ACID: Action-Conditional Implicit Visual Dynamics for Deformable Object Manipulation

Bokui Shen[1,*] Zhenyu Jiang[2] Christopher Choy[3] Silvio Savarese[1] Leonidas J. Guibas[1]
Anima Anandkumar[3,4] Yuke Zhu[2,3]
[1]Stanford University [2]The University of Texas at Austin [3]NVIDIA [4]Caltech

***Abstract*—Manipulating volumetric deformable objects in the real world, like plush toys and pizza dough, bring substantial challenges due to infinite shape variations, non-rigid motions, and partial observability. We introduce ACID, an action-conditional visual dynamics model for volumetric deformable objects based on structured implicit neural representations. ACID integrates two new techniques: implicit representations for action-conditional dynamics and geodesics-based contrastive learning. To represent deformable dynamics from partial RGB-D observations, we learn implicit representations of occupancy and flow-based forward dynamics. To accurately identify state change under large non-rigid deformations, we learn a correspondence embedding field through a novel geodesics-based contrastive loss. To evaluate our approach, we develop a simulation framework for manipulating complex deformable shapes in realistic scenes and a benchmark containing over 17,000 action trajectories with six types of plush toys and 78 variants. Our model achieves the best performance in geometry, correspondence, and dynamics predictions over existing approaches. The ACID dynamics models are successfully employed to goal-conditioned deformable manipulation tasks, resulting in a 30% increase in task success rate over the strongest baseline. For more results and information, please visit https://b0ku1.github.io/acid/.**

I. Introduction

From playing with plush toys to making pizza dough, deformable objects are prevalent in many daily activities. Perceiving realistic deformable objects and modeling their dynamics over time play an integral role in building autonomous robots to interact with soft objects [1]. However, estimating the geometry of deformable objects from raw visual signals and predicting their motions present significant challenges owing to their infinite continuous configuration spaces, complex non-linear dynamics, partial observability, and self-collision.

Decades of robotics research have developed dynamics models that characterize complex motions of non-rigid objects. Conventional approaches rely on high-fidelity mathematical models [2]–[5] or recently their learned approximations [6], [7]. These models, however, are hard to estimate from raw sensory data, hindering their applicability. In recent years, data-driven methods have demonstrated preliminary success in modeling 1D deformable linear objects like ropes [8]–[12], 2D flat deformable objects like cloth [13]–[18], or primitive 3D objects like sponges [19]–[22]. Nonetheless, existing methods struggle to handle volumetric deformable shapes, of which the 3D volumes and dynamics have to be inferred from partially observed visual data. Meanwhile, the 3D vision community has made great strides in modeling deformation from visual data. Specifically, implicit neural representations have shown great promise to model deformable objects and their dynamics with coordinate-based neural networks in high fidelity [23]–[26]. Nonetheless, these methods focus on limited object categories and constrained motions rather than action-conditioned dynamics required for manipulation planning.

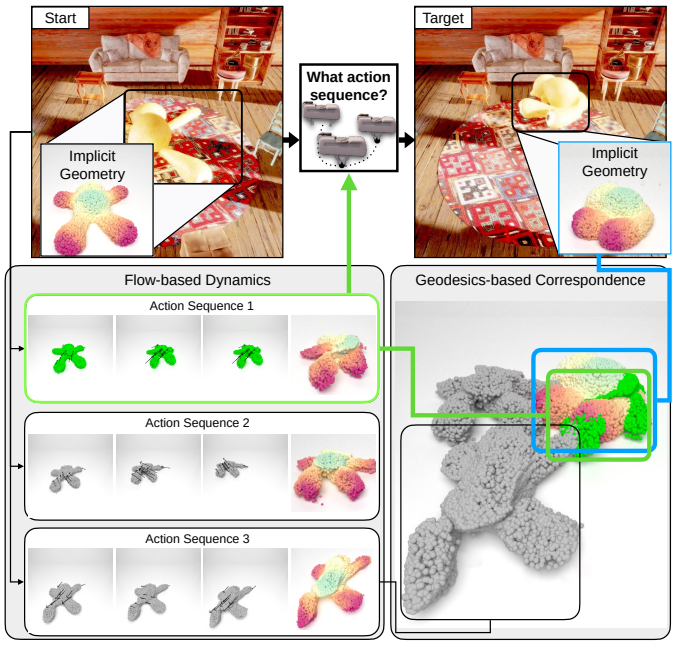

Fig. 1: Perceiving deformable objects from raw visual signals and modeling their dynamics are challenging due to partial observations and complex non-rigid dynamics, making it difficult to determine what action sequence can manipulate an object into the desired configuration. We use implicit representations and geodesics-based correspondence learning to learn an end-to-end action-conditional visual dynamics model for realistic volumetric deformable objects manipulation.

We introduce an action-conditional visual dynamics model for volumetric deformable objects manipulation in realistic 3D scenes, named ACID (**A**ction-**C**onditional **I**mplicit **D**ynamics). Our model learns joint representations for geometry, flow-based dynamics, and correspondence embedding fields end-to-end, as illustrated in Fig. 1. Unlike explicit 3D representations (e.g. point cloud or voxels) widely used in dynamics modeling [18], [27]–[29], we use implicit neural representations to reconstruct high-fidelity full geometry and predict flow-based

∗ Work done during an internship at NVIDIA Research.

dynamics field from RGB-D observations. Moreover, measuring state changes accurately under non-rigid transformations benefits from establishing a dense correspondence over time. We propose a geodesic-aware contrastive learning loss to learn a correspondence embedding field to establish point-wise correspondence. The learned correspondence improves dynamics prediction and informs manipulation planning.

We develop a new simulation framework for manipulating deformable objects in realistic scenes using state-of-the-art physical simulation and photorealistic rendering techniques to train and evaluate ACID. Our framework is built with the PhysX engine in NVIDIA's Omniverse Kit. We use the framework to generate a deformable visual dynamics dataset containing 17,000+ action trajectories with six types of plush toys of 78 different variants.

Our model significantly outperforms existing dynamics models using explicit 3D representations, with a 23% increase in shape mIoU and a 24% decrease in dynamics mean-squared error. Our geodesics-based correspondence shows a 15% increase in a matching metric. We use our model for model-based control of volumetric deformable object manipulation, resulting in 30% increase in task success rate.

II. Related Work

Learning Deformable Dynamics Models for Manipulation. Robot manipulation of deformable objects has been a prolific field with wide-ranging applications in diverse industries. Various works have proposed approaches to extract intermediate geometric representations of deformable objects and model their dynamics for manipulation. Such methods have shown success in 1D deformable linear objects (*e.g.*, ropes) as line segments [8]–[12], 2D flat deformable objects (*e.g.*, cloth) as a graph or a surface mesh [13]–[18]. Others model simple 3D deformable objects (*e.g.*, sponge) as their 2D contours [19]–[22]. Several comprehensive reviews [1], [2] have captured the rich history in modeling deformable objects for robot manipulation. However, learning a full 3D representation in an end-to-end framework for realistic deformable objects is still beyond reach.

Alternative approaches try to tackle deformable objects manipulation with learned dynamics models in pixel [31]–[33] or latent [34]–[37] spaces, rather than explicit physical computation. Pixel-space models directly predict future observations and measure prediction quality with the reconstruction loss or the perceptual loss. Latent-space models encode raw observations into latent state vectors (*e.g.*, using self-supervision like reconstruction [35], [37], contrastive loss [34] or keypoints [36]), and learn the dynamics in the latent state space. These pixel-space and latent-space models can be learned end-to-end, but they do not explicitly represent the underlying 3D structures of deformable objects. In comparison, we introduce a new simulation framework with 3D supervision and study how 3D dynamics can be faithfully captured with structured representations.

Other works from related fields developed learning models to approximate physics simulator [6], [7], [38]–[41]. With access to underlying physical state (*e.g.*, mass, velocity, acceleration), these works approximate the explicit calculations in physics simulator with learned models. However, these approaches are difficult to estimate from raw visual input.

Implicit Representations of Geometry and Motion. Numerous works have demonstrated impressive performance in shape prediction using voxels [42]–[44], depth predictions [45], [46], point clouds [47], [48], and meshes [49]. Recently, implicit representations [50]–[55] have shown impressive performances representing shapes. Moreover, the implicit approach has shown impressive performance in beyond-3D tasks like spatial-temporal reconstruction [23], [56] and shape deformation estimation [24]. Prior works focus on rigid shapes from a large-scale shape repository [57] or deformable shapes of a specific category (*e.g.*, humans) [58]. Our work focuses on extending these approaches to realistic deformable objects of different morphology (*e.g.*, plush snake vs. plush teddy), where geometry and motion must be learned jointly with actions. Furthermore, we consider deformable objects in realistic scenes where an agent must take obstacles into account for interaction with the object of interest.

Geometric Correspondence Learning. Learning geometric correspondence has been widely studied [59], [60]. We discuss two main branches of work. First is learning shape correspondences between deformable objects like human and animals [61]–[69]. These works take in surface meshes of two objects and learn their correspondences. In contrast to ours, these works learn correspondences between objects of the same category and morphology. Moreover, they need access to complete surface shapes, which are unavailable in real camera observations due to (self) occlusion. The second is learning dense features of point clouds for registration [70]–[72]. They used a partial point cloud for input, but these works have not been examined for deformable objects.

