3D Implicit Transporter for Temporally Consistent Keypoint Discovery

Chengliang Zhong[1,2], Yuhang Zheng[3], Yupeng Zheng[4], Hao Zhao[2*], Li Yi[5], Xiaodong Mu[1],
Ling Wang[1], Pengfei Li[2], Guyue Zhou[2], Chao Yang[6], Xinliang Zhang[2], Jian Zhao[7,8,9]

[1]Xi'an Research Institute of High-Tech, [2]AIR, Tsinghua University, [3]Beihang University,
[4]Institute of Automation, Chinese Academy of Sciences, [5]IIIS, Tsinghua University,
[6]Shanghai AI Laboratory, [7]Institute of North Electronic Equipment,
[8]Intelligent Game and Decision Laboratory, [9]Peng Cheng Laboratory

zhongcl19@mails.tsinghua.edu.cn, zhaohao@air.tsinghua.edu.cn

Abstract

Keypoint-based representation has proven advantageous in various visual and robotic tasks. However, the existing 2D and 3D methods for detecting keypoints mainly rely on geometric consistency to achieve spatial alignment, neglecting temporal consistency. To address this issue, the Transporter method was introduced for 2D data, which reconstructs the target frame from the source frame to incorporate both spatial and temporal information. However, the direct application of the Transporter to 3D point clouds is infeasible due to their structural differences from 2D images. Thus, we propose the first 3D version of the Transporter, which leverages hybrid 3D representation, cross attention, and implicit reconstruction. We apply this new learning system on 3D articulated objects and non-rigid animals (humans and rodents) and show that learned keypoints are spatio-temporally consistent. Additionally, we propose a closed-loop control strategy that utilizes the learned keypoints for 3D object manipulation and demonstrate its superior performance. Codes are available at https://github.com/zhongcl-thu/3D-Implicit-Transporter.

1. Introduction

The ability to establish correspondences in temporal inputs is a hallmark of the human visual system, and this ability has been verified by developmental biologists [52] as the enabling factor of object perception. Specifically, infants can naturally separate different objects by considering pixels that move together. Meanwhile, establishing dense correspondences from image sequences (i.e., optical flow) [5, 6, 9, 13, 54, 55, 53] is also one of the oldest computer vision topics, dating back to the birth of this discipline [19].
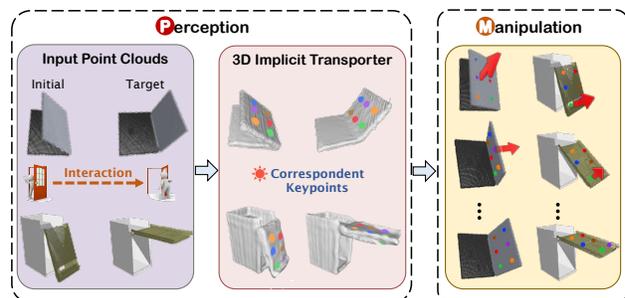

Figure 1. Given paired point clouds, our 3D Implicit Transporter leverages the object/part motion to discover temporally consistent keypoints and recover the underlying shape for each input. Moreover, the learned keypoints can serve downstream robotics tasks, such as articulated object manipulation.

On the other hand, keypoints are preferred as a compact mid-level representation in many scenarios, like visual recognition [51], pose estimation[72], reconstruction [37], and robotic manipulation [43]. Sparse correspondence from keypoints [30, 7, 10, 45] is another fundamental vision topic. Most of the 2D and 3D keypoint detection methods [73, 25] depend on the consistency under geometric transformations to achieve spatial alignment of keypoints. However, these methods are limited in their ability to identify temporally consistent keypoints, which is crucial for representing movable or deformable objects such as the human body whose shape and topology may vary over time. So does there exist a generic principle that reflects how humans extract spatiotemporally consistent keypoints? One (possible) such principle is that good mid-level representation can be used to re-synthesize raw visual inputs. This principle has been explored in legacy methods like FRAME [77], but limited by the modeling power of generative models at that age, and this principle has not seen much success.

Recently, a method named Transporter [23] has been proposed in the 2D domain, successfully connecting keypoint extraction and correspondence establishment in a self-

---

*Corresponding author

supervised manner, using exactly the aforementioned principle. Thanks to the strong power of modern image reconstruction networks, this method can extract meaningful keypoints from image sequences without using any human annotation. Transporter is both a useful tool and an elegant formulation that (potentially) mimics how the human visual system extracts keypoints. However, to the best of our knowledge, this methodology has not been translated into the 3D domain. Because the process of 2D feature transportation is implemented on regular data formats, such as 2D image grids, it is not applicable to point clouds that allow for non-uniform point spacing. Another reason we believe is that 3D reconstruction from keypoints is a more challenging setting.

As such, we propose the first 3D Transporter in the literature, based upon three core components: hybrid 3D representation architectures for 3D feature transportation, cross-attention for better keypoint discovery, and an implicit geometry decoder for 3D reconstruction. Our method takes as input two point clouds containing moving objects or object parts (Fig. 1 left panel). Then by watching these two states solely, our 3D Implicit Transporter extracts temporally consistent keypoints and a surface occupancy field for each state, in a self-supervised manner (Fig. 1 middle panel). The method reconstructs the shape of the target state by transporting explicit feature grids from the initial state according to the locations of detected keypoints. Via extensive evaluations on the PartNet-Mobility dataset [65] and ITOP dataset [18], we demonstrate significantly improved perception performance in terms of spatiotemporal consistency of keypoints over state-of-the-art counterparts. The qualitative results on the Rodent3D[40] dataset also show our keypoints consistently capture the rodent's skeletal structure.

Besides, we also explore how well our self-supervised mid-level representation (3D keypoints) can serve downstream robotic applications (Fig. 1 right panel). We choose articulated object manipulation, which requires sophisticated 3D reasoning about the kinematic structure and part movement, as the benchmark. Existing methods often rely on object-agnostic affordance-based representation [67, 35, 62]. Our 3D keypoint representation is also object-agnostic, but we demonstrate that our approach offers two distinct advantages over these methods: 1) the efficient learning formulation of ours does not involve costly trial-and-error interaction in simulators; 2) we leverage spatio-temporally aligned 3D keypoints to provide a structured understanding of objects, enabling the design of an effective closed-loop manipulation strategy.

To summarize, we have the following contributions:

- We propose the first 3D Implicit Transporter formulation that extracts 3D correspondent keypoints from temporal point cloud inputs, using 3D feature grid

transportation, attentional keypoint detection, and target shape reconstruction.

- Based on the extracted 3D keypoint representation, we build a closed-loop manipulation strategy and demonstrate it successfully addresses the manipulation of many articulated objects in an object-agnostic setting.

- We extensively benchmark the perception and manipulation performance of 3D Implicit Transporter and report state-of-the-art results on public benchmarks.

2. Related Work

2.1. 3D Keypoint Detection

Detecting 3D keypoints from point clouds has drawn a lot of attention in vision and robotics [44, 57, 4, 25, 70, 76]. Traditional hand-crafted methods predict salient points according to local geometric statistics of inputs, such as density [75] and curvature [24]. Modern learning-based approaches employ the consistency of keypoint coordinates [25] or saliency scores [74] under rigid transformations to formulate keypoint detection as a self-supervised task. However, they are unable to ensure temporally consistent keypoint detection of non-rigid objects whose shapes are significantly changed after movement. To achieve that, recent research has explored discovering temporally aligned keypoints from given image videos. Most of them [34, 56, 17, 11, 23, 60] consider the keypoint learning problem as a signal reconstruction process. For example, Minderer *et al.* [34] and Jennifer [56] proposed to reconstruct a future frame by using features of the current frame and future keypoints. Despite their better results, these methods all concentrate on 2D keypoint discovery. As far as we know, there is seldom work investigating the task of 3D temporally consistent keypoint detection.

2.2. Neural Implicit Representation

Recently, multiple works [29, 33, 48, 26, 39, 50] have focused on implicit geometric representation. It intends to parameterize a signal as a continuous function by a neural network that could decode complex shape topologies of discrete inputs [39, 41] It shows great achievements of the implicit neural function on the grasp pose generation [21], articulated model estimation [20, 36], and object pose representation [49]. More recently, it has been proved that entangling an implicit shape decoder [74] instead of a coordinate decoder [69] encourages the model to predict more semantically consistent keypoints. Inspired by these works, we exploit the implicit occupancy function to reconstruct the underlying shape of the transported 3D objects.

