IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

Manipulating Elasto-Plastic Objects With 3D Occupancy and Learning-Based Predictive Control

Zhen Zhang[1,2], Xiangyu Chu[1,2], Yunxi Tang[1], Lulu Zhao[3], Jing Huang[1,2], Zhongliang Jiang[4], and K. W. Samuel Au[1,2]

***Abstract*—Manipulating elasto-plastic objects remains a significant challenge due to severe self-occlusion, difficulties of representation, and complicated dynamics. This work proposes a novel framework for elasto-plastic object manipulation with a quasi-static assumption for motions, leveraging 3D occupancy to represent such objects, a learned dynamics model trained with 3D occupancy, and a learning-based predictive control algorithm to address these challenges effectively. We build a novel data collection platform to collect full spatial information and propose a pipeline for generating a 3D occupancy dataset. To infer the 3D occupancy during manipulation, an occupancy prediction network is trained with multiple RGB images supervised by the generated dataset. We design a deep neural network empowered by a 3D convolution neural network (CNN) and a graph neural network (GNN) to predict the complex deformation with the inferred 3D occupancy results. A learning-based predictive control algorithm is introduced to plan the robot's actions, incorporating a novel shape-based action initialization module specifically designed to improve the planner's efficiency. The proposed framework in this paper can successfully shape the elasto-plastic objects into a given goal shape and has been verified in various experiments both in simulation and the real world.**

***Index Terms*—Deformable Object Manipulation, 3D Occupancy, Elasto-Plastic Objects**

I. Introduction

DEFORMABLE object manipulation (DOM) is essential for many applications in the real world, such as household, industrial, and healthcare. These applications often involve handling various types of deformable objects, which can be broadly categorized into two major categories: thin-shell structural objects, such as clothes [1], [2], and ropes [3]; and volumetric objects, such as plasticine [4], [5] and dough [6]. In this paper, we focus on the robot manipulation of elasto-plastic volumetric objects, which deform elastically under small stress and undergo permanent plastic deformation under larger stress, and the objects' dynamics are assumed to follow a quasi-static motion model. Unlike rigid objects which can be represented by 6D pose [7], deformable objects pose challenges in compact state representation and dynamics modeling due to their high degrees of freedom, complex deformations, and significant self-occlusion. Although the geometric representation like surface-level point clouds can provide rich spatial information, it is sparse and semantics-lacking.

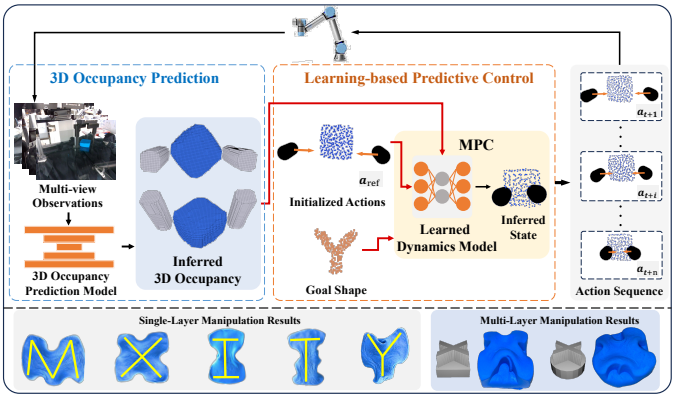


Fig. 1: **Overview of our framework and results.** We predict 3D occupancy from RGB images, learn the dynamics with a deep neural network, and apply learning-based predictive control with shape-based action initialization to deform the object into a goal shape.

In this work, we propose a novel framework to represent 3D deformable objects with dense voxel-grid 3D occupancy and develop modeling and control methods based on this representation. To demonstrate this paradigm, we choose 3D elasto-plastic objects (e.g., plasticine) as our study example, since their deformation is irreversible which poses a significant challenge in shaping them to a desired shape. Specifically, a novel elasto-plastic object manipulation framework is proposed, as shown in Fig. 1. A 3D occupancy prediction network is proposed to generate inferred 3D occupancy from RGB images. Then, a learned dynamics model based on 3D occupancy is obtained via training a deep neural network empowered by 3D convolution neural network (CNN) and graph neural network (GNN). Finally, with a shape-based action initialization module, the model predictive control (MPC) is used to generate an action sequence for manipulation. Extensive experiments are conducted both in the simulator and the real world to evaluate our work. The key contributions are:

- We propose a novel framework to represent and manipulate elasto-plastic objects based on 3D occupancy, which includes a 3D occupancy prediction model, a learned dynamics model, and learning-based predictive control.
- We propose a shape-based action initialization module that uses the geometrical information between the current state and the goal to select actions for MPC to achieve the desired deformations.
- We build a novel and low-cost 3D data collection platform with a transparent operating plane to collect the data of

Manuscript received: January 17, 2025; Revised: April 17, 2025; Accepted: May 20, 2025. This letter was recommended for publication by Editor Júlia Borràs Sol upon evaluation of the reviewers' comments. (*Corresponding author: Xiangyu Chu*)

This work was supported in part by the Multi-scale Medical Robotics Center, AIR@InnoHK and in part by Direct Grants (The Chinese University of Hong Kong) No. 4055245. [1]: Department of Mechanical and Automation Engineering, The Chinese University of Hong Kong, Hong Kong SAR, China. [2]: Multi-scale Medical Robotics Center, Hong Kong SAR, China. [3]: School of Artificial Intelligence, Beijing Normal University, China. [4]: Computer Aided Medical Procedures, Technical University of Munich, Germany.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

3D deformable objects with full spatial information, and propose a pipeline for 3D occupancy dataset generation of deformable objects from multi-view RGB-D.

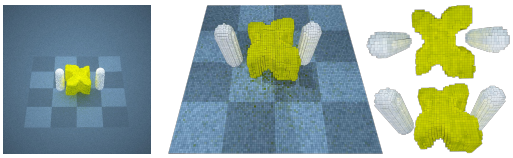
II. Related Work

A. Deformable Object State Representation

In recent works, data-driven manipulation methods have drawn more attention to representing deformable objects in low dimensions. For some thin-shell deformable objects, such as cable and cloth, mesh embedding [8], latent space features [9], and keypoint embedding [3], [10] have been commonly used for their representation. For volumetric deformable objects like clay and tissue, particle-based and point cloud-based representations have gained popularity. For example, Li et al. [11] and Shi et al. [4] proposed using particle-based representation to learn dynamics. Thach et al. [12], [13] and Bartsch et al. [14] employed deep neural networks to process the point cloud of deformable objects. Although these representations can provide 3D spatial information, both particles and point clouds are sparse and semantics-lacking compared with RGB images. To bridge such a gap, Li et al. [15] introduced a representation model that combines a PointNet encoder with a conditional neural radiance field to process point clouds and RGB-Ds. However, this approach introduces the challenge of cross-modality feature fusion, i.e., efficiently integrating features from different modalities. In this work, we propose to use voxel-grid 3D occupancy, a predictable, fine-grained, semantic-rich, and inherently volumetric representation, to indicate the state of deformable objects and facilitate both dynamics modeling and planning for 3D DOM.

