LaSeSOM
A Latent and Semantic Representation Framework for Soft Object Manipulation

Zhou, Peng; Zhu, Jihong; Huo, Shengzeng; Navarro-Alarcon, David

DOI
10.1109/LRA.2021.3074872

Publication date
2021

Document Version
Accepted author manuscript

Published in
IEEE Robotics and Automation Letters

LaSeSOM: A Latent and Semantic Representation Framework for Soft Object Manipulation

Peng Zhou, *Student Member, IEEE*, Jihong Zhu, *Member, IEEE*, Shengzeng Huo, *Student Member, IEEE*, 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*—Bimanual Manipulation; Representation Learning; Shape Deformation Planning; Latent Space and Manifolds; Geodesic Interpolation.**

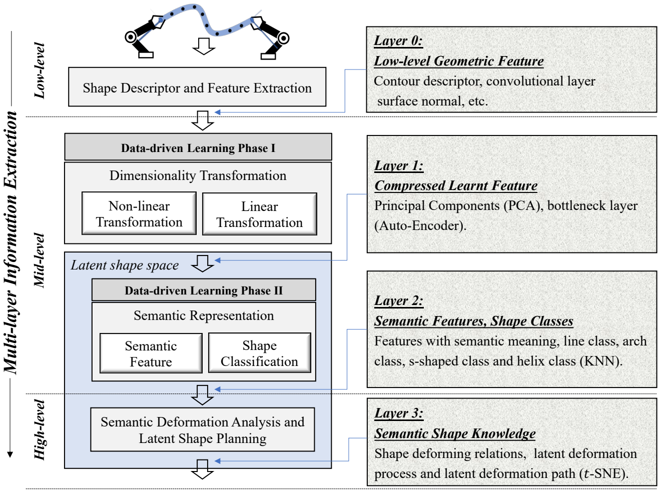
Low-level
Mid-level
High-level
Multi-layer Information Extraction
Shape Descriptor and Feature Extraction
Data-driven Learning Phase I
Dimensionality Transformation
Non-linear Transformation
Linear Transformation
Latent shape space
Data-driven Learning Phase II
Semantic Representation
Semantic Feature
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.

I. Introduction

RECENT studies have shown that the manipulation of soft objects is crucial and indispensable to achieve high autonomy in robots [1]. 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 [2] 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 [3] (simple, but cannot generalize). Other classical methods are based on geometric features e.g. angles, curvatures, catenaries [4], [5]; Its disadvantage is that they are case-specific, thus, can only be used to perform a single shaping action. Some works have addressed this issue by developing generic representations that only require sensory data. For example, [6], [7], and [8] 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 [9], [10]) and combine them with dimension reduction techniques, as in e.g. [11], [12]. Data-driven based shape analyses [13], [14] 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 [15], [16], and [17]). 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 [18], relative attributes [19], and

Manuscript received December 24, 2020; Revised March 21, 2021; Accepted April 13, 2021. This paper was recommended for publication by Editor Hong Liu upon evaluation of the Associate Editor and Reviewers' comments. This work is supported by the Research Grants Council under Grant 14203917, in part by the PROCORE-France/Hong Kong Joint Research Scheme under Grant F-PolyU503/18, in part by the Key-Area Research and Development Program of Guangdong Province 2020 under project 76, in part by the Jiangsu Industrial Technology Research Institute Collaborative Research Program Scheme under Grant ZG9V, and in part by PolyU under Grants 252047/18E, ZZHJ, and UAKU. *(Corresponding author: David Navarro-Alarcon.)*

P. Zhou, S. Huo and D. Navarro-Alarcon are with The Hong Kong Polytechnic University, KLN, Hong Kong (e-mail: jeffery.zhou@connect.polyu.hk; kyle-sz.huo@connect.polyu.hk; dna@ieee.org)

J. Zhu is with Delft University of Technology, Mekelweg 2, 2628CD, The Netherlands. (e-mail: j.zhu-3@tudelft.nl)

Digital Object Identifier (DOI): see top of this page.

semantic image color palette editing [20]. Several works [21], [22] further combine shape analysis and semantic attributes for a in-depth deformation analysis.

Latent space approaches have recently achieved many successful results in image analysis [23], 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 [24] 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 [25] is formulated by:

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

for $\mathbf{J}_{\mathbf{z_0}} = \frac{\partial g}{\partial \mathbf{z}}\Big|_{\mathbf{z}=\mathbf{z_0}}$ and $\Delta_{12} = \delta_1 - \delta_2$, which indicates that the normal distance in $\mathcal{Z}$ space changes locally as it is determined by the local Jacobian. Consequently, seeking the shortest curve along a curved surface, a manifold, manifold is a more reasonable way to compute the interpolation and generate undistorted samples.

As a feasible solution to these problems, we present a general data-driven representation framework — **LaSeSOM** for semantic soft object manipulation depicted in Fig. 1, which 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. 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 presents the representation models. Section III shows the experimental results. Section IV gives final conclusions.