2.3. Perception and Manipulation of Articulated Objects

Previous research has explored various techniques to understand and represent articulated objects, such as kinematic graphs [22, 1], 6D pose estimation [27, 63], part segmentation [61, 65, 68], deformation flow [68], joint parameters [20, 38] and so on. However, most of them require ground truth knowledge of the object or are category-dependent. Unlike them, we use sparse correspondence from keypoints to capture the critical part of articulated objects without human labels, which can also generalize to unseen categories. Albeit the fruitful progress in vision, one could not directly infer an action from perceptual outputs to manipulate articulated objects. Therefore, recent works have proposed manipulation-centric visual representations, like visual affordance [35, 67, 64] or dense articulation flow [14]. However, they need substantial trial-and-error interactions in simulation or ground-truth geometric knowledge. In contrast, the keypoint learning of ours is unsupervised and effective, and the correspondences between keypoints can also serve robotic manipulation well.

3. Method

Our perception method presents a new formulation to discover temporally and spatially consistent 3D keypoints of a moving object or object part in a sequence of point clouds in a self-supervised manner. Once trained, the learned keypoints are used to devise a strategy for the manipulation of an articulated object from its starting state to a goal state, thus avoiding the costly trial-and-error interaction typically utilized in [67].

3.1. Temporal 3D Keypoint Discovery

In accordance with the formulation in [23], we consider a dataset comprising pairs of frames extracted from a series of trajectories, where each frame is represented as a 3D point cloud instead of an image. Two frames in a pair are distinguished by differences solely in the pose/geometry of the objects. Our goal is to find the corresponding keypoints describing the object or object part motion from the source to the target frame. We tackle this problem by reconstructing the underlying shape of the target frame from the source frame. Fig. 2 summarizes our method, and the subsequent subsections provide further details on its components.

3.1.1 3D Feature Transporter

Hybrid 3D Representation. As per the formulation in 2D Transporter, feature transportation occurs between uniform data, such as 2D images, which is not feasible for point clouds that are in an irregular format. A straight way is to convert point clouds to uniform 3D voxel grids before inputting them to a neural network. However, converting raw

point clouds into voxels inevitably introduces quantization errors that break the intrinsic geometric patterns (e.g., isometry) of 3D data. Although a high-resolution volumetric representation could compensate this information loss, both computational cost and memory requirements escalate cubically with voxel resolution. On the contrary, point-based models [15, 42] lead to a significant reduction in memory usage due to the sparse representation. As such, we utilize the point-based backbone to extract local features from sparse points and then leverage the voxel-based model for transporting local features.

Given a frame $\mathbf{o} \in \mathbb{R}^{N_1 \times 3}$, where N_1 is the number of input points, we exploit a PointNet [42] P to get point features $P(\mathbf{o}) \in \mathbb{R}^{N_1 \times C_1}$, where C_1 is the dimension of features. These features are then locally pooled and projected into structured volumes $\mathbf{v} \in \mathbb{R}^{C_2 \times C_h \times C_w \times C_d}$, where C_h, C_w and C_d are the number of voxels in three orthogonal axes. Afterwards, the feature volumes are processed with a 3D UNet [12] U, resulting in the outputs $U(\mathbf{v}) \in \mathbb{R}^{C_3 \times C_h \times C_w \times C_d}$. The above point and voxel-based models are denoted as the feature encoder Φ in Fig. 2.

Attentional Keypoint Detection. When we are asked to find moving objects or object parts between paired frames, we adopt an iterative process whereby we inspect and sift through multiple tentative regions in both frames. However, extracting keypoints on mobile parts using a single frame, as done by 2D Transporter[1], can be inherently ambiguous, particularly when multiple potential mobile parts exist in each frame. Therefore, inspired by [47, 20, 28, 71], we propose to use a cross-attention module to aggregate geometric features from both frames to locate keypoints.

Specifically, we utilize a point-based model (not shared with Φ) to extract multi-level features for the input point clouds and correlate paired inputs at a coarse level to reduce computational costs, as done in [20]. Given a frame pair $\mathbf{o}_s, \mathbf{o}_t$, we exploit a shared PointNet++ [42] $\hat{P}$ to get two down-sampled point features $\mathbf{f}_s = \hat{P}(\mathbf{o}_s)$ and $\mathbf{f}_t = \hat{P}(\mathbf{o}_t)$, where $\mathbf{f}_s, \mathbf{f}_t \in \mathbb{R}^{N_2 \times C_4}$. Then, a cross attention block [59] is used to mix point features of paired inputs, achieved by:

$$\mathbf{z}_s = \text{softmax}(\frac{\mathbf{f}_s \mathbf{f}_t^T}{\sqrt{C_4}})\mathbf{f}_s, \qquad \mathbf{f}_s' = [\mathbf{f}_s, \mathbf{z}_s], \qquad (1)$$

$$\mathbf{z}_t = \text{softmax}(\frac{\mathbf{f}_t \mathbf{f}_s^T}{\sqrt{C_4}})\mathbf{f}_t, \qquad \mathbf{f}_t' = [\mathbf{f}_t, \mathbf{z}_t]. \qquad (2)$$

The output of this block is the concatenation of input features and attended features. Then, we upsample $\mathbf{f}_s', \mathbf{f}_t'$ to get dense features $\mathbf{f}_s^d, \mathbf{f}_t^d \in \mathbb{R}^{N_1 \times C_5}$ using a PointNet++ decoder. Following the principle in the above, we convert these dense point features into the keypoint saliency volumes by projection and a 3D UNet U_k. Suppose the full detection module is denoted as Ψ, the outputs are named as $\Psi(\mathbf{o}_s, \mathbf{o}_t)_s, \Psi(\mathbf{o}_s, \mathbf{o}_t)_t \in \mathbb{R}^{m \times C_h \times C_w \times C_d}$. Then, we

---

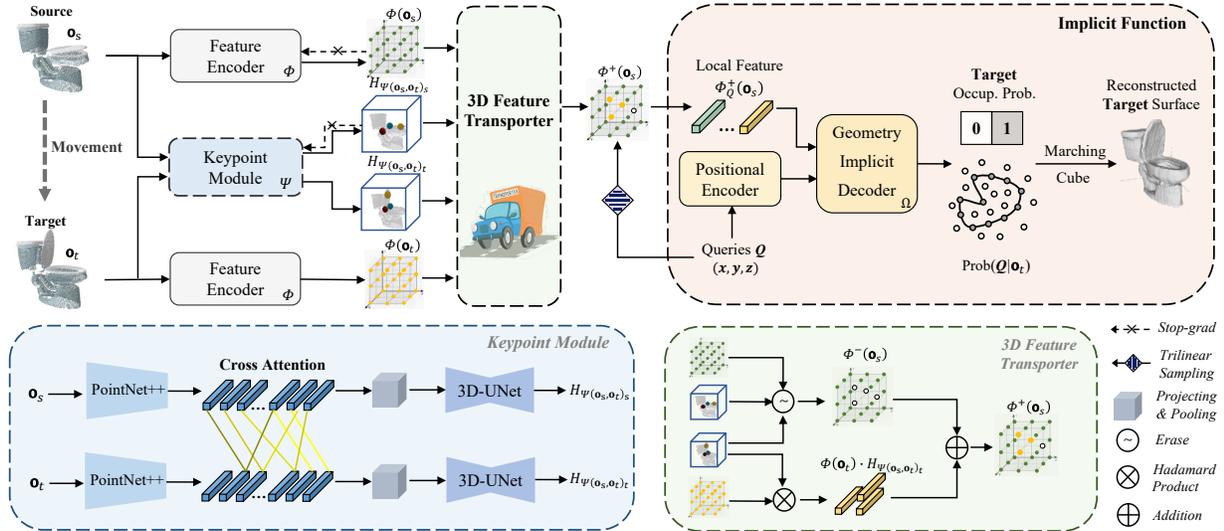[1]The reconstruction objective requires two frames.

Figure 2. **Architecture of our 3D Implicit Transporter.** The network consists of (1) a feature encoder Φ that extracts features for transportation; (2) a keypoint module Ψ to indicate where to transport; (3) a 3D feature transporter that reconstruct the feature volume of interest; and (4) a geometry implicit decoder that allows self-supervision using solely input point clouds. The overall process involves extracting m corresponding 3D keypoints from two frames and using them to transport the features of the target frame to the source frame based on their keypoint locations. The transported features are then fed into the implicit decoder to reconstruct the target shape.

can marginalize (see section 3.1 of [23]) the saliency volumes along three orthogonal axes to extract m 3D keypoints $\mathbf{k}_s, \mathbf{k}_t \in \mathbb{R}^{m \times 3}$, as visualized in the blue panel in Fig. 2. Here, the i-th keypoint in $\mathbf{k}_s$ and $\mathbf{k}_t$ correspond to each other ($i \in [1, m]$).