B. Deformable Object Manipulation

In addition to representation, modeling state transitions under action is critical for control. Prior works [16]–[20] have explored diverse dynamics models for object manipulation, but face challenges when applied to complex 3D deformable objects. Methods such as Suh and Tedrake [16] and Transporter Networks [19] relied on 2D image-space representations, limiting their ability to capture volumetric deformations and handle occlusions. Xue et al. [17] and Wang et al. [18] targeted granular materials, neglecting cohesive media like clay. The Koopman-based model [20] assumed globally linearizable dynamics, which is unrealistic for elasto-plastic behavior. Recent efforts have focused on learning graph-based dynamics models from low-dimensional features for 3D deformable object manipulation. Prior works [4], [21] utilized GNNs trained on down-sampled internal particles or surface mesh vertices, while Bartsch et al. [22] proposed a GNN-based dynamics prediction pipeline using point cloud clusters. However, these methods either rely on accurate RGB-D-based mesh reconstruction during manipulation [4], [21] or operate solely on surface-level information [21], [22], neglecting internal structural cues. This spatial information sparsity constrains contextual representation, ultimately degrading the performance of the learned dynamics. In contrast, our approach leverages dense occupancy-based representations and 3D CNNs to retain internal spatial information, enabling more accurate modeling of 3D deformable dynamics.



(a) Observation of plasticine (b) 3D occupancy result visualization

Fig. 2: **Inferred 3D occupancy of manipulation scenario in the simulator.** (a) RGB observation. (b) Inferred 3D occupancy visualization. The grey, yellow, and dark blue parts represent the gripper, plasticine, and operating plane, respectively.

III. Approach

A. Problem Formulation

In this work, we aim to use a paralleled 2-finger gripper to manipulate an elasto-plastic object (i.e., plasticine) into a given goal shape s_{goal} in a quasi-static setting with the observation from the multi-view cameras $I = \{I_1, I_2, ..., I_n\}$, where n is the number of cameras. Specifically, we propose to use a dense, fine-grained, and volumetric representation inferred by a neural network with multi-view RGB images as input to indicate the state s of plasticine, which can be represented as $s = \mathcal{N}(I)$, where $\mathcal{N}$ is a neural network. To manipulate the plasticine into the given goal shape s_{goal}, a sequence of actions $\mathcal{A} = \{a_0, a_1, ..., a_{t-1}\}$ will be applied upon the plasticine whose initial state is s_0, and the state of the plasticine will finally transit from s_0 to s_t, where $s_t \approx s_{goal}$ is expected and measured by quantitative metrics. The complex transition of plasticine can be learned with a neural network. Specially, given the inferred current plasticine state s_{t-1} and gripper action a_{t-1}, the state s_t at time step t can be represented as:

$$s_t = \mathcal{G}\left(s_{t-1}, a_{t-1}\right) = \mathcal{G}\left(\mathcal{N}\left(I_{t-1}\right), a_{t-1}\right) \tag{1}$$

where $\mathcal{G}$ represents a trained neural network to approximate the state transition function. We can formulate the manipulation task as a model predictive control with the learned dynamics model. The cost function $\mathcal{L}$ is defined to measure the error between the current state s_t and the given goal state s_{goal}. The sequence of actions $\mathcal{A} = \{a_0, a_1, ..., a_{t-1}\}$ can be optimized by minimizing the cost function: $\mathcal{A}^* = \arg\min \mathcal{L}\left(\mathcal{G}\left(s_0, \mathcal{A}\right), s_{goal}\right)$.

B. 3D Occupancy-based State Representation for DOs

We utilize a voxel grid structure to represent the 3D occupancy of deformable objects. 3D occupancy is an effective representation for reconstructing multi-camera deformable object manipulation scenarios, as shown in Fig. 2. First, 3D occupancy is an inherently dense and fine-grained representation that can provide detailed, semantic (e.g., classes), and spatial information compared with RGB-D images and point clouds, which can be leveraged to extract high-dimensional features for learning the complex state transition dynamics model. Secondly, trained with occluded samples, the neural network can predict occluded areas between grippers and the deformable object according to the rich semantic information without providing depth information. Given multi-view RGB images I^t, where t is the time stamp, the 3D occupancy prediction result can be represented as:

$$\mathcal{O}_t = \mathcal{N}(I_1^t, I_2^t, ..., I_n^t) \tag{2}$$

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

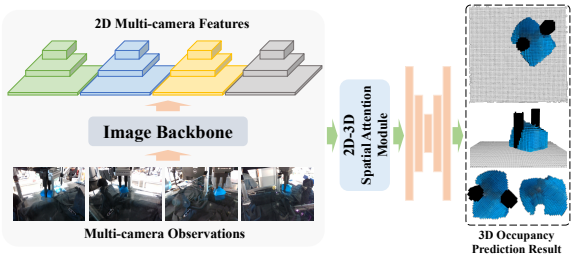


Fig. 3: **3D occupancy prediction framework.** We use four cameras to extract multi-scale features, fuse them with 2D-3D spatial attention, and predict 3D occupancy, supervised by the ground truth.

where $\mathcal{O}_t$ is the 3D occupancy. Its value is between 0 and 1, representing the occupied probability of the voxel grids. Furthermore, we use the farthest point sampling (FPS) algorithm [23] to down-sample the 3D occupancy $\mathcal{O}$ into a sparse voxel set $K = \{v_0, v_1, ..., v_k\}$ to indicate the 3D occupancy-based state s of the plasticine. Therefore, the state of the plasticine can be represented as $s_t = FPS(\mathcal{O}_t) = \{v_0^t, v_1^t, ..., v_k^t\}$. The framework of 3D occupancy prediction is shown in Fig. 3. Specifically, we first use an image backbone network (i.e., ResNet-50 [24]) to extract multi-scale features of multi-camera images I. The 2D-3D spatial attention module and multi-scale occupancy prediction proposed by [25] are used to fuse multi-camera features and output 3D occupancy prediction. The prediction network is supervised by the dense 3D occupancy ground truth generated by the pipeline proposed in Sec. IV.

C. Learning-based Dynamics Model

To model the plasticine's dynamics, we construct a voxel-based state graph by down-sampling from the 3D occupancy of the plasticine and use a deep neural network empowered by a 3D CNN and GNN, inspired by [4]. To be specific, we first down-sample the 3D occupancy of the current state to construct a voxel-based state graph. Then, we use a 3D sparse CNN to extract the multi-scale features of the dense 3D occupancy, and use voxel set abstraction module [26] to project the down-sampled voxels into the voxel grids and collect the features of each node for enhancing the spatial and semantic information. Lastly, we encode the node feature through a state encoder, and fuse the voxel-wised features and node features for the dynamics model learning. The overall structure is shown in Fig. 4.

1) Voxel-based State Graph Construction: We construct a voxel-based state graph with the 3D occupancy-based state s, as $\mathcal{S} = (\mathcal{V}, \mathcal{E})$. Its vertices $\mathcal{V}$ are the voxel set K and the edges $\mathcal{E}$ between two vertices indicate their spatial relationship, which changes dynamically over time. For each vertex, all its neighbors within a predefined distance are connected as an edge, and each edge contains the receiver and sender particle index, object internal relation, and finger-to-object relation.