II. Methods

In **LaSeSOM**, we first introduce two shape features extracted from two data formats for shape description (marker points and point clouds), and then two dimensionality transformation techniques for building latent space. With this latent space, we design several semantic analysis algorithms to describe soft object deformations and 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, Let $\mathcal{S} = \left\{\mathcal{S}_1, \ldots, \mathcal{S}_p \mid \mathcal{S}_i \in \mathbb{R}^{q\times 3}\right\}$ be set of a complete soft object deformation, and $\mathcal{S}_i$ denotes the i-th shape during the deformation process. Using q marker points, $\mathcal{S}_i = \left\{\mathbf{x_1}, \ldots, \mathbf{x_q} \mid \mathbf{x_i} \in \mathbb{R}^{\mathbf{3}}\right\}$ can be determined by an ordered 3D points set. Consequently, the entire deformation can represented as a shape matrix $\mathbf{X}_{in} \in \mathbb{R}^{p\times 3q}$, where the coordinates of the markers have been fatten so each row with q markers has $3q$ features and the number of total shapes during this deformation is denoted by p. To approximate the contour composed of 3D marker points, Fourier approximation [26] is selected considering that this descriptor can depict the shape with arbitrary precision. However, instead of using its common 2D modeling form, we expand this descriptor into a 3D configuration as below:

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

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. The coefficients of the n-th harmonic are denoted by $a_n, b_n, \ldots, f_n$, which can be solved with expressions in [26] to constitute the description of the shape.

A deformable shape $\mathcal{S}_i$ can also be represented as a point cloud data $\mathcal{P}_i$. With farthest point sampling algorithm used in PointNet++ [27], 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 [28]: increasing the features with convolutional 1D layers (thus, 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:

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

B. Dimensionality Transformation

To seek optimal and concise features for shape representations, two typical techniques are used to embed shape features in a latent space. First, Principal components analysis

(PCA) [29] is selected to provide a sequence of optimal linear transformations for high-dimensional ordered shape 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 $\boldsymbol{Y} = \boldsymbol{PX}$. One efficient solution for the PCA problem is known as the singular value decomposition (SVD) [30]. Since semantic analysis of **LaSeSOM** needs the reconstructed shapes from the low-dimensional latent variables. For this reason, the inverse sample, $\boldsymbol{X}_{rec}$ reconstructed from the compressed feature is needed, which can be solved by $\boldsymbol{X}_{rec} = \boldsymbol{P}^{-1}\boldsymbol{Y} + \boldsymbol{\mu}$, where $\boldsymbol{\mu}$ is the mean of normalization. Besides, to select an appropriate number of components, the explained variance is defined as: $v_{exp} = \sum_{i=1}^{k} v_i / \sum_{i=1}^{n} v_i$.

Second, The auto-encoder (AE) [31] is used to compress shape features with non-linear transformations. Formally, an AE takes an n-dimensional soft object shape vector $\boldsymbol{x}$ as its input, which is mapped to its k-dimensional 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:

$$\boldsymbol{\theta}^*, \boldsymbol{\theta}'^* = \arg\min_{\boldsymbol{\theta}, \boldsymbol{\theta}'} \frac{1}{n} \sum_{i=1}^{n} L\left(\boldsymbol{x}^{(i)}, g_{\boldsymbol{\theta}'}\left(f_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right)\right)\right) \tag{4}$$

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.

C. Latent Shape Space

With dimensionality transformations, we embed the low-level features of the collected shapes in a low-dimensional latent shape space. In deep generative models, as shown in Fig. 2, a manifold $\mathcal{M}$ is formed through a generator g mapping linear coordinates of variables in latent space $\mathcal{Z}$ ($\mathcal{Z} \subseteq \mathbb{R}^k$) into the curvilinear coordinates of originally high-dimensional shape space $\mathcal{X}$ ($\mathcal{X} \subseteq \mathbb{R}^n$, $k \ll n$). Normally, g is a composition function of numerous layers, $g = g^{(1)} \circ g^{(2)} \circ \ldots \circ g^{(\ell)}$, with ℓ indexing the layer. Combined with a nonlinear activation function ϕ, it can be represented as below:

$$g_k^{(l)}\left(z^{(l)}\right) = \phi\left(W_k^{(l)} z^{(l)} + b^{(l)}\right) \tag{5}$$

where $g_k^{(l)}$ and $W_k^{(l)}$ denote the kth component of the output and kth row of the weight matrix, respectively. The image of g could be a smooth (i.e., C^{∞}), k-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, $\mathcal{M}$ is a locally differentiable but globally intersected k-dimensional Euclidean space (*immersed* manifold).

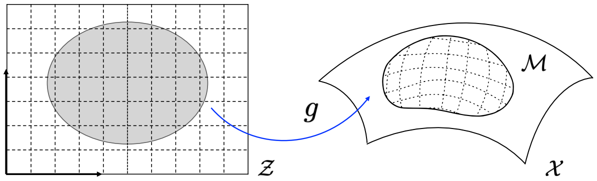

Fig. 2. Conceptual representation of a generator g as a mapping from low-dimensional latent space $\mathcal{Z}$ into a manifold in input data space $\mathcal{X}$.