Feature Transportation. Similar to 2D Transporter, the next step involves feature transportation for reconstructing $\mathbf{o}_t$ from $\mathbf{o}_s$. We transport the features in $\Phi(\mathbf{o}_t)$ around $\mathbf{k}_t$ into $\Phi(\mathbf{o}_s)$ and suppress features in $\Phi(\mathbf{o}_s)$ around $\mathbf{k}_t$ and $\mathbf{k}_s$. As shown in the green panel in Fig. 2, we first erase features at both sets of keypoints in $\Phi(\mathbf{o}_s)$ to get $\Phi^-(\mathbf{o}_s)$, then extract the feature surrounding keypoints $\mathbf{k}_t$ from $\mathbf{o}_t$, and finally combine both to generate $\Phi^+(\mathbf{o}_s)$, which is formulated by:

$$\Phi^+(\mathbf{o}_s) = (1 - H_{\Psi(\mathbf{o}_s, \mathbf{o}_t)_s}) \cdot (1 - H_{\Psi(\mathbf{o}_s, \mathbf{o}_t)_t}) \cdot \Phi(\mathbf{o}_s) + H_{\Psi(\mathbf{o}_s, \mathbf{o}_t)_t} \cdot \Phi(\mathbf{o}_t), \quad (3)$$

where H_Ψ represents a 3D heatmap made out of fixed-variance σ isotropic Gaussians centered at each of the m keypoint coordinates indicated by Ψ, and $(1 - H_{\Psi(\mathbf{o}_s, \mathbf{o}_t)_s}) \cdot (1 - H_{\Psi(\mathbf{o}_s, \mathbf{o}_t)_t}) \cdot \Phi(\mathbf{o}_s)$ is denoted as $\Phi^-(\mathbf{o}_s)$ in Fig. 2.

It's worth noting that the exact values of feature dimensions in our experiments are detailed in the supplementary.

3.1.2 Geometry Implicit Decoder

Since the geometry except for moving parts remains the same between the source and target frame, building transported features using detected corresponding keypoints of moving parts can enable the re-synthesis of the target visual inputs. As the 2D Transporter does not change the data

structure after transportation, it is easily achievable to reconstruct the input image by a CNN-based decoder. This is not viable for irregular 3D data, so our 3D Transporter utilizes implicit neural representations to reconstruct the underlying shape of the target instead of the raw point clouds. This is motivated by recent studies demonstrating the effectiveness of deep implicit functions for 3D reconstruction. By mapping the irregular point clouds into volumetric features, we find that using implicit shape decoding is more effective compared to sparse reconstruction (see Tab. 3).

Given a point $\mathbf{q} \in \mathbb{R}^3$ from a query set Q, our method encodes it into a C_e-dimensional vector $\mathbf{q}_e$ using a multi-layer perceptron. Then, the local feature $\Phi_{\mathbf{q}}^+(\mathbf{o}_s)$ is queried from the transported feature volume $\Phi^+(\mathbf{o}_s)$ via trilinear interpolation. Our implicit decoder Ω maps the concatenation of feature $\mathbf{q}_e$ and $\Phi_{\mathbf{q}}^+(\mathbf{o}_s)$ to a target surface occupancy probability $\text{Prob}(\mathbf{q}|\mathbf{o}_t) \in [0, 1]$, as formulated by:

$$\Omega(\mathbf{q}_e, \Phi_{\mathbf{q}}^+(\mathbf{o}_s)) \rightarrow \text{Prob}(\mathbf{q}|\mathbf{o}_t). \quad (4)$$

3.1.3 Loss Function

All modules could be optimized by a surface reconstruction loss. As we claim that we have no access to any information other than the given videos, we solely use the input point clouds for training the implicit decoder. Specifically, we define occupied points are those lying on the input surface, while all other points are considered unoccupied, including those inside and outside the surface.

The binary cross-entropy loss between the predicted target surface occupancy $\text{Prob}(\mathbf{q}|\mathbf{o}_t)$ and the ground-truth la-

bels of the target frame $\text{Prob}^{\text{gt}}(\mathbf{q}|\mathbf{o}_t)$ is used. If $\mathbf{q}$ is from the input target point clouds, the $\text{Prob}(\mathbf{q}|\mathbf{o}_t)$ would be 1, otherwise be 0. We randomly sample queries Q from the volume of size $C_h \times C_w \times C_d$ and the target point clouds, then average the results over all queries:

$$\mathcal{L}_{\text{occ\_t}} = \frac{1}{|Q|} \sum_{\mathbf{q} \in Q} l_{\text{BCE}}\big(\text{Prob}(\mathbf{q}|\mathbf{o}_t), \text{Prob}^{\text{gt}}(\mathbf{q}|\mathbf{o}_t)\big), \quad (5)$$

where $|Q|$ is the number of queries Q.

We also incorporate an additional loss term, $\mathcal{L}_{\text{occ\_s}}$, to aid the source frame reconstruction process by leveraging its own feature grids, $\Phi(\mathbf{o}_s)$. This loss term, formulated in the supplementary, leads to improved perception results.

3.2. Manipulation using Consistent Keypoints

The use of keypoints as a mid-level representation of objects is an appropriate way for contact-rich robotic manipulation tasks that occur within a 3D space, such as tool manipulation [43], object grasp [32], cloth folding [31] and generic visuomotor policy learning [16]. However, previous works either focus solely on 2D keypoint representation or struggle to detect temporally consistent 3D keypoints when faced with shape variations and changes in object topology. Owing to the long-term consistency of 3D Transporter keypoints, our method is well-suited for handling 3D manipulation tasks. To demonstrate that, we choose articulated object manipulation as the benchmark.

The task is formulated in UMPNET [67]: Given a goal state $\mathbf{o}_g$, a robot with an end-effector aims to generate a set of actions by which the articulated object can be moved from the current state $\mathbf{o}_c$ to $\mathbf{o}_g$, as shown in Fig. 3. In this study, each state is in the form of point clouds instead of RGB-D images as used in [67]. Here, we use a suction-based gripper that can grasp any point on the object surface as [14, 67] used.

Before manipulation, we leverage the geometry prior about articulation to design two additional losses during keypoint learning for improving the performance of 3D Transporter keypoint estimates:

Keypoint Correspondence Loss Since the predicted keypoints are expected to scatter on the mobile part, we can use them to generate the pose hypothesis of the rigid part motion between the source and target, which is given by:

$$\hat{\mathbf{R}}, \hat{\mathbf{t}} = \min_{\mathbf{R}, \mathbf{t}} \sum_{i=1}^{m} \left\| \mathbf{k}_g^i - \left(\mathbf{R} \cdot \mathbf{k}_c^i + \mathbf{t} \right) \right\|^2. \quad (6)$$

This can be computed in closed form using SVD [8]. We enforce all correspondent keypoints to meet this rigid transformation to make keypoints geometrically aligned:

$$\mathcal{L}_{\text{corr}} = \sum_{i=1}^{m} \left\| \mathbf{k}_g^i - \left(\hat{\mathbf{R}} \cdot \mathbf{k}_c^i + \hat{\mathbf{t}} \right) \right\|^2. \quad (7)$$
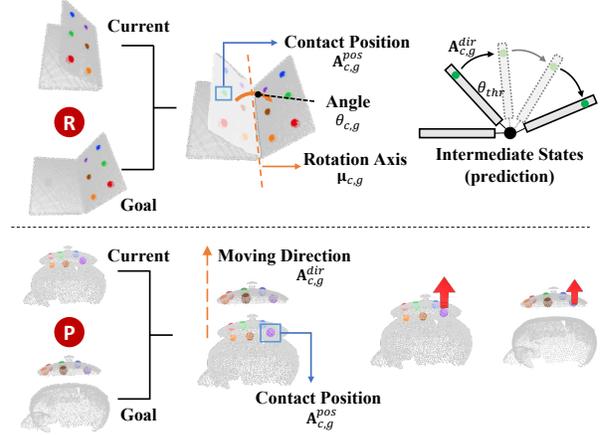

Figure 3. Illustrations of our manipulation policy based on correspondent keypoints. R means revolute and P means prismatic. The length of the red arrow is proportional to the action distance.

