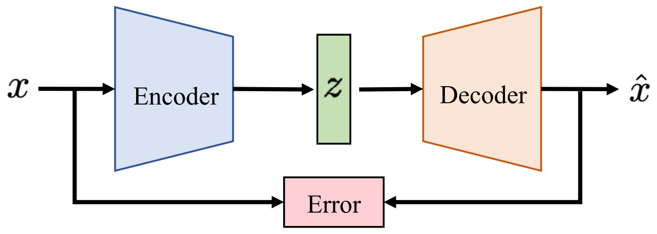


Fig. 2. Conceptual representation of the Auto-Encoder network structure.

bottleneck layer $\boldsymbol{y}$ through the deterministic equation $\boldsymbol{y} = f_{\boldsymbol{\theta}}(\boldsymbol{x}) = s(\boldsymbol{W}\boldsymbol{x} + \boldsymbol{b})$, which in turn is parameterized by $\boldsymbol{\theta} = \{\boldsymbol{W}, \boldsymbol{b}\}$. $\boldsymbol{W}$ is a $k \times n$ weight matrix, $\boldsymbol{b}$ is a vector of bias, and s is a $sigmoid$ activation function, $s(x) = \frac{1}{1+e^{-x}}$. The hidden representation is then traced back to a reconstruction $\boldsymbol{z}$ with n dimensions, which is sometimes referred to as the latent representation, where $\boldsymbol{z} = g_{\boldsymbol{\theta}'}(\boldsymbol{y}) = s\left(\boldsymbol{W}'\boldsymbol{y} + \boldsymbol{b}'\right)$, with $\boldsymbol{\theta}' = \{\boldsymbol{W}', \boldsymbol{b}'\}$. The parameters $\boldsymbol{\theta}, \boldsymbol{\theta}'$ for the model are designed to minimize the average error of reconstruction, which is defined as:

$$\boldsymbol{\theta}^*, \boldsymbol{\theta}'^* = \arg\min_{\boldsymbol{\theta},\boldsymbol{\theta}'} \frac{1}{n}\sum_{i=1}^{n} L\left(\boldsymbol{x}^{(i)}, g_{\boldsymbol{\theta}'}\left(f_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right)\right)\right) \tag{9}$$

where the loss function L needs to be changed depending on the property of input features. For example, if the input feature is the ordered features extracted by Fourier descriptor, then L could be normal mean square error (MSE). However, for the unordered point cloud features, the permutation-invariant metric defined in Eq. 3 is needed to calculate a reconstruction loss. Besides, AE probably learns the identity function and hardly extracts valid features from the input. Consequently, in most cases, a regularized empirical risk function on a data set with D_n shape samples defined as follows is required:

$$\hat{R}_\lambda\left(f_\lambda, D_n\right) = \left(\sum_{i=1}^{D_n} L\left(f_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right), \boldsymbol{z}^{(i)}\right)\right) + \lambda\Omega(\boldsymbol{\theta}) \tag{10}$$

where all parameters more or less have been penalized by Ω, and $\lambda \geq 0$ regulates the regularization degree. The AE training algorithm with stochastic gradient descent is explained in [41].

D. k-Nearest Neighbor

Based on the output of Phase I in mid-level representation, a kNN [42] classifier is introduced to distinguish different shapes. The reason for choosing this classifier is because the similar shapes are relatively close in raw input data space during the deformation. Additionally, by extending kNN with geodesic path distance, the pattern could keep unchanged in the latent space generated by AE, which makes kNN more flexible to work under different spaces. $\mathcal{D}_k$ is defined as the k-nearest shape neighbors of query shape $\boldsymbol{x}^{[q]}$ in the required space:

$$\mathcal{D}_k = \left\{\left\langle \boldsymbol{x}^{[1]}, f\left(\boldsymbol{x}^{[1]}\right)\right\rangle, \ldots, \left\langle \boldsymbol{x}^{[k]}, f\left(\boldsymbol{x}^{[k]}\right)\right\rangle\right\} \tag{11}$$

where $\mathcal{D}_k$ can be generated with the Euclidean distance measurement when $f(\boldsymbol{x})$ represents a mapping to target variable in observation space, while it can also be a set with the Riemannian distance measurement if $f(\boldsymbol{x})$ is a continuous transform function to a latent space. The kNN hypothesis remains consistent and is formulated as follows:

$$h\left(\boldsymbol{x}^{[q]}\right)=\arg\max_{y\in\{1,\ldots,t\}}\sum_{i=1}^{k}\delta\left(y,f\left(\boldsymbol{x}^{[i]}\right)\right) \tag{12}$$

where δ denotes the Kronecker Delta function. Before using the kNN, the number of nearest neighbors should be determinated. An effective strategy is to select the k with the lowest mean square error (MSE) in n-fold cross validation, which can be defined as:

$$\mathrm{CV}(n)=\frac{1}{n}\sum_{i=1}^{n}\mathrm{MSE}_i \tag{13}$$

E. Latent Shape Space

With dimensionality transformation techniques, we embed the low-level features of the collected shapes in a low-dimensional latent shape space. This subsection will illustrate the connection between the latent space and the derived manifold, which serves the foundation to develop a algorithm for geodesic path generation with Riemannian metrics. In deep generative models (i.e. AEs [43], VAEs [44]), as shown in Fig. 3, a manifold $\mathcal{M}$ is formed through a generator g mapping linear coordinates of variables in latent space $\mathcal{Z}$ ($\mathcal{Z}\subseteq\mathbb{R}^d$) into the curvilinear coordinates of originally high-dimensional data space $\mathcal{X}$ ($\mathcal{X}\subseteq\mathbb{R}^D$, normally, $d\ll D$). In fact, g is a composition function of numerous layers, which can be formulated as $g=g^{(1)}\circ g^{(2)}\circ\ldots\circ g^{(l)}$ with superscripts to denote the index of layer. For each layer, $g^{(l)}$ is an affine transformation. Combined with a nonlinear activation function ϕ, it can be represented as below:

$$g_k^{(l)}\left(y^{(l)}\right)=\phi\left(W_k^{(l)}y^{(l)}+b^{(l)}\right) \tag{14}$$

where $g_k^{(l)}$ denotes the kth component of the output and $W_k^{(l)}$ represents kth row of the weight matrix. The image of g could be regarded as a smooth (i.e., C^∞), d-dimensional immersed manifold on condition that the Jacobian $J_g(z)$ of g at every point $z\in\mathcal{Z}$ has rank d. According to the chain rules of neural nets, the condition would be satisfied if we choose a smooth and monotonic activation function, ϕ, and weight matrix has full column rank. The condition of activation function can be ensured by choosing a correct activation function in the phrase of network construction. Therefore, we can make sure that $\mathcal{M}$ is an *immersed* manifold, which means that it is locally differentiable but globally intersected d-dimensional Euclidean space.

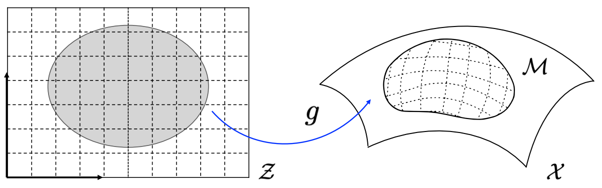


Fig. 3. Conceptual representation of a generator as a mapping from low-dimensional latent space into a manifold in input data space.

