CaDeX: Learning Canonical Deformation Coordinate Space for Dynamic Surface Representation via Neural Homeomorphism

Jiahui Lei  
University of Pennsylvania  
leijh@seas.upenn.edu

Kostas Daniilidis  
University of Pennsylvania  
kostas@cis.upenn.edu

Figure 1. We model the deformable surface through a learned canonical shape (the middle shape) and factorize the deformation (orange arrow) by learnable continuous bijective canonical maps (green bidirectional arrows) that provide the cycle consistency and topology preservation. Visual results are from the test set on dynamic animals, human bodies and articulated objects.

Abstract

While neural representations for static 3D shapes are widely studied, representations for deformable surfaces are limited to be template-dependent or to lack efficiency. We introduce Canonical Deformation Coordinate Space (CaDeX), a unified representation of both shape and non-rigid motion. Our key insight is the factorization of the deformation between frames by continuous bijective canonical maps (homeomorphisms) and their inverses that go through a learned canonical shape. Our novel deformation representation and its implementation are simple, efficient, and guarantee cycle consistency, topology preservation, and, if needed, volume conservation. Our modeling of the learned canonical shapes provides a flexible and stable space for shape prior learning. We demonstrate state-of-the-art performance in modeling a wide range of deformable geometries: human bodies, animal bodies, and articulated objects. 1

1. Introduction

Humans perceive, interact, and learn in a continuously changing real world. One of our key perceptual capabilities is the modeling of a dynamic 3D world. Such geometric intelligence requires sufficiently general neural representations that can model different dynamic geometries in 4D sequences to facilitate solving robotics [55], computer vision [29], and graphics [44] tasks. Unlike the widely studied 3D neural representations, a dynamic representa-

1https://www.cis.upenn.edu/~leijh/projects/cadex

tion has to be able to associate (for example, finding correspondence) and aggregate (for example, reconstruction and texturing) information across the deformation states of the world. Directly extending a successful static 3D representation (for example, [33]) to each deformed frame leads to low efficiency [36], and the inability to model the information flow across frames, which is critical when solving ill-posed problems as in [44]. Our desired dynamic representation needs to simultaneously represent a global surface (canonical/reference shape) across all frames and the consistent deformation (correspondence/flow/motion) between any frame pair (Fig. 1), so that we can recover the dynamic geometry by reconstructing only one reference surface and generating the rest of the deformed surfaces by using the consistent deformation representation as well as associate and aggregate information across frames (Fig. 2A).

The majority of dynamic representations that satisfy the above desired properties are model-based and rely on parametric models for specific categories like human bodies [1, 31] (Fig. 2B), faces [4, 30], or hands [47]. On the contrary, recent model-free methods like the implicit flow [36, 50] (Fig. 2C) apply one universal 4D representation but model the canonical shape in an ad hoc chosen frame [36, 50] that complicates the shape prior. Alternatively, the choice of an approximate mean/neutral shape [58] as the canonical shape can limit the shape expressibility. Modeling of the deformation is done by either MLPs [50, 58] that ignore the real world deformation properties, or by ODEs [36] that are inefficient for space deformation, or by an optimized embedded graph [6] or Atlas [2] that are sequence specific.

In this work, we introduce a novel and general archi-
tecture and representation that enable a competitive re-
construction of every frame and the recovery of consist-
tent correspondence across frames. Our approach is rooted
in the factorization of deformation (Sec. 3.1). If we as-
sume that the topology does not change during deformation,
all deformed surfaces of one instance can be regarded as
equivalent through continuous bijective mappings (home-
omorphisms). This allows us to factorize the deforma-
tion between two frames by the composition of two con-
tinuous invertible functions such that one maps the source
frame into a common 3D Canonical Deformation Coordi-
nate Space (CaDeX) while another maps it back to the des-
tination frame. Such a factorization and its implementa-
tion (Sec. 3.2) is novel, simple, and efficient (compared to
ODEs [36]) while it guarantees cycle consistency, topology
preservation, and, if necessary, volume conservation
(Sec. 3.3). The canonical shape embedded in the CaDeX
can be regarded as the representative element, while the
associated invertible mappings that transform between de-
formed frames and the CaDeX are the canonical maps.
Therefore, we model the reference surface directly in the
CaDeX via an implicit field [33] (Sec. 3.4), which can be
optimized together with the canonical maps during training.