Joint Consistent Loss During an articulated motion, we note that the axis orientation of a certain joint under different states should remain the same or parallel. Given the predicted pose by Eq. (6), we can calculate its axis orientation $\boldsymbol{\mu}$ and angle θ, through Rodrigues' rotation formula. We penalize the difference in axis orientation of a certain joint at different time steps:

$$\mathcal{L}_{\text{axis}} = \min(1 - \boldsymbol{\mu}_{12} \boldsymbol{\mu}_{23}^T, 1 + \boldsymbol{\mu}_{12} \boldsymbol{\mu}_{23}^T), \quad (8)$$

where $\boldsymbol{\mu}_{ab}$ is the predicted axis orientation between observations at time-step a and b.

The overall training loss is:

$$\mathcal{L} = \mathcal{L}_{\text{occ\_t}} + \mathcal{L}_{\text{occ\_s}} + \lambda_1 \mathcal{L}_{\text{corr}} + \lambda_2 \mathcal{L}_{\text{axis}}, \quad (9)$$

where λ_1 and λ_2 are the loss weights (exact values are detailed in the supplementary).

After training, we develop an object-agnostic manipulation policy based on 3D Implicit Transporter keypoints, avoiding the notoriously inefficient exploration used in [67]. We first define each action $\mathbf{A}_{c,g}$, that moves the object from its current state to the goal state, as a 6-Dof pose which indicates the suction position $\mathbf{A}_{c,g}^{pos} \in \mathbb{R}^3$ and moving direction $\mathbf{A}_{c,g}^{dir} \in \mathbb{R}^3$. Then, our policy consists of two parts:

Position and Direction Inference The first step is to obtain predicted keypoints $\mathbf{k}_c, \mathbf{k}_g$, axis $\boldsymbol{\mu}_{c,g}$ and angle $\theta_{c,g}$. Then, we compute the sparse articulation flow $\mathbf{F}_{c,g}^i = \mathbf{k}_g^i - \mathbf{k}_c^i$ from correspondent keypoints. To efficiently actuate the moving part, we select the keypoint location $\mathbf{k}_c^s$ with the highest magnitude flow as the suction point according to *the principle of leverage*, denoted as $\mathbf{A}_{c,g}^{pos} = \mathbf{k}_c^s$.

For a revolute joint (Fig. 3 top panel), $\mathbf{k}_c^s$ is restricted to move on a 2D circle with a radius $\mathbf{r}^s$ perpendicular to the axis of rotation, where $\mathbf{r}^s$ is the shortest vector from $\mathbf{k}_c^s$ to the joint link. Therefore, the ideal action direction

$\mathbf{A}_{c,g}^{dir_s}$ for $\mathbf{k}_c^s$ is tangent to the circle formed by $\mathbf{r}^s$. If the range of motion between the current and goal state is small, w.r.t. $\theta_{c,g} \leq \theta_{thr}$, where θ_{thr} is a threshold, $\mathbf{F}_{c,g}^s$ is approximately parallel to $\mathbf{A}_{c,g}^{dir_s}$, which can be set as the action direction. However, when the rotation is increased, the difference between them gets more significant. This issue can be alleviated by interpolating some intermediate states according to the predicted axis. Specifically, we use the axis $\mathbf{\mu}_{c,g}$ and θ_{thr} to rotate $\mathbf{k}_c^{s'}$ (the result of moving center of $\mathbf{k}_c^s$ to origin coordinates) to generate $\mathbf{k}_{c_1}^{s'}$. Then $\mathbf{F}_{c,c_1}^{s'}$ can be computed as the action direction at $\mathbf{k}_c^s$.

For a prismatic joint (Fig. 3 bottom panel), $\mathbf{A}_{c,g}^{dir_s}$ is parallel to the ground-truth articulation flow of each moving point. Therefore, we can directly use $\mathbf{F}_{c,g}^s$ as the articulate direction. Since the motion of a prismatic joint always satisfies the condition $\theta_{c,g} \leq \theta_{thr}$, and the rule for direction estimates is the same as the revolute joint, **there is no need to classify the joint type**.

Closed-loop Manipulation In contrast to using a single-step action to reach the target, we generate a sequence of actions over multiple steps to gradually change the articulation state. To achieve this, we adopt a closed-loop control system that relies on feedback to adjust the current action. Specifically, we predict the next action based on the object's current and goal state. Unlike the previous method [67] that used a constant moving distance, we leverage the magnitude of articulation flow $\mathbf{F}_{c,g}^s$ to adjust the moving distance dynamically. Inspired by the idea of a PID controller [3], we set $||\mathbf{A}_{c,g}^{dir_s}|| = \lambda \cdot ||\mathbf{F}_{c,g}^s||$ at the current state, where λ is a proportional coefficient.

4. Experiments

We demonstrate the effectiveness of our approach in both perception and manipulation tasks. We begin with an evaluation of 3D correspondent keypoint detection methods on both synthetic and real-world datasets. Additionally, we conduct an ablation study to investigate the impact of each design choice in our approach. Next, we demonstrate our method's ability to perform goal-conditioned manipulation in simulation and further verify our approach's practicality by showcasing its performance on a real-world platform. More details regarding our implementation and hyper-parameters can be found in the supplementary.

4.1. Datasets

PartNet-Mobility [65]: We adopt a similar approach to Xu *et al.* [67] for selecting synthetic object models from PartNet-Mobility to generate our data, except for two categories with an insufficient number of instances. Thus, we train on 10 categories and test on 9 categories, with specific object classes listed in the footnote of Tab. 1. For the perception task, we load one instance at a time into the Py-
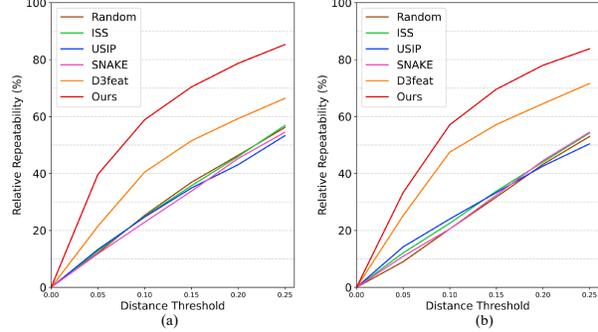

Figure 4. Relative repeatability (%) of novel instances in train categories (a) and test categories (b). Our method significantly surpasses other baselines. Note that our counterparts need a segmentation mask to filter the keypoints on the mobile part.

bullet simulator with random pose and joint configurations, and gradually change the articulation states of a randomly selected joint to mimic human-object interaction. For the manipulation task, we use the same joint configurations of testing objects at the initial and goal states as [67] set. At each state in both tasks, we integrate three-view rendered depth images into point clouds. Examples of generated data are visualized in the supplementary.

ITOP [18]: The ITOP dataset, which consists of depth map sequences capturing diverse real human actions, is used in our perception task. Specifically, we select 9k training frames and 1k testing frames from the dataset.

Rodent3D [40]: The Rodent3D dataset contains 240 minutes of multimodal (RGB, depth, and thermal) video recordings depicting rodents exploring an arena in a laboratory. The dataset is leveraged to develop a model aimed at accurately tracking the 3D pose of animals.

4.2. Baselines

For the perception task, we compare our method with several 3D keypoint detection methods, which contains random guess, hand-crafted detectors: ISS [75], and deep learning-based unsupervised detectors: USIP [25] and SNAKE [74]. To find correspondence between keypoints of two observations, we need both keypoint detectors and descriptors. As such, we use an off-the-shelf and generic descriptor FPFH [46] with the abovementioned keypoint detectors. Moreover, the matching is based on the nearest neighbor search. We also choose a joint learning method for 3D keypoint detection and description: D3feat [4]. All the baselines are pre-trained on our dataset.

For the manipulation task, UMPNet [67] is selected as a strong baseline, which proposed to use a universal strategy to handle various objects for goal-conditioned manipulation. We also compare with a single-step action model proposed by Agrawal *et al.* [2].

Table 1. The perception performance of ours and other baselines.