Mathematically, for every point z in the latent space $\mathcal{Z}$, $J_g(z)$, the Jacobian matrix of g, maps $T_z\mathcal{Z}$, the tangent space of $\mathcal{Z}$ at z, to $T_{g(z)}\mathcal{M}$, the tangent space of $\mathcal{M}$ at $g(z)$. In AE, $J_g(z)$ is a $d\times D$ partial derivative matrix calculated by backpropagation algorithm. Since a Riemannian metric offers the format of an inner product structure between tangent vectors in their tangent spaces $T_x\mathcal{M}$, so the induced metric would be used from the ambient data space $\mathcal{X}$. Consider two vectors $u,v\in T_x\mathcal{M}$ as living in a linear subspace of $\mathcal{X}$, the Euclidean dot product of x could be used to compute the Riemannian metric $\langle u,v\rangle$. Intuitively, the metric denotes the curvature of a Riemannian manifold and measures the extent to which deviates from being Euclidean. See standard definitions of Riemannian geometry for a detailed mathematical explaining of curvature(e.g. []). Note that a manifold $\mathcal{M}$ with zero curvature doesn't mean that it it linear. For example, bending a sheet of paper with a straight line make the surface highly nonlinear but zero curvature, because the straight line changes into a geodesic curve with the same arc length.

F. Geodesic Path on Manifolds

Through the mapping g, all the concepts (tangent vectors, tangent spaces, curves, etc.) defined in the latent space $\mathcal{Z}$ have a unique counterpart on the manifold $\mathcal{M}$. For each point $z\in\mathcal{Z}$, the Riemannian metric is defined as below:

$$G(z)=J_g(z)^TJ_g(z) \tag{15}$$

Therefore, the inner product of two tangent vectors $u,v\in T_zZ$ is $\langle u,v\rangle=u^TG(z)v$. Consider a smooth curve in the latent space $\gamma_t:[a,b]\to\mathcal{Z}$, then it has length $\int_a^b\|\dot{\gamma}_t\|\,\mathrm{d}t$, where $\dot{\gamma}_t=\mathrm{d}\gamma_t/\mathrm{d}t$ denotes the velocity of the curve. The length of this curve L lying on the manifold ($g\circ\gamma(t)\in M$) is computed as:

$$L\left[g(\gamma_t)\right]=\int_a^b\|\dot{g}\left(\gamma_t\right)\|\,\mathrm{d}t=\int_a^b\|\mathbf{J}_{\gamma_t}\dot{\gamma}_t\|\,\mathrm{d}t \tag{16}$$

where $\mathbf{J}_{\gamma_t}=\left.\frac{\partial g}{\partial\mathbf{z}}\right|_{\mathbf{z}=\gamma_t}$ and the last step follows from Taylor's Theorem, which implies the length of a curve γ_t along the surface can be computed directly in the latent space using below defined norm:

$$\|\mathbf{J}_\gamma\dot{\gamma}\|=\sqrt{\dot{\gamma}^\top\left(\mathbf{J}_\gamma^\top\mathbf{J}_\gamma\right)\dot{\gamma}}=\sqrt{\dot{\gamma}^\top\mathbf{M}_\gamma\dot{\gamma}} \tag{17}$$

Here, $\mathbf{M}_\gamma=\mathbf{J}_\gamma^\top\mathbf{J}_\gamma$ and it is a symmetric and positive definite matrix, that gives rise to the definition of a Riemannian metric for each point z in the latent space $\mathcal{Z}$. The arc length with metric $\mathbf{M}_\gamma$ can be re-expressed as:

$$L(\gamma)=\int_a^b\sqrt{\dot{\gamma}_t^\top\mathbf{M}_{\gamma_t}\dot{\gamma}_t}\mathrm{d}t \tag{18}$$

To obtain a geodesic curve, the curve length $L(\gamma)$ is locally minimized through an energy functional $E(\gamma)$ defined as:

$$E(\gamma)=\frac{1}{2}\int_a^b\dot{\gamma}(t)^TG_{\gamma(t)}\dot{\gamma}(t)dt \tag{19}$$

In Riemannian geometry, taking a variation of the geodesic energy function can lead to the Euler-Lagrange equation calculated as:

$$\frac{d^2\gamma^\mu}{dt^2}=-\Gamma_{\alpha\beta}^\mu\frac{d\gamma^\alpha}{dt}\frac{d\gamma^\beta}{dt} \tag{20}$$

where $\Gamma_{\alpha\beta}^{\mu}$ is the Christoffel symbol of the metric G, which is defined as:

$$\Gamma_{\alpha\beta}^{\mu} = \frac{1}{2} G^{v\mu} \left(\frac{\partial G_{v\beta}}{\partial \gamma^{\alpha}} + \frac{\partial G_{v\alpha}}{\partial \gamma^{\beta}} - \frac{\partial G_{\alpha\beta}}{\partial x^{\mu}} \right) \tag{21}$$

where $G^{v\mu}$ is the inverse of $G_{v\mu}$. Typically, a geodesic path can be given by calculating a numerical integration of the ordinary differential equation defined in (20). But calculation of the Christoffel symbols is considerably expensive, because this process involves not only second derivatives of the g and but also an inverse of G. However, in practice, instead of getting the entire geodesic path, we only need to calculate out few discrete points along on the geodesic path. Therefore, directly starting from discrete geodesic energy (19) can avoid the abovementioned expensive calculations.

Formally, consider a discretized curve denoted by a series of coordinates $z_0, z_1, \ldots, z_N \in \mathcal{Z}$, it is approximating a continuous curve, $\gamma : [0,1] \to \mathcal{Z}$. Then with T time steps, a sequence of discrete time intervals of $\delta t = 1/N$ is generated, which corresponds to a discretized curve on the manifold $\mathcal{M}$ as $g(z_i)$. With a forward finite small shift, the velocity of the curve at $g(z_i)$ can be approximated as $v_i = \left(g\left(z_{i+1}\right) - g\left(z_i\right)\right)/\delta t$. Similarly, the energy of this curve can be given:

$$E_{z_i} = \frac{1}{2} \sum_{i=0}^{N} \frac{1}{\delta t} \left\| g\left(z_{i+1}\right) - g\left(z_i\right) \right\|^2 \tag{22}$$

Fixing the first and last points, z_0 and z_N, as the beginning and ending points of our geodesic path, minimizing this discrete geodesic energy function would result in a approximated geodesic path. To obtain this finalized curve, we will perform a gradient descent in the rest points, $z_1, \ldots, z_{N-1}$, along this curve. The gradient at z_i is computed as:

$$\nabla_{z_i} E = -\frac{1}{\delta t} J_g^T\left(z_i\right)\left(g\left(z_{i+1}\right) - 2g\left(z_i\right) + g\left(z_{i-1}\right)\right) \tag{23}$$

Therefore, by implementing a gradient descent algorithm, the calculating process of a discretized geodesic path can avoid the expensive calculations of Christoffel symbols. The detailed procedures is illustrated in Algorithm 1.

```
Algorithm 1: Geodesic Path Generation
  Input: Two coordinates, $z_0, z_N \in \mathcal{Z}$;
  learning rate $\alpha \in \mathbb{R}_+$
  Output: Discrete geodesic path, $z_0, z_1, \ldots, z_N \in \mathcal{Z}$
1 Initialize $z_i$ as linear interpolation between $z_0$ and $z_N$
  while $\sum_i \|\nabla_{z_i} E\|^2 > \epsilon$ do
2     for $i \in \{1, \ldots N-1\}$ do
3         Compute gradient $\nabla_{z_i} E$ using (23)
4         $z_i \leftarrow z_i - \alpha \nabla_{z_i} E$
5 return $z_0, z_1, \ldots, z_N$
```

G. Semantic Analysis

To make the deformation process of soft objects explainable, semantic analysis techniques are introduced to the high-level representation. This semantic analysis can be divided into three parts, namely semantic feature identification, latent deformation analysis, and latent shape planning. By comparing the geometric changes of the visualized inverse samples reconstructed from the low-dimensional latent variables, the effect on each shape dimension can be identified; Establishing a mapping from real soft object deformations to a path in latent shape space allows us to explore some useful knowledge behind the common deformations; In contrast to this forward mapping, an inverse mapping from latent shape space to the real shape deformation can help carry out a shape planning.