In summary, our main contributions are: (1) A novel
general representation and architecture for dynamic sur-
faces that jointly solve the canonical shape and consistent
deforation problems. (2) Learnable continuous bijective
canonical maps and canonical shapes that jointly factorize
the shape deformation, and are novel, simple, efficient, and
guarantee cycle consistency and topology preservation. (3)
A novel solution to the dynamic surface reconstruction and
correspondence tasks given sparse point clouds or depth
views based on the proposed representation. (4) We demon-
strate state-of-the-art performance on modelling different
defeasible categories: Human bodies [5], Animals [57]
and Articulated Objects [53].

2. Related Work

Proposed neural representations for static 3D geo-
metry [9,11,17,18,20,26,33,34,37,40,43,56] are promising,
but most of them do not involve modeling of deformations.
A few recent approaches represent or process 3D shapes via
deformation [13,22,23,25,59], but they focus on static 3D
shape collections that do not meet the requirements (e.g.
efficiency) for processing 4D data. We will focus our related
work on dynamic representations of deformable geometry.

Model-Based Dynamic Representation: Many success-
ful 3D parametric models for specific shape categories have
been introduced, for example, the morphable model [4] and
FLAME [30] for faces, SCAPE [1] and SMPL [31] for hu-
man bodies, and MANO [47] for hands. These model-
based representations (Fig. 2B) suffer from limited expres-
sivity, which can be mitigated by neural networks. Net-
works can express detailed template shapes based on tem-
plate meshes [32,38,48,52] or skeletons [12,27,51], and
can learn more detailed skinning functions [8,48,52] or for-
ward deformations [38]. However, they rely on the strong
assumption of canonicalization through the pose, skeleton,
and template mesh, which makes them limited to specific
categories, and insufficient for modeling the rich dynamic
3D world. Our method does not rely on any hard-wired
template mesh or skeleton, and the same architecture is uni-
versal for all shape categories.

Model-Free Dynamic Representation: Recent works [6,
24,36,50] extend the success of static 3D representa-
tions [9,33,40] to 4D by modeling the deformation be-
tween frames. Fig. 2C illustrates how the two closest works
to ours, O-Flow [36] and LPDC [50], are related to our
problem formulation. First, our method differs from O-
Flow [36] and LPDC [50] in the representation of the space
deforation. We represent the deformation through a novel
canonical map factorization that is efficient and guarantees
real world properties based on conditional neural home-
omorphisms [14,15], while O-Flow [36] uses a Neural-
ODE [7] that also guarantees the production of a well-
behaved deformation (see [22] for details) but with higher
computational complexity than ours. LPDC [50] replaces
the Neural-ODE [7] by a Multilayer Perceptron (MLP) to
learn correspondences in parallel. However, the MLP de-
formation [20,41,44,50] has difficulty to model a home-
omorphism or express real world deformation properties.
Note that both O-Flow [36] and LPDC [50] compute the
reference surface in the first frame, which turns out to be a
random choice, since the shape can be in an arbitrary de-
formation state in the first frame. Our reference shape is
modeled by Neural-ODEs [36] or MLPs [50].

I3DM [58] learns a near neutral/mean canonical tem-
plate from human head scans, which limits its expressive
ability. CASPR [46] and Garment Nets [10] learn a canoni-
calization of deformable objects, but rely on the ground
truth canonical coordinate supervision, which is often inaccessible. Other neural dynamic representations include the learned embedded graph [6] first proposed in [49] and parametric atlases [2]. Beyond 4D data, A-SDF [35] models the general articulated objects with a specially designed disentanglement network, but it cannot model correspondence. Instead, our method achieves stronger disentanglement by explicitly modeling the deformation.

**Invertible Networks for 3D representation:** Many works [3, 7, 14, 15, 19, 28, 39] have been proposed to construct invertible networks for generative models. In 3D deep learning, Neural-ODE [7] is widely used as a good model of deformation [22, 23, 35, 25, 36] or transformation of point cloud [56]. ShapeFlow [25] learns a “Hub-and-spoke” surface deformation for 3D shape collection via ODEs, but is inefficient when applied to the 4D data since every frame needs to be lifted to the “hub” through integration. Besides ODEs, I-ResNet [3] is used in [21] to build invertible deformation for shape editing. Our method is inspired by Neural-Parts [42] where Real-NVP [15] is used to model the deformation from a sphere primitive to a local part. While we also use Real-NVP [15] for its simplicity and efficiency, we have two distinct differences compared to [42]: Our canonical shape is a learned implicit surface instead of a fixed sphere that can only capture local parts; we use the inverse of the Real-NVP to close the factorization cycle, while [42] uses the inverse in a complementary training path.