Mathematically, $\forall z \in \mathcal{Z}$, the Jacobian matrix of g, $J_g(z)$, maps the tangent space of $\mathcal{Z}$ at z, $T_z\mathcal{Z}$, to the tangent space of $\mathcal{M}$ at $g(z)$, $T_{g(z)}\mathcal{M}$. In AE, backpropagation algorithm will calculate out a $k \times n$ partial derivative matrix, $J_g(z)$. Consider two vectors $p, q \in T_x\mathcal{M}$ in a linear subspace of $\mathcal{X}$, as a riemannian metric offers the format of an inner product for different tangent vectors in $T_x\mathcal{M}$, therefore, the Riemannian metric of $\langle u, v\rangle$ can be re-expressed with the dot product of x in the Euclidean space. 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 [32].

D. 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 an equivalent variable on the manifold $\mathcal{M}$. For each point $z \in \mathcal{Z}$, the Riemannian metric is defined as below:

$$G(z) = J_g(z)^T J_g(z) \tag{6}$$

Therefore, the inner product of two tangent vectors $u, v \in T_z Z$ is $\langle u, v\rangle = u^T G(z) v$. Consider a smooth curve in the latent space $\gamma_t : [a, b] \rightarrow \mathcal{Z}$, 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:

$$L\left[g(\gamma_t)\right] = \int_a^b \left\|\dot{g}\left(\gamma_t\right)\right\| \mathrm{d}t = \int_a^b \left\|\mathbf{J}_{\gamma_t} \dot{\gamma}_t\right\| \mathrm{d}t \tag{7}$$

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:

$$\|\mathbf{J}_\gamma \dot{\gamma}\| = \sqrt{\dot{\gamma}^\top \left(\mathbf{J}_\gamma^\top \mathbf{J}_\gamma\right) \dot{\gamma}} = \sqrt{\dot{\gamma}^\top \mathbf{M}_\gamma \dot{\gamma}} \tag{8}$$

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:

$$L(\gamma)=\int_a^b \sqrt{\dot{\gamma}_t^\top \mathbf{M}_{\gamma_t}\dot{\gamma}_t}\mathrm{d}t \tag{9}$$

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

$$E(\gamma)=\frac{1}{2}\int_a^b \dot{\gamma}(t)^T G_{\gamma(t)}\dot{\gamma}(t)dt \tag{10}$$

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

$$\frac{d^2\gamma^\mu}{dt^2}=-\Gamma^\mu_{\alpha\beta}\frac{d\gamma^\alpha}{dt}\frac{d\gamma^\beta}{dt} \tag{11}$$

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

$$\Gamma^\mu_{\alpha\beta}=\frac{1}{2}G^{v\mu}\left(\frac{\partial G_{v\beta}}{\partial\gamma^\alpha}+\frac{\partial G_{v\alpha}}{\partial\gamma^\beta}-\frac{\partial G_{\alpha\beta}}{\partial x^\mu}\right) \tag{12}$$

where $G^{v\mu}$ is the inverse of $G_{v\mu}$. However, calculation of the Christoffel symbols is considerably expensive, because this process involves the inverse of G and second order derivatives of the g. Thus, instead of getting the entire geodesic path, we only calculate out few discrete points along on the geodesic path with discrete geodesic energy (10) to avoid expensive calculations. Formally, consider a discretized curve $\gamma:[0,1]\rightarrow\mathcal{Z}$ denoted by a series of coordinates $z_0,z_1,\ldots,z_N\in\mathcal{Z}$. With T time steps, a sequence of discrete time intervals, $\delta t=1/N$, is generated, which matches a discretized points on the manifold $\mathcal{M}$, $g(z_i)$. With a small shift, the velocity of $g(z_i)$ can be formulated by $v_i=\left(g\left(z_{i+1}\right)-g\left(z_i\right)\right)/\delta t$. Similarly, the energy of this curve can be given:

$$E_{z_i}=\frac{1}{2}\sum_{i=0}^{N}\frac{1}{\delta t}\left\|g\left(z_{i+1}\right)-g\left(z_i\right)\right\|^2 \tag{13}$$

Fixing the first and last points, z_0 and z_N, as the beginning and ending points of the geodesic curve, minimizing this energy function would result in an approximated geodesic path, which can be obtained by performing a gradient descent algorithm for $z_1,\ldots,z_{N-1}$, along this curve. The gradient at z_i is computed as:

$$\nabla_{z_i}E=-\frac{1}{\delta t}J_g^T\left(z_i\right)\left(g\left(z_{i+1}\right)-2g\left(z_i\right)+g\left(z_{i-1}\right)\right) \tag{14}$$