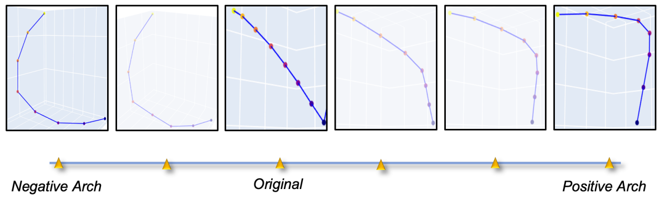


Fig. 4. The effect of changing a semantic feature that is identified to describe the arch degree of a shape on soft object deformation.

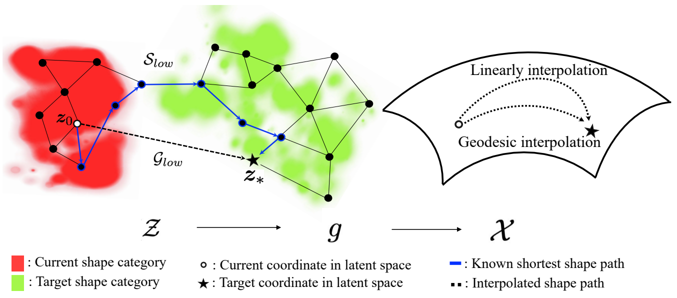
Semantic Feature Analysis: Given a compressed features $\boldsymbol{z}_0$ encoded by function h, we design Alg. 2 for a semantic feature analysis. In this algorithm, we gradually increase the p-th feature value with a short step δ for $\boldsymbol{z}_0$ to form a set of changed coordinates, $\mathcal{G}_{low}^{(p)}$, and then we need to update this set based on the whether generator g is not linear. At last, we reconstruct the inverse samples $\{\boldsymbol{x}_1', \boldsymbol{x}_2', \ldots, \boldsymbol{x}_n'\}$ for the soft object. The visualization of these inverse samples allows us to identify the semantic meanings for each dimension of the compressed feature in order to support our high-level semantic shape analysis (see Fig. 4 for an example of how semantic feature works). Note that the semantic functions on certain feature dimension will vary depending on the applied dimensionality technique and the distribution of data samples. Thus, visualization is a stable approach to accomplish this goal.

```
Algorithm 2: Semantic Feature Analysis
  Input: Shape vector $\boldsymbol{x}_0$, order $p$, step $\delta$, iteration $N$,
         encoder $h$, decoder $g$
  Output: Semantic deformation trace of $p$-th dim $\mathcal{D}_s^{(p)}$
1 Compute the coordinate $\boldsymbol{z}_0$ with $\boldsymbol{z}_0 = h(\boldsymbol{x}_0)$
2 $\mathcal{G}_{low}^{(p)} = \{\boldsymbol{z}_0, \boldsymbol{z}_1, \ldots, \boldsymbol{z}_N\}$ = Interpolation$(\boldsymbol{z}_0, p, \delta, N)$
3 if $g$ is not linear then
4     Update $\mathcal{G}_{low}^{(p)}$ with geodesic Alg. (1)
5 $\mathcal{G}_{high}^{(p)} = \{\boldsymbol{x}_1', \boldsymbol{x}_2', \ldots, \boldsymbol{x}_n'\} = g(\mathcal{G}_{low}^{(p)})$
6 $\mathcal{D}_s^{(p)} = $ Visualizer$(\mathcal{G}_{high}^{(p)})$
7 return $\mathcal{D}_s^{(p)}$
```

Semantic Deformation Analysis: This part examines the formation of a deformation path that serves as the basis of inverse shape planning. Intuitively, the raw shape data have certain correlations with the latent variables in latent shape space, and if the dimensionality reduction technique is invertible, then the shapes will share the same patterns and rules in the compressed feature space. Given that the deformation process of soft objects is continuous in real-world applications, its deformation process is also continuous in the latent feature space. Additionally, by constructing different shape network in the latent shape space, this continuous deformation path will travel through different spaces of shape classes, and manifests some rules of shape deformations in real-world applications.

Algorithm 3: Latent Shape Planning

```
Input: Current shape $\boldsymbol{x}_0$, target shape $\boldsymbol{x}_*$, iteration $N$,
       encoder $h$, decoder $g$
Output: Planned deformation trace $\mathcal{D}_p$
1 Compute the coordinates using $(\boldsymbol{z}_0, \boldsymbol{z}_*) = h(\boldsymbol{x}_0, \boldsymbol{x}_*)$
2 $\mathcal{S}_{low} = \{\boldsymbol{z}_0, \boldsymbol{z}_1, \ldots, \boldsymbol{z}_*\}$ = ShortestPath$(\boldsymbol{z}_0, \boldsymbol{z}_*)$
3 $\mathcal{S}_{high} = g(\mathcal{S}_{low})$
4 $\mathcal{G}_{low} = \{\boldsymbol{z}_0, \boldsymbol{z}'_1, \ldots, \boldsymbol{z}_*\}$ = Interpolation$(\boldsymbol{z}_0, \boldsymbol{z}_*, N)$
5 if $g$ is not linear then
6     Update $\mathcal{G}_{low}$ with geodesic Alg. (1)
7 $\mathcal{G}_{high} = g(\mathcal{G}_{low})$
8 $\mathcal{D}_p = \{\text{Visualizer}(\mathcal{S}_{high}), \text{Visualizer}(\mathcal{G}_{high})\}$
9 return $\mathcal{D}_p$
```



$\mathcal{S}_{low}$
z_0
$\mathcal{G}_{low}$
z_*
Linearly interpolation
Geodesic interpolation
$\mathcal{Z}$
g
$\mathcal{X}$
🟥 : Current shape category
🟩 : Target shape category
○ : Current coordinate in latent space
★ : Target coordinate in latent space
: Known shortest shape path
: Interpolated shape path

Fig. 5. Depiction of the deformation planning in latent shape space. According to Alg. 3, geodesic interpolated path is generated based on the results of linear interpolation in the latent space.

Latent Shape Planning: Latent shape planning presents a solution for an important scenario where the robotic manipulator attempts to manipulate soft objects into the target shape from the current shape. We design an algorithm to illustrate this process. Assume that the current shape is represented with shape vector $\boldsymbol{x}_0$, and the target shape is $\boldsymbol{x}_*$, after dimensionality transformation, the input shapes are transformed to a k-dimensional latent shape space ($\mathcal{Z} \subseteq \mathbb{R}^k$). With a encoder g, the encoded coordinates of $\boldsymbol{z}_0$ and $\boldsymbol{z}_*$ are readily known in this latent sspace. As Fig. 5 shows, shapes are represented as nodes in the latent space. In this space, these nodes are connected to form different neighbor networks rendered by different colors based on the kNN algorithm. With the implementation of shortest path searching algorithm in the latent shape space, the deformation path from the location of current shape to the location of target shape based on the known shape network can be achieved. Let $\mathcal{S}_{low}$ denote the shapes lying on the shortest path from $\boldsymbol{z}_0$ to $\boldsymbol{z}_*$ and let $\mathcal{S}_{high}$ denote the same shape vectors but with high dimensions reconstructed from $\mathcal{S}_{low}$. However, the shortest path algorithm can only find out a shortest path built on known shape data set in the shape space. This shape feature space should include a large quantity of shapes which are not yet collected and remain unseen to the dataset . Based on this idea, a straight line can be draw from $\tilde{\boldsymbol{x}}^\circ$ to $\tilde{\boldsymbol{x}}^*$, which may represent another shortest deformation path. To judge the rationality of this path, we must reconstruct and visualize these possible shapes. Accordingly, n intervals are set to generate $n+1$ intermediate shape statuses denoted by $\mathcal{G}_{low}$ and then update it according to geodesic path generation if the generator is not a linear transformation. At last, a shape set $\mathcal{G}_{high}$ comprising transitional deformation is formed. Finally, these two deformation paths pass through a visualizer and output the deformation set $\mathcal{D}_p$.