### 3. Method

As in Fig. 2A, given a sequence$^2$ of point cloud observations of one deforming instance, our goal is to reconstruct a sequence of surfaces. Instead of directly solving a correspondence map between two frames during reconstruction, we propose an architecture (Fig. 3) where the interframe correspondence is computed via a pivot canonical shape. We will call the map between a surface in any deformed frame and the canonical shape a canonical map.

#### 3.1. CaDeX and Canonical Map

Let us denote $[x^t, y^t, z^t] \in \mathbb{R}^3$ as the 3D coordinates of the input 3D space$^3$, in which a deformed surface $S_t$ at time $t_t$ is embedded. Consider a continuous bijective mapping (homeomorphism) $H_t : \mathbb{R}^3 \mapsto \mathbb{R}^3$ at time $t_t$ that maps each deformed coordinate to its global (shared over different time frames) 3D coordinate $[u, v, w] = H_t([x^t, y^t, z^t])$. Note that $[u, v, w]$ has no index of time and can be seen as a globally consistent indicator of each correspondence trajectory across time in the input 3D space. Hence, we name $[u, v, w]$ the canonical deformation coordinates of the position $[x^t, y^t, z^t]$ at time $t_t$ and call the $uvw$ 3D space the Canonical Deformation Coordinate Space (CaDeX) of the sequence $S = \{S_t\}$. The homeomorphisms $H_t$ that transform $[x^t, y^t, z^t]$ to $[u, v, w]$ are called canonical maps.

Since CaDeX is globally shared across time, we model the canonical shape (surface) $U$ directly in CaDeX instead of selecting an input frame as is the case in [36, 50]. Taking advantage of neural fields [54], we model $U$ as a level set of an occupancy field [33]:

$$U = \{ [u, v, w] \mid \text{OccField}( [u, v, w] ) = l \} ,$$

where $l$ is the level surface. Using the inverse of each canonical map $H_t$ at time $t_t$, we can directly obtain each deformed surface at time $t_t$ in the input 3D space as:

$$S_t = \{ H_t^{-1}([u, v, w]) \mid \forall [u, v, w] \in U \} .$$

The correspondence/deformation $F_t$ that associates any coordinate (for both surface and non-surface points) from the 3D space at time $t_t$ to the 3D space at time $t_j$ can be factorized by the canonical maps as:

$$[x^t, y^t, z^t] = F_{tj}([x^j, y^j, z^j]) = H^{-1}_j \circ H_t([x^t, y^t, z^t]) .$$

Note that $H_t$ must be invertible; otherwise, the above deformation function cannot be defined. By now, any surface that is topologically isomorphic to the deformable instance satisfies the above definitions, leading to infinitely many valid canonical shapes and maps. In the following, we will optimize the canonical shapes and maps predicted by the architecture in Fig. 3 subject to the priors from the dataset.

#### 3.2. Canonical Map Implementation

**Neural Homeomorphism**

One key technique of our implementation is an efficient way to parameterize and learn the homeomorphism between coordinate spaces. Unfortunately, the widely used Neural-ODEs [7] do not meet our efficiency requirements since a full integration would have to be applied to every frame. Inspired by [42], we utilize the Conditional Real-NVP [15] (Real-valued Non-Volume Preserving) or the NICE [14] (Nonlinear Independent Component Estimation) normalizing flow implementations to learn the homeomorphism. Taking the more general NVP [15] as an example (Fig. 3-C), with the network being a stack of Coupling Blocks [15], we apply NVP to 3D coordinates. During initialization, each block is randomly assigned an input split pattern; for example, a block always splits $[x, y, z]$ to $[x, y]$ and $[z]$. Given a condition latent code $c$, each block takes in 3D coordinates $[x, y, z]$ and outputs the transformed coordinates $[x', y', z']$ by changing one part based on the other part in the input coordinate split:

$$[x', y', z'] = [x, y, z \exp(s_\theta(x, y|c)) + t_\theta(x, y|c)]$$

where $s_\theta(\cdot|c) : \mathbb{R}^2 \mapsto \mathbb{R}$ and $t_\theta(\cdot|c) : \mathbb{R}^2 \mapsto \mathbb{R}$ are scale and translation predicted by any network conditioned on $c$.