2) Features Aggregation for Graph Nodes: The spatial resolution of the voxel-grid 3D occupancy is $L \times W \times H$, where the features of the non-empty voxels are the 3D coordinates x, y, z, the colors r, g, b, and the class c. We use a plain sparse 3D CNN backbone network to extract the multi-scale semantic features. By default, the plain sparse CNN backbone network utilizes a series of $3 \times 3 \times 3$ 3D sparse convolution and has 4 levels, with the feature strides $\{1, 2, 4, 8\}$. We name the output sparse features $\{F_1, F_2, F_3, F_4\}$ of each level, respectively. For a voxel $v_i \in K$ with coordinate $(x_{v_i}, y_{v_i}, z_{v_i})$, the sparse features of the voxel from the last three levels $\{F_4, F_3, F_2\}$ can be represented as $f_4^{v_i} = F_4\left(\frac{x_{v_i}}{2^3}, \frac{y_{v_i}}{2^3}, \frac{z_{v_i}}{2^3}\right), f_3^{v_i} = F_3\left(\frac{x_{v_i}}{2^2}, \frac{y_{v_i}}{2^2}, \frac{z_{v_i}}{2^2}\right), f_2^{v_i} = F_2\left(\frac{x_{v_i}}{2^1}, \frac{y_{v_i}}{2^1}, \frac{z_{v_i}}{2^1}\right)$. The 3D coordinate $(x_{v_i}, y_{v_i}, z_{v_i})$ of the voxel v_i is projected to the 2D bird's-eye-view coordinate system, and we utilize bilinear interpolation to obtain the features $f_{v_i}^{bev}$ from the bird-view feature maps. Therefore, the multi-scale semantic feature f_{v_i} for each voxel v_i is generated by concatenating the features from different levels and BEV features following

$$f_{v_i} = [f_4^{v_i}, f_3^{v_i}, f_2^{v_i}, f_{v_i}^{bev}], \text{for } i = 1, ..., n. \tag{3}$$

3) Loss Functions and Evaluation Metrics: We use three loss functions to evaluate the similarity between the predicted and the ground truth state.

Earth Mover's Distance (EMD). The Earth Mover's Distance L_{EMD} between two distributions $\mathcal{V}_p, \mathcal{V}_{gt} \subseteq \mathbb{R}^3$ can be defined as $L_{EMD}(\mathcal{V}_p, \mathcal{V}_{gt}) = \min_{\mu:\mathcal{V}_p \to \mathcal{V}_{gt}} \sum_{x \in \mathcal{V}_p} \|x - \mu(x)\|_2$, where $\mu : \mathcal{V}_p \to \mathcal{V}_{gt}$ is the a bijection [27]. Here, $\mathcal{V}_p$ and $\mathcal{V}_{gt}$ are assumed to have the same volume.

Chamfer Distance (CD). Chamfer Distance L_{CD} between two distributions $\mathcal{V}_p, \mathcal{V}_{gt} \subseteq \mathbb{R}^3$ is: $L_{CD}(\mathcal{V}_p, \mathcal{V}_{gt}) = \frac{1}{|\mathcal{V}_p|} \sum_{x \in \mathcal{V}_p} \min_{y \in \mathcal{V}_{gt}} \|x - y\|_2 + \frac{1}{|\mathcal{V}_{gt}|} \sum_{y \in \mathcal{V}_{gt}} \min_{x \in \mathcal{V}_p} \|y - x\|_2$.

Density-aware Chamfer Distance (DCD). Since CD only considers its nearest neighbor in the other set while ignoring the surroundings, [28] proposed DCD. The Density-aware Chamfer Distance L_{DCD} between two distributions $\mathcal{V}_p, \mathcal{V}_{gt} \subseteq \mathbb{R}^3$ can be calculated: $L_{DCD}(\mathcal{V}_p, \mathcal{V}_{gt}) = \frac{1}{2}(\frac{1}{|\mathcal{V}_p|} \sum_{x \in \mathcal{V}_p}(1 - \frac{1}{n_{\hat{y}}^{\lambda}} e^{-\alpha \|x - \hat{y}\|_2}) + \frac{1}{|\mathcal{V}_{gt}|} \sum_{y \in \mathcal{V}_{gt}}(1 - \frac{1}{n_{\hat{x}}^{\lambda}} e^{-\alpha \|y - \hat{x}\|_2}))$, where $\hat{y} = \min_{y \in \mathcal{V}_{gt}} \|x - y\|_2$, $\hat{x} = \min_{x \in \mathcal{V}_p} \|y - x\|_2$, $n_{\hat{y}}^{\lambda} = |\mathcal{V}_p^{\hat{y}}|$, $n_{\hat{x}}^{\lambda} = |\mathcal{V}_{gt}^{\hat{x}}|$ and α denotes a temperature scalar.

The total loss in our experiments is the weighted sum of the three distance functions mentioned above: $L(\mathcal{V}_p, \mathcal{V}_{gt}) = w_1 \times L_{EMD}(\mathcal{V}_p, \mathcal{V}_{gt}) + w_2 \times L_{DCD}(\mathcal{V}_p, \mathcal{V}_{gt}) + w_3 \times L_{CD}(\mathcal{V}_p, \mathcal{V}_{gt})$.

D. Learning-based Predictive Control

After obtaining the learned dynamics model, the model predictive control is used to control the gripper to manipulate the plasticine.

1) Action Space and Goal-Conditioned MPC: We use a paralleled 2-finger gripper to manipulate the plasticine, and the action space of two fingers can be denoted by two fingers' geometry center positions as $\{p_r = (x_r, y_r, z_r), p_l = (x_l, y_l, z_l)\}$. The action space of the gripper can be defined as $\{x, y, z, r_z, l\}$. r_z is the gripper's rotation about the z axis (i.e., perpendicular to the operating plane), which is $\arctan(\frac{y_l - y_r}{x_l - x_r})$. There are no rotations about the x and y axes. We define $x = \frac{x_r + x_l}{2}$, $y = \frac{y_r + y_l}{2}$, $z = \frac{z_r + z_l}{2} = z_r = z_l$. l represents the distance between 2 gripper fingers, which is $\|p_r - p_l\|_2$.

Given a goal state $\mathcal{S}_g$ of the plasticine and the action sequence $\mathcal{A}_{0 \to t-1} = \{a_0, a_1, ..., a_{t-1}\}$ sampled from the gripper action space, where t is the time horizon, we denote the resulting state $\mathcal{S}$ changing after applying the control inputs