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 shape coordinates, $z_0, z_N \in \mathcal{Z}$;
learning rate $\alpha \in \mathbb{R}_+$
Output: discretized geodesic points,
        $z_0, z_1, \ldots, z_N \in \mathcal{Z}$
Initialize $z_i$ by a linear interpolation between $z_0$ and
  $z_N$ while $\sum_i \|\nabla_{z_i}E\|^2 > \epsilon$ do
    for $i \in \{1,\ldots,N-1\}$ do
        Calculate $\nabla_{z_i}E$ using (14)
        $z_i \leftarrow z_i - \alpha\nabla_{z_i}E$
    end
end
return $z_0, z_1, \ldots, z_N$
```

E. Semantic Analysis

To make the deformation process of soft objects explainable, semantic analysis techniques are introduced to the high-level representation in **LaSeSOM**. First, to identify the effect on each shape dimension, Alg. 2 is designed. In this algorithm, given a latent variable $\boldsymbol{z}_0$ encoded by function h, 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. Second, *semantic deformation analysis* is introduced to establish a mapping from soft object deformations to latent variables in latent shape space. Intuitively, if the dimensionality reduction technique is invertible, then we can explore deformation rules between different shape classes by observing the latent shape space. With performing classification on the latent variables encoded from collected shapes, this path will travel through different spaces enclosed by pre-defined shape classes, thus revealing some rules of shape deformations in real-world applications. Third, *latent shape planning* presents a solution to the shape planning problem for soft objects from the current shape to the target shape. Let the current shape and target shape be $\boldsymbol{x}_0$ and $\boldsymbol{x}_*$, respectively. After dimensionality transformation, the input shapes are transformed to a k-dimensional latent shape space ($\mathcal{Z}\subseteq\mathbb{R}^k$). With a encoder h, the encoded coordinates of $\boldsymbol{z}_0$ and $\boldsymbol{z}_*$ are readily known in this latent space. As Fig. 3 shows, shapes are represented as nodes in the latent space and these nodes are connected to form different neighbor networks with

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 $\mathcal{D}_s^{(p)}$
Compute the coordinate $\boldsymbol{z}_0$ with $\boldsymbol{z}_0 = h(\boldsymbol{x}_0)$
$\mathcal{G}_{low}^{(p)} = \{\boldsymbol{z}_0, \boldsymbol{z}_1, \ldots, \boldsymbol{z}_N\}$ = Interpolation$(\boldsymbol{z}_0, p, \delta, N)$
if $g$ is not linear then
    Update $\mathcal{G}_{low}^{(p)}$ with geodesic Alg. (1)
end
$\mathcal{G}_{high}^{(p)} = \{\boldsymbol{x}_1', \boldsymbol{x}_2', \ldots, \boldsymbol{x}_n'\} = g(\mathcal{G}_{low}^{(p)})$
$\mathcal{D}_s^{(p)}$ = Visualizer$(\mathcal{G}_{high}^{(p)})$
return $\mathcal{D}_s^{(p)}$
```

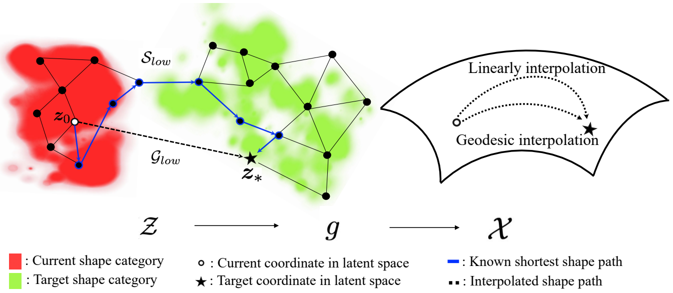
$\mathcal{S}_{low}$
z_0
$\mathcal{G}_{low}$
z_*
Linearly interpolation
Geodesic interpolation
$\mathcal{Z}$
g
$\mathcal{X}$
🟥 : Current shape category
🟩 : Target shape category
○ : Current coordinate in latent space
★ : Target coordinate in latent space
— : Known shortest shape path
‥ : Interpolated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

Fig. 3. 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.

different colors based on the prediction from kNN algorithm. With the implementation of shortest path searching algorithm in the latent shape space, the shape 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, $\mathcal{S}_{low}$ can only find out a shortest path built on known shape data set. This latent shape space contains numerous shapes unknown to the dataset. Thus, we first link $\tilde{\boldsymbol{x}}^{\circ}$ to $\tilde{\boldsymbol{x}}^{*}$ with a straight line. Accordingly, n intervals are set to generate $n+1$ intermediate shape statuses denoted by $\mathcal{G}_{low}$ and then could be updated to obtain a shorter geodesic path if the generator is not a linear transformation. Note that the linear interpolated path is an intermediate state of geodesic interpolation and they are not exclusive approaches. 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 $\mathcal{D}_p$.