---

$^2$Or a set, but for conciseness, we will only refer to the sequence.

$^3$The superscript $i$ refers to the time index.
Figure 3. (A) **Canonical Map** (Sec. 3.2): A sequence (or a set) of input point clouds is first sent to the Deformation Encoder generating deformation embeddings $c_i$ for each frame. Then the canonical map $H$ can transform any coordinate (e.g. a yellow point in the point cloud, a blue query position for implicit field or a purple source point for correspondence) from any deformed frame to the canonical coordinate via $H$ conditioning on the corresponding deformation embedding. The correspondence prediction (right bottom) can be obtained by directly mapping back the canonical coordinate through $H^{-1}$.

(B) **Canonical Shape Encoder-Decoder** (Sec. 3.4): All input multi-frame point clouds are first transformed to the canonical space via $H$ and are directly unioned to aggregate a canonical observation. The global geometry embedding $g$ (unique across frames) is encoded via a PointNet $\phi$, and the occupancy value for the canonical coordinate of a query position at $t_i$ (blue point) is predicted through a standard OccNet $\psi$. During training, the occupancy is supervised by $L_R$, and the correspondence can be optionally supervised by $L_C$ (Sec. 3.5).

(C) The **Real-NVP** [15] invertible architecture of $H$ (Sec. 3.2). Such a block models a bijection since the inverse can be immediately derived as:

$$[x, y, z] = \left[ x', y', z' - t_\theta(x', y'|c) \exp(s_\theta(x', y'|c)) \right].$$ (5)

Therefore, the whole stack of blocks is invertible. If the activation functions in each block are continuous, then the whole network models a homeomorphism. *NICE* [14] is simply removing the scale freedom from the NVP block, i.e.: $s_\theta(\cdot|c) \equiv 0$. Note that the inverse of NVP and NICE is as simple as the forward, which induces our desired efficiency and simplicity, and enables using the definition in Eq.3.

**H architecture** Note that in Eq. 2, each deformed surface $S_i$ has a different canonical map $H_i$ that associates $S_i$ with $U$. We implement them with the conditional real-NVP or NICE, denoted by $H$ (noncalligraphic). Given the vector $c_i$ that encodes the deformation information at time $t_i$ such that $H_i(\cdot) \equiv H(\cdot; c_i)$, where the network $H$ is shared across different time frames. The canonical deformation coordinates are predicted as (Fig. 3-A, boxes marked with $H$):

$$[u, v, w] = H([x^i, y^i, z^i]; c_i).$$ (6)

Note that on the right side of Eq. 6, the input coordinates and the deformation embedding have the index $t_i$ since they come from each deformed frame. However, after application of the canonical map, the coordinates on the left side are independent of the index because there is only one global CaDeX for this sequence. Finally, the correspondence/deformation between two deformed frames (Eq. 3) can be implemented as:

$$[\hat{x}^j, \hat{y}^j, \hat{z}^j] = H^{-1}(H([x^i, y^i, z^i]; c_i); c_j)$$ (7)

where $[\hat{x}^j, \hat{y}^j, \hat{z}^j]$ is the mapped position at time $t_j$ of the original position $[x^i, y^i, z^i]$ at time $t_i$. Regarding the choice of $H$, Real-NVP [15] can provide more flexible deformation since it has one more degree of freedom (scale); *NICE* [14] guarantees volume conservation (Sec. 3.3) that results in a more regularized deformation.

**Deformation Encoder** To obtain the per-frame deformation embedding $c_i$ that is used as the condition of $H$, we demonstrate two kinds of inputs and three encoder types (Fig. 3-A, orange box). One direct approach is to employ a PointNet that summarizes the deformation code separately per frame ($PF$). If the input is a sequence of point clouds,
we can alternatively use the ST-PointNet variant proposed in [50] to get the deformation code \( ST \). The ST encoder processes the 4D coordinates and applies the pooling spatially and temporally. If the input is a set without order, we develop a 2-phase PointNet to obtain a global set deformation code \( SET \), and then use a 1-D code query network to output the deformation embedding \( c_t \) taking the query articulation angle and the global deformation code as input. Since these are not our main contributions, we refer the reader to the supplementary for details on these encoders.