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

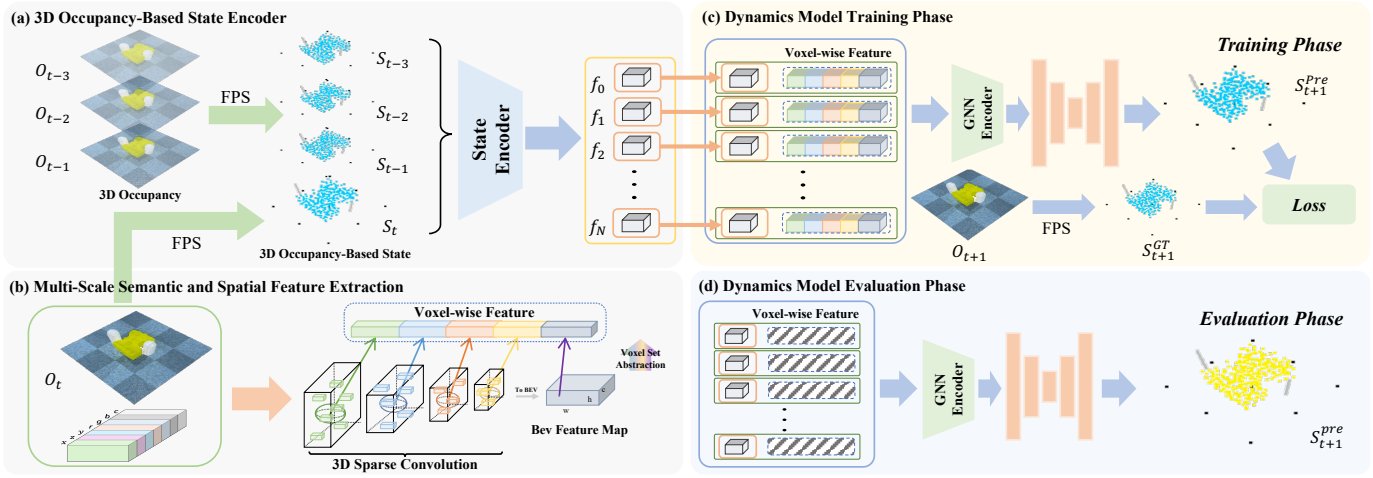


Fig. 4: **Overall structure of our proposed 3D CNN-Based dynamics model.** (a) The 3D occupancy of the current step t and previous three steps $t-3 \cdots t-1$ is down-sampled to construct a voxel-based state graph, and node features f_j are extracted through a state encoder. (b) The 3D occupancy of the current step t is fed into a 3D sparse CNN to learn multi-scale semantic and spatial features. The learned voxel-wise features of each graph node are then retrieved and summarized into a feature set from multiple levels through a voxel set abstraction module. (c) During the training phase, the aggregated node features are concatenated to the encoded node features f_j and then fed into the GNN for dynamics model training. (d) During evaluation, voxel-wise features are masked and only used at the initial step ($t = 0$).

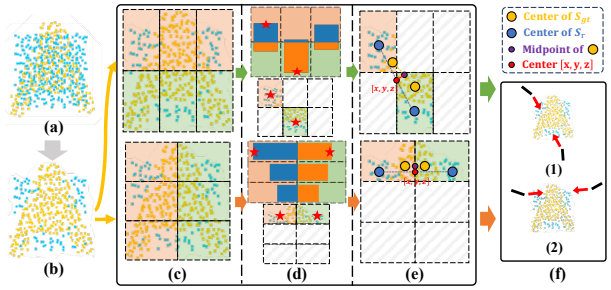


Fig. 5: **Pipeline of our shape-based action initialization for the gripper.** The point cloud s_p colored blue is the initial state of plasticine and the point cloud s_{gt} colored yellow is the goal of plasticine (i.e., letter "A"). (a) Align s_p to s_{gt}. (a)→(b) Filter out points close to s_{gt} (c) Segment the point clouds into $N \times 2$ regions along the x-axis and y-axis as two branches. (d) Calculate the cost of moving those points using Euclidean distance in each region, respectively. (e) Select two parts with the maximum cost from two branches respectively, and calculate the direction and center of the line connecting the two fingers as the reference initialization. (f) Visualization of initialized actions.

(i.e., action sequence $\mathcal{A}_{0\to t-1}$) as $\mathcal{S}_{0\to t} = \{\mathcal{S}_0, ..., \mathcal{S}_t\}$. The task is to obtain the actions that minimize the loss $\mathcal{L}(\mathcal{S}_t, \mathcal{S}_g)$ between the actual state $\mathcal{S}_t$ and the target state $\mathcal{S}_g$. Thus, L-BFGS [29], a gradient-based optimization algorithm, is used to optimize $\mathcal{A}_{0\to t-1}$ with the gradient of $\mathcal{L}(\mathcal{S}_t, \mathcal{S}_g)$, which is the same as the training loss of our dynamics model.

2) Shape-based Action Initialization: While using MPC, we find that action initialization will significantly impact completion quality and repetition consistency as the deformation of plasticine is irreversible. To this end, a shape-based action initialization module is proposed, as shown in Fig. 5. First, we align the current state with the center of the target state (Fig. 5 (a)) and remove the point set $\{p_0, p_1, \cdots, p_i\} \in s_p$ which is close to s_{gt} (Fig. 5 (b)), to get the rest of point cloud s_r. Then, the point cloud is segmented into $m \times 2$ regions along the x-axis and y-axis (Fig. 5 (c)) and calculate the cost of moving those points using Euclidean distance in each region respectively (Fig. 5 (d)), where m is within the range of 3 to 6 in our experiments. Finally, we choose the two largest cost regions and obtain the center positions of point cloud s_r and s_{gt} in the two regions respectively. The rotation of the gripper r_z is defined as the direction of the line connecting two center points of s_r, the center x, y, z as the projection position of the midpoint of the line connecting two center points of s_{gt} onto the line connecting the two center points of s_r (Fig. 5 (e)). Finally, we can get the initial actions for MPC (Fig. 5 (f)).

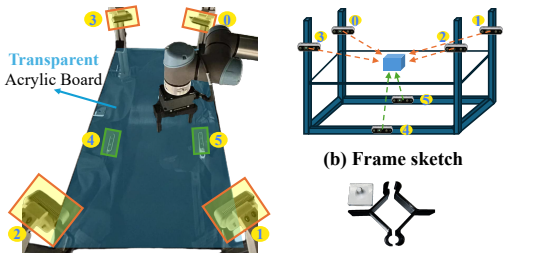
IV. 3D Occupancy Ground Truth For DOM

In our experimentwe find that the network supervised by RGB-D images is unable to predict dense enough occupancy. Thus, we need to generate dense occupancy labels for training. To this end, we built a novel data collection platform and proposed a pipeline to generate fine-grained dense occupancy ground truth based on raw RGB-D images.

A. Data Collection Platform

Conventional data collection platforms [30] can only collect the data of the objects above the plane while ignoring the bottom information. In practice, RGB-D images from the above views are unable to provide enough point clouds to build an enclosed mesh. To address it, we used aluminum profiles to build a frame and installed a transparent acrylic board in the middle of the frame as the operating plane, allowing us to collect full spatial information (i.e., top, side, and bottom information) of the manipulated object. Six cameras are installed at different positions and angles to refine the mesh reconstruction. Specifically, we set 4 cameras on top of it and 2 cameras on the bottom to collect multi-view calibrated images of the manipulation scenario to obtain dense point clouds, as shown in Fig. 6. The bottom-view observations significantly enhance mesh reconstruction and occupancy quality by capturing occluded regions that top and side views miss. To minimize distortion, we use a high-transmittance acrylic board and depth sensors with robust filtering and careful calibration. For more details, we refer readers to [31].

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.



(a) Hardware setup (b) Frame sketch (c) Plasticine shape resetting tool

Fig. 6: **Hardware setup of our data collection platform.** The blue part is the acrylic board serving as the operating plane. The orange parts are four cameras above the plane for collecting the top and side images. The green parts are two cameras below the plane for collecting the bottom images.