Deformable Object Simulation. Research in computer graphics has made huge progress in deformable object simulation [4], [73]–[76]. Such progress has produced many robust physics simulation engines, and learning-based components have been incorporated for acceleration [77]–[81]. Such efforts have in turn powered recent simulators for deformable object manipulation [82]–[84]. By leveraging the GPU-accelerated finite element methods in Omniverse [85], we extend existing simulators from primitive geometries to realistic objects in complex scenes.

III. Action-Conditional Implicit Dynamics

In this section, we introduce ACID (Fig. 2), our action-conditional visual dynamics model for deformable object manipulation using structured implicit neural representations. We discuss the problem formulation and the three main components of ACID: geometry prediction, dynamics prediction, and correspondence learning. Lastly, we discuss applying ACID to a deformable manipulation task using model-based planning.

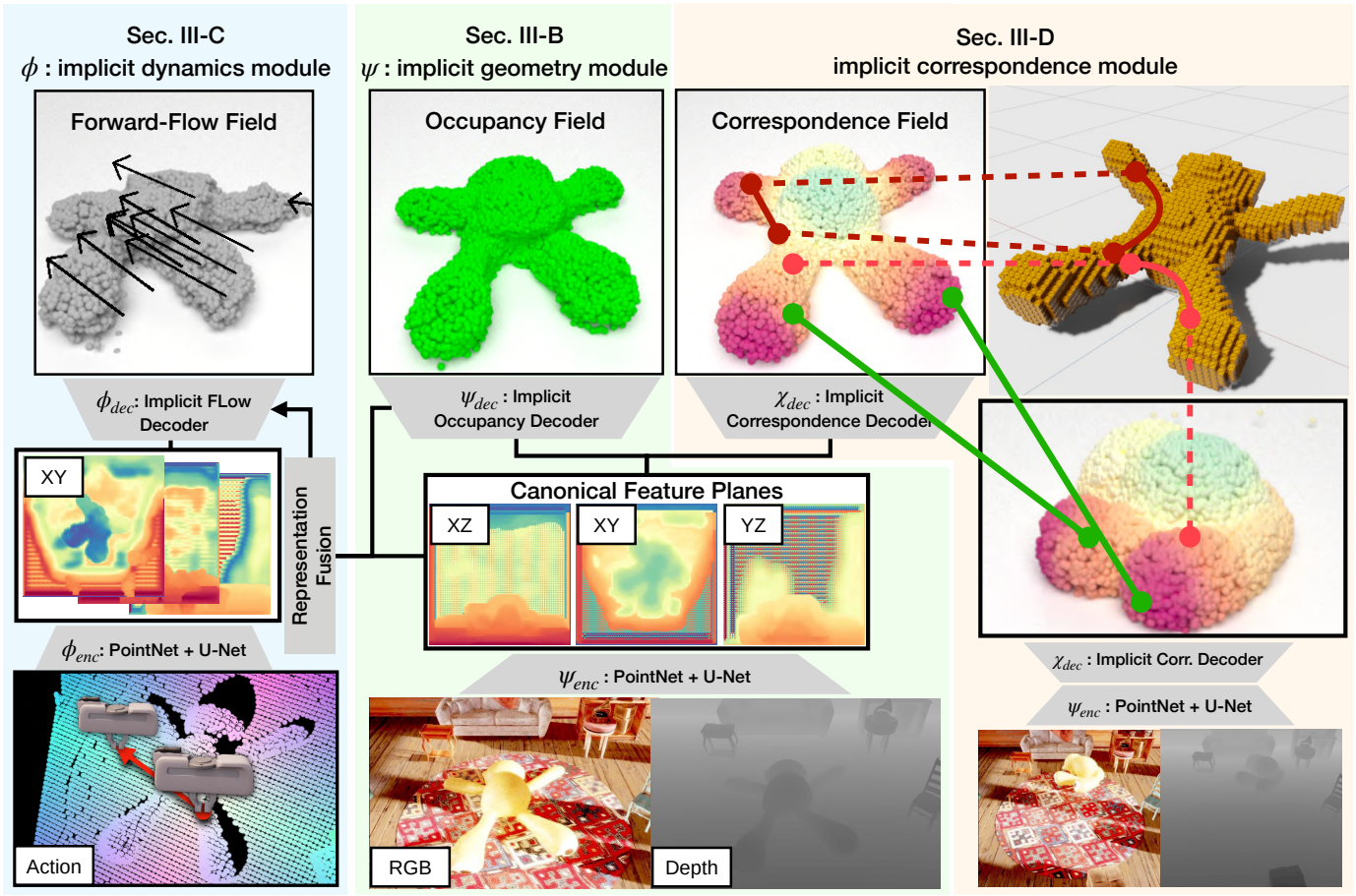

Fig. 2: We introduce **ACID**, an end-to-end action-conditional visual dynamics model that jointly predicts geometry (Sec. III-B), dynamics (Sec. III-C), and correspondence (Sec. III-D) as structured implicit representations. We extract state representation as explicit 3D shapes (point clouds) from implicit representations and visualize them here. First, the two encoders ψ_{enc} for RGB-D inputs and ϕ_{enc} for robot actions produce three canonical feature planes, respectively. The Implicit Occupancy and Correspondence Decoders bilinearly interpolate the encoded feature planes at different 3D locations and generate: 1) an occupancy field and 2) a correspondence field trained with a geodesics-based contrastive loss. We illustrated the mechanism of contrastive learning on the upper right. We show here the t-SNE [30] color-coded features overlaid on an example pair. The Implicit Flow Decoder fuses the representations from the two encoders and predicts a one-step forward flow field.

A. Problem Formulation

We focus on manipulating deformable objects in clutter. Specifically, we build a robotic gripper controller that rearranges deformable objects from an initial position to the target position from RGB-D observations. We denote the observation at time t as $\mathbf{o}_t \in \mathcal{O}$, where $\mathcal{O}$ is the space of observations. We denote the action of a gripper at time t as $\mathbf{a}_t = (p_g, p_r) \in \mathcal{A}$, where $\mathcal{A}$ is the space of actions. The action represents grasping a target object at point $p_g \in \mathbb{R}^3$, moving the gripper to $p_r \in \mathbb{R}^3$, and releasing the gripper.

We use a model-based approach. Thus, building a robust model of states and state transitions is of primal importance. To this end, we learn state representations as both a full object geometry state $\mathbf{s}_t \in \mathcal{S}$ and a high-dimensional correspondence embedding space using neural networks. Here we denote $\mathcal{S}$ as the state space. Learning this model requires three different functions: state representation, state transition, and state distance measure for planning. We use neural networks for state representation $\psi : \mathcal{O} \to \mathcal{S}$ (Sec. III-B) and state transition $\phi : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ (Sec. III-C). Lastly, $d(\cdot)$ is a state distance measure (Sec. III-E). Conventional distance measures include non-parameterized metrics like mIoU or Chamfer distance, which do not take correspondences into account. Instead, we propose a learning-based correspondence module to compute more accurate distances under deformation (Sec. III-D).

B. Predicting Geometry from Partial Observation

To perceive deformable objects and reason about their dynamics in 3D space from partial observations, prior work has used various explicit 3D representations, among which partial point clouds [18], [27] and voxels [28], [29] are two of the most popular choices. These representations, however, have to trade-off between memory footprint and geometric resolution crucial for accurate dynamics modeling. Instead, we use an implicit geometry module $\psi : \mathcal{O} \to \mathcal{S}$ that takes a partial RGB-D

observation $\mathbf{o}_t$ as input and performs both continuous implicit encoding as well as 3D completion. The module contains two components: an observation encoder $\psi_{enc} : \mathcal{O} \to \mathbb{R}^{H\times W\times D}$ that encodes partial observation into a feature map and an Implicit Occupancy Decoder $\psi_{dec} : \mathbb{R}^3 \times \mathbb{R}^D \to [0, 1]$ that maps a coordinate and its queried feature into an occupancy probability.

We follow the architecture of Peng *et al*. [55] to create the observation encoder ψ_{enc} that maps a point cloud to a set of 2D feature maps. We first back-project an RGB-D image into the 3D space of per-point RGB colors and use a shallow PointNet [47] with local pooling to map the three-dimensional input point coordinates and their colors into a latent feature space. These point cloud features are then orthographically projected onto three canonical feature planes (xy, xz, yz planes). Lastly, we pass these three canonical features maps of size $H \times W \times D$ through a U-Net [86]. Specifically, we use a hidden layer dimension of $D = 64$ and U-Net of depth 4 to generate three feature planes of size $128 \times 128 \times 64$.