### 3.3. Properties of the Canonical Map

The novel factorization and its implementation induce the following desired properties of real world deformation:

**Cycle consistency:** The deformation/correspondence between deformed frames predicted by our factorization (Eq. 3, 7) is cycle consistent (path-invariant). The reason is that every canonical map maps any deformed frame in the sequence (or set) to the global CaDeX of this sequence (or set), and the canonical maps are invertible:

\[
F_{jk} \circ F_{ij} = H_{kj}^{-1} \circ H_{ij} \circ H_{ji} = H_{ki}^{-1} \circ H_{i} = F_{ik}.
\]

**Topology preserving deformation:** Since our factorization (Eq. 3, 7) is a composition of two homeomorphisms, the induced deformation function is thus a homeomorphism as well, and therefore never changes the surface topology.

**Volume conservation (NICE):** If \( H \) is implemented by NICE [14], then the predicted deformation preserves the volume of the geometry, which can be proved by the fact that the determinant of the Jacobian off the Jacobi system of every coupling block in NICE [14] is 1 (see Supp. for more details).

**Continuous deformation if \( c \) is continuous:** Some applications require the sequence \( S = \{ S \} \) to be dense on time axis, for example, modeling continuous deformation across time [36]. In this case, the deformation codes \( c \) become a function of time \( c(t) \). Since all activation functions we are using in the canonical map are continuous, it is obvious that if \( c(t) \) is continuous, then the predicted deformation in Eq. 7 must be continuous across \( t \).

### 3.4. Representing Canonical Shape

**Geometry Encoder** We represent the canonical shape in the CaDeX by a standard Occupancy Network [33]. The canonical map brings additional benefits for encoding the global geometry embedding (Fig. 3-B). Denote the observed point cloud at time \( t_i \) as \( X_i = \{ [x_{ij}^i, y_{ij}^i, z_{ij}^i] \mid j = 0, 1, \ldots, N_i \}^4 \), where \( [x_{ij}^i, y_{ij}^i, z_{ij}^i] \) is the 3D coordinate of each point in the point cloud. The observations from different \( t_i \)’s are partial, noisy, and not aligned. We overcome such irregularity by using the same canonical map (Sec.3.2) to obtain a canonical aggregated observation. Given the deformation embedding \( c_t \) per-frame, the canonical observations are merged via set union as:

\[
\tilde{X} = \bigcup_{t_i} \{ H([x_{ij}^i, y_{ij}^i, z_{ij}^i]; c_t) \mid [x_{ij}^i, y_{ij}^i, z_{ij}^i] \in X_i \}.
\]

The global geometry embedding \( g \) of the sequence \( S \) is encoded by a PointNet \( \phi : g = \phi(\tilde{X}) \).

**Geometry Decoder** Given the global geometry embedding \( g \), we obtain the canonical shape encoded in \( g \) via an occupancy network [33] that takes as well as the query position \([u, v, w]\) in the CaDeX as input, and predicts the occupancy in the CaDeX: \( \hat{\psi} = \psi([u, v, w]; g) \), where the decoder \( \psi \) is an MLP. However, the ground truth \((u, v, w, o^*)\) supervision pair is unavailable in the CaDeX since the canonical shape is not known in advance and is learned during training. Available types of supervision are the query-occupancy pairs \([(x^i, y^i, z^i), o^*]) \) in each deformed coordinate space where the deformed surface \( S_i \) is embedded at each time \( t_i \). Therefore, we predict the occupancy field of any deformed frame through the canonical map via Eq. 6:

\[
\hat{\psi} = \psi(H([x^i, y^i, z^i]; c_t); g)
\]