B. Data Collection and Processing

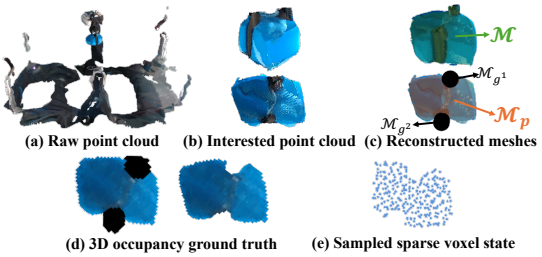
In real-world scenarios, we collect multi-view RGB-D images and then convert them into dense point clouds $P = \{P_1, P_2, \cdots, P_n\}$ with RGBs using the extrinsic and intrinsic parameters, where $n = 6$. We use the physical prior (i.e., size) of the platform to remove the points far from the platform and restrain the point cloud in the range of the platform (i.e., the platform size $[l, w, h]$ is $[1.0\,\mathrm{m}, 1.0\,\mathrm{m}, 0.2\,\mathrm{m}]$). Therefore, the raw point cloud of the manipulation scenario can be updated with $P_{raw} = M(P)$, as shown in Fig. 7 (a), where M is the mask to filter the points out of the platform. For the manipulation task, the region of interest (ROI) is only related to the deformable object and the gripper, so we cropped the point cloud P_{raw} to P_{roi}, as shown in Fig. 7 (b). We use color information as a color-based filter to remove the noisy points and segment the point cloud P_{do} of the plasticine and the point cloud P_g of the gripper from P_{roi}. For P_g, DBSCAN [32] is used to cluster the point clouds belonging to two primitives $\{P_g^1, P_g^2\}$ of the gripper.

C. Refinement with Geometry Prior

Although a color-based filter is implemented, some missed points P_m belonging to two primitives remain in P_{do} due to occlusion and reflection. This is worse when the primitives are in contact with the plasticine. So we use P_{xyz} and $[R, L]$ of two primitives to refine the point cloud of the plasticine, where P_{xyz} is the 3D position of two primitives in the cartesian coordinate system in 3D space, R and L is the radius and the length of the primitive, respectively. We implement tiny dilation to remove the missed points of the two primitives $[P_m^1, P_m^2]$ in P_{do}. Therefore, the representation of the two primitives and the DO will be updated as $\{P_g^1 = [P_g^1, P_m^1], P_g^2 = [P_g^2, P_m^2]\}$ and $P_{do} = \{P_{do} - [P_m^1, P_m^2]\}$, respectively.

D. Surface Reconstruction

Instead of only considering the gripper and plasticine as one point cloud to construct a whole mesh $\mathcal{M}$, we also construct the meshes of plasticine $\mathcal{M}_p$ and two primitives $\{\mathcal{M}_{g^1}, \mathcal{M}_{g^2}\}$ using their point cloud P_{do}, P_g^1 and P_g^2 shown in Fig. 7 (c), via Poisson Surface Reconstruction [33], a learning-free method that uses smoothness prior of surface geometry to create watertight surfaces from oriented point sets.



(a) Raw point cloud (b) Interested point cloud (c) Reconstructed meshes

(d) 3D occupancy ground truth (e) Sampled sparse voxel state

Fig. 7: **Pipeline for occupancy ground truth generation.** (a) Converting RGB-Ds into dense and well-registered point clouds P_{raw} filtered by M. (b) Point cloud P_{roi} of DO and gripper. (c) Reconstruct deformed meshes $\mathcal{M}$, $\mathcal{M}_p$, $\mathcal{M}_{g^1}$ $\mathcal{M}_{g^2}$. (d) Dense 3D occupancy ground truth. (e) Down-sampled voxel state from 3D occupancy.

E. Dense Occupancy Generation and Semantic Labeling

The obtained mesh $\mathcal{M} = \{\mathcal{V}, \mathcal{E}\}$ fills up the holes of point clouds with evenly distributed vertices $\mathcal{V}$, so we can further convert the mesh into dense voxel set V_d shown in Fig. 7 (d). Specifically, first, the range $\{[x_{min}, y_{min}, z_{min}], [x_{max}, y_{max}, z_{max}]\}$ of $\mathcal{M}$ is obtained from the evenly distributed vertices $\mathcal{V}$. Second, the space $\mathcal{R} = \{x_{min} \le x \le x_{max}, y_{min} \le y \le y_{max}, z_{min} \le z \le z_{max}\}$ is voxelized with a fixed voxel size $[s_x, s_y, s_z]$. After voxelization, we compute the signed distance function (SDF) of $\mathcal{M}$ for each voxel and remove the voxels outside $\mathcal{M}$. Finally, we use the plasticine mesh $\mathcal{M}_p$ and primitives $\{\mathcal{M}_{g^1}$ $\mathcal{M}_{g^2}\}$ to label the voxels inside them repeating the previous step. For the operating plane, we consider it as a thin shell, a cube of size $[l, w, h]$, where l, w, h is the length, width, and height, and h equals to the voxel size s_z. The color of each voxel is defined as the average of the RGB values of the six vertices obtained by projecting the voxel center onto the mesh surface along the x, y, and z axis. Following the proposed method, we can get dense volumetric occupancy with fine semantic labels without huge human efforts of annotation.

V. Experiments and Results

In this section, we present a comprehensive set of experiments to evaluate the effectiveness of our proposed method. Specifically, we evaluate the quality of state representation derived from voxels down-sampled from the predicted 3D occupancy and the performance of the dynamics model trained on the representation. Additionally, we perform ablation studies to highlight the benefits of fusing dense 3D occupancy features and the proposed shape-based action initialization module. Finally, we validate the performance of our method through extensive hardware experiments, showcasing its ability to manipulate plasticine into a given target shape.

A. Experimental Setup

1) Simulation Setup: The simulation [34] features two capsule-shaped fingers, each with a radius of $0.045\,\mathrm{m}$ and a length of $0.25\,\mathrm{m}$, interacting with a square plasticine of size $0.25 \times 0.25 \times 0.2$ m. Six RGB-D cameras are strategically placed at diverse viewpoints, including top and bottom perspectives, to capture RGB-Ds at a resolution of 512×512.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

2) Hardware Setup: As shown in Fig. 6 (a), we build this platform to conduct real-world experiments. We utilize a 6-DoF UR5e robotic arm equipped with a gripper (DH-Robotics PGI-140-80), and six RGB-D cameras (Real-sense D453i). Specifically, we 3D-printed two fingers for the gripper, each with a length of $0.15\,\mathrm{m}$ and a radius of $0.0075\,\mathrm{m}$. Additionally, we calibrated the six RGB-D cameras, which operate at a frame rate of $6\,\mathrm{Hz}$ and a resolution of 640×480. For better understanding, a schematic of the platform is provided in Fig.6 (b). Additionally, we designed and 3D-printed a specialized tool, illustrated in Fig.6 (c). We control the UR5e robotic arm and the gripper under the Robot Operating System (ROS). The transformations between the two fingers and the robot base are published at a frequency of $15\,\mathrm{Hz}$.