These three feature planes encode the shape implicitly. To obtain the occupancy at each point in $\mathbb{R}^3$, we use bilinear interpolation to compute a local feature of this point from each of the canonical feature planes and pass the summed feature to a multi-layer perceptron (MLP) to extract occupancy. Technically, let $p \in \mathbb{R}^3$ be a query point and we project p to three canonical planes and sum three features from the planes to get $\psi_{enc}(\mathbf{o}_t)|_p$. Then we pass it through ψ_{dec}, an MLP with skip connections, to predict the occupancy probability:

$$\psi_{dec}(p, \psi_{enc}(\mathbf{o}_t)|_p) \in [0, 1] \tag{1}$$

Given the implicit geometry module ψ, we can extract state $\mathbf{s}_t$ which is the explicit 3D shape of the object of interest. Specifically, we extract an interior 3D point cloud from the Implicit Occupancy Decoder by aggregating the coordinates whose occupancy probabilities are above a threshold τ:

$$\mathbf{s}_t = \{p \in \mathbb{R}^3 | \psi_{dec}(p, \psi_{enc}(\mathbf{o}_t)|_p) > \tau\} \subseteq \mathbb{R}^3 \tag{2}$$

C. Action-conditional Flow-based Dynamics Field

Using implicit representations allows us to model geometry beyond the grid resolution. Similarly, we adopt a coordinate-based neural model to model high-fidelity dynamics [23], [24]. We use an implicit dynamics module ϕ to model the action-conditioned dynamics $\phi : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$. The implicit dynamics module consists of two components similar to the implicit geometry module: an encoder $\phi_{enc} : \mathcal{O} \times \mathcal{A} \to \mathbb{R}^{H\times W\times D}$ that encodes the partial observation and an action into a feature map, and the Implicit Flow Decoder $\phi_{dec} : \mathbb{R}^3 \times \mathbb{R}^D \to \mathbb{R}^3$ that maps a coordinate and its queried feature into a one-step prediction of forward flow.

We encode the action jointly with the input partial observation using ϕ_{enc}, with the same network architecture as our geometry module's encoder ψ_{enc}. Similar to geometry feature encoding, we first back-project the RGB-D image to a partial point cloud. Instead of using per-point RGB color as the point feature, here we fuse the action command with the partial point cloud into a point feature. Given an action $\mathbf{a}_t = (p_g, p_r)$, for each p_i in the partial point cloud, we calculate its distance to the grasp location and assign per-point feature as $(p_g - p_i, p_r)$. This way we ensure that per-point features can capture the point's relative position to the action location. This featurized partial point cloud is similarly encoded into three canonical features planes of $128 \times 128 \times 64$.

For point $p \in \mathbb{R}^3$, we query the feature vector $\phi_{enc}(\mathbf{o}_t, \mathbf{a}_t)|_p$ in the same way as the Implicit Occupancy Decoder. Finally, the one-step forward flow is computed with the Implicit Flow Decoder ϕ_{dec}:

$$\phi_{dec}(p, \phi_{enc}(\mathbf{a}_t, \mathbf{o}_t)|_p) \in \mathbb{R}^3 \tag{3}$$

Representation fusion: We learn the implicit dynamics module ϕ jointly with the implicit geometry module ψ (Sec. III-B) through representation fusion. When predicting occupancy for point p, the encoder generates a per-point feature $\psi_{enc}(\mathbf{o}_t)|_p$ (Sec. III-B). The Implicit Occupancy Decoder, which contains multiple fully-connected layers, further processes $\psi_{enc}(\mathbf{o}_t)|_p$ into a set of per-point intermediate representations, which we denote as $\mathbf{r}_p$ for the point p. We feed $\mathbf{r}_p$ and $\psi_{enc}(\mathbf{o}_t)|_p$ as additional inputs into the Implicit Flow Decoder. Consequently, the one-step forward flow prediction becomes:

$$\phi_{dec}(p, \phi_{enc}(\mathbf{a}_t, \mathbf{o}_t)|_p, \psi_{enc}(\mathbf{o}_t)|_p, \mathbf{r}_p) \in \mathbb{R}^3 \tag{4}$$

D. Geodesics-based Contrastive Learning

When reasoning about a deformable object in varying configurations, a critical challenge is to establish dense correspondences between two visual observations of the same object in drastically different configurations. Prior work [71], [72] has used contrastive learning for point cloud correspondence features. But they focus on 1) rigid objects, 2) static contrastive margin, and 3) Euclidean distance to define positive pairs. For deformable objects, we have to take deformation into account and the simple Euclidean distance is no longer accurate. For example, when a teddy bear's arm bends and touches its body, the arm and the body have zero Euclidean distance, but they belong to two separate parts that we must distinguish. Therefore, we propose to use the geodesic distance defined on the surface of a deformable object. It allows the arm and the body to have great distances even though they touch each other. In this work, we use such a geodesic distance as the contrastive margin to learn more discriminative features.

For a pair of states $\mathbf{s}_t, \mathbf{s}_{t\prime} \subseteq \mathbb{R}^3$ under non-rigid deformation, we have a set of correspondences between the sets of points as $\mathcal{C} = \{(p, q) | p \in \mathbf{s}_t, q \in \mathbf{s}_{t\prime}\}$. In a contrastive learning setting [71], we can learn a point embedding $\mathbf{f}$ that for points p, q minimize the loss:

$$\begin{aligned} L(\mathbf{f}_p, \mathbf{f}_q) =& I_{pq}[D(\mathbf{f}_p, \mathbf{f}_q) - m_{pos}]_+^2 \\ &+ \bar{I}_{pq}[m_{neg} - D(\mathbf{f}_p, \mathbf{f}_q)]_+^2 \end{aligned} \tag{5}$$

where $D(\cdot, \cdot)$ is a distance measure; $I_{pq} = 1$ if $(p, q) \in \mathcal{C}$ and 0 otherwise, $\bar{\cdot}$ is the NOT operator, m_{pos}, m_{neg} are margins

for positive and negative pairs. To incorporate geodesics of the original shape manifold, we extend L as:

$$L_{geo}(\mathbf{f}_p, \mathbf{f}_q) = I^g_{pq}[D(\mathbf{f}_p, \mathbf{f}_q) - m_{pos}]^2_+ + \bar{I}^g_{pq}\big[\log\left(\frac{d_O(p,q)}{d_{thres}}\right) + m_{neg} - D(\mathbf{f}_p, \mathbf{f}_q)\big]^2_+ \quad (6)$$

where $d_O(p,q)$ is the geodesic function described below, d_{thres} is a geodesic threshold, and $I^g_{pq} = 1$ if $d_O(p,q) < d_{thres}$ and 0 otherwise.

We represent $\mathbf{f}$ as an embedding field in $\mathbb{R}^{32}$. The correspondence field is learned jointly with the occupancy prediction, sharing the same encoder ψ_{enc}. We use an Implicit Correspondence Decoder χ_{dec} to predict the point embedding:

$$\chi_{dec}(p, \psi_{enc}(\mathbf{o}_t)|_p) \rightarrow \mathbf{f}_p \in \mathbb{R}^{32} \quad (7)$$

To measure $d(\mathbf{s}_t, \mathbf{s}_{t\prime}) \rightarrow v \in \mathbb{R}$, we first extract a correspondence ξ_{corr} between $\mathbf{s}_t$ and $\mathbf{s}_{t\prime}$ that minimize the aggregated feature distance among all possible matching:

$$\xi_{corr} := \underset{\xi:\mathbf{s}_t \rightarrow \mathbf{s}_{t\prime}}{\arg\min} \sum_{p \in \mathbf{s}_t} D(\mathbf{f}_p, \mathbf{f}_{\xi(p)}) \quad (8)$$

Given the matching ξ_{corr}, we can calculate the mean corresponded distance as:

$$d_{corr}(\mathbf{s}_t, \mathbf{s}_{t\prime}, \xi_{corr}) := \frac{1}{|\mathbf{s}_t|} \sum_{p \in \mathbf{s}_t} \|p - \xi_{corr}(p)\|^2 \quad (9)$$

where $|\mathbf{s}_t|$ denotes the number of points in shape $\mathbf{s}_t$.

Geodesics computation: For each volumetric deformable object O, we approximate it as a graph G of connected tetrahedrons in a fixed resolution. For two arbitrary points $p, q \in O$, we can retrieve their corresponding tetrahedrons $t_p, t_q \in G$, and we can calculate the geodesic distance $d_O(p,q)$ between p and q as the shortest path distance between t_p and t_q in graph G. During non-rigid deformation, though each tetrahedron is deformed, the connectivity of the structure remains fixed, which establishes d_O as a deformation-consistent metric.

E. Planning with Dynamics Model

We now explain how we use ACID for model-based manipulation planning following a target-driven setup, where the target is specified by an image of the goal state. Given the current and target configurations specified as RGB-D images $\mathbf{o}_0, \mathbf{o}_{target}$ respectively, model-based planning requires a cost function on a sequence of actions $\texttt{cost}(\mathbf{a}_1, ..., \mathbf{a}_n)$, which guides the selection of actions that minimize the cost. Given the implicit geometry module, we first recover initial state $\mathbf{s}_0$ as the explicit full geometry at the current time using Eq. (2) with the threshold $\tau = 0.75$. Then, for state $\mathbf{s}_t$ at time t, we can estimate the one-step forward flow of each $p \in \mathbf{s}_t$ with the implicit dynamics module, described in Sec. III-C:

$$\mathbf{s}_{t+1} = \{p + \phi_{dec}(p, \phi_{enc}(\mathbf{a}_t, \mathbf{o}_t)|_p, \psi_{enc}(\mathbf{o}_t)|_p, \mathbf{r}_p) | p \in \mathbf{s}_t\} \quad (10)$$

Note that the function above takes as input $\mathbf{o}_t$, but for a future time step $t \neq 0$, the observation $\mathbf{o}_t$ is unavailable. We tackle this challenge as follows: First, we make the simplification that for a given point p, its features from the implicit geometry module ψ ($\psi_{enc}(\mathbf{o}_t)|_p$ and $\mathbf{r}_p$) remain fixed across time steps. We make such approximation as point features are supervised with a correspondence loss, encouraging the same point to have a similar feature across different configurations. Thus we only need to evaluate ψ once using $\mathbf{o}_0$ as input, to obtain $\psi_{enc}(\mathbf{o}_0)|_p$ and $\mathbf{r}_p$. Second, since $\phi_{enc}(\mathbf{a}_t, \mathbf{o}_t)|_p$ does not depend on the per-point RGB feature (Sec. III-C), we observe that we can approximate $\mathbf{o}_t$ without color by performing camera projection over the full object geometry $\mathbf{s}_t$. We then can randomly sample future actions from the projected point cloud of the object of interest. Thus, the calculation of Eq. (10) can be performed iteratively to roll out future states under a given action sequence.