	Novel Instances in Train Categories											Test Categories									
	fridge	folding chair	laptop	stapler	trashcan	microwave	toilet	window	cabinet	kettle	Avg.	box	phone	dishwasher	safe	oven	washing machine	table	kitchen pot	door	Avg.
Average correspondent keypoint distance (ACKD) ↓																					
Random	0.24	0.26	0.34	0.31	0.27	0.24	0.21	0.28	0.24	0.13	0.25	0.32	0.30	0.32	0.25	0.25	0.18	**0.21**	0.24	0.31	0.26
ISS[75]	0.26	0.24	0.31	0.28	0.26	0.24	0.20	0.25	0.25	0.14	0.24	0.31	0.29	0.33	0.23	0.26	0.19	0.22	0.20	0.30	0.26
USIP[25]	0.23	0.28	0.34	0.36	0.26	0.23	0.25	0.24	**0.23**	0.13	0.26	0.33	0.34	0.38	0.19	0.29	0.21	0.23	0.21	0.26	0.27
SNAKE[74]	0.25	0.26	0.31	0.29	0.25	0.28	0.21	0.27	0.25	0.16	0.25	0.29	0.31	0.31	0.24	0.24	0.18	0.22	0.26	0.29	0.26
D3feat[4]	**0.20**	0.21	0.22	0.21	0.25	0.23	**0.11**	0.24	0.25	0.14	0.21	0.19	0.22	0.24	0.17	0.20	0.13	**0.21**	0.16	0.19	0.19
Ours	0.21	**0.07**	**0.07**	0.11	0.15	0.11	0.11	0.14	0.23	**0.06**	**0.13**	0.13	**0.06**	0.12	0.15	0.18	0.12	0.22	**0.07**	0.13	**0.13**
Average distance on pose estimation (ADD) ↓																					
Random	0.26	0.26	0.31	0.34	0.27	0.27	0.21	0.28	0.26	0.18	0.26	0.31	0.29	0.34	0.28	0.26	0.18	0.22	0.32	0.33	0.28
ISS[75]	0.27	0.24	0.32	0.30	0.26	0.28	0.22	0.23	0.26	0.18	0.26	0.29	0.30	0.35	0.24	0.26	0.19	0.22	0.30	0.33	0.27
USIP[25]	0.28	0.25	0.33	0.38	0.26	0.27	0.23	0.27	0.27	0.17	0.27	0.29	0.33	0.38	0.25	0.28	0.24	0.24	0.30	0.29	0.29
SNAKE[74]	0.26	0.25	0.33	0.33	0.28	0.29	0.24	0.24	0.26	0.19	0.27	0.29	0.32	0.37	0.24	0.26	0.21	**0.21**	0.37	0.29	0.29
D3feat[4]	0.21	0.22	0.26	0.24	0.27	0.27	0.16	0.24	0.28	0.19	0.23	0.21	0.26	0.34	0.19	0.27	0.16	**0.21**	0.21	0.18	0.23
Ours	**0.18**	**0.08**	**0.07**	**0.10**	**0.16**	**0.11**	**0.11**	**0.13**	0.21	**0.07**	**0.12**	**0.14**	**0.06**	**0.11**	0.15	**0.15**	**0.12**	0.24	**0.07**	**0.12**	**0.13**

Classes: fridge, folding chair, laptop, stapler, trashcan, microwave, toilet, window, cabinet, kettle, box, phone, dishwasher, safe, oven, washing machine, table, kitchen pot, door.
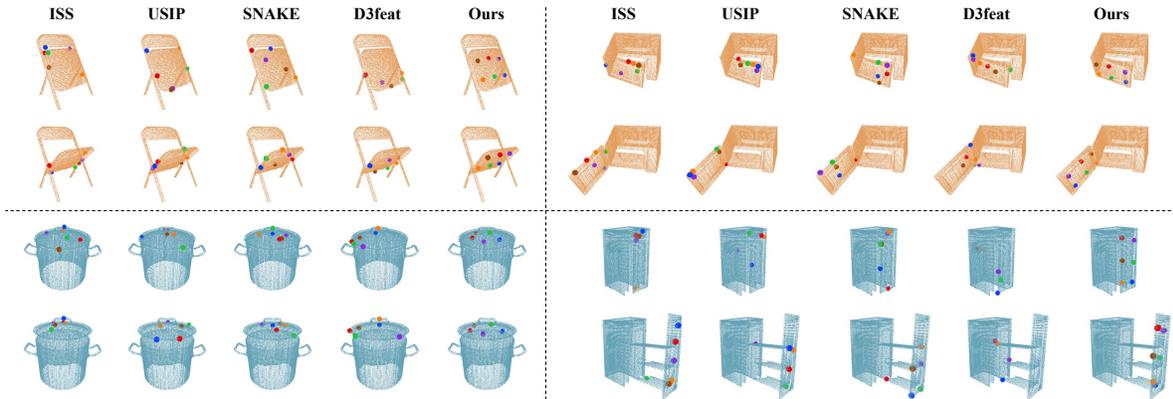

Figure 5. Temporal consistency of keypoints on articulated objects in different articulation states. The object in yellow has a revolute joint, and the blue one has a prismatic joint. The keypoint in the same color is correspondent.

Table 2. Results on the ITOP dataset under two settings: w/ and w/o a mask to filter keypoints of the human body at the test.

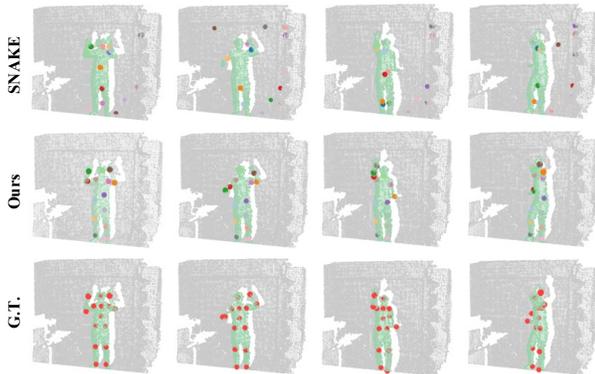|	Random	ISS	USIP	SNAKE	D3feat	Ours
w/ mask	0.53	0.29	0.23	0.27	0.22	**0.13**
w/o mask	0.74	0.89	0.87	0.69	0.78	**0.14**

Figure 6. Keypoint consistency on an articulated human.

4.3. Correspondent Keypoint Detection

We compare the keypoint consistency of two different articulation states of the same instance. We mainly focus on keypoints of the moving part to demonstrate the tempo-
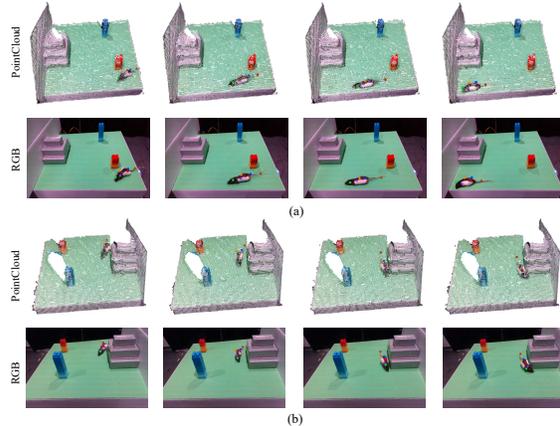

Figure 7. Keypoint predictions of ours on the Rodent3D dataset.

ral consistency. Note that we need to use the ground-truth segmentation mask to select keypoints to meet the above requirement for all baselines. For a fair comparison, we set a fixed keypoint number m of 6 for each method and ours.

Metrics We exploit the following metrics for evaluation: 1) *Average correspondent keypoint distance* (ACKD): CKD is the Euclidean distance of correspondent keypoints in the same coordinate system. ACKD is the average CKD of all
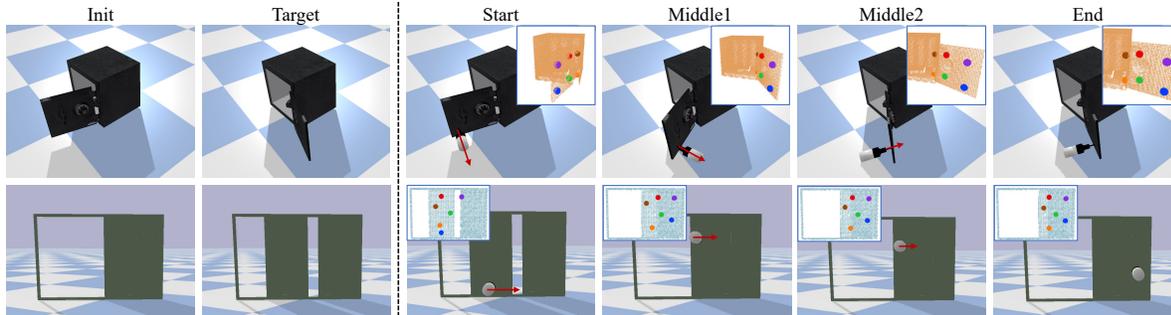
Figure 8. Visualizations of the goal-conditioned task, and the perception and manipulation results of our method in each step.

keypoints. 2) *Relative repeatability* (RR): A keypoint is repeatable if its CKD is below a distance threshold. RR is the percentage of repeatable points in a total number of detected keypoints. 3) *Average distance on pose estimation* (ADD): According to Eq. (6), the part motion of an articulated object can be solved. Following PoseCNN [66], we adopt ADD to compare the distance between predicted pose $[\hat{\mathbf{R}}, \hat{\mathbf{t}}]$ and ground-truth pose $[\mathbf{R}, \mathbf{t}]$.