3) Dataset Collection and 3D Occupancy Ground Truth Generation: In the simulator [34],we collected a total of 70 episodes, comprising 8400 frames, where 50 episodes are used for training and 20 episodes are for evaluation. In the real-world experiments, we collect 50 episodes, consisting of 6000 frames, as training data to fine-tune the pre-trained model for the real-world experiments. Specifically, each frame includes six RGB-D images and the positions of the two fingers w.r.t. the robot arm's base frame. During each episode, the plasticine was pinched three times, with data recorded when the gripper and plasticine were aligned on the same plane. To ensure consistent initial conditions, a plasticine shape resetting tool shown in Fig. 6 (c) is used to reshape the plasticine to a uniform starting shape and localized at the same position by clamping the tool to the gripper before each episode. Following the pipeline described in Sec. IV, we generated the 3D occupancy ground truth using six RGB-Ds to supervise the training of the 3D occupancy prediction model.

B. Experimental Results

1) 3D Occupancy-based State Representation: To model the plasticine dynamics, we down-sample the 3D occupancy of the plasticine to get a sparse voxel state as shown in Fig. 7 (e) and construct a voxel-based state graph. First, we train a 3D occupancy prediction network using our generated 3D occupancy dataset on an Nvidia GeForce RTX-4080 with 16G memory. The occupancy prediction range is $[-0.1\,\mathrm{m}, 0.1\,\mathrm{m}]$ for X, Y axis and $[-0.01\,\mathrm{m}, 0.07\,\mathrm{m}]$ for Z axis. The shape of predicted occupancy is $100 \times 100 \times 40$ with the voxel size of $[0.002\,\mathrm{m}, 0.002\,\mathrm{m}, 0.002\,\mathrm{m}]$. Only specific parts of predicted occupancy are labeled as the plasticine, operating plane, and gripper, while the others would be considered noise. Then the trained model is employed to infer the 3D occupancy from four camera views, which is downsampled into a voxel set K ($k = 300$) representing the 3D occupancy-based state of the plasticine. The similarity between the down-sampled results and the ground truth indicates the quality of state representation, which will determine the performance of the trained dynamics model. So we compare the 3D occupancy-based method with two baseline methods proposed in [4], patch-based and crop-based method (four camera views) on the simulation dataset, averaged over 160 frames (i.e., four pinches) using EMD, DCD, and CD. Specifically, the patch-based method patches the sampled results using points from the previous frame, and the crop-based method directly crops out the object point cloud and then down-samples it. For the 3D occupancy-based method, "Prediction" refers to the state that is down-sampled from the inferred occupancy with four cameras, while "Ground Truth" denotes the state that is down-sampled from the generated occupancy dataset with six cameras. Table. I shows that the 3D occupancy-based method has a more precise representation, achieving lower loss across all metrics.

Methods		Average Over Four Pinches		
		$\mathbf{EMD}_{\times 1e^{-3}} \downarrow$	$\mathbf{DCD}_{\times 1e^{-4}} \downarrow$	$\mathbf{CD}_{\times 1e^{-3}} \downarrow$
Patch-Based		32.7 ± 3.86	39.4 ± 2.05	38.6 ± 2.12
Crop-Based		30.5 ± 1.62	36.7 ± 1.26	37.1 ± 0.953
3D Occ-based	Prediction	28.6 ± 1.23	34.5 ± 1.03	36.0 ± 0.667
	Ground Truth	$\mathbf{27.9 \pm 1.14}$	$\mathbf{34.2 \pm 0.998}$	$\mathbf{35.8 \pm 0.676}$

TABLE I: **Quality comparison of different state representation methods.** We compared patch-based, crop-based, and 3D occupancy-based (3D Occ-based) methods in terms of EMD, DCD, and CD, respectively, over four pinches (160 frames).

2) 3D CNN-GNN Dynamics Model Training: Based on our experiments, we set the hyper-parameters of L_{DCD} as $\alpha = 500$ and $\lambda = 0.5$ for evaluation, and $\lambda = 0.1$ and $\alpha = 20$ for training. Additionally, the hyper-parameters of total loss $L(\mathcal{V}_p, \mathcal{V}_{gt})$ are selected as $w_1 = 0.5$, $w_2 = 0.4$ and $w_3 = 0.1$. We train our model with 3D occupancy-based state representation for 24 epochs and evaluate the performance of the learned dynamics model. We compare our method against two strong baselines introduced in RoboCraft [4] and RoboCook [21]. Both baselines employ GNN-only architectures and are trained for 100 epochs only using EMD and CD. The state representation in RoboCook is referred to as the surface-based method, where the representation is obtained by down-sampling the vertices of the object's surface mesh. All the dynamics models used in our comparisons can be summarized as follows:

- RoboCraft [4]: Crop-based (state) + GNN (model)
- RoboCook [21]: Surface-based (state) + GNN (model)
- Ours: 3D Occ-based (state) + 3D CNN-GNN (model)

The evaluation results using EMD, DCD, and CD are shown in Table. II, and our model trained with a 3D occupancy-based state representation consistently outperforms baselines across all metrics. These results can be attributed to the structural advantages of our representation. RoboCook uses surface-only nodes with limited connectivity, while our 3D occupancy includes internal nodes, enabling richer structural understanding. RoboCraft considers internal structure, but its crop-based method yields a sparse, local receptive field that lacks global spatial context. In contrast, our dense 3D features capture multi-scale spatial context, leading to more accurate predictions of the dynamics during manipulation. Note that, for the evaluation of the surfaced-based method in [21], the mesh is reconstructed from surface-level points, and $k = 300$ internal points are then down-sampled for the quantitative comparison.

3) Manipulation Results: We conduct experiments both in the simulator and real world. We conduct three manipulation trials with letters "X", "T", and "K", and Fig. 8 shows both qualitative and quantitative comparison results in the simulator with RoboCraft [4] for fair comparison, showing the superior performance of our method. We present the manipulation

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

Methods	Dynamics Model Average Evaluation Loss		
	$\text{EMD}_{\times 1e^{-3}} \downarrow$	$\text{DCD}_{\times 1e^{-4}} \downarrow$	$\text{CD}_{\times 1e^{-3}} \downarrow$
RoboCraft	24.3 ± 2.82	27.3 ± 2.30	34.7 ± 1.64
RoboCook	25.5 ± 2.92	33.2 ± 2.02	36.3 ± 1.45
Ours	$\mathbf{22.8 \pm 2.30}$	$\mathbf{25.3 \pm 1.52}$	$\mathbf{33.5 \pm 0.951}$

TABLE II: **Mean and standard deviation of the dynamics model's performance comparison.**

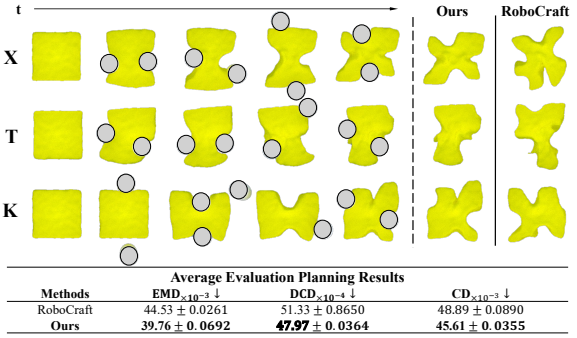


Fig. 8: **Qualitative and quantitative results from three simulation trials.** Trials are performed with letters 'X', 'T', and 'K'.