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 $\mathcal{D}_p$
Compute the coordinates using $(\boldsymbol{z}_0, \boldsymbol{z}_*) = h(\boldsymbol{x}_0, \boldsymbol{x}_*)$
$\mathcal{S}_{low} = \{\boldsymbol{z}_0, \boldsymbol{z}_1, \ldots, \boldsymbol{z}_*\}$ = ShortestPath$(\boldsymbol{z}_0, \boldsymbol{z}_*)$
$\mathcal{S}_{high} = g(\mathcal{S}_{low})$
$\mathcal{G}_{low} = \{\boldsymbol{z}_0, \boldsymbol{z}'_1, \ldots, \boldsymbol{z}_*\}$ = Interpolation$(\boldsymbol{z}_0, \boldsymbol{z}_*, N)$
if $g$ is not linear then
    Update $\mathcal{G}_{low}$ with geodesic Alg. (1)
end
$\mathcal{G}_{high} = g(\mathcal{G}_{low})$
$\mathcal{D}_p = \{$Visualizer$(\mathcal{S}_{high})$, Visualizer$(\mathcal{G}_{high})\}$
return $\mathcal{D}_p$
```

III. Results

In this section, the data collection for building **LaSeSOM** is described first, and afterwards the framework is used to present different representation results in a robotic teleoperated soft object manipulation task via Leap Motion [33] demonstration.

A. Data Collection

As shown in Fig. 4, two different soft objects (a foam bar and a foam sheet) were used to collect deformed shapes.

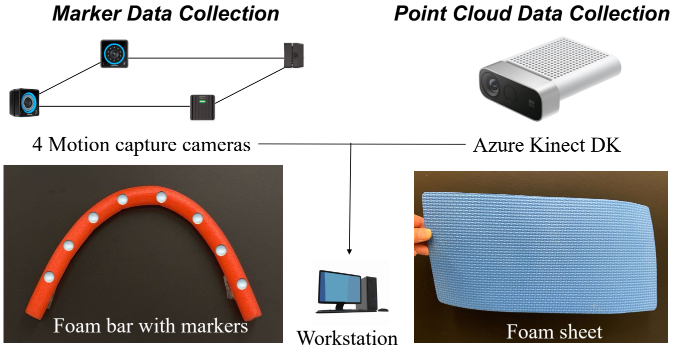

Fig. 4. Experimental setups of the shape data collection to build **LaSeSOM**. Left shows the setup to collect ordered marker data for a foam bar, while right is used to collect unordered point cloud data for a foam sheet.

For the foam bar, the Prime 13 motion tracking system was used to track the position of each marker mounted on the its surface in 30 FPS. 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). Fig. 5 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

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	2250

B. Semantic Feature Analysis

1) Shape Features: To examine the fitting performance, the coefficient of determination R^2 [34], defined as: $1 - \sum_i (y_i - f_i) / \sum_i (y_i - \bar{y})^2$, is used to quantify the amount of variability explained by Fourier approximation. As shown in Fig. 7(a), the shape descriptor becomes more accurate along with the increasing number of harmonics. Specifically, the line and arch class shapes demonstrate better performance than the other class shapes under the same number of harmonics, because the S-shaped and helix class shapes are more complex to represent with same number of harmonics.

2) Reduced Dimensions: With PCA performed on Fourier coefficients of marker data, the number of components is set as 4 ($\text{Var}_{exp} \geq 95\%$ when $k = 4$) to investigate following semantic analysis. In the semantic analysis algorithm, parameters are set with iteration $T = 10$, $k = 4$, and $t = 1$ and Fig. 6 (a) to 6 (d) visually presents the individual semantic effect of the four features. Generally, 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

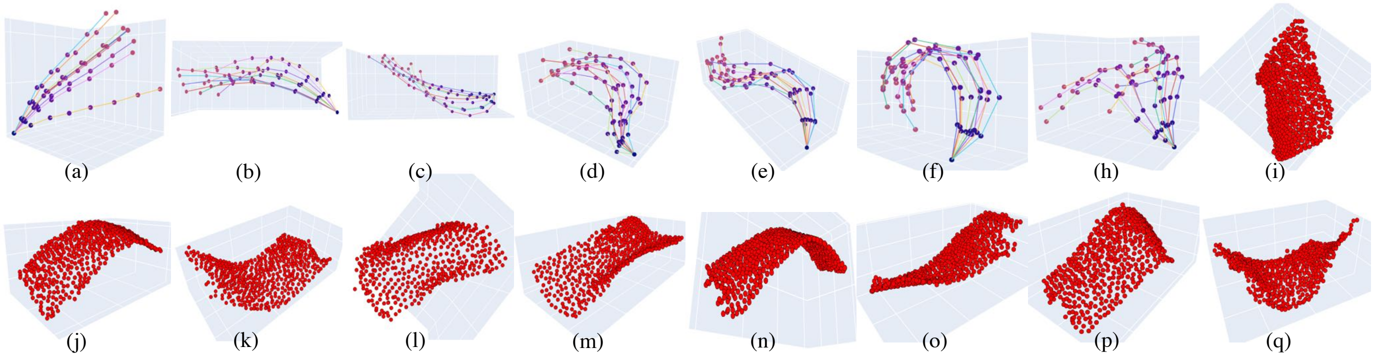

Fig. 5. Visualizations of shape samples of predefined categories. Figures (a) to (h) shows the seven classes for the foam bar deformation, and figures (i) to (q) presents the nine classes for the foam sheet deformation.

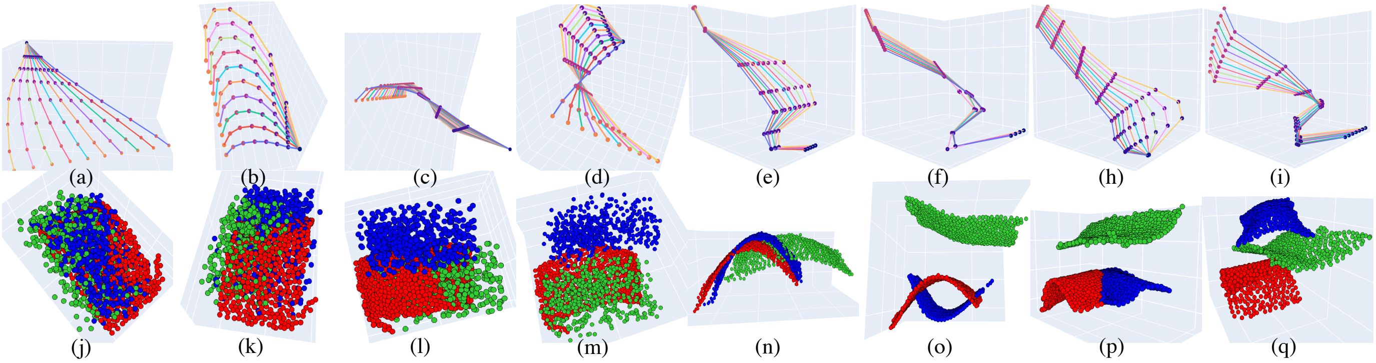

Fig. 6. Visual comparison of the semantic features from different dimensionality transformation techniques, where figures (a) to (d) and (e) to (i) respectively shows the results of the foam bar from PCA and AE. Figures (j) to (q) shows the visualization results of eight (total in *64-dim*) semantic features.

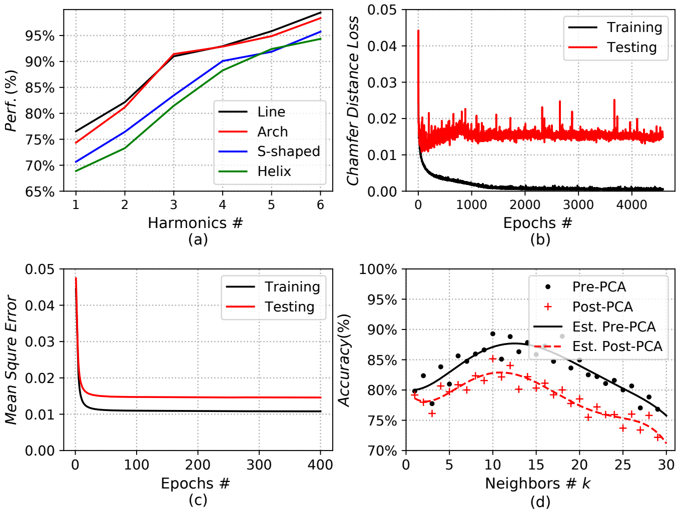