Evaluation Results The quantitative results on the *PartNet-Mobility* dataset are provided in Tab. 1 and Fig. 4. Our method demonstrates superior performance on both novel instances within the training set and test categories. Baselines perform poorly on temporal keypoint detection. The reason may be twofold. 1) Their core principle is to find repeatable keypoints under view variations, which makes sense only for rigid objects. And they did not design a strategy to adapt to significant shape disturbances of the same object. 2) The bottleneck is the mismatch between keypoint detector and descriptor. The performance gap between the D3feat and other baselines is the apparent evidence to verify that. Owing to the design of the 3D feature transport and implicit shape reconstruction, our method can leverage object motion to discover keypoints scattered on the mobile parts without segmentation masks, which is different from all baselines. As shown in Fig. 5, our keypoints show good temporal alignment compared to other methods.

We compute the Chamfer Distance for the *ITOP* dataset using human-annotated semantic points, and the results are listed in Tab. 2. Note that all methods do not have access to masks of the human body points during training. Despite this, our proposed method achieves satisfactory performance in real-world scenarios compared to the baselines. The counterparts cannot distinguish keypoints on the moving body from those in the background without segmentation masks. The visualization in Fig. 6 illustrates that our keypoints are close to the human labels.

Notably, the Rodent3D dataset does not furnish valid pose annotations, and therefore, we restrict ourselves to presenting qualitative outcomes of our 3D Implicit Transporter, as shown in Fig. 7. Our results demonstrate that the keypoint predictions generated by our method effectively cap-

ture the rodent's skeletal structure and display spatiotemporal coherence. Overall, our self-supervised method has the potential to significantly improve behavior analysis by addressing the difficulty of obtaining accurate 3D keypoint annotations for animals. More intuitive performance can be found in the supplementary video.

Ablation Studies 1) *Network and loss functions.* Tab. 3 shows the average perception performance on test categories from the PartNet-Mobility dataset w.r.t. designs of our method. (Row 1-2) Using a point-based decoder like TopNet [58] to recover the target point cloud, rather than employing the implicit reconstruction results in a decrease in performance. (Row 2-3) It shows that the keypoint performance improves when we utilize the cross-attention for feature fusion. (Row 3-5) The two loss functions are essential to make keypoints temporally consistent. 2) *Keypoint parameters.* The keypoint number m and Gaussian variance σ are involved with the spatial range of transported features. The results for different settings of both parameters are shown in supplementary, which indicates that $m = 6$ and $\sigma = 0.15$ are the best choice in our settings. 3) *Volume size.* The higher volumetric resolution of feature grids improves keypoint detection performance, as shown in the supplementary. However, this comes at the cost of increased memory usage. So we choose the voxel size of 64 to balance the memory cost and keypoint performance. 4) *Query sampling.* Increasing the number of queries can improve the performance of keypoint detection as shown in Tab. 4. But this increases the cost of memory and training time as well. Besides, it is crucial to sample negative queries randomly.

Table 3. Ablations for the designs of our method.

Imp. Rec.	Cro. Att.	$\mathcal{L}_{corr}$	$\mathcal{L}_{axis}$	RR ↑	ACKD ↓	ADD ↓
				0.170	0.286	0.317
✓				0.457	0.199	0.186
✓	✓			0.530	0.153	0.143
✓	✓	✓		0.602	0.130	0.123
✓	✓	✓	✓	**0.611**	**0.127**	**0.109**

Table 4. Analysis of query sampling strategies. 'R' and 'U' means random and uniform sampling of negative queries.

Query #	5k(R)	2k(R)	1k(R)	2k(U)
Repeatability ↑	**0.663**	0.611	0.597	0.058

Table 5. The manipulation performance of ours and other baselines.

	Novel Instances in Train Categories										Avg.	Test Categories									Avg.
Success rate ↑																					
ISS[75]	0.43	0.44	0.52	**0.99**	0.73	0.85	0.45	0.61	0.55	0.68	0.63	0.68	0.48	0.90	0.67	0.59	0.78	0.25	0.78	0.64	0.64
SNAKE[74]	0.44	0.32	0.33	0.71	0.32	0.64	0.32	0.44	0.41	0.21	0.41	0.45	0.42	0.44	0.62	0.38	0.14	0.23	0.34	0.67	0.41
D3feat[4]	0.55	0.57	0.80	**0.99**	0.43	0.60	0.52	0.73	0.34	0.67	0.62	0.56	0.67	0.26	0.61	0.09	0.41	0.55	0.88	0.76	0.53
INVERSE[2]	0.43	0.68	0.72	0.55	0.63	0.89	0.78	0.65	0.61	0.83	0.68	0.67	0.59	0.80	0.73	0.58	0.83	0.67	**1.00**	0.68	0.73
UMPNet[67]	0.67	0.78	0.90	0.73	0.68	0.86	0.90	0.58	0.63	0.79	0.75	0.68	0.89	0.86	0.76	0.62	0.80	**0.68**	**1.00**	**0.79**	0.79
Ours	**0.77**	**0.81**	**1.00**	0.98	**0.93**	0.90	**0.91**	0.64	**0.90**		**0.87**	**0.77**	**0.98**	**0.97**	**0.87**	0.78	**0.89**	0.57	0.95	0.78	**0.83**
Normalized distance to target ↓																					
ISS[75]	0.53	0.54	0.43	**0.01**	0.25	0.14	0.54	0.42	0.37	0.44	0.37	0.26	0.50	0.10	0.33	0.37	0.20	0.75	0.20	0.34	0.34
SNAKE[74]	0.52	0.62	0.63	0.28	0.67	0.33	0.67	0.54	0.58	0.78	0.56	0.54	0.53	0.55	0.35	0.61	0.86	0.77	0.66	0.31	0.58
D3feat[4]	0.42	0.41	0.16	**0.01**	0.57	0.39	0.46	0.27	0.65	0.32	0.37	0.41	0.35	0.74	0.36	0.91	0.58	0.44	0.08	0.20	0.45
INVERSE[2]	0.30	0.21	0.32	0.31	0.27	0.17	0.28	**0.09**	0.27	**0.09**	0.23	0.25	0.32	0.09	0.17	0.27	0.15	**0.21**	**0.00**	0.27	0.19
UMPNET[67]	**0.20**	0.19	0.05	0.19	0.23	0.16	0.12	0.13	0.28	0.11	0.17	0.26	0.03	0.06	0.15	**0.21**	0.16	0.22	**0.00**	0.17	**0.14**
Ours	**0.20**	**0.09**	**0.00**	**0.01**	**0.05**	**0.08**	**0.07**	0.10	0.35	**0.09**	**0.11**	**0.19**	**0.02**	**0.02**	**0.08**	**0.21**	**0.11**	0.42	0.05	0.21	0.15

4.4. Goal Conditioned Manipulation

The manipulation task is formulated in section 3.2. We set the same initial and goal state for test objects, and the maximum action step as [67] used. We adopt our proposed manipulation strategy for other keypoint-based methods.

Metrics Following UMPNet, we exploit 1) *normalized distance to target state* and 2) *success rate* as manipulation metrics. The first metric means the distance between the end and target state divided by the distance between the initial and target state, denoted as d. A success means $d<0.1$.