### 3.5. Losses, Training, Inference

Our model is fully differentiable and is trained end-to-end. Following Eq. 10, the main loss function is the reconstruction loss in each deformed frame:

\[
\mathcal{L}_R = \frac{1}{T} \sum_{i=1}^{T} \frac{1}{M_i} \sum_{j=1}^{M_i} BCE(\psi(H(p_i^j; c_i); g), o_{ij}^*)
\]

where \( T \) is the total number of frames that have occupancy field supervision and \( M_i \) is the number of queried positions at each frame. We denote by \( \psi(p_i^j) \) the \( j \)th query position in frame \( t_i \), and by \( o_{ij}^* \) the corresponding ground truth occupancy state. Optionally, if the ground truth correspondence pairs are given, we can utilize them as a supervision signal via Eq. 7. The additional correspondence loss reads:

\[
\mathcal{L}_C = \frac{1}{|Q|} \sum_{(p_k^l, p_{l}^*) \in Q} \|H^{-1}(H(p_k^l; c_i); c_j) - p_{l}^* \|_l^2
\]

where \( Q \) is the set of ground truth correspondence pairs: \( p_k^l \) is the source position (Fig. 3 purple coordinate) in frame \( t_i \) and \( p_{l}^* \) is the ground truth corresponding position in frame \( t_j \); and \( k \) is the index of all supervision pairs. We denote by \( l \) the order of the error norm. Note that the cycle consistency guaranteed by our method (Sec. 3.3) does not depend on \( \mathcal{L}_C \). The overall loss function is \( \mathcal{L} = w_R \mathcal{L}_R + w_C \mathcal{L}_C \), where \( w_C \) can be zero if no correspondence supervision is provided. Note that there is no loss directly applied to the
predicted canonical deformation coordinates \([u, v, w]\). This gives the maximum freedom to the canonical shape to form a pattern that helps the prediction accuracy. All patterns of the canonical shape emerge automatically during training (see the Supplement for an additional discussion).

During training, our model is trained directly from scratch with the mandatory reconstruction loss (Eq. 11). For efficiency, at each training iteration, we randomly select a subset of frames in the input sequence and supervise the occupancy prediction. If the ground-truth correspondence supervision is also provided, we predict the corresponding position of surface points in the first frame for every other frame and minimize the correspondence loss in Eq. 12.

During inference, our model generates all surfaces of a sequence in parallel after a single marching cubes mesh extraction. Directly marching the CaDeX is intractable since it is learned. However, by using Eq. 10 as a query function, we can extract the mesh \((V_0, E_0)\) in the first frame, which is equivalent to marching the CaDeX given the canonical map. The equivalent canonical mesh in the CaDeX is \((V_c, E_c) = (H(V_0; c_0), E_0)\). Then any mesh in other frames can be extracted as:

\[
(V_i, E_i) = (H^{-1}(V_c; c_i), E_c).
\]

(13)

Note that all meshes above share the same connectivity \(E_0\), so the mesh correspondence is produced. Eq. 13 can be implemented in batch to achieve better efficiency.

4. Results

To demonstrate CaDeX as a general and expressive representation, we investigate the performance in modeling three distinct categories: human bodies (Sec. 4.1), animals (Sec. 4.2) and articulated objects (Sec. 4.3). Finally, we examine the effectiveness of our design choice in Sec. 4.4.

Metrics: To measure our performance for shape and correspondence modeling, we follow the paradigm of [36, 50] and use the same metrics: evaluating the reconstruction accuracy using the IoU and Chamfer Distance, and the motion accuracy by correspondence \(l_2\)-distance error.

Baselines: We compare with the closest model-free dynamic representations. The main baselines described in Sec. 2 are: O-Flow [36] and LPDC [50] for sequence inputs and A-SDF [35] for articulated object set inputs.

4.1. Modeling Dynamic Human Bodies

We first demonstrate the power of modeling the dynamic human body across time. We use the same experiment setup, dataset, and split as [24,36,50]. The data are generated from D-FAUST [5], a real 4D human scan dataset. Following the setting of [36], the input is a randomly sampled sparse point cloud trajectory (300 points) of 17 frames evenly sampled across time. The ground-truth occupancy field as well as the optional surface point correspondence are provided. Our default model is configured by using the ST-encoder (Sec. 3.2) and an NVP homeomorphism. The following tables and sections assume such a configuration if not otherwise specified. We test the performance of the PF-encoder and the NICE homeomorphism variants as well. The experiments are divided into training without correspondence (Tab. 1) and training with correspondence (Tab. 2) tracks for fair comparison between methods. The testing set has two difficulty levels: unseen motion and unseen individuals [36].