To compute the cost with Eq. (9), we first recover the target state geometry as $\mathbf{s}_{target}$ with Eq. (2). We can establish correspondence ξ_{corr} between $\mathbf{s}_0$ and $\mathbf{s}_{target}$ as in Eq. (8). We then calculate the cost as the state distance between the roll-out state after n actions $\mathbf{s}_n$ and the target state as:

$$\texttt{cost}(\mathbf{a}_1, ..., \mathbf{a}_n) = d_{corr}(\mathbf{s}_n, \mathbf{s}_{target}, \xi_{corr}) \quad (11)$$

The action sequence of the lowest cost is chosen.

F. Implementation Details

In realistic scenes, the object of interest only occupies a small portion of the 3D space. At training time, we sample query points $p \in \mathbb{R}^3$ from a multivariate normal distribution with a mean of object's center-of-mass and with standard deviation proportional to the size of the object bounding box. The object of interest is specified via a 2D instance segmentation mask that is jointly passed into the network with RGB-D image. The Implicit Occupancy Decoder ψ_{dec} is supervised by the binary cross-entropy loss between the predicted and the ground-truth value at the sampled 3D coordinates. The Implicit Flow Decoder ϕ_{dec} is supervised by the mean squared error. The Implicit Correspondence Decoder χ_{dec} is supervised with geodesic-based contrastive loss L_{geo} in Eq. (6).

Decoder Architecture: the decoders for occupancy, correspondence and flow all have a hidden layer dimension of 32. For flow prediction, representation fusion is performed before each fully-connected layer. The intermediate features in the Implicit Flow Decoder are concatenated with the corresponding layer's feature from the Implicit Occupancy Decoder, and passed through a fully-connected layer.

Training Procedures: we use PyTorch [87] for model training and use the Adam optimizer [88] with a learning rate of 10^{-3} and weight decay of 10^{-4}. All methods are trained for 36 epochs. We use the Adam optimizer [88] with a learning rate of 10^{-3} and weight decay of 10^{-4} for all methods. We perform evaluations on the validation set every 4,000 iterations and select the model with the lowest loss for dynamics prediction. We use a batch size of 12 for all baselines and all variants of our ACID model. Geodesic distances between pairs of points are pre-calculated prior to training. Training the ACID model takes 2 days on a TITAN RTX GPU.

Fig. 3: We introduce **PlushSim**, a manipulation environment of realistic volumetric deformable objects. Left: example images from the six types of plush toys, with lighting and layout randomization. Right: tetrahedral approximation used for finite element method simulation.

IV. PlushSim Environment

To quantitatively evaluate deformable objects manipulation in realistic scenes, we use a simulation environment where we can manipulate volumetric deformable objects and store the action information and the resulting ground-truth object motions. We develop a new framework that contains realistic actions and sensor simulation data for deformable plush toys in a visually and layout-wise randomized home scene.

PlushSim Environment Our framework is built with the PhysX engine in NVIDIA's Omniverse Kit. Omniverse features PhysX with GPU-accelerated finite element method (FEM) simulation that represents a deformable body volumetrically as a graph of connected tetrahedrons [85]. We use a fixed resolution for the object's tetrahedral approximation, ensuring consistent object simulation behavior. For physics simulation, we use a fixed time-step resolution of $\frac{1}{150}$ second, ensuring an accurate and consistent behavior for object-to-object and object-to-self collisions. On top of this engine, we created a realistic randomized attic scene and generated six types of plush toys with 78 variations for simulation. The attic scene contains 15 randomized furniture obstacles. For each action sequence, the object pose, scene obstacles layout, and lighting are randomized to avoid overfitting (Fig. 3).

We further implemented a set of manipulation API utilizing a Franka Emika Panda gripper over the simulation engine. The API contains parameterized control commands of `grasp`, `move`, and `release`. The `grasp` command takes the 3D location of the grasp center as input. The `move` command takes in a 3D displacement vector and moves the gripper in a straight line at a constant speed by the displacement. The `release` command takes no parameter. For collecting the dataset, we sample control commands as follows. For `grasp`, we sample uniformly among the object's visible points from the observations. For `move`, we sample a displacement vector in spherical coordinates (r, θ, ϕ), where r has a mean of 2.4 meters and a standard deviation of 0.8 meters; ϕ is uniform among $(0, 2\pi]$; θ has a mean of $\frac{\pi}{4}$ and a standard deviation of $\frac{\pi}{6}$. The gripper is moving at a constant speed of 0.5 meters per second. We reset the scene and perform domain randomization after every 15 commands. We collect the data in parallel with 6 Titan RTX GPUs for 72 hours, resulting in a dataset of 17,665 actions. We show in Fig. 4 an example action sequence where the gripper performs `grasp`, `move`, and `release`.

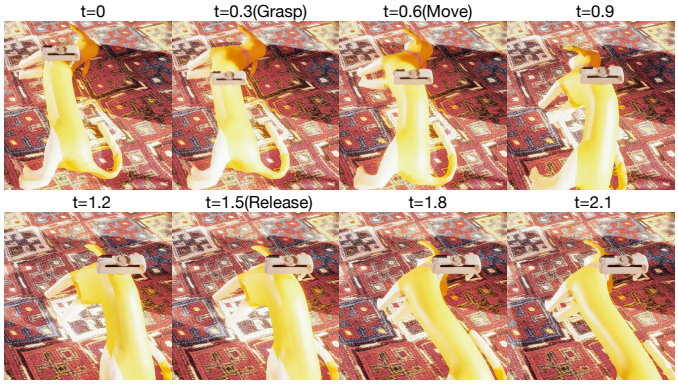

Fig. 4: An example control command sequence. The sequence consists of `grasp`, `move`, and `release`. We visualize the scene every 0.3 seconds. We show more example sequences in the supplementary video.

	Action-Conditional Visual Dynamics				
	Dynamics		Correspondence		Ranking
Method	vis↓	full↓	FMR↑	acc.↑	Kendall τ↑
Pixel-flow [27]	0.525	-	-	-	-0.002
Voxel-flow [29]	0.492	0.488	-	-	0.278
ACID-No Corr	0.434	0.385	-	-	0.534
+Deep shells [67]	-	-	0.085	0.035	0.516
+FPFH [89]	-	-	0.027	0.052	0.511
+InfoNCE[72]	-	-	0.081	0.061	0.510
+FCGF [71]	-	-	0.665	0.204	0.526
ACID-e2e-w/ [72]	0.486	0.434	0.543	0.191	0.512
ACID-e2e-w/ [71]	0.486	0.436	0.751	0.240	0.527
ACID-w/o fusion	0.457	0.409	0.718	0.174	0.541
ACID (Chamfer)	-	-	-	-	**0.547**
ACID (Ours)	**0.425**	**0.372**	**0.815**	**0.257**	0.544

TABLE I: **Visual Dynamics Quantitative Comparison.** We report dynamics prediction, correspondence prediction, and action-ranking performances for baselines and our model in the test set. The results indicate that our implicit representations formulation consistently outperform existing approaches.

Train/Test split. For visual dynamics model evaluation, we use five types of plush toys for training and one for testing, which leads to 14,524 actions for training and 3,141 for testing. For the manipulation evaluation, we randomly sample 216 start and target configurations for rearrangement, each of which is equipped with 256 sampled action sequences.

V. Experiments

We evaluate each component of our ACID model: geometry, dynamics, and correspondence learning. Specially, we aim at answering the following key questions:

- (Sec. V-A) does our model with implicit representations of geometry and dynamics outperform prior work using explicit representations?
- (Sec. V-B) does our geodesics-based correspondence module outperform existing shape correspondence methods?