Evaluation Results We report the results in Tab. 5. Our method surpasses other keypoint-based baselines on both seen and unseen categories, demonstrating the better performance of our predicted keypoints. Our method also outperforms manipulation-centric methods Inverse [2] and UMP-Net [67] in most categories. It is noteworthy that, although UMPNET produces comparable results, it necessitates prohibitively costly trial-and-error simulations for pixel-wise affordance learning. The training time of UMPNet is 5-7 times of ours when using the same hardware, which can be found in the supplementary. Notably, they also need color and surface normal for each point. It is suggested that the learning formulation of the 3D Implicit Transporter is efficient and the manipulation strategy based on correspondent keypoints is also effective. We provide the qualitative results of our method in Fig. 8. With the correspondent keypoints, we can compute the axis-angle and translation direction, which direct us to generate a future action to interact. In contrast to moving an end-effector a fixed length every time as UMPNet used, we dynamically adjust the moving distance according to the articulation flow $\mathbf{F}_{c,g}^{s}$ between the current and goal state.

4.5. Real-World Experiments

Finally, to examine the effectiveness of our method, we conduct a real-world experiment. We design a robot system that consists of a 6-DoF robotic arm, a pneumatic suction gripper as the end-effector, and a calibrated RGB-D camera, as visualized in Fig. 9. We choose a laptop as the manipula-tion object. Results on more real-world objects can be found in the supplementary. For training data generation, we interact with the laptop to change articulation states under different camera view-points. In contrast to synthetic data, real data contains human motion which encourages keypoints to locate on the human body according to the principle of the Transporter. Therefore, we manually label an bounding box in the first frame of each video to coarsely filter points of the laptop in all frames. Fig. 9 illustrates that our method could find temporally consistent keypoints and the proper suction positions and action directions.
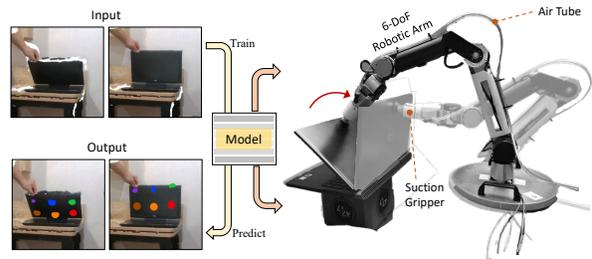

Figure 9. Real-world experiment. We train the 3D Transporter on the real data and transfer the model to a real robotic application.

5. Conclusions

Our work presents 3D Implicit Transporter, a self-supervised method to discover temporally correspondent 3D keypoints from point cloud sequences. We introduce three novel components that extend the 2D Transporter to the 3D domain. Extensive evaluations show that our keypoints are temporally consistent and generalizable to unseen object categories. Moreover, we develop a manipulation policy for downstream tasks that utilizes the Transporter keypoints and demonstrate that they are suitable for 3D manipulation tasks.

References

[1] Hameed Abdul-Rashid, Miles Freeman, Ben Abbatematteo, George Konidaris, and Daniel Ritchie. Learning to infer kinematic hierarchies for novel object instances. In *ICRA*, 2022. 3

[2] Pulkit Agrawal, Ashvin V Nair, Pieter Abbeel, Jitendra Malik, and Sergey Levine. Learning to poke by poking: Experiential learning of intuitive physics. In *NeurIPS*, 2016. 6, 9

[3] Kiam Heong Ang, Gregory Chong, and Yun Li. Pid control system analysis, design, and technology. *IEEE transactions on control systems technology*, 13(4):559–576, 2005. 6

[4] Xuyang Bai, Zixin Luo, Lei Zhou, Hongbo Fu, Long Quan, and Chiew-Lan Tai. D3feat: Joint learning of dense detection and description of 3d local features. In *CVPR*, 2020. 2, 6, 7, 9

[5] Simon Baker, Daniel Scharstein, JP Lewis, Stefan Roth, Michael J Black, and Richard Szeliski. A database and evaluation methodology for optical flow. *IJCV*, 92(1):1–31, 2011. 1

[6] John L Barron, David J Fleet, and Steven S Beauchemin. Performance of optical flow techniques. *IJCV*, 12(1):43–77, 1994. 1

[7] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *CVIU*, 110(3):346–359, 2008. 1

[8] Paul J Besl and Neil D McKay. Method for registration of 3-d shapes. In *Sensor fusion IV: control paradigms and data structures*, volume 1611, pages 586–606. Spie, 1992. 5

[9] Daniel J Butler, Jonas Wulff, Garrett B Stanley, and Michael J Black. A naturalistic open source movie for optical flow evaluation. In *ECCV*, 2012. 1

[10] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. Brief: Binary robust independent elementary features. In *ECCV*, 2010. 1

[11] Boyuan Chen, Pieter Abbeel, and Deepak Pathak. Unsupervised learning of visual 3d keypoints for control. In *ICML*, 2021. 2

[12] Özgün Çiçek, Ahmed Abdulkadir, Soeren S Lienkamp, Thomas Brox, and Olaf Ronneberger. 3d u-net: learning dense volumetric segmentation from sparse annotation. In *MICCAI*, 2016. 3

[13] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Haussser, Caner Hazirbas, Vladimir Golkov, Patrick Van Der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. In *ICCV*, 2015. 1

[14] Ben Eisner*, Harry Zhang*, and David Held. Flowbot3d: Learning 3d articulation flow to manipulate articulated objects. In *RSS*, 2022. 3, 5

[15] Haoqiang Fan, Hao Su, and Leonidas J Guibas. A point set generation network for 3d object reconstruction from a single image. In *CVPR*, 2017. 3

[16] Peter Florence, Lucas Manuelli, and Russ Tedrake. Self-supervised correspondence in visuomotor policy learning. *IEEE Robotics and Automation Letters*, 5(2):492–499, 2019. 5

[17] Anand Gopalakrishnan, Sjoerd van Steenkiste, and Jürgen Schmidhuber. Unsupervised object keypoint learning using local spatial predictability. *arXiv preprint arXiv:2011.12930*, 2020. 2

[18] Albert Haque, Boya Peng, Zelun Luo, Alexandre Alahi, Serena Yeung, and Li Fei-Fei. Towards viewpoint invariant 3d human pose estimation. In *ECCV*, 2016. 2, 6

[19] Berthold KP Horn and Brian G Schunck. Determining optical flow. *Artificial intelligence*, 17(1-3):185–203, 1981. 1

[20] Zhenyu Jiang, Cheng-Chun Hsu, and Yuke Zhu. Ditto: Building digital twins of articulated objects from interaction. *arXiv preprint arXiv:2202.08227*, 2022. 2, 3

[21] Zhenyu Jiang, Yifeng Zhu, Maxwell Svetlik, Kuan Fang, and Yuke Zhu. Synergies between affordance and geometry: 6-dof grasp detection via implicit representations. *arXiv preprint arXiv:2104.01542*, 2021. 2

[22] Dov Katz and Oliver Brock. Manipulating articulated objects with interactive perception. In *ICRA*, 2008. 3

[23] Tejas D Kulkarni, Ankush Gupta, Catalin Ionescu, Sebastian Borgeaud, Malcolm Reynolds, Andrew Zisserman, and Volodymyr Mnih. Unsupervised learning of object keypoints for perception and control. In *NeurIPS*, 2019. 1, 2, 3, 4

[24] Chang Ha Lee, Amitabh Varshney, and David W Jacobs. Mesh saliency. In *SIGGRAPH*. 2005. 2

[25] Jiaxin Li and Gim Hee Lee. Usip: Unsupervised stable interest point detection from 3d point clouds. In *CVPR*, 2019. 1, 2, 6, 7

[26] Pengfei Li, Ruowen Zhao, Yongliang Shi, Hao Zhao, Jirui Yuan, Guyue Zhou, and Ya-Qin Zhang. Lode: Locally conditioned eikonal implicit scene completion from sparse lidar. *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8269–8276, 2023. 2

[27] Xiaolong Li, He Wang, Li Yi, Leonidas J Guibas, A Lynn Abbott, and Shuran Song. Category-level articulated object pose estimation. In *CVPR*, 2020. 3

[28] Yang Li and Tatsuya Harada. Lepard: Learning partial point cloud matching in rigid and deformable scenes. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5554–5564, 2022. 3

[29] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. In *NeurIPS*, 2020. 2

[30] David G Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004. 1

[31] Xiao Ma, David Hsu, and Wee Sun Lee. Learning latent graph dynamics for visual manipulation of deformable objects. In *ICRA*. 5

[32] Lucas Manuelli, Wei Gao, Peter Florence, and Russ Tedrake. kpam: Keypoint affordances for category-level robotic manipulation. In *Robotics Research: The 19th International Symposium ISRR*. 5