IV. Results

In this section, we describe the experiment setup shown in Fig. 6 to collect soft object data in order to build LaSeSOM. During this process, the soft object is only manipulated by human hands. After building LaSeSOM, this framework is used to present different levels of representation results in a robotic teleoperated manipulation task on soft object via Leap Motion [45] demonstration.

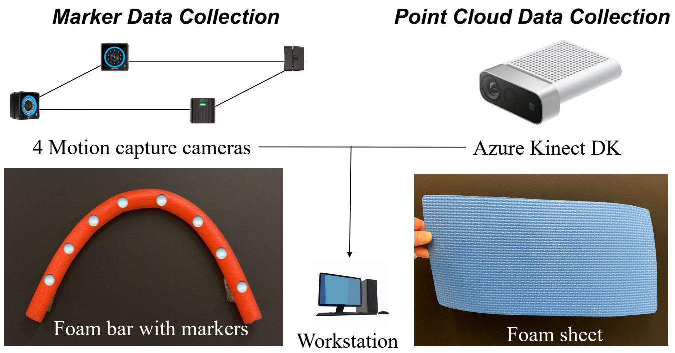


Fig. 6. Experimental setups of the shape data collection to build LeSeSOM. Left shows the setup to collect ordered marker data for a foam bar in a motion capture system, while right is used to collect unordered point cloud data for a foam sheet.

A. Data Collection

As shown in Fig. 6, two different soft objects (a foam bar and a foam sheet) were used to collect deformed shapes. For the foam bar, the Prime 13 motion tracking system was used to track the position of each marker mounted on the its surface. Specifically, four synchronized cameras were installed around the foam bar which were evenly labeled with eight markers. The 3D positions of these markers were calculated once the overlapping position data was transmitted to workstation connected with all cameras through a hub. During the collection, 30 FPS was set to conduct the manipulation. Whereas, the deformations of the foam sheet were captured with a same 30 FPS in a format of point clouds by an RGB-D camera (Azure Kinect DK) However, some issues need to be solved with the raw collected data. For marker data, due to limitations of the capture system and environmental influences, the system could generate a percentage (around 8%) of shape frames with more than or less than eight marker points. Give the small percentage and high cost to correct them, we simply removed them. Another issue for marker data is the consistence of the order of the markers for every shape frame during the deformation. It was solved by rearranging the orders based on the computation of the distance during neighboring frames.

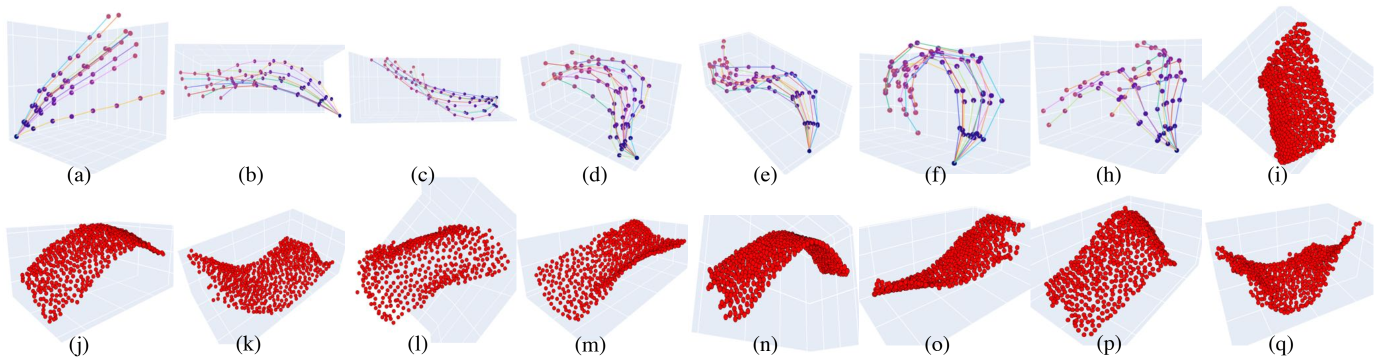


Fig. 7. Visualizations of shape samples of predefined categories. Figures (a) to (h) shows the seven classes for the foam bar deformation, and figures (i) to (q) presents the nine classes for the foam sheet deformation.

Afterward, all eight marker position coordinates were subtracted by the first marker coordinate to ensure that all shape frames start from the same position known as the original point when visualizing the shape. However, for the point cloud data of the foam sheet, the similar manipulation is not allowed because point cloud is totally unordered data. But in order to unify the input dimension for AE net, farthest point sampling (FPS [37]) was used to ensure that all the point clouds have a same number of points (512 points). Finally, data collection process outputted two datasets as Tab. I summarized. Based on the basic statistics on the shape of two manipulated objects, shape frames were divided into different categories as illustrated in the table. Fig. 7 displayed few samples for each corresponding categories. Note that the positive and negative categories would be combined or separated based on different analytical needs.

TABLE I
DATA SUMMARY

Category	Set1	Set2
Line	857	57
Arch Pos.	1038	825
Arch Neg.	1339	0
S Pos.	1570	200
S Neg.	1482	100
Helix Pos.	1005	110
Helix Neg.	957	0
Total	8248	1292

(a) Marker dataset

Category	Set
Plane	250
Blend #1 Pos.	250
Blend #1 Neg.	250
Blend #2 Pos.	250
Blend #2 Neg.	250
Fold #1 Pos.	250
Fold #1 Neg.	250
Fold #2 Pos.	250
Fold #2 Neg.	250
Total	8248

(b) Point cloud dataset

B. Representation Results

1) Low-level: To examine the fitting performance, the coefficient of determination R^2 [46], which is defined as Equ. (24), is used to quantify the amount of variability explained by Fourier approximation. As shown in Fig. 9(a), the shape descriptor becomes more accurate along with the increasing number of harmonics. For example, the line class shape starts from 0.7654 when the number of harmonics is set to 1 and ends up with 0.9942 when the model has 6 harmonics. Specifically, the line and arch class shapes demonstrate better performance than the other class shapes under the same number of harmonics. The main reason is that the S-shaped and helix class shapes are too complex to represent by using the Fourier descriptor. However, a Fourier descriptor with more harmonics entails a higher computing cost. Therefore, a suitable number of harmonics should be identified to balance the shape accuracy with the computing cost. In the following experiment, the number of harmonics is always set to 5.

$$R^2 \equiv 1 - \frac{\sum_i (y_i - f_i)}{\sum_i (y_i - \bar{y})^2} \tag{24}$$

2) Mid-level: After performing PCA on Fourier coefficients of marker data from low-level, a suitable number of components k is selected according the explained variance, and the explained variance exceeds 95% when k equals 3 and 4, respectively. Based on this observations, $k = 4$ is chosen to investigate following semantic analysis. In the designed semantic analysis algorithm, parameters are set with iteration $T = 10$, $k = 4$, and $t = 1$ and Fig. 8 (a) to 8 (d) visually presents the individual semantic effect of the four features. The reconstructed shapes are represented with a series of markers, which are linked with splines in different colors and the blue line shows the original shape. To summarize, the first component tries to maintain the same shape and alter the angle as the feature value increases, whereas the second component tries to describe the arch shape. The third component is trend to depict the degree of "S" shape, whereas the fourth component tries to capture helix shape. Though sometimes the results shows partially combined semantic effect (not single effect), each feature dimension has a dominant semantic effect, respectively.