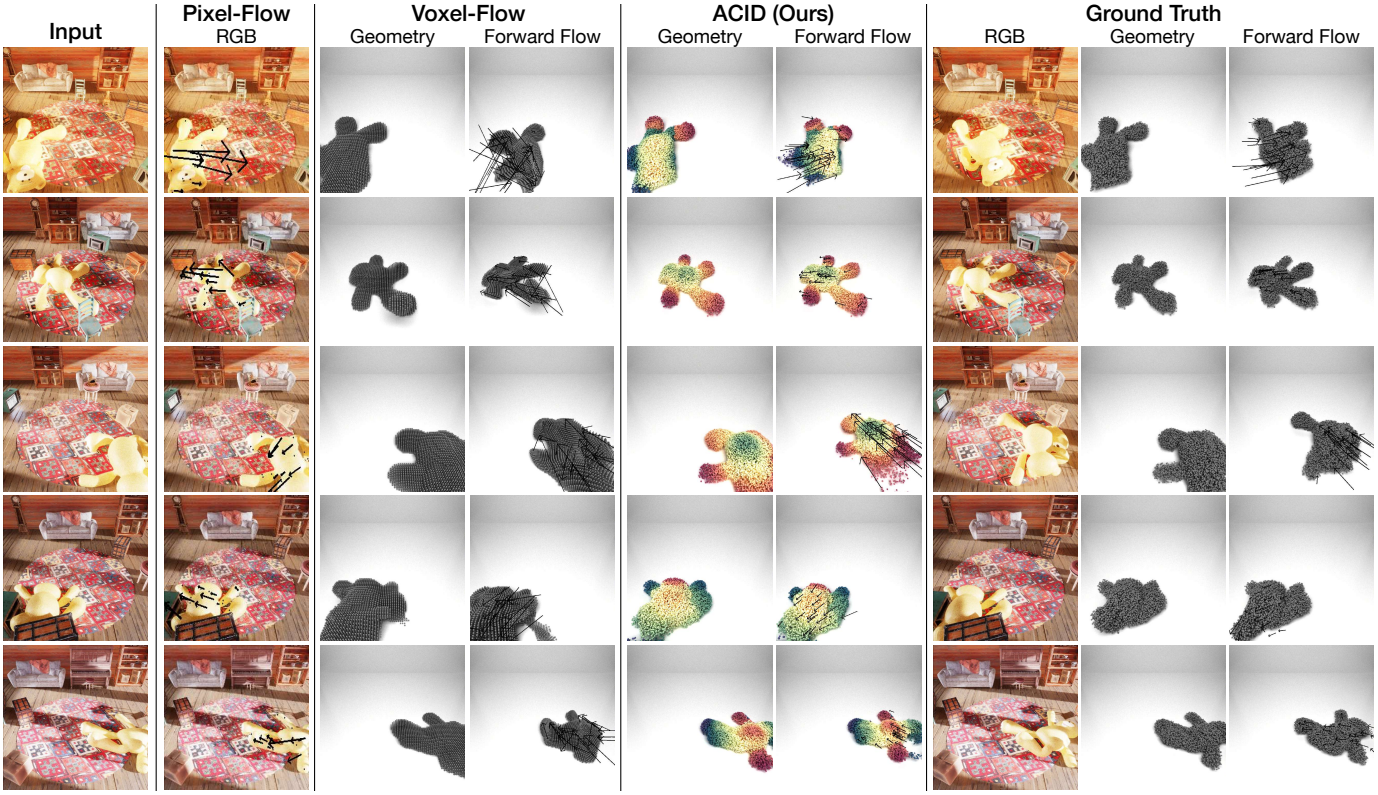

Fig. 5: **Action-Conditional Visual Dynamics**. Qualitative comparison of ACID with `Pixel-Flow` and `Voxel-Flow` on the unseen toy type (bear). Flows are are visualized as black arrows. For ACID, we additionally show visualization of the t-SNE color-coded correspondence feature.

- (Sec. V-C) does the high performance of each component translate to manipulation success?

Overall, we show that ACID outperforms baselines in all three components. Moreover, combining these components increases the task success rate in a volumetric deformable object manipulation task by 30% over the strongest baseline. Finally, we discuss ablation studies and our limitations.

A. Geometry and Dynamics Evaluations

We conduct quantitative evaluations for geometry and dynamics prediction (Table I) and visualize qualitative results of different models in Fig. 5.

Geometry Evaluation: We adapt the state-of-the-art voxel-based action-conditional visual dynamics models for rigid-body into a deformable setting as `Voxel-flow` [29], which represents the entire scene as a dense voxel grid of size $128 \times 128 \times 48$. The partial point cloud observation is encoded into a truncated signed distance function (TSDF), which is used to predict the instance mask of the object's full geometry and the scene flow of each pixel. To control the effect of jointly learning correspondences, we compare ACID model without contrastive supervision `ACID-No Corr`. We follow the standard procedure and report shape reconstruction quality with the mean Intersection over Union (mIoU) metric. `Voxel-flow` achieves a mIoU of 0.643 in the test set, while `ACID-No Corr` achieves a mIoU of 0.796. The results suggest that our implicit occupancy representation of geometry is more suitable for deformable shape reconstruction than previous explicit representation.

Dynamics Evaluation: Following prior work [29], we use the Mean Squared Error (MSE) to evaluate the predicted 3D scene flow and provide two metrics: MSE averaged over visible surfaces of the object of interest (*vis*), and MSE averaged over all points with the object of interest (*full*). We additionally compare with image-based approaches `Pixel-flow`, adapted from SE3-Nets [27]. As shown in Table I, `ACID-No Corr` achieves the best flow prediction performance of 0.434 for visible surface and 0.385 for full object. `Voxel-flow` performs significantly worse, with 0.492 for *vis* and 0.488 for *full*. This further suggests the usefulness of an implicit representation when modeling deformable objects and their dynamics. `Pixel-flow` performs the worst with 0.525 MSE for *vis*, corroborating that modeling deformable object dynamics benefits from reasoning over full object shape.

Further, we observe that `ACID (Ours)`, which is trained jointly with correspondence, achieves better flow prediction performance of 0.425 for *vis* and 0.372 for *full*. By fusing the representations from the implicit correspondence decoder as well as the final embedding, the model can leverage auxiliary information for dynamics prediction. We ablate the effect of representation fusion by training ACID without representation fusion `ACID-w/o fusion`, which indeed achieves an infe-

rior dynamics prediction of 0.457 for *vis* and 0.409 for *full*.

B. Correspondence Evaluations

We evaluate our geodesics-based correspondence against state-of-the-art shape correspondence baselines:

- `FPFH` [89]: uses an oriented histogram on pairwise geometric properties, which is widely adopted in robotics frameworks like ROS [90].
- `InfoNCE` [72]: state-of-the-art per-point dense feature learnt from a contrastive PointInfoNCE Loss.
- `FCGF` [71]: convolutional contrastive geometric features that achieves state-of-the-art performances in registration.
- `Deep-Shells` [67]: state-of-the-art mesh-based correspondence model.

To evaluate the baselines, we used our model to perform shape reconstruction and retrieve the full object point clouds. For `Deep-Shells`, we use marching cube to generate the full mesh. We used author-provided pretrained weight for `Deep-Shells`. Following FCGF [71], we evaluate the feature-match recall with standard parameters [70], which measures the percentage of matched pairs that can be recovered with high confidence. We additionally measure correspondence accuracy. A point is counted as matched accurately if its match is 5cm within the ground truth match. As indicated in Table I, our geodesics approach performs the best with an FMR of 0.815 and a correspondence accuracy of 0.257, significantly outperforms prior works. `FCGF` achieves the best performance amongst the baseline, with an FMR of 0.665 and accuracy of 0.204.

We additionally examine different contrastive loss variations from prior works without geodesics [71], [72]. The models are trained by only substituting the loss term for correspondence while keeping everything else unchanged. `ACID-e2e-w/`[71] and `ACID-e2e-w/`[72] perform worse than our geodesics-based contrastive loss. Interestingly, we no longer see the benefit of dynamics prediction for the ablated models. In fact, jointly training dynamics with correspondence [71], [72] decreases dynamics prediction performance. It indicates that understanding geodesics plays a vital role in dynamics prediction. We made an interesting observation that jointly training dynamics and correspondence also improves correspondence prediction (`Ours` vs.`w/o-fusion`), further suggesting the synergies between correspondence and dynamics learning.

C. Manipulation Evaluations

Task Setup: we evaluate our models and all baselines in a volumetric deformable object manipulation task (Fig. 6). The scene consists of a deformable object of interest of a held-out type randomly posed in a randomized environment. The robot performs grasp-move-release actions as discussed in Sec. III-A. The task is to rearrange the object of interest into the target configuration specified by an image. We define task success as the final configuration is within $0.5m$ from the target configuration.

Following prior work [29], for each start and target configuration, we sample 256 action sequences around the object of interest with a length of 3. We compute the cost for all action sequences and select the one with the lowest cost for each model. For models with correspondence prediction, the cost is measured as the corresponded distance d_{corr} (Eq. (9)) between rolled-out states and target states. For models without correspondence, the Chamfer distance is used for the cost. We further ablate our full model that was trained with correspondence prediction to use Chamfer distance as the cost measure `ACID (Chamfer)`.

	Manipulation Performance				
	Plan Execution Result				Success Rate↑
Method	mIoU↑	F1↑	Chamfer↓	d_{corr}↓	
Pixel-flow [27]	0.247	0.435	1.210	1.720	19.4
Voxel-flow [29]	0.281	0.495	1.005	1.459	22.2
ACID-No Corr	0.401	0.647	0.548	1.045	47.2
+Deep shells [67]	0.341	0.592	0.627	1.174	28.5
+FPFH [89]	0.304	0.558	0.678	1.284	25.7
+InfoNCE[72]	0.314	0.566	0.660	1.268	28.5
+FCGF [71]	0.382	0.628	0.580	1.082	45.7
ACID-e2e-w/ [72]	0.363	0.602	0.650	1.196	41.6
ACID-e2e-w/ [71]	0.330	0.574	0.676	1.273	30.5
ACID-w/o fusion	0.358	0.605	0.605	1.136	36.1
ACID (Chamfer)	0.435	0.701	0.451	0.947	54.3
ACID (Ours)	**0.438**	**0.713**	**0.426**	**0.936**	**55.6**

TABLE II: **Manipulation Quantitative Comparison.** We report the performances of plan execution and success rate in a manipulation task for baselines and variations of our model.