results from real-world experiments in Fig. 9, showcasing five trials with varying levels of shaping difficulties utilizing a parallel two-finger gripper, such as the relatively easy case "X", which exhibits completely symmetrical (vertical, horizontal, and 45° rotational symmetry), and the more challenging case "Y" and "M", which are symmetrical only about the vertical axis without rotational symmetry. Our proposed framework demonstrates the ability to successfully manipulate the plasticine into a shape closely resembling the goal shape. With our proposed framework, the robot can efficiently shape plasticine to meet different difficulty levels. To further justify the use of the 3D voxel-based representation, we design the experiments of pinching a two-layer stamp with a given height ratio $\frac{H1}{H2}$ between layers. As shown in Fig. 10, our proposed framework successfully pinches out the given two-layer stamps with a height ratio close to the desired. To show our method's generalization in terms of initial shapes, colors, and materials, we test the method in three more initial shapes (circle, triangle, and irregular polygon), two more colors (pink and yellow), and one more material (foam clay), as shown in Fig. 11. All final shapes are close to our goal, the "X" shape.

Furthermore, manipulation experiments of RoboCraft [4] are conducted in the real world, and human manipulation trials are conducted using a 3D-printed device (Fig.12 (b)), which operates in the same manner as the gripper (Fig.12 (a)). Amateurs are allowed to pinch the plasticine the same number of times as the robot. As shown in Fig.12 (c), our method achieves comparable performance to human demonstrations and outperforms RoboCraft [4].

4) Ablation Study: To discern the contribution of proposed methods to the framework, we assess the contribution of the 3D CNN in modeling the dynamics, contribution of shape-based action initialization in MPC, and the DCD loss.

3D CNN Module: Table III presents the ablation results for the 3D CNN module in the learned dynamics model. Without this module, the dynamics model is trained directly with the encoded state features of the sparse voxels. However, this straightforward approach performs worse than the method that integrates features derived from the 3D CNN module.

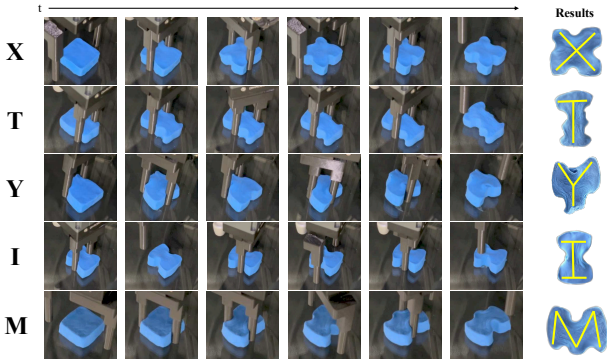


Fig. 9: **Manipulation results of single-layer objects in real world.**

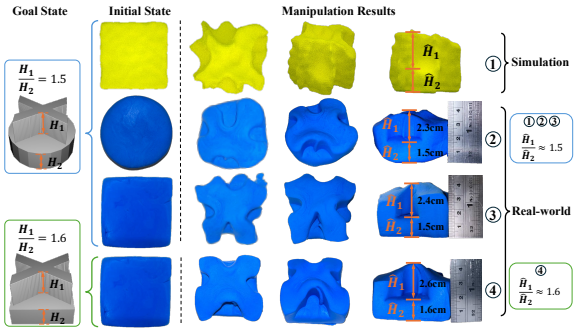


Fig. 10: **Manipulation results of multi-layer objects in real world.** The initial heights of trials ②, ③, and ④ are approximately 3.5 cm, 3.0 cm, and 3.2 cm, respectively.

Moreover, the ablation study indicates that the 3D CNN-based dynamics model achieves prior performance since it can effectively preserve multi-scale 3D spatial information.

Shape-based Actions Initialization: The results in Fig. 13 demonstrate the importance of using shape-based initialization. As shown in Fig. 13 (a), for the goal shape "X", without this module, the actions are randomly initialized, resulting in unreasonable and irreversible deformation. Additionally, we conduct experiments 10 times in identical experimental configurations with and without this module. We evaluate the shape similarity to the goal after the first action applied using EMD, DCD, and CD, and the quantitative results are shown

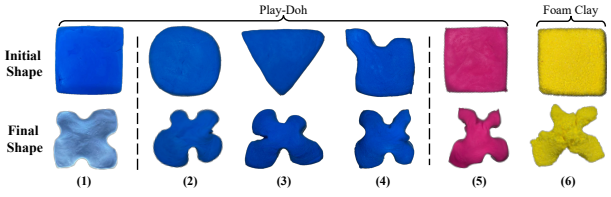


Fig. 11: **Generalization demonstrations in terms of initial shapes, colors, and materials.** Six trials are conducted in four initial shapes, three colors, and two materials (dense play-doh vs. spongy form clay).

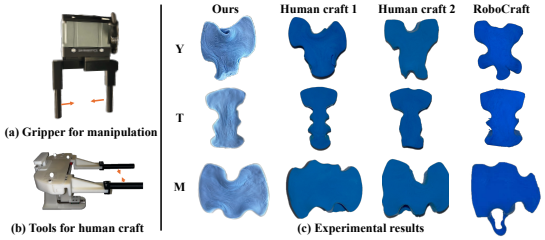
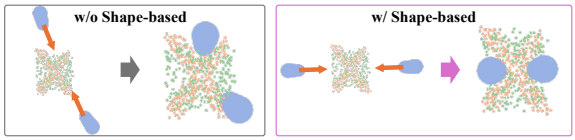


Fig. 12: **Comparison of experimental results.** (a) The gripper for robot manipulation. (b) Tool for human manipulation inspired by [35]. (c) The shaping results of our method, two amateur humans, and RobotCraft [4].

Method	$\text{EMD}_{\times 10^{-3}} \downarrow$	$\text{DCD}_{\times 10^{-4}} \downarrow$	$\text{CD}_{\times 10^{-3}} \downarrow$
w/o 3D CNN	24.5±5.07	27.2±2.25	34.3±2.60
w/o DCD loss	23.4±3.94	26.7±3.96	34.2±2.36
Ours	$\mathbf{22.8 \pm 2.30}$	$\mathbf{25.3 \pm 1.52}$	$\mathbf{33.5 \pm 0.951}$

TABLE III: **Ablation results in terms of the 3D CNN module and the DCD loss.**



(a) Visualization of action initialization results

Method	EMD × 10−3 ↓	DCD × 10−4 ↓	CD × 10−3 ↓
w/o Shape-based	46.9 ± 4.54	51.1 ± 2.54	50.7 ± 4.40
Ours	**38.9 ± 0.734**	**45.6 ± 1.25**	**43.4 ± 1.10**

(b) Evaluation results using EMD, DCD, and CD

Fig. 13: **Ablation study of shape-based action initialization.** "w/o shape-based" means that the actions are randomly initialized during manipulation. (a) Visualization of actions initialization results. (b) The evaluation results using EMD, DCD, and CD after the first action.

in Fig. 13 (b).