TABLE II
NETWORK ARCHITECTURE

Marker data (Form Bar)
Input 8×3
Flatten
FC 8, BatchNorm, ReLU
FC 4, BatchNorm, ReLU
FC 8, BatchNorm, Sigmoid
FC 24, Sigmoid
Reshape 8×3

Point cloud (Form Sheet)
Input 512×3
3×1 conv, 8, BatchNorm, ReLU
8×1 conv, 32, BatchNorm, ReLU
32×1 conv, 64, BatchNorm, ReLU
Max pool
FC 256, Batch norm, Sigmoid
FC 512, Batch norm, Sigmoid
FC 1536, Sigmoid
Reshape 512×3

To compare with the effect of PCA, AE is also performed on the marker data with an implementation by using *Pytorch*.

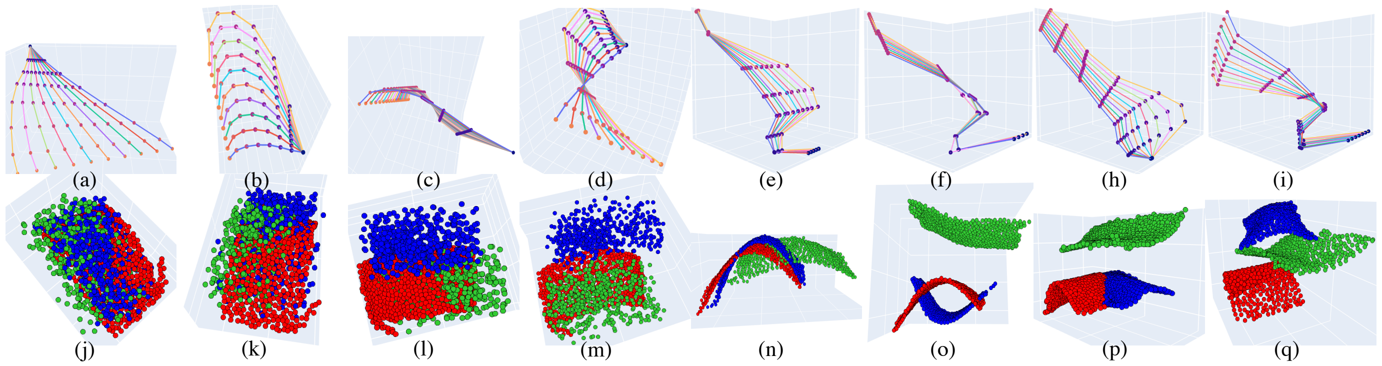


Fig. 8. Visual comparison of the semantic features from different dimensionality transformation techniques, where figures (a) to (d) and (e) to (i) respectively shows the results of the foam bar from PCA and AE. Figures (i) to (q) shows the visualization results of eight (total in *64-dim*) semantic features.

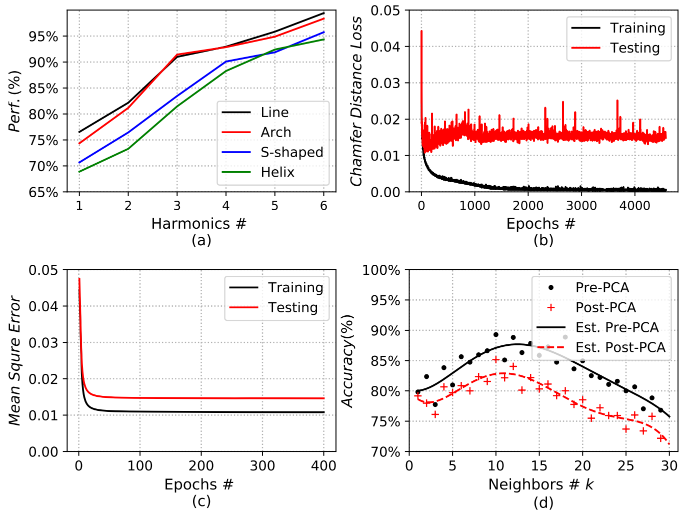


Fig. 9. (a) The performance of Fourier approximation for four main shape classes of the foam bar by different harmonics; (b) and (c) respectively show the training and validation errors for the corresponding soft objects; (d) presents the pre-PCA and post-PCA classification accuracy for the foam bar.

The corresponding architecture is shown in Tab. II. The architecture for the generator ($g : \mathcal{Z} \to \mathcal{X}$) is reverse of the encoder architecture with two fully connected layers that outputs reshaped marker coordinates. The latent dimension is kept at 4 for marker's dataset. By performing the similar semantic feature analysis on this latent dimensions, Figs. 8 (e) to 8 (i) visually presents the individual semantic effect of the four dimensions for code layer of AE. Unlike the result of PCA, these four dimensions mainly depict "S" shapes from different perspectives. This is due to the four neural units in the code layer receive a linear combination from all input data and the S-shaped category accounts for the majority of the training dataset. During the training process, the epoch is set to 400, the batch size is set to 100, and the learning rate is set to 0.0002, taking with *Adam* as the model optimizer and *MSE* as the loss function. 20% of the samples are separated to test validation performance. Fig. 9 (c) shows that the model converges quickly on both training data and validation data.

As for point cloud data of the foam sheet, the latent dimension is kept at 64 with the network architecture as shown in Tab. I. During the training, the epoch for the model is set to 5000 with *Adam* as the optimizer, but the loss function is set to Chamfer distance specified for point sets. The learning rate is set to 0.0005 and Fig. 9 (b) shows the corresponding loss trend for training and testing. With the same implementation of semantic feature analysis, Figs. 9 (j) to (q) shows the eight reconstructed results out of total the *64-dim* code layer. The red points represent the raw shape and the blue and green one shows the results of increasing and decreasing feature value, respectively. The former four mainly describe translation of the sheet, whereas the latter four capture the degree of curvature for the foam sheet.

To define the best number k for kNN, *test_size* is set to 0.4, and a *5-fold* cross-validation is applied to show a stable precision for each number of neighbors k. As Fig. 9 (d) shows, both pre- and post-PCA kNN models are built with k iterating from 1 to 30. Both of these models share a similar trend and reach a peak under the same k, thereby validating the optimal number of neighbors. Most importantly, a tiny loss in precision and a huge reduction in dimensions is especially meaningful for processing large-scale soft object shape data, such as point cloud data.

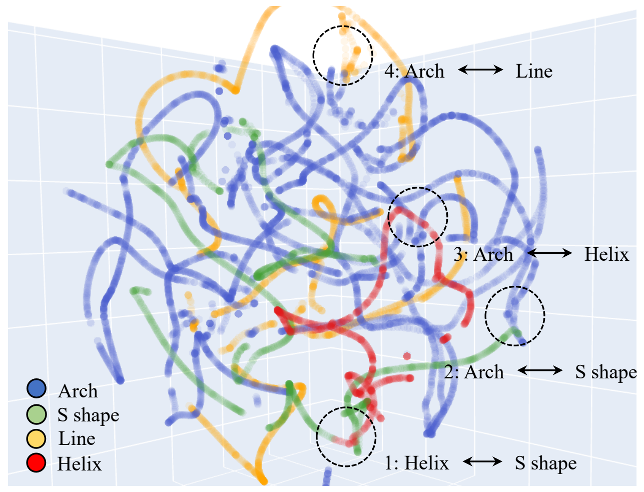


Fig. 10. Visualization of shape deformation by using t-SNE in a 3D shape space, where each shape frame is represented as a node in the space, and is colored according to the different shape labels predicted by the kNN classifier. (b) shows the linking result of shape reachness relations based on t-SNE.