	Trained Types					Test
Method						
Pixel-flow [27]	19.4	36.1	11.1	2.7	22.2	19.4
Voxel-flow [29]	27.8	27.9	13.8	11.1	8.3	22.2
ACID-No Corr	34.3	45.7	14.2	11.4	31.4	36.1
ACID (Ours)	**41.6**	**50.0**	**16.7**	**13.9**	**38.9**	**55.6**

TABLE III: Manipulation task success rate by the toy type.

Roll-out Ranking: we first evaluate how accurately each model ranks the action sequences. We report the mean Kendall's τ [91] across 256 sampled sequences. It measures how well the predicted ranking agrees with the ground truth, with values close to 1 indicating strong agreement and -1 indicating strong disagreement. Variations of ACID model consistently show strong agreement with ground truth ranking, with an 0.26 increase over `Voxel-Flow`.

Plan Execution Results: each model selects an action sequence based on the predicted lowest cost, we evaluate the selected action sequence's performance after being executed. We report mIoU, F-score [92] (F1), Chamfer distance, and d_{corr} in Eq. 9 using the ground truth correspondence mapping ξ_{gt} provided by the simulator. Table II shows that our model achieves the best execution results. As indicated by our performance over ablation baseline *ACID (Chamfer)*, beyond the benefit of learning better feature representations, using d_{corr} for action sequence cost measure also results in better model performance. Our model also achieves the best success rate, with a 30% increase over the strongest existing visual dynamics model, `Voxel-Flow`. Visualization of example

Fig. 6: **Example action sequence roll-outs of our model:** Start and target configurations are specified as images. Future states are rolled out conditioned on action sequence. A cost measure selects the best action sequence, which is executed to get the final result.

roll-outs are shown in Fig. 6.

Per-Type Performance: We further report the planning success rate for five training toy types (see Table III). As the table indicates, our model consistently outperforms prior approaches and self baseline. Interestingly, inter-type performances have a high variance, with smaller objects (dogs and rabbits) showing a lower success rate. It implies further challenges of modeling objects of various scales and reasoning at different resolutions.

D. Discussion of Limitations

Several limitations of ACID can be investigated and improved in future work.

First, although we build simulation environments of volumetric deformable objects with a high degree of visual and physical realism, applying ACID to real-world deformable objects requires careful engineering. Sim2real transfer of volumetric deformable object manipulation presents unique challenges compared to rigid bodies. First, object dynamics are higher dimensional and complex (flow fields *vs*. SE(3)), which could enlarge the reality gap. Secondly, primitive actions like grasping for volumetric deformable object remains a challenging problem [93]. Nonetheless, we believe recent advances in sim2real transfer methods for 1D and 2D deformable objects [33], [94], [95] could offer insight into closing the reality gap for volumetric deformable objects.

Second, since we randomly sample action trajectories, the dataset lacks variety towards the long-tail of fine-grained manipulation behaviors (*e.g.*, twisting the toy snake into a loop). Manipulating object into a complex configuration is difficult for the current approach, and incorporating long-tail actions into training poses an exciting challenge for future work.

VI. Conclusion

We introduce ACID, an action-conditional visual dynamics model for volumetric deformable objects manipulation based on structured implicit neural representations. We proposed novel techniques for learning joint representations of dynamics field, occupancy field, and correspondence embedding field for action-conditional dynamics. Our results suggest that these techniques combined led to better generalization and higher manipulation performance. In a broader scope, these promising results shed light on how to build intelligent robots that can effectively reason about realistic deformable objects and their dynamics. One of the possible future directions is to model finer-grained dynamics and deploy the learned model on physical hardware.

Acknowledgement: We would like to sincerely thank Michelle Lu, Philipp Reist, and Cheng Low for help with Omniverse Kit simulation. We would like to sincerely thank De-An Huang, Linxi Fan, Zhiding Yu, the NVIDIA AI-ALGO team, other NVIDIA colleagues and colleagues from the Stanford Geometric Computation Group for the discussion and constructive suggestions.

REFERENCES

[1] Jose Sanchez, Juan-Antonio Corrales, Belhassen-Chedli Bouzgarrou, and Youcef Mezouar. Robotic manipulation and sensing of deformable objects in domestic and industrial applications: a survey. *The International Journal of Robotics Research*, 37(7):688–716, 2018.

[2] P Jiménez. Survey on model-based manipulation planning of deformable objects. *Robotics and computer-integrated manufacturing*, 28(2):154–163, 2012.

[3] Fouad F Khalil and Pierre Payeur. *Dexterous robotic manipulation of deformable objects with multi-sensory feedback-a review*. IntechOpen, 2010.

[4] Patricia Moore and Derek Molloy. A survey of computer-based deformable models. In *International Machine Vision and Image Processing Conference (IMVIP 2007)*, pages 55–66. IEEE, 2007.

[5] Mozafar Saadat and Ping Nan. Industrial applications of automatic manipulation of flexible materials. *Industrial Robot: An International Journal*, 2002.

[6] Yunzhu Li, Jiajun Wu, Russ Tedrake, Joshua B. Tenenbaum, and Antonio Torralba. Learning particle dynamics for manipulating rigid bodies, deformable objects, and fluids. In *ICLR*, 2019.

[7] Yunzhu Li, Jiajun Wu, Jun-Yan Zhu, Joshua B Tenenbaum, Antonio Torralba, and Russ Tedrake. Propagation networks for model-based control under partial observation. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 1205–1211. IEEE, 2019.

[8] Mark Moll and Lydia E Kavraki. Path planning for deformable linear objects. *IEEE Transactions on Robotics*, 22(4):625–636, 2006.

[9] John Schulman, Alex Lee, Jonathan Ho, and Pieter Abbeel. Tracking deformable objects with point clouds. In *2013 IEEE International Conference on Robotics and Automation*, pages 1130–1137. IEEE, 2013.

[10] Olivier Roussel, Andy Borum, Michel Taix, and Timothy Bretl. Manipulation planning with contacts for an extensible elastic rod by sampling on the submanifold of static equilibrium configurations. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3116–3121. Ieee, 2015.

[11] Te Tang, Yongxiang Fan, Hsien-Chung Lin, and Masayoshi Tomizuka. State estimation for deformable objects by point registration and dynamic simulation. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2427–2433. IEEE, 2017.

[12] Mengyuan Yan, Yilin Zhu, Ning Jin, and Jeannette Bohg. Self-supervised learning of state estimation for manipulating deformable linear objects. *IEEE robotics and automation letters*, 5(2):2372–2379, 2020.

[13] Stephen Miller, Jur Van Den Berg, Mario Fritz, Trevor Darrell, Ken Goldberg, and Pieter Abbeel. A geometric approach to robotic laundry folding. *The International Journal of Robotics Research*, 31(2):249–267, 2012.

[14] Yinxiao Li, Yan Wang, Michael Case, Shih-Fu Chang, and Peter K Allen. Real-time pose estimation of deformable objects using a volumetric approach. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1046–1052. IEEE, 2014.

[15] Yinxiao Li, Chih-Fan Chen, and Peter K Allen. Recognition of deformable object category and pose. In *ICRA*, pages 5558–5564. IEEE, 2014.

[16] Ioannis Mariolis, Georgia Peleka, Andreas Kargakos, and Sotiris Malassiotis. Pose and category recognition of highly deformable objects using deep learning. In *2015 International conference on advanced robotics (ICAR)*, pages 655–662. IEEE, 2015.

[17] Yinxiao Li, Yan Wang, Yonghao Yue, Danfei Xu, Michael Case, Shih-Fu Chang, Eitan Grinspun, and Peter K Allen. Model-driven feedforward prediction for manipulation of deformable objects. *IEEE Transactions on Automation Science and Engineering*, 15(4):1621–1638, 2018.

[18] Xingyu Lin, Yufei Wang, and David Held. Learning visible connectivity dynamics for cloth smoothing. In *CoRL*, 2021.

[19] David Navarro-Alarcon, Yun-hui Liu, Jose Guadalupe Romero, and Peng Li. On the visual deformation servoing of compliant objects: Uncalibrated control methods and experiments. *The International Journal of Robotics Research*, 33(11):1462–1480, 2014.

[20] Antoine Petit, Vincenzo Lippiello, and Bruno Siciliano. Real-time tracking of 3d elastic objects with an rgb-d sensor. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3914–3921. IEEE, 2015.

[21] David Navarro-Alarcon, Hiu Man Yip, Zerui Wang, Yun-Hui Liu, Fangxun Zhong, Tianxue Zhang, and Peng Li. Automatic 3-d manipulation of soft objects by robotic arms with an adaptive deformation model. *IEEE Transactions on Robotics*, 32(2):429–441, 2016.

[22] David Navarro-Alarcon and Yun-Hui Liu. Fourier-based shape servoing: A new feedback method to actively deform soft objects into desired 2-d image contours. *IEEE Transactions on Robotics*, 34(1):272–279, 2017.

[23] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Occupancy flow: 4d reconstruction by learning particle dynamics. In *ICCV*, pages 5379–5389, 2019.

[24] Chiyu Jiang, Jingwei Huang, Andrea Tagliasacchi, and Leonidas Guibas. Shapeflow: Learnable deformations among 3d shapes. In *NeurIPS*, 2020.

[25] Keunhong Park, Utkarsh Sinha, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Steven M Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5865–5874, 2021.

[26] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10318–10327, 2021.

[27] Arunkumar Byravan and Dieter Fox. Se3-nets: Learning rigid body motion using deep neural networks. In *ICRA*, pages 173–180, 2017.