[33] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 2

[34] Matthias Minderer, Chen Sun, Ruben Villegas, Forrester Cole, Kevin P Murphy, and Honglak Lee. Unsupervised

learning of object structure and dynamics from videos. In *NeurIPS*, 2019. 2

[35] Kaichun Mo, Leonidas J Guibas, Mustafa Mukadam, Abhinav Gupta, and Shubham Tulsiani. Where2act: From pixels to actions for articulated 3d objects. In *CVPR*, 2021. 2, 3

[36] Jiteng Mu, Weichao Qiu, Adam Kortylewski, Alan Yuille, Nuno Vasconcelos, and Xiaolong Wang. A-sdf: Learning disentangled signed distance functions for articulated shape representation. In *CVPR*, 2021. 2

[37] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE transactions on robotics*, 31(5):1147–1163, 2015. 1

[38] Neil Nie, Samir Yitzhak Gadre, Kiana Ehsani, and Shuran Song. Structure from action: Learning interactions for articulated object 3d structure discovery. *arXiv preprint arXiv:2207.08997*, 2022. 3

[39] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *CVPR*, 2019. 2

[40] Mahir Patel, Yiwen Gu, Lucas C Carstensen, Michael E Hasselmo, and Margrit Betke. Animal pose tracking: 3d multimodal dataset and token-based pose optimization. *International Journal of Computer Vision*, 131(2):514–530, 2023. 2, 6

[41] Songyou Peng, Michael Niemeyer, Lars Mescheder, Marc Pollefeys, and Andreas Geiger. Convolutional occupancy networks. In *ECCV*, 2020. 2

[42] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *NeurIPS*, 2017. 3

[43] Zengyi Qin, Kuan Fang, Yuke Zhu, Li Fei-Fei, and Silvio Savarese. Keto: Learning keypoint representations for tool manipulation. In *ICRA*. 1, 5

[44] Hossein Rahmani, Arif Mahmood, Q Du Huynh, and Ajmal Mian. Hopc: Histogram of oriented principal components of 3d pointclouds for action recognition. In *ECCV*, 2014. 2

[45] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *ICCV*, 2011. 1

[46] Radu Bogdan Rusu, Nico Blodow, and Michael Beetz. Fast point feature histograms (fpfh) for 3d registration. In *ICRA*, 2009. 6

[47] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *CVPR*. 3

[48] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. Graf: Generative radiance fields for 3d-aware image synthesis. In *NeurIPS*, 2020. 2

[49] Anthony Simeonov, Yilun Du, Andrea Tagliasacchi, Joshua B Tenenbaum, Alberto Rodriguez, Pulkit Agrawal, and Vincent Sitzmann. Neural descriptor fields: Se (3)-equivariant object representations for manipulation. In *ICRA*, 2022. 2

[50] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representa-tions with periodic activation functions. In *NeurIPS*, 2020. 2

[51] Josef Sivic and Andrew Zisserman. Video google: A text retrieval approach to object matching in videos. In *ICCV*, 2003. 1

[52] Elizabeth S Spelke. Principles of object perception. *Cognitive science*, 14(1):29–56, 1990. 1

[53] Deqing Sun, Charles Herrmann, Fitsum Reda, Michael Rubinstein, David J Fleet, and William T Freeman. Disentangling architecture and training for optical flow. In *ECCV*, 2022. 1

[54] Deqing Sun, Stefan Roth, and Michael J Black. Secrets of optical flow estimation and their principles. In *CVPR*, 2010. 1

[55] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *CVPR*, 2018. 1

[56] Jennifer J Sun, Serim Ryou, Roni H Goldshmid, Brandon Weissbourd, John O Dabiri, David J Anderson, Ann Kennedy, Yisong Yue, and Pietro Perona. Self-supervised keypoint discovery in behavioral videos. In *CVPR*, 2022. 2

[57] Supasorn Suwajanakorn, Noah Snavely, Jonathan J Tompson, and Mohammad Norouzi. Discovery of latent 3d keypoints via end-to-end geometric reasoning. In *NeurIPS*, 2018. 2

[58] Lyne P Tchapmi, Vineet Kosaraju, Hamid Rezatofighi, Ian Reid, and Silvio Savarese. Topnet: Structural point cloud decoder. In *CVPR*. 8

[59] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 3

[60] Xudong Wang, Long Lian, and Stella X Yu. Unsupervised visual attention and invariance for reinforcement learning. In *CVPR*, 2021. 2

[61] Xiaogang Wang, Bin Zhou, Yahao Shi, Xiaowu Chen, Qinping Zhao, and Kai Xu. Shape2motion: Joint analysis of motion parts and attributes from 3d shapes. In *CVPR*, 2019. 3

[62] Yian Wang, Ruihai Wu, Kaichun Mo, Jiaqi Ke, Qingnan Fan, Leonidas J Guibas, and Hao Dong. Adaafford: Learning to adapt manipulation affordance for 3d articulated objects via few-shot interactions. In *ECCV*. 2

[63] Yijia Weng, He Wang, Qiang Zhou, Yuzhe Qin, Yueqi Duan, Qingnan Fan, Baoquan Chen, Hao Su, and Leonidas J Guibas. Captra: Category-level pose tracking for rigid and articulated objects from point clouds. In *CVPR*, 2021. 3

[64] Ruihai Wu, Yan Zhao, Kaichun Mo, Zizheng Guo, Yian Wang, Tianhao Wu, Qingnan Fan, Xuelin Chen, Leonidas Guibas, and Hao Dong. Vat-mart: Learning visual action trajectory proposals for manipulating 3d articulated objects. *arXiv preprint arXiv:2106.14440*, 2021. 3

[65] Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, et al. Sapien: A simulated part-based interactive environment. In *CVPR*, 2020. 2, 3, 6

[66] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. Posecnn: A convolutional neural network for

6d object pose estimation in cluttered scenes. *arXiv preprint arXiv:1711.00199*, 2017. 8

[67] Zhenjia Xu, He Zhanpeng, and Shuran Song. Umpnet: Universal manipulation policy network for articulated objects. *RA-L*, 7:2447–2454, 2022. 2, 3, 5, 6, 9

[68] Li Yi, Haibin Huang, Difan Liu, Evangelos Kalogerakis, Hao Su, and Leonidas Guibas. Deep part induction from articulated object pairs. *arXiv preprint arXiv:1809.07417*, 2018. 3

[69] Yang You, Wenhai Liu, Yanjie Ze, Yong-Lu Li, Weiming Wang, and Cewu Lu. Ukpgan: A general self-supervised keypoint detector. In *CVPR*, 2022. 2

[70] Andy Zeng, Shuran Song, Matthias Nießner, Matthew Fisher, Jianxiong Xiao, and Thomas A. Funkhouser. 3dmatch: Learning local geometric descriptors from rgb-d reconstructions. In *CVPR*, 2017. 2

[71] Yiming Zeng, Yue Qian, Zhiyu Zhu, Junhui Hou, Hui Yuan, and Ying He. Corrnet3d: Unsupervised end-to-end learning of dense correspondence for 3d point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6052–6061, 2021. 3

[72] Jian Zhao, Jianshu Li, Hengzhu Liu, Shuicheng Yan, and Jiashi Feng. Fine-grained multi-human parsing. *International Journal of Computer Vision*, 128:2185–2203, 2020. 1

[73] Chengliang Zhong, Chao Yang, Fuchun Sun, Jinshan Qi, Xiaodong Mu, Huaping Liu, and Wenbing Huang. Sim2real object-centric keypoint detection and description. In *AAAI*. 1

[74] Chengliang Zhong, Peixing You, Xiaoxue Chen, Hao Zhao, Fuchun Sun, Guyue Zhou, Xiaodong Mu, Chuang Gan, and Wenbing Huang. Snake: Shape-aware neural 3d keypoint field. *Advances in Neural Information Processing Systems*, 35:7052–7064, 2022. 2, 6, 7, 9

[75] Yu Zhong. Intrinsic shape signatures: A shape descriptor for 3d object recognition. In *ICCV Workshops*, 2009. 2, 6, 7, 9

[76] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Fast global registration. In *ECCV*, 2016. 2

[77] Song Chun Zhu, Yingnian Wu, and David Mumford. Filters, random fields and maximum entropy (frame): Towards a unified theory for texture modeling. *IJCV*, 27(2):107–126, 1998. 1