3) High-level: Since it's hard to visualize a high dimensional shape data space, here only a 3D shape space for foam bar is visualized through an implementation of *t-SNE* as shown in Fig. 10. The *random_state* is fixed to ensure a stable beginning status. The *perplexity* is set to its default value. In this chart, each point represents a shape of the manipulated foam bar and simultaneously it is colored according to different shape categories. Due to the deformation process is continuous, therefore the distributed points accordingly form a continuous curve in this shape space. Specifically, the linkings between different categories denotes that a shape is changing between different shape categories in the corresponding space. By collecting all the connections in the dashed circle, the transformation relation of deforming shapes can be discovered. For example, arch, S-shaped and helix shape categories can be transformed. However, the line shape category needs to be transformed into an arch shape before reaching a s-shape or helix shape. All of the above findings are aligned with our knowledge for sponge bar manipulation.

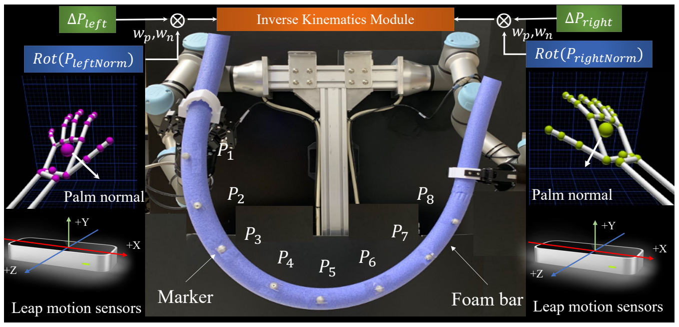


Fig. 11. Architecture of the teleoperated robotic soft object manipulation task with Leap Motion sensors for the validation of LaSeSOM.

C. Latent Shape Planning

To validate the effectiveness of LaSeSOM in a soft object manipulation task, we presented a real-time teleoperated manipulation of a foam bar via Leap Motion [47] demonstration and the corresponding experimental setup is shown in Fig. 11. Two Leap Motion sensors were used to capture the hand gestures due to that the interaction space provided by a single sensor is very limited. With the Leap Motion *SDK*, we can compute the displacement of palm position δp and the rotation of palm normal $Rot(\mathbf{n}_p)$ between the previous and current frames captured by sensors. Multiplied by suitable weights w_p and w_n, δp and $Rot(\mathbf{n}_p)$ are adjusted to appropriately map from the palm space to the robotic working space. Subsequently, two 6-DOF robotic manipulators (UR3) received the weighted δp and $Rot(\mathbf{n}_p)$ to compute the poses with the inverse kinematics module. The Table I shows the summary of shape frames collected from this manipulation task.

1) Semantic Deformation: As Fig. 12 shows, all the shape frames of this manipulation task for the foam bar are encoded into a latent shape space built from AE on the Fourier coefficients. In this space, the resulting deformation path of the manipulation is represented as a red curve. Different shape categories of dataset #1 were organized with *mesh3D* from *Plotly*, and then rendered with different colors according to the predicted labels from kNN. The beginning shape located at the position of the triangle marker, and then the foam bar started from the line category area denoted by a blue color. As the shape deformed, the current point moved continuously toward the positive arch category denoted by the yellow color in area #1, and then moved to the negative S-shaped category denoted

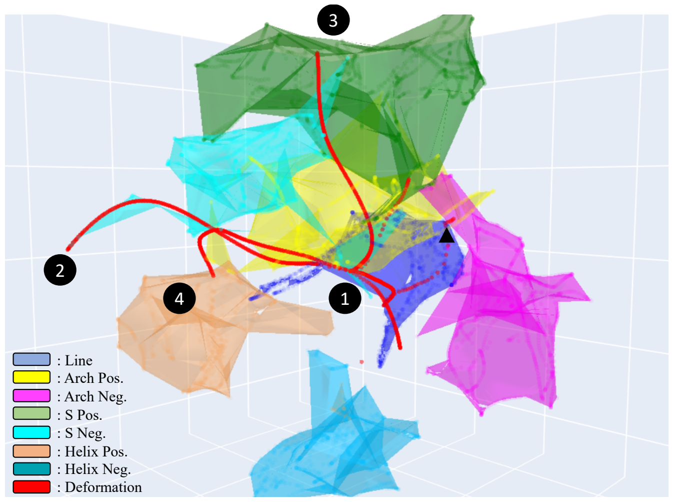


Fig. 12. 3D visual deformation trace during the soft object manipulation task in a compressed shape space, where the red trace travels from the beginning line category to different shape categories synchronized with the manipulation task.

by the cyan color in area #2. Subsequently, the foam bar went back to the positive arch shape from area #2 which form a identical but inverse path. And so forth, the deformed foam bar then traveled to a positive S-shaped category, a positive helix category, and ended up with its original shape state. Therefore, the entire trace semantically reflects the entire process of shape deformation in a latent space when manipulating a soft object and shows a possibility of guiding a manipulator to do shape planning.

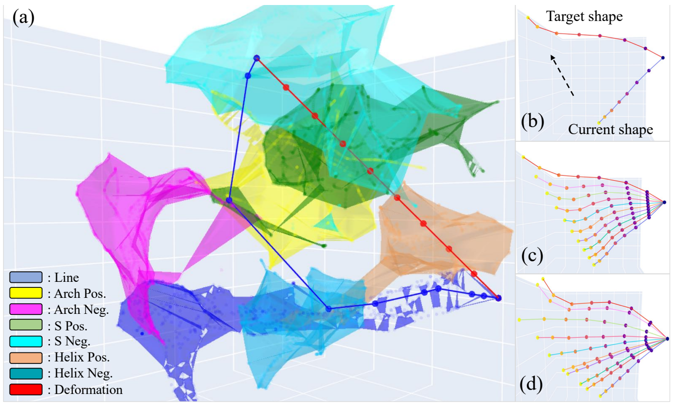


Fig. 13. Visualization of the process of latent shape planning for the foam bar. Figure (b) shows the beginning shape x_0 and the target shape x_*, and figures (c) and (d) present the shape deformations $\mathcal{G}_{high}$ and $\mathcal{S}_{high}$, which are generated from the shortest path searching on the collected shape set and the geodesic path-based interpolation on data manifold, respectively. (a) presents their corresponding deformation pathes in a 3D shape space.

2) Shapes on Manifolds: We use Algorithm 3 to perform a shape planning through a generator ($g : \mathcal{Z} \rightarrow \mathcal{X}$), to map paths calculated in the latent space into shapes on the generated manifold ($\mathcal{M}$). Fig. 13(b) shows a beginning shape $\boldsymbol{x}_0$ and target shape $\boldsymbol{x}_*$ of a foam bar. With the encoder h (illustrated in Table II), we can get encoded shapes in $\mathcal{Z}$ space, which are respectively represented as $\boldsymbol{z}_0$ and $\boldsymbol{z}_*$. Then, two sets of shapes are generated based on different calculations in $\mathcal{Z}$ space. One shape set $\mathcal{S}_{low}$ is calculated by performing an shortest path search algorithm based on collected data, which is denoted by the blue spline