[28] Hsiao-Yu Fish Tung, Zhou Xian, Mihir Prabhudesai, Shamit Lal, and Katerina Fragkiadaki. 3d-oes: Viewpoint-invariant object-factorized environment simulators. In *CoRL*, 2020.

[29] Zhenjia Xu, Zhanpeng He, Jiajun Wu, and Shuran Song. Learning 3d dynamic scene representations for robot manipulation. In *CoRL*, 2020.

[30] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[31] Chelsea Finn and Sergey Levine. Deep visual foresight for planning robot motion. In *ICRA*, pages 2786–2793, 2017.

[32] Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *ICML*, pages 2555–2565. PMLR, 2019.

[33] Ryan Hoque, Daniel Seita, Ashwin Balakrishna, Aditya Ganapathi, Ajay Kumar Tanwani, Nawid Jamali, Katsu Yamane, Soshi Iba, and Ken Goldberg. Visuospatial foresight for physical sequential fabric manipulation. 2021.

[34] Wilson Yan, Ashwin Vangipuram, Pieter Abbeel, and Lerrel Pinto. Learning predictive representations for deformable objects using contrastive estimation. In *CoRL*, 2020.

[35] Martina Lippi, Petra Poklukar, Michael C Welle, Anastasiia Varava, Hang Yin, Alessandro Marino, and Danica Kragic. Latent space roadmap for visual action planning of deformable and rigid object manipulation. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5619–5626. IEEE, 2020.

[36] Yunzhu Li, Antonio Torralba, Anima Anandkumar, Dieter Fox, and Animesh Garg. Causal discovery in physical systems from videos. *NeurIPS*, 33, 2020.

[37] Yunzhu Li, Shuang Li, Vincent Sitzmann, Pulkit Agrawal, and Antonio Torralba. 3d neural scene representations for visuomotor control. In *CoRL*, 2021.

[38] Peter W Battaglia, Razvan Pascanu, Matthew Lai, Danilo Rezende, and Koray Kavukcuoglu. Interaction networks for learning about objects, relations and physics. *arXiv preprint arXiv:1612.00222*, 2016.

[39] Damian Mrowca, Chengxu Zhuang, Elias Wang, Nick Haber, Li Fei-Fei, Joshua B Tenenbaum, and Daniel LK Yamins. Flexible neural representation for physics prediction. In *NeurIPS*, 2018.

[40] Benjamin Ummenhofer, Lukas Prantl, Nils Thuerey, and Vladlen Koltun. Lagrangian fluid simulation with continuous convolutions. In *ICLR*, 2019.

[41] Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter Battaglia. Learning to simulate complex physics with graph networks. In *International Conference on Machine Learning*, pages 8459–8468. PMLR, 2020.

[42] Christopher B Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *ECCV*. Springer, 2016.

[43] Gernot Riegler, Ali Osman Ulusoy, and Andreas Geiger. Octnet: Learning deep 3d representations at high resolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3577–3586, 2017.

[44] Zak Murez, Tarrence van As, James Bartolozzi, Ayan Sinha, Vijay Badrinarayanan, and Andrew Rabinovich. Atlas: End-to-end 3d scene re-

construction from posed images. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VII 16*, pages 414–431. Springer, 2020.

[45] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. Mvsnet: Depth inference for unstructured multi-view stereo. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 767–783, 2018.

[46] Clément Godard, Oisin Mac Aodha, and Gabriel J Brostow. Unsupervised monocular depth estimation with left-right consistency. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 270–279, 2017.

[47] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017.

[48] Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *arXiv preprint arXiv:1706.02413*, 2017.

[49] Angjoo Kanazawa, Shubham Tulsiani, Alexei A Efros, and Jitendra Malik. Learning category-specific mesh reconstruction from image collections. In *ECCV*, pages 371–386, 2018.

[50] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 165–174, 2019.

[51] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *CVPR*, pages 4460–4470, 2019.

[52] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *CVPR*, pages 5939–5948, 2019.

[53] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene representation networks: continuous 3d-structure-aware neural scene representations. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 1121–1132, 2019.

[54] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *Advances in Neural Information Processing Systems*, 33, 2020.

[55] Songyou Peng, Michael Niemeyer, Lars Mescheder, Marc Pollefeys, and Andreas Geiger. Convolutional occupancy networks. In *ECCV*, pages 523–540. Springer, 2020.

[56] Boyang Deng, John P Lewis, Timothy Jeruzalski, Gerard Pons-Moll, Geoffrey Hinton, Mohammad Norouzi, and Andrea Tagliasacchi. Nasa neural articulated shape approximation. In *ECCV*, pages 612–628, 2020.

[57] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[58] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. Smpl: A skinned multi-person linear model. *ACM TOG*, 34(6):1–16, 2015.

[59] Oliver Van Kaick, Hao Zhang, Ghassan Hamarneh, and Daniel Cohen-Or. A survey on shape correspondence. In *Computer graphics forum*, volume 30, pages 1681–1707. Wiley Online Library, 2011.

[60] Yusuf Sahillioğlu. Recent advances in shape correspondence. *The Visual Computer*, 36(8):1705–1721, 2020.

[61] Davide Boscaini, Jonathan Masci, Emanuele Rodolà, and Michael Bronstein. Learning shape correspondence with anisotropic convolutional neural networks. *NeurIPS*, 29, 2016.

[62] Or Litany, Tal Remez, Emanuele Rodola, Alex Bronstein, and Michael Bronstein. Deep functional maps: Structured prediction for dense shape correspondence. In *ICCV*, pages 5659–5667, 2017.

[63] Federico Monti, Davide Boscaini, Jonathan Masci, Emanuele Rodola, Jan Svoboda, and Michael M Bronstein. Geometric deep learning on graphs and manifolds using mixture model cnns. In *CVPR*, pages 5115–5124, 2017.

[64] Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. 3d-coded: 3d correspondences by deep deformation. In *ECCV*, pages 230–246, 2018.

[65] Oshri Halimi, Or Litany, Emanuele Rodola, Alex M Bronstein, and Ron Kimmel. Unsupervised learning of dense shape correspondence. In *CVPR*, pages 4370–4379, 2019.

[66] Nicolas Donati, Abhishek Sharma, and Maks Ovsjanikov. Deep geometric functional maps: Robust feature learning for shape correspondence. In *CVPR*, pages 8592–8601, 2020.

[67] Marvin Eisenberger, Aysim Toker, Laura Leal-Taixé, and Daniel Cremers. Deep shells: Unsupervised shape correspondence with optimal transport. In *NeurIPS*, 2020.

[68] Zhangsihao Yang, Or Litany, Tolga Birdal, Srinath Sridhar, and Leonidas Guibas. Continuous geodesic convolutions for learning on 3d shapes. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 134–144, 2021.

[69] Marvin Eisenberger, David Novotny, Gael Kerchenbaum, Patrick Labatut, Natalia Neverova, Daniel Cremers, and Andrea Vedaldi. Neuromorph: Unsupervised shape interpolation and correspondence in one go. In *CVPR*, pages 7473–7483, 2021.

[70] Haowen Deng, Tolga Birdal, and Slobodan Ilic. Ppfnet: Global context aware local features for robust 3d point matching. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 195–205, 2018.

[71] Christopher Choy, Jaesik Park, and Vladlen Koltun. Fully convolutional geometric features. In *ICCV*, pages 8958–8966, 2019.

[72] Saining Xie, Jiatao Gu, Demi Guo, Charles R Qi, Leonidas Guibas, and Or Litany. Pointcontrast: Unsupervised pre-training for 3d point cloud understanding. In *ECCV*, pages 574–591, 2020.

[73] Demetri Terzopoulos and Kurt Fleischer. Deformable models. *The visual computer*, 4(6):306–331, 1988.

[74] Demetri Terzopoulos and Dimitri Metaxas. Dynamic 3d models with local and global deformations: deformable superquadrics. *IEEE Transactions on pattern analysis and machine intelligence*, 13(7):703–714, 1991.

[75] Matthias Müller, Bruno Heidelberger, Marcus Hennix, and John Ratcliff. Position based dynamics. *Journal of Visual Communication and Image Representation*, 18(2):109–118, 2007.

[76] Andrew Nealen, Matthias Müller, Richard Keiser, Eddy Boxerman, and Mark Carlson. Physically based deformable models in computer graphics. In *Computer graphics forum*, 2006.

[77] Miles Macklin, Matthias Müller, Nuttapong Chentanez, and Tae-Yong Kim. Unified particle physics for real-time applications. *ACM Transactions on Graphics (TOG)*, 33(4):1–12, 2014.

[78] Tom Erez, Yuval Tassa, and Emanuel Todorov. Simulation tools for model-based robotics: Comparison of bullet, havok, mujoco, ode and physx. In *ICRA*, pages 4397–4404. IEEE, 2015.

[79] Yuanming Hu, Tzu-Mao Li, Luke Anderson, Jonathan Ragan-Kelley, and Frédo Durand. Taichi: a language for high-performance computation on spatially sparse data structures. *ACM Transactions on Graphics (TOG)*, 38(6):1–16, 2019.

[80] Yuanming Hu, Luke Anderson, Tzu-Mao Li, Qi Sun, Nathan Carr, Jonathan Ragan-Kelley, and Fredo Durand. Difftaichi: Differentiable programming for physical simulation. In *ICLR*, 2019.

