Journal Title
XX(X):1–15

DOI: 10.1177/ToBeAssigned
www.sagepub.com/
SAGE

RoboCraft: Learning to See, Simulate, and Shape Elasto-Plastic Objects in 3D with Graph Networks

Haochen Shi[1*], Huazhe Xu[1*†], Zhiao Huang[2], Yunzhu Li[1,3], and Jiajun Wu[1]

Abstract

Modeling and manipulating elasto-plastic objects are essential capabilities for robots to perform complex industrial and household interaction tasks (e.g., stuffing dumplings, rolling sushi, and making pottery). However, due to the high degrees of freedom of elasto-plastic objects, significant challenges exist in virtually every aspect of the robotic manipulation pipeline, e.g., representing the states, modeling the dynamics, and synthesizing the control signals. We propose to tackle these challenges by employing a particle-based representation for elasto-plastic objects in a model-based planning framework. Our system, RoboCraft, only assumes access to raw RGBD visual observations. It transforms the sensory data into particles and learns a particle-based dynamics model using graph neural networks (GNNs) to capture the structure of the underlying system. The learned model can then be coupled with model predictive control (MPC) algorithms to plan the robot's behavior. We show through experiments that with just 10 minutes of real-world robot interaction data, our robot can learn a dynamics model that can be used to synthesize control signals to deform elasto-plastic objects into various complex target shapes, including shapes that the robot has never encountered before. We perform systematic evaluations in both simulation and the real world to demonstrate the robot's manipulation capabilities.

Keywords