Fig. 7. (a) The performance of Fourier approximation for four shape classes by different harmonics; (b) and (c) respectively show the training and validation errors for the corresponding soft objects; (d) presents the pre-PCA and post-PCA classification accuracy for the foam bar.

TABLE II
NETWORK ARCHITECTURE

Marker data (Form Bar)	Point cloud (Foam Sheet)
Input 8×3	*Input 512×3*
Flatten	3×1 conv, 8, BatchNorm, ReLU
FC 8, BatchNorm, ReLU	8×1 conv, 32, BatchNorm, ReLU
FC 4, BatchNorm, ReLU	32×1 conv, 64, BatchNorm, ReLU
FC 8, BatchNorm, Sigmoid	Max pool
FC 24, Sigmoid	FC 256, Batch norm, Sigmoid
Reshape 8×3	FC 512, Batch norm, Sigmoid
	FC 1536, Sigmoid
	Reshape 512×3

arch shape. The third component is trend to depict the degree of "S" shape, whereas the fourth component tries to capture helix shape. Though, the results shows partially combined semantic effect (not single effect), each feature dimension has a dominant semantic effect, respectively. Note that the results of the foam sheet with PCA are not presented because PCA can only perform the ordered data.

To compare with PCA, we implement AE on both marker data and point cloud data with the structure in Tab. II. The latent dimension is kept at 4 for marker's dataset. By performing the similar semantic feature analysis on this latent dimensions, Figs. 6 (e) to 6 (i) visually present the individual semantic effect of the four dimensions for the code layer. Unlike PCA, these four dimensions mainly depict "S" shapes from different perspectives, because the 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. As for point cloud data of the foam sheet, the latent dimension is kept at 64 with the network architecture as shown in Tab. II. and Fig. 7 (b) shows the corresponding loss trend for training and testing. With the same implementation of semantic feature analysis, Figs. 7 (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

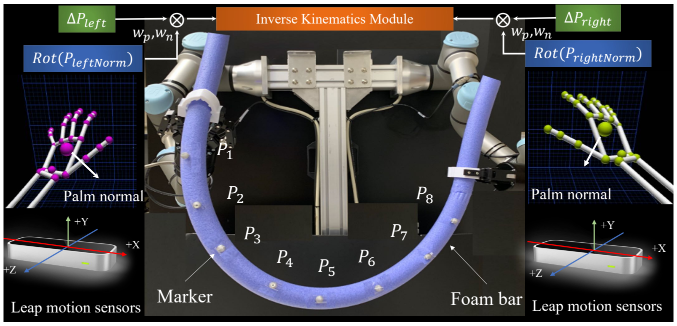

Fig. 8. Architecture of the teleoperated system with Leap Motion sensors for the validation of **LaSeSOM**. The new pose is computed with the displacement of palm position δp and variance of the hand orientation $Rot(\mathbf{n}_p)$ between 10 frames, multiplied by appropriate weights w_p and w_n.

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. In summary, PCA shows more meaningful semantic analysis results than AE, but it suffers from an unordered data structure. However, AE can perform both ordered and unordered data but hard to explain the semantic meaning of encoded features. Fig. 7 (d) shows the best number k for kNN to classify the shapes in latent space with a *5-fold* cross-validation and both pre- and post-PCA kNN models share a similar trend and reach a peak under the same $k = 12$.

C. Latent Shape Space

To imitate the soft object manipulation with human hands, and validate the effectiveness of realtime feedback in a robotic soft object manipulation task, a hand gesture-based teleoperation using Leap Motion [35] is an appropriate technique to extract the control signals from hand gestures to teleoperate the soft object in a real-time manner, and the corresponding experimental setup is shown in Fig. 8, where the robot grippers are fully constrained [36] as during the data collection stage. The related shape dataset of this manipulation task is shown in the Table I (dataset #2).

1) Semantic Deformation: As Fig. 9 (a) shows, all the shapes collected from the foam bar (dataset #1) are encoded into a 3D latent shape space with t-SNE built from AE. In this space, the deformation path generated from gesture controls is represented as a red curve and different shape categories of dataset #1 were organized with *mesh3D* from *Plotly* and rendered with different colors according to the prediction of 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 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 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.