[81] Yuanming Hu, Jiancheng Liu, Andrew Spielberg, Joshua B Tenenbaum, William T Freeman, Jiajun Wu, Daniela Rus, and Wojciech Matusik. Chainqueen: A real-time differentiable physical simulator for soft robotics. In *ICRA*, pages 6265–6271. IEEE, 2019.

[82] Xingyu Lin, Yufei Wang, Jake Olkin, and David Held. Softgym: Benchmarking deep reinforcement learning for deformable object manipulation. In *CoRL*, 2020.

[83] Chuang Gan, Jeremy Schwartz, Seth Alter, Martin Schrimpf, James Traer, Julian De Freitas, Jonas Kubilius, Abhishek Bhandwaldar, Nick Haber, Megumi Sano, et al. Threedworld: A platform for interactive multi-modal physical simulation. 2020.

[84] Zhiao Huang, Yuanming Hu, Tao Du, Siyuan Zhou, Hao Su, Joshua B Tenenbaum, and Chuang Gan. Plasticinelab: A soft-body manipulation benchmark with differentiable physics. In *ICLR*, 2021.

[85] Nvidia, omniverse.

[86] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[87] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *NeurIPS*, 32:8026–8037, 2019.

[88] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[89] Radu Bogdan Rusu, Nico Blodow, and Michael Beetz. Fast point feature histograms (fpfh) for 3d registration. In *ICRA*, pages 3212–3217. IEEE, 2009.

[90] Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, Andrew Y Ng, et al. Ros: an open-source robot operating system. In *ICRA workshop on open source software*, volume 3, page 5. Kobe, Japan, 2009.

[91] Maurice G Kendall. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93, 1938.

[92] Maxim Tatarchenko, Stephan R Richter, René Ranftl, Zhuwen Li, Vladlen Koltun, and Thomas Brox. What do single-view 3d reconstruction networks learn? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3405–3414, 2019.

[93] Isabella Huang, Yashraj Narang, Clemens Eppner, Balakumar Sundaralingam, Miles Macklin, Tucker Hermans, and Dieter Fox. Defgraspsim: Simulation-based grasping of 3d deformable objects. *arXiv preprint arXiv:2107.05778*, 2021.

[94] Yilin Wu, Wilson Yan, Thanard Kurutach, Lerrel Pinto, and Pieter Abbeel. Learning to manipulate deformable objects without demonstrations. *arXiv preprint arXiv:1910.13439*, 2019.

[95] Priya Sundaresan, Jennifer Grannen, Brijen Thananjeyan, Ashwin Balakrishna, Michael Laskey, Kevin Stone, Joseph E Gonzalez, and Ken Goldberg. Learning rope manipulation policies using dense object descriptors trained on synthetic depth data. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9411–9418. IEEE, 2020.

Supplementary Material for *ACID: Action-Conditional Implicit Visual Dynamics for Deformable Object Manipulation*

In this supplementary document, we provide detailed definitions for all evaluation metrics for geometry prediction, dynamics prediction, correspondence evaluation, ranking evaluation, and plan execution evaluation (Section I). Additional qualitative results for comparison with existing action-conditional visual dynamics models can be found in Section II. Additional action sequence roll-outs for our ACID model can be found in Section III.

I. Metrics

In this section, we provide the formal definitions of the metrics that we use for evaluation. We define `mIoU` for geometry evaluation; `vis` and `full` mean-squared error for dynamics evaluation; `FMR` and `acc` for correspondence evaluation; `Kendall's` τ for ranking evaluation; `F-score` and `Chamfer` for planning execution evaluation. d_{corr} and `success rate` have been defined in the main paper.

A. Geometry Metric

mIoU: we follow Peng *et al.*[**?**] for volumetric interaction over union calculation. Let $\mathcal{S}_{pred}$ and $\mathcal{S}_{GT}$ be the set of all points that are inside of the predicted and ground-truth shapes, respectively. The volumetric IoU is the volume of two shapes' intersection divided by the volume of their union:

$$\texttt{IoU}(\mathcal{S}_{pred}, \mathcal{S}_{GT}) = \frac{|\mathcal{S}_{pred} \cap \mathcal{S}_{GT}|}{|\mathcal{S}_{pred} \cup \mathcal{S}_{GT}|}$$

We randomly sample 100k points from the bounding boxes and determine if the points lie inside or outside $\mathcal{S}_{pred}$ and $\mathcal{S}_{GT}$, respectively. The `mIoU` is the mean volumetric intersection over union across test set.

B. Dynamics Metric

vis and full MSE: we follow Xu *et al.* [**?**] for flow mean-squared error. Let $\mathcal{S}_{GT}$ be the set of all points that are inside of the ground-truth shapes, and $\mathcal{V}_{GT}$ be the set of all points that are visible in the camera for the ground-truth shapes. And let f_p be the predicted flow $\in \mathbb{R}^3$, and $\hat{f}_p$ be the ground-truth flow $\in \mathbb{R}^3$. The `vis` and `full` mean-squared error is defined as:

$$\texttt{vis} = \sum_{p \in \mathcal{V}_{GT}} \frac{(f_p - \hat{f}_p)^2}{|\mathcal{V}_{GT}|} \quad \texttt{full} = \sum_{p \in \mathcal{S}_{GT}} \frac{(f_p - \hat{f}_p)^2}{|\mathcal{S}_{GT}|}$$

C. Correspondence Metric

FMR (Feature-match Recall): we follow Deng *et al.*[**?**] for feature-match recall calculation, which measures the percentage of fragment pairs that is recovered with high confidence. Let ξ_{gt} be the ground truth mapping and ξ_{pred} be the predicted mapping. Mathematically, FMR is calculated as:

$$R = \mathbb{1}([\frac{1}{\mathcal{S}_{GT}} \sum_{p \in \mathcal{S}_{GT}} \mathbb{1}(\|\xi_{gt}(p) - \xi_{pred}(p)\| < \tau_1)] > \tau_2)$$

And R is averaged across test set. $\tau_1 = 0.1m$ inlier distance threshold and $\tau_2 = 0.05$ or 5% is the inlier recall threshold, following prior work [**?**], [**?**].

acc. (accuracy): is defined as the percentage of points that are correctly matched, where being correctly matched means that the predicted matched point is within $0.05m$ from the ground-truth matched point:

$$\texttt{acc.} = \frac{1}{\mathcal{S}_{GT}} \sum_{p \in \mathcal{S}_{GT}} \mathbb{1}(\|\xi_{gt}(p) - \xi_{pred}(p)\| < 0.05$$

D. Ranking Metric

Kendall's τ: we use `Kendall's` τ [**?**] for ranking evaluation, which has been examined in a computer vision context before [**?**]. Kendall's τ is a measure of the correspondence between two rankings, with values close to 1 indicating strong agreement, and values close to -1 indicating strong disagreement. In our scenarios, we have a predicted ranking of sampled action sequence r_{pred} and a ground-truth ranking of the sampled action sequence r_{gt}. For action sequence i, j, the quadruplet of rank indices $(r_{pred}(i), r_{pred}(j), r_{gt}(i), r_{gt}(j))$ is said to be concordant if $r_{pred}(i) < r_{pred}(j)$ and $r_{gt}(i) < r_{gt}(j)$ or $r_{pred}(i) > r_{pred}(j)$ and $r_{gt}(i) < r_{gt}(j)$. Otherwise it is said to be discordant. And `Kendall's` τ is defined as:

$$\texttt{Kendall's } \tau = \frac{P - Q}{P + Q}$$

where P is the number of concordant pairs, Q the number of discordant pairs.

E. Planning Execution Metric

Chamfer distance: The Chamfer distance is widely used for point set distance calculation [**?**]. For two point sets $S_1, S_2 \subseteq \mathbb{R}^3$, the Chamfer distance is defined as:

$$\texttt{chamfer} = \sum_{x \in S_1} \min_{y \in S_2} \|x - y\|_2^2 + \sum_{y \in S_2} \min_{x \in S_1} \|x - y\|_2^2$$

For us, S_1 represents the object shape specified in the target configuration, S_2 represents the shape of the object after actions are executed.

F-Score: we follow Tatarchenko *et al*.[**?**] for F-score calculation. `Recall` is defined as the number of points in the ground-truth shape that lie within a certain distance to the source shape. `Precision` counts the percentage of points on the source shape that lie within a certain distance to the ground truth. The F-Score is then defined as the harmonic mean between precision and recall:

$$\texttt{F-Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

For us, the source shape is the final shape of the object after the action sequence is executed.

II. Action-Conditional Visual Dynamics Visualizations

We show here more example visualizations of our model comparing to the baselines `Pixel-Flow` and `Voxel-Flow`. As shown in Fig. 1 and Fig. 2, our method consistently outperforms action-condition visual dynamics model baselines.

III. Roll-outs Visualizations

We show here more example visualizations of ACID's planning results. We visualize the roll-outs of the selected action sequences with lowest cost under different start and target configurations. As shown in Fig. 3, our method can estimate the dynamics of the object over multiple action commands and select reasonable action sequence.

Fig. 1: **Action-Conditional Visual Dynamics.** Comparison of ACID with `Pixel-Flow` and `Voxel-Flow` on unseen category.

Fig. 2: **Action-Conditional Visual Dynamics.** More Examples.

Fig. 3: **Examples of the selected action sequence roll-outs of our model.** Start and target configurations are specified as images. Future states are predicted by our visual dynamics model conditioned on 256 sequences of 3-action commands. A cost measure selects the best action sequence to execute, of which the roll-out and final result are visualized in this figure.