DCD Loss: We conduct experiments of dynamics model training in identical experimental configurations with and without DCD loss. We evaluate the performance of the learned dynamics model, the results are shown in Table. III.

VI. Conclusion and Discussions

In this paper, we propose a framework to manipulate the elasto-plastic object (e.g., plasticine) into a desired goal shape with 3D occupancy and a learning-based predictive control algorithm enhanced by a shape-based action initialization module. Meanwhile, we build a novel platform for full spatial data collection during manipulation and propose a pipeline to generate dense 3D occupancy ground truth and train a 3D occupancy prediction network supervised by the generated occupancy ground truth to infer the 3D occupancy during manipulation. Through our comprehensive experiments, we demonstrate the superiority of 3D occupancy-based state representation and the proposed framework can successfully manipulate the plasticine into a given 3D shape with different levels of shaping difficulties and multi-layer structures, and achieve comparable performance to humans.

Our current framework relies on full-space observations enabled by transparent plane; without it, occlusions significantly degrade the quality of mesh reconstruction for 3D occupancy generation. Given the limited quality of point clouds from employed RGB-D cameras, Poisson surface reconstruction struggles to capture fine topological changes such as splits and merges, making them difficult to model accurately. Compared to segmented observations that commonly require separate processing, jointly leveraging RGB and category features allows for more efficient and semantically meaningful feature extraction. Due to disparities in material properties between simulation and the real world, we find fine-tuning on real data is essential. Finally, the computational demands of our method require GPU acceleration, limiting deployment on CPU-only systems and resource-constrained applications.

References

[1] R. Wu *et al.*, "Learning foresightful dense visual affordance for deformable object manipulation," in *ICCV*, pp. 10947–10956, 2023.

[2] K. Mo *et al.*, "Foldsformer: Learning sequential multi-step cloth manipulation with space-time attention," *RAL*, vol. 8, no. 2, pp. 760–767, 2022.

[3] Y. Tang *et al.*, "Learning-based MPC with safety filter for constrained deformable linear object manipulation," *RAL*, 2024.

[4] H. Shi *et al.*, "RoboCraft: Learning to see, simulate, and shape elasto-plastic objects in 3D with graph networks," *IJRR*, vol. 43, no. 4, pp. 533–549, 2024.

[5] S. Chen *et al.*, "DiffSRL: Learning dynamical state representation for deformable object manipulation with differentiable simulation," *RAL*, vol. 7, no. 4, pp. 9533–9540, 2022.

[6] C. Qi *et al.*, "Learning closed-loop dough manipulation using a differentiable reset module," *RAL*, vol. 7, no. 4, pp. 9857–9864, 2022.

[7] A. Mousavian *et al.*, "6-DoF GraspNet: Variational grasp generation for object manipulation," in *ICCV*, pp. 2901–2910, 2019.

[8] Q. Tan *et al.*, "Realtime simulation of thin-shell deformable materials using CNN-based mesh embedding," *RAL*, vol. 5, no. 2, pp. 2325–2332, 2020.

[9] J. Matas *et al.*, "Sim-to-real reinforcement learning for deformable object manipulation," in *CoRL*, pp. 734–743, 2018.

[10] Y. Deng *et al.*, "Learning visual-based deformable object rearrangement with local graph neural networks," *Complex & Intelligent Systems*, vol. 9, no. 5, pp. 5923–5936, 2023.

[11] Y. Li *et al.*, "Learning particle dynamics for manipulating rigid bodies, deformable objects, and fluids," *arXiv:1810.01566*, 2018.

[12] B. Thach *et al.*, "DefGoalNet: Contextual goal learning from demonstrations for deformable object manipulation," *ICRA*, pp. 3145–52, 2024.

[13] B. Thach *et al.*, "DeformerNet: Learning bimanual manipulation of 3D deformable objects," *arXiv:2305.04449*, 2023.

[14] A. Bartsch *et al.*, "SculptDiff: Learning robotic clay sculpting from humans with goal conditioned diffusion policy," in *IROS*, pp. 7307–7314, 2024.

[15] C. Li *et al.*, "DeformNet: Latent space modeling and dynamics prediction for deformable object manipulation," *arXiv:2402.07648*, 2024.

[16] H. J. T. Suh *et al.*, "The surprising effectiveness of linear models for visual foresight in object pile manipulation," in *Algorithmic Foundations of Robotics XIV*, pp. 347–363, 2021.

[17] S. Xue *et al.*, "Neural field dynamics model for granular object piles manipulation," in *CoRL*, pp. 2821–2837, 2023.

[18] Y. Wang *et al.*, "Dynamic-resolution model learning for object pile manipulation," *arXiv:2306.16700*, 2023.

[19] A. Zeng *et al.*, "Transporter networks: Rearranging the visual world for robotic manipulation," in *CoRL*, pp. 726–747, 2021.

[20] Y. Li *et al.*, "Learning compositional Koopman operators for model-based control," *arXiv:1910.08264*, 2019.

[21] H. Shi *et al.*, "RoboCook: Long-horizon elasto-plastic object manipulation with diverse tools," *arXiv:2306.14447*, 2023.

[22] A. Bartsch *et al.*, "Sculptbot: Pre-trained models for 3D deformable object manipulation," in *ICRA*, pp. 12548–12555, 2024.

[23] C. Moenning *et al.*, "Fast marching farthest point sampling," Tech. Rep., University of Cambridge, Computer Laboratory, 2003.

[24] K. He *et al.*, "Deep residual learning for image recognition," in *CVPR*, pp. 770–778, 2016.

[25] Y. Wei *et al.*, "SurroundOcc: Multi-camera 3D occupancy prediction for autonomous driving," in *ICCV*, pp. 21729–21740, 2023.

[26] S. Shi *et al.*, "PV-RCNN: Point-voxel feature set abstraction for 3D object detection," in *CVPR*, pp. 10529–10538, 2020.

[27] Y. Rubner *et al.*, "A metric for distributions with applications to image databases," in *ICCV*, pp. 59–66, 1998.

[28] T. Wu *et al.*, "Density-aware Chamfer Distance as a comprehensive metric for point cloud completion," *arXiv:2111.12702*, 2021.

[29] R. Fletcher, "Practical Methods of Optimization," John Wiley & Sons, 2013.

[30] J. Obrist *et al.*, "PokeFlex: Towards a real-world dataset of deformable objects for robotic manipulation," *arXiv:2409.17124*, 2024.

[31] Z. Zhang *et al.*, "DOFS: A real-world 3D deformable object dataset with full spatial information for dynamics model learning," *arXiv:2410.21758*, 2024.

[32] M. Ester *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proceedings of KDD*, vol. 96, no. 34, pp. 226–231, 1996.

[33] M. Kazhdan *et al.*, "Poisson surface reconstruction," in *Proc. Eurographics Symp. Geometry Processing (SGP)*, vol. 7, no. 4, 2006.

[34] Z. Huang *et al.*, "PlasticineLab: A soft-body manipulation benchmark with differentiable physics," *arXiv:2104.03311*, 2021.

[35] C. Chi *et al.*, "Universal Manipulation Interface: In-the-wild robot teaching without in-the-wild robots," *arXiv:2402.10329*, 2024.