Quantitative comparisons in Tab. 1, 2 indicate that our method outperforms state-of-the-art methods by a significant margin. The qualitative comparison in Fig. 4 shows the advantage of our method in capturing fast moving parts and shape details (marked with red). We attribute such improvements to two main reasons: First, our factorization of the deformation and its implementation provide a strong regularization that other approaches like O-flow [36] can only achieve with an ODE integration. Additionally, supervising a per-frame implicit reconstruction in our model is equivalent to the dense cross-frame reconstruction supervision in [50]. Second, our shape prior is stored in the learned canonical space (marked green in Fig. 4), which is relatively stable across different sequences as shown in the figure. When training without correspondence (Tab. 1), our method can learn the correspondence implicitly and reach a similar reconstruction performance as [50] in Tab. 2, which is trained with dense parallel correspondence supervision. Comparing the different configurations of our method in Tab. 2, the NICE [14] version has a performance drop since the deformation is strongly regularized to conserve the volume, but
Figure 4. Left: Human body modeling (Sec. 4.1); Right: Animal body modeling (Sec. 4.2). The left top figure marked in the green box is our canonical shape, the first input is not displayed. The colors of the meshes encode the correspondence. More results are in the the Supp.

<table>
<thead>
<tr>
<th>Input Method</th>
<th>Seen individual</th>
<th>Unseen individual</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IoU↑</td>
<td>CD↓</td>
</tr>
<tr>
<td>O-Flow [36]</td>
<td>70.6%</td>
<td>0.104</td>
</tr>
<tr>
<td>PCL</td>
<td>72.4%</td>
<td>0.085</td>
</tr>
<tr>
<td>Ours</td>
<td>80.3%</td>
<td>0.061</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Input Method</th>
<th>Seen individual</th>
<th>Unseen individual</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IoU↑</td>
<td>CD↓</td>
</tr>
<tr>
<td>O-Flow [36]</td>
<td>63.0%</td>
<td>0.131</td>
</tr>
<tr>
<td>LPDC [50]</td>
<td>58.4%</td>
<td>0.160</td>
</tr>
<tr>
<td>Ours</td>
<td>71.1%</td>
<td>0.094</td>
</tr>
</tbody>
</table>

Table 3. Results on DeformingThings4D [57] animal bodies. PCL and Dep correspond to the input types. This is achieved by freezing half of the capacity (scale freedom). We note that the naive per-frame encoder (PF) works better compared to the spatial-temporal encoder [50] (ST). A potential reason is that the per-frame encoder provides a higher canonicalization level since when deciding the deformation code \( c_i \), no information from other frames can be considered, so the PF encoder might avoid overfitting.

4.2. Modeling Dynamic Animals

We experiment with a more challenging setting: modeling different categories of animals with one model. We generate the same supervision types as Sec. 4.1 based on the DeformingThings4D-Animals [57] dataset (DT4D-A). We use 17 animal categories and generate 2 types of input observations: Sparse point cloud input as Sec. 4.1 as well as the monocular depth video input from a randomly posed static camera. We assume that the camera viewpoint estimation problem is solved so all partial observations live in one global world frame. All models are trained across all animal categories. We refer the reader to the supplementary material for more details. Such a setting is more challenging because animals have both large shape and motion variance across categories. Additionally, the models are required to aggregate information across time and hallucinate the missing parts in depth observation inputs. Quantitative results of both the sparse point cloud input and the depth input in Tab. 3 as well as the qualitative results in Fig. 4 indicate that our method outperforms state-of-the-art methods in these challenging settings. In addition to the reasons mentioned in Sec. 4.1, the improvement when predicting from depth observation can be attributed to our design of the canonical observation encoder (Sec. 3.4) that explicitly aggregates observations in the CaDeX.