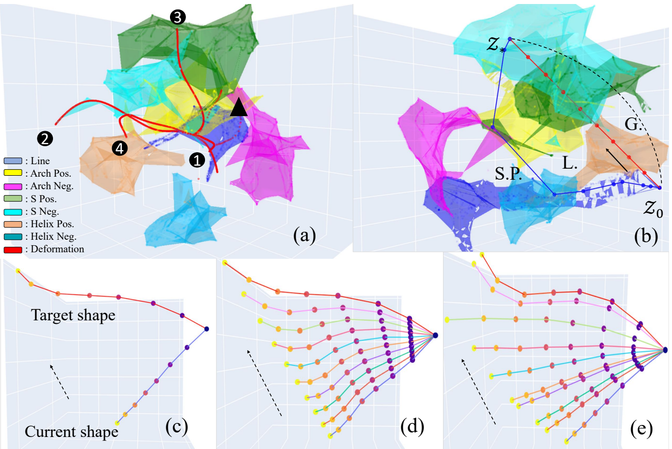

Fig. 9. Visualization of the process of latent shape planning for the foam bar. (a) Deformation trace of the manipulation task with Leap motion in latent shape space; (c) shows the beginning shape and the target shape; figures (d) and (e) present the planned shape deformations; (b) presents their corresponding deformation paths with shape planning algorithm.

2) Latent Shape Planning: 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. 9(b) shows a beginning line and target S-shape of a foam bar. With the encoder h (illustrated in Table II), we can get encoded shapes in $\mathcal{Z}$ space (see Fig. 9-a), 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. Shape set $\mathcal{S}_{low}$ denoted by the blue spline is calculated by a shortest path search algorithm on collected data. In dataset #1, $\mathcal{S}_{low}$ is a sequence of shape index, $\{x_{540}, x_{532}, x_{530}, x_{526}, x_{568}, x_{777}, x_{774}, x_{1929}, x_{5812}, x_{5040}\}$. Another shape set $\mathcal{G}_{high}$ is generated by a linear interpolation denoted by the red spline between $\boldsymbol{z}_0$ and $\boldsymbol{z}_*$ at first, and then an iterative updating on each coordinate with geodesic path illustrated in Alg. 1. Figs. 9 (c) and (d) show the resulting deformation processes from a geodesic interpolation and shortest path, respectively. We can clearly observed that the geodesic path-based interpolation deformation process is smoother compared with the process with a shortest path.

To compare geodesic path-based interpolation to its intermediate state (pure linear interpolation), Fig. 10 shows two groups of point clouds (foam sheet) generated with the shortest path, linear interpolation, geodesic interpolation and the corresponding arc lengths. The first column represents the current shapes and the last for the target shapes. The geodesic path-based interpolation has a shorter arc length on data manifold and smoother morphing process compared with the shortest path and linear interpolation methods, which is supported by the morphing processes marked by green boxes. In contrast, the shortest path-based and linear interpolation methods show several results (marked by red boxes) with unsatisfied physical feasibility, which may cause excessive stretching and damage the object. Although, the geodesic curve on the manifold presents a shorter arc length compared

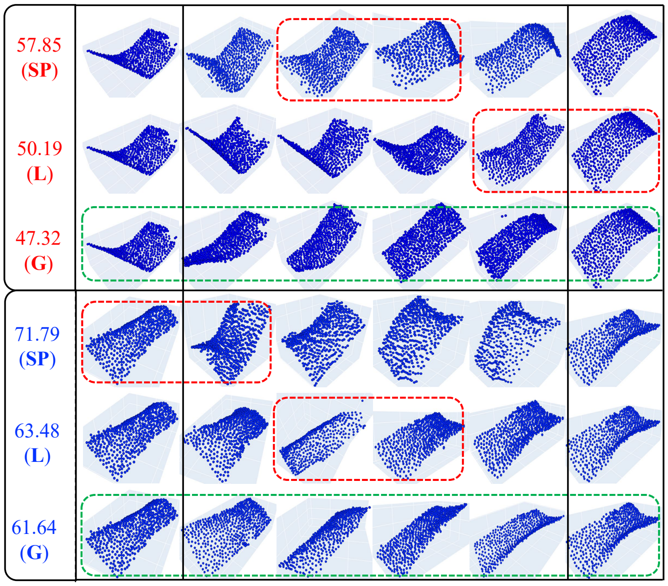

Fig. 10. Shape planning results of shortest path, linear interpolation, and geodesic interpolation for foam sheet dataset. Column 1: arc length; Rows 1, 4: shortest path; Rows 2, 5: linear; Rows 3, 6: geodesic.

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.

IV. Conclusions

In this paper, we present a generic latent representation framework for semantic soft object manipulation tasks. With dimensionality transformations, we embed the shapes of soft objects from the originally high-dimensional shape space into a semantically low-dimensional latent shape space and solve the shape planning with designed geodesic path-based algorithms on the data manifold. The numerical and experimental results have validated the effectiveness of the proposed framework. As future research, we plan to implement a manipulator with **LaSeSOM** based feedback control for soft objects and transfer learning for soft object representation.