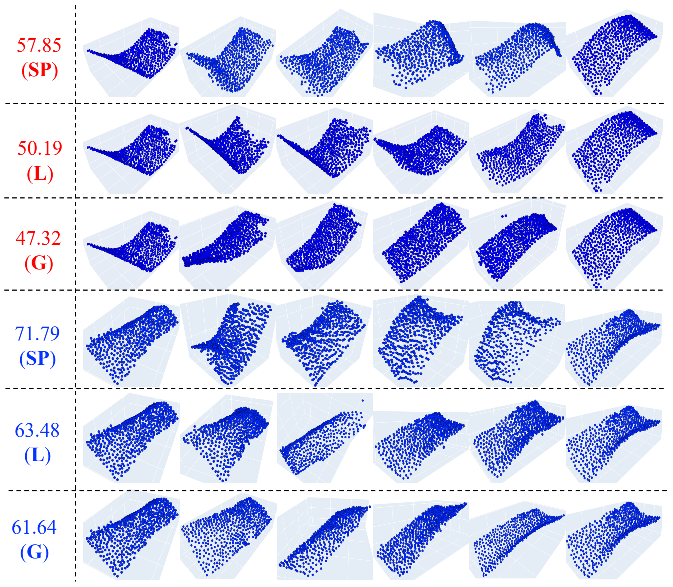


Fig. 14. Shape planning results of shortest path, linear interpolation, and geodesic interpolation for foam sheet dataset. Column 1: arc length; Rows 1, 4: shortest path; Rows 2, 5: linear; Rows 3, 6: geodesic.

in the latent space. This shortest path can be represented as a series of indexes of shape frame in the dataset, $\{x_{540}, x_{532}, x_{530}, x_{526}, x_{568}, x_{777}, x_{774}, x_{1929}, x_{5812}, x_{5040}\}$. With decoder g, $\mathcal{S}_{low}$ can be mapped to the manifold of shape data, denoted by $\mathcal{S}_{high}$ in the shape data space $\mathcal{X}$. Another shape set $\mathcal{G}_{high}$ is generated by a linear interpolation between z_0 and z_* at first, and then an iterative updating on each coordinate with geodesic path illustrated in Alg. 1. This initial linear interpolated path is marked by a red spline in the figure. Figs. 13 (c) and (d) show the resulting deformation processes from a geodesic interpolation and shortest path, respectively. We can clearly observed that the deformation process with geodesic path interpolation is smoother compared with the one generated by a shortest path. Fig. 14 shows a few point clouds on the shortest path, linear interpolation, and geodesic interpolation curves along with their arc lengths for the foam sheet. The shortest path gives the longest arc length compared with the other two methods, because the collected dataset could not collect all the equally changed shape on deformations. Although, the geodesic curve on the manifold presents a shorter arc length compared with linear interpolation, their difference is not significant, which indicates that the manifold generated by generator architecture for form sheet has little curvature, even non-linear. For more experimental results, please refer to the video via: https://sites.google.com/view/lasesom.

V. Conclusions

In this paper, we present a generic data-driven representation framework for soft objects in bimanual manipulation tasks. This 3-level framework can be divided into 4 layers. Layer 0 forms low-level shape features for soft objects, and layers 1 and 2 mainly deals with dimensionality transformations to produce a semantic internal representation for soft objects. The last layer detects deformation knowledge in a latent space and solves shape planning with geodesic path generation on the data manifold.

The experimental results show the viability of using this framework for different levels of representation. Specifically, the low-level layer can extract valid features from both ordered data and unordered data. In mid-level layers, semantic techniques can identify the semantic meaning for latent variables; High-level layers can extract the accurate deforming relations between different shape categories; In addition, a geodesic path-based interpolation algorithm has validate the effectiveness for soveling a shape planning problem.

The proposed framework has several limitations. For example, it is still hard to clearly explain the semantic meaning for several dimensions in AE code layer with currently designed semantic analysis algorithms. Besides, a data-driven framework mainly depends on an elaborate data collection procedure. However, when the environment of manipulation tasks has been changed, this framework needs some time to fit the models so as to form a valid representation again.

As future research, the implementation of a manipulator with LaSeSOM based feedback control for 3D soft objects will be carried out. Currently, we are also working on the development of shape deformation planning with constraints (obstacles, occluders, etc.). Another extension of this paper is to explore some learning paradigms on soft object manipulation, such as imitation learning from a real-time human action or video recordings.

References

[1] H. B. Amor, A. Saxena, *et al.*, “Special issue on autonomous grasping and manipulation,” *Auton. Robots*, vol. 36, no. 1-2, pp. 1–3, 2014.

[2] S. Tokumoto and S. Hirai, “Deformation control of rheological food dough using a forming process model,” in *Proc. IEEE Int. Conf. Robot Autom.*, vol. 2, 2002, pp. 1457–1464.

[3] M. Cusumano-Towner, A. Singh, S. Miller, J. F. O’Brien, and P. Abbeel, “Bringing clothing into desired configurations with limited perception,” in *IEEE Int. Conf. Robot Autom.*, 2011, pp. 3893–3900.

[4] E. J. Park and J. K. Mills, “Static shape and vibration control of flexible payloads with applications to robotic assembly,” *IEEE/ASME Trans. Mechatronics*, vol. 10, no. 6, pp. 675–687, 2005.

[5] H. Wakamatsu *et al.*, “Knotting/unknotting manipulation of deformable linear objects,” *Int. J Robot Res.*, vol. 25, no. 4, pp. 371–395, 2006.

[6] P. Culmer, J. Barrie, R. Hewson, M. Levesley, M. Mon-Williams, D. Jayne, and A. Neville, “Reviewing the technological challenges associated with the development of a laparoscopic palpation device,” *Int. J. Med. Robot Comp.*, vol. 8, no. 2, pp. 146–159, 2012.

[7] J. Sanchez, J.-A. Corrales, *et al.*, “Robotic manipulation and sensing of deformable objects in domestic and industrial applications: a survey,” *Int. J. Robot. Res.*, vol. 37, no. 7, pp. 688–716, 2018.

[8] S. Hirai and T. Wada, “Indirect simultaneous positioning of deformable objects with multi-pinching fingers based on an uncertain model,” *Robotica*, vol. 18, no. 1, pp. 3–11, Jan. 2000.

[9] D. Navarro-Alarcon, Y.-H. Liu, J. G. Romero, and P. Li, “Model-free visually servoed deformation control of elastic objects by robot manipulators,” *IEEE Trans. Robot.*, vol. 29, no. 6, pp. 1457–1468, 2013.

[10] D. Navarro-Alarcon, Y.-h. Liu, *et al.*, “On the visual deformation servoing of compliant objects: Uncalibrated control methods and experiments,” *Int. J. Robot. Res.*, vol. 33, no. 11, pp. 1462–1480, 2014.

[11] D. Navarro-Alarcon, H. M. Yip, *et al.*, “Automatic 3-d manipulation of soft objects by robotic arms with an adaptive deformation model,” *IEEE Trans. Robot.*, vol. 32, no. 2, pp. 429–441, 2016.

[12] M. Laranjeira, C. Dune, and V. Hugel, “Catenary-based visual servoing for tether shape control between underwater vehicles,” *Ocean Engineering*, vol. 200, pp. 1–19, 2020.

[13] D. Navarro-Alarcon and Y.-H. Liu, “Fourier-based shape servoing: A new feedback method to actively deform soft objects into desired 2D image shapes,” *IEEE Trans. Robot.*, vol. 34, no. 1, pp. 272–1279, 2018.