4.3. Modeling Articulated Objects

We extend CaDeX from modeling the 4D nonrigid surface sequence to representing semi-nonrigid articulated object sets. We generate the dataset and inputs as in Sec. 4.2 from [35] based on Shape2Motion [53], which contains 7 categories of articulated objects with 1 or 2 deformable angles. We configure the model with the SET encoder (Sec. 3.2) that produces the global dynamic code and then use the articulation angle to query the deformation code for each frame (for details, see the Supplement). During training, we input the sparse point cloud of 4 randomly sampled deformed frames of one object and then use the ground-truth angle to query the deformation code for each frame (for details, see the Supplement). During training, we input the sparse point cloud of 4 randomly sampled deformed frames of one object and then use the ground-truth angle to query per-frame deformation codes; finally, the model predicts the occupancy field for 4 seen (input) frames and 4 unseen frames. We supervise both \( \mathcal{L}_R \) and \( \mathcal{L}_C \). For completeness, we also predict the articulation angles of the input frames by a small head in the encoder and supervise them. Each category is trained separately for all methods. Note that A-SDF [35] demonstrates the auto-decoder setup, but it only solves half of our problem without correspondence. Simultaneously solving the shape and the correspondence leads to difficulties when applying an auto-decoder with optimization during testing, so we leave this as a future direction. For fair comparison, we adapt A-SDF.
Figure 5. Articulated objects modeling (Sec. 4.3) with 7 distinct categories. The left-top figure marked in the green box is our canonical shape, the four small figures next to it are the inputs. The first row is the reconstruction of an observed deformation angle and the second is for an unobserved angle. Note that A-SDF has no correspondence so is not colored.

with a similar encoder as our model and adapt the decoder to predict the occupancy. We also compare with LPDC [50] which is also adapted to use a similar encoder as ours.

Tab. 4 summarizes the average performance across 7 object categories while Fig. 5 presents the qualitative comparison. Both of them show our state-of-the-art performance on modeling general articulated objects. We produce an accurate reconstruction, while providing the correspondence prediction that A-SDF [35] can not predict. Thus, the marching cube is needed for each frame in [35] and results in a longer inference time as shown in Tab. 4. Note that our method preserves the topology when deforming the objects (Fig. 5 oven) while [35] does not have such guarantees. This is the main reason that our method has a performance drop on eyeglasses category since the dataset contains many unrealistic deformations where the legs of the eyeglasses get crossed. Additionally, our method models more details in the moving parts (e.g. the inner side of the refrigerator door in Fig. 5) due to the learned canonical space, which provides a stable container for the shape prior.

4.4. Ablation Study

We show the effectiveness of our design as the following: First, we replace the invertible canonical map with a one-way MLP that maps the deformed coordinates to the canonical space (such setting is similar to [13, 58, 59]). Since the mapping is one-way, we supervise the correspondence by enforcing the consistency in the canonical space. Every frame needs a separate application of marching cubes to extract meshes in this version. Second, we remove the geometry encoder in the canonical space and obtain the global geometry embedding via a latent fusion using the ST-encoder [50]. We demonstrate the performance of the deer subcategory from DT4D-A [57] with point cloud inputs (Sec. 4.2). Tab. 5 shows the performance difference, where we observe a significant performance decrease as well as longer inference times when using MLP instead of homeomorphisms. Additionally, we observe the drop in reconstruction accuracy when removing the geometry encoder in the canonical space (Sec. 3.4). We present more details in the supplementary material.

5. Limitations

Our method guarantees several desirable properties and achieves state-of-the-art performance on a wide range of shapes, but still has limitations that need future exploration. Although we can produce continuous deformation across time if \( c(t) \) is continuous, the continuity of \( c \) is not guaranteed in the ST-encoder [50] that we use. Therefore, when the input undergoes a large discontinuity, we do observe a trembling in the output of both LPDC [50] and our method. Another issue is that although our method preserves the topology, sometimes the real world deformation also results in topology changes. Future work can explore how to selectively preserve or alter the topology. Finally, it is currently nontrivial to adapt our method in an auto-decoder framework [35, 40] since it requires simultaneously optimizing the canonical map (deformation) and the canonical shape during testing, which future work can explore.

6. Conclusion

We introduced a novel and general representation for dynamic surface reconstruction and correspondence. Our key insight is the factorization of the deformation by continuous bijective canonical maps through a learned canonical shape. We prove that our representation guarantees cycle consistency and topology preservation, as well as (if desired) volume conservation. Extensive experiments on reconstructing humans, animals, and articulated objects demonstrate the effectiveness and versatility of our approach. We believe that CaDeX enables more possibilities for future research on modeling and learning from our dynamic real world.

Acknowledgement: The authors appreciate the support of the following grants: ARL MURI W911NF-20-1-0080, NSF TRIPODS 1934960, NSF CPS 2038873, ARL DCIST CRA W911NF-17-2-0181, and ONR N00014-17-1-2093.
References