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. 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.

[3] Z. Wang, X. Li, D. Navarro-Alarcon, and Y. Liu, "A unified controller for region-reaching and deforming of soft objects," in *Int. Conf. Intelligent Robots and Systems*, 2018, pp. 472–478.

[4] 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.

[5] 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.

[6] D. Navarro-Alarcon *et al.*, "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.

[7] 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.

[8] 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.

[9] C. Collewet and E. Marchand, "Photometric visual servoing," *IEEE Trans. Robot.*, vol. 27, no. 4, pp. 828–834, 2011.

[10] E. Marchand, "Subspace-based direct visual servoing," *IEEE Robot. Autom. Lett.*, vol. 4, no. 3, pp. 2699–2706, 2019.

[11] K. M. Digumarti, B. Trimmer, A. T. Conn, and J. Rossiter, "Quantifying dynamic shapes in soft morphologies," *Soft Robot.*, 2019.

[12] J. Zhu, D. Navarro-Alarcon, R. Passama, and A. Cherubini, "Vision-based manipulation of deformable and rigid objects using subspace projections of 2d contours," *arXiv preprint arXiv:2006.09023*, 2020.

[13] K. Xu, V. G. Kim *et al.*, "Data-driven shape analysis and processing," in *SIGGRAPH ASIA 2016 Courses*, 2016, pp. 1–38.

[14] 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.

[15] A. Golovinskiy and T. Funkhouser, "Consistent segmentation of 3d models," *Computers Graphics*, vol. 33, no. 3, pp. 262–269, 2009.

[16] 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.

[17] 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.

[18] L. Tao, L. Yuan, and J. Sun, "Skyfinder: attribute-based sky image search," *ACM Trans. Graphic.*, vol. 28, no. 3, pp. 1–5, 2009.

[19] D. Parikh and K. Grauman, "Relative attributes," in *Int. Conf. Comput. Vis.*, 2011, pp. 503–510.

[20] 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.

[21] 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.

[22] M. Attene *et al.*, "Characterization of 3d shape parts for semantic annotation," *Comput. Aided Des.*, vol. 41, no. 10, pp. 756–763, 2009.

[23] 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.

[24] 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.

[25] G. Arvanitidis *et al.*, "Latent space oddity: On the curvature of deep generative models," in *Int. Conf. Learn Represent.*, 2018.

[26] 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.

[27] 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.

[28] 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.

[29] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," *Chemometr. Intell. Lab.*, vol. 2, no. 1-3, pp. 37–52, 1987.

[30] 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.

[31] G. E. Hinton *et al.*, "Reducing the dimensionality of data with neural networks," *science*, vol. 313, no. 5786, pp. 504–507, 2006.

[32] M. P. d. Carmo, *Riemannian geometry*. Birkhäuser, 1992.

[33] I. Jang, J. Carrasco, A. Weightman, and B. Lennox, "Intuitive barehand teleoperation of a robotic manipulator using virtual reality and leap motion," in *Annu. Conf. Auton. Robot. Syst.*, 2019, pp. 283–294.

[34] 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.

[35] 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.

[36] D. Navarro-Alarcon and Y. Liu, "A dynamic and uncalibrated method to visually servo-control elastic deformations by fully-constrained robotic grippers," in *IEEE Int. Conf. on Robotics and Automation*, 2014, pp. 4457–4462.