[14] J. Zhu, B. Navarro, P. Fraisse, A. Crosnier, and A. Cherubini, "Dual-arm robotic manipulation of flexible cables," in *IEEE/RSJ Int. Conf. on Robots and Intelligent Systems*, 2018, pp. 479–484.
[15] Z. Hu, P. Sun, and J. Pan, "Three-dimensional deformable object manipulation using fast online gaussian process regression," *IEEE Robot. Autom. Lett.*, vol. 3, no. 2, pp. 979–986, 2018.
[16] C. Collewet and E. Marchand, "Photometric visual servoing," *IEEE Trans. Robot.*, vol. 27, no. 4, pp. 828–834, 2011.
[17] E. Marchand, "Subspace-based direct visual servoing," *IEEE Robot. Autom. Lett.*, vol. 4, no. 3, pp. 2699–2706, 2019.
[18] K. M. Digumarti, B. Trimmer, A. T. Conn, and J. Rossiter, "Quantifying dynamic shapes in soft morphologies," *Soft Robot.*, 2019.
[19] J. Zhu, D. Navarro-Alarcon, R. Passama, and A. Cherubini, "Vision-based manipulation of deformable and rigid objects using subspace projections of 2d contours," 2020.
[20] K. Xu, V. G. Kim, *et al.*, "Data-driven shape analysis and processing," in *SIGGRAPH ASIA 2016 Courses*, 2016, pp. 1–38.
[21] H. Zhang, A. Sheffer, D. Cohen-Or, Q. Zhou, O. Van Kaick, and A. Tagliasacchi, "Deformation-driven shape correspondence," in *Computer Graphics Forum*, vol. 27, no. 5, 2008, pp. 1431–1439.
[22] A. Golovinskiy and T. Funkhouser, "Consistent segmentation of 3d models," *Computers Graphics*, vol. 33, no. 3, pp. 262–269, 2009.
[23] O. Sidi, O. van Kaick, Y. Kleiman, H. Zhang, and D. Cohen-Or, "Unsupervised co-segmentation of a set of shapes via descriptor-space spectral clustering," in *Proc. SIGGRAPH Asia Conf.*, 2011, pp. 1–10.
[24] O. Van Kaick, K. Xu, H. Zhang, Y. Wang, S. Sun, A. Shamir, and D. Cohen-Or, "Co-hierarchical analysis of shape structures," *ACM Trans. Graphic.*, vol. 32, no. 4, pp. 1–10, 2013.
[25] L. Tao, L. Yuan, and J. Sun, "Skyfinder: attribute-based sky image search," *ACM Trans. Graphic.*, vol. 28, no. 3, pp. 1–5, 2009.
[26] D. Parikh and K. Grauman, "Relative attributes," in *Int. Conf. Comput. Vis.*, 2011, pp. 503–510.
[27] P.-Y. Laffont, Z. Ren, X. Tao, C. Qian, and J. Hays, "Transient attributes for high-level understanding and editing of outdoor scenes," *ACM Trans. Graph.*, vol. 33, no. 4, pp. 1–11, 2014.
[28] G. Leifman, R. Meir, and A. Tal, "Semantic-oriented 3d shape retrieval using relevance feedback," *Visual Comput.*, vol. 21, no. 8-10, pp. 865–875, 2005.
[29] M. Attene *et al.*, "Characterization of 3d shape parts for semantic annotation," *Comput. Aided Des.*, vol. 41, no. 10, pp. 756–763, 2009.
[30] P. D. Hoff, A. E. Raftery, and M. S. Handcock, "Latent space approaches to social network analysis," *J. Am. Stat. Assoc*, vol. 97, no. 460, pp. 1090–1098, 2002.
[31] P. Achlioptas, O. Diamanti, I. Mitliagkas, and L. Guibas, "Learning representations and generative models for 3d point clouds," in *Int. conf. Machine Learning*. PMLR, 2018, pp. 40–49.
[32] G. Arvanitidis *et al.*, "Latent space oddity: On the curvature of deep generative models," in *Int. Conf. Learn Represent.*, 2018.
[33] Y. Zhong, "Intrinsic shape signatures: A shape descriptor for 3d object recognition," in *IEEE Int. Conf. Comput. Vis.*, 2009, pp. 689–696.
[34] F. Tombari, S. Salti, and L. Di Stefano, "A combined texture-shape descriptor for enhanced 3d feature matching," in *18th IEEE Int. Conf. Image Process.* IEEE, 2011, pp. 809–812.
[35] J. Xie, Y. Fang, F. Zhu, and E. Wong, "Deepshape: Deep learned shape descriptor for 3d shape matching and retrieval," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 1275–1283.
[36] D. Zhang, G. Lu, *et al.*, "A comparative study of fourier descriptors for shape representation and retrieval," in *Proc. 5th Asian Conf. Comput. Vis.*, 2002, p. 35.
[37] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," in *Adv. Neural Inf. Process Syst.*, 2017, pp. 5099–5108.
[38] C. R. Qi, H. Su, , *et al.*, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 652–660.
[39] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," *Chemometr. Intell. Lab.*, vol. 2, no. 1-3, pp. 37–52, 1987.
[40] M. E. Wall, A. Rechtsteiner, and L. M. Rocha, "Singular value decomposition and principal component analysis," in *A practical approach to microarray data analysis*, 2003, pp. 91–109.
[41] G. E. Hinton *et al.*, "Reducing the dimensionality of data with neural networks," *science*, vol. 313, no. 5786, pp. 504–507, 2006.
[42] G. Guo, H. Wang, *et al.*, "Knn model-based approach in classification," in *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"*, 2003, pp. 986–996.
[43] Y. Wang, H. Yao, and S. Zhao, "Auto-encoder based dimensionality reduction," *Neurocomputing*, vol. 184, pp. 232–242, 2016.
[44] Y. Pu, Z. Gan, R. Henao, X. Yuan, C. Li, A. Stevens, and L. Carin, "Variational autoencoder for deep learning of images, labels and captions," in *Adv. Neural Inf. Process Syst.*, 2016, pp. 2352–2360.
[45] I. Jang, J. Carrasco, A. Weightman, and B. Lennox, "Intuitive bare-hand teleoperation of a robotic manipulator using virtual reality and leap motion," in *Annu. Conf. Auton. Robot. Syst.*, 2019, pp. 283–294.
[46] N. J. Nagelkerke *et al.*, "A note on a general definition of the coefficient of determination," *Biometrika*, vol. 78, no. 3, pp. 691–692, 1991.
[47] L. E. Potter, J. Araullo, and L. Carter, "The leap motion controller: a view on sign language," in *Proc. 25th Au. Conf. Comput. Hum. Interact.*, 2013, pp. 175–178.

Peng Zhou (S'20) received the M.Sc. degree in software engineering from Tongji University, Shanghai, China, in 2017. Since 2019, he is pursuing his the Ph.D. degree in robotics in The Hong Kong Polytechnic University, Kowloon, Hong Kong. His research interests include data-driven robot manipulation, machine learning, and robot learning.

Jihong Zhu received the M.Sc. in Systems and Control in 2015 from TU Delft and the Ph.D. in Robotics from University of Montpellier in 2020. He conducted his doctoral research at LIRMM. In 2019, he was a visting PhD student at The Hong Kong Polytechnic University. He is currently a postdoc at Cognitive Robotics department, TU Delft. His research interests include: sensor-based control, and manipulation of soft objects.

Shengzeng Huo received his B.Sc. degree in vehicle engineering from the South China University of Technology, China in 2019 and is currently pursuing his the M.Sc. degree in mechanical engineering in The Hong Kong Polytechnic University, Kowloon, Hong Kong. His research interests include automation, data-driven visual servoing and robot learning.

David Navarro-Alarcon (GS'06–M'14–SM'19) received the Ph.D. degree in mechanical and automation engineering from The Chinese University of Hong Kong (CUHK), NT, Hong Kong, in 2014. From 2015 to 2017, he was an Assistant (Research) Professor at the CUHK T Stone Robotics Institute. Since 2017, he has been with The Hong Kong Polytechnic University, KLN, Hong Kong, where he is an Assistant Professor at the Department of Mechanical Engineering and the Principal Investigator of the Robotics and Machine Intelligence Laboratory. His current research interests include perceptual robotics and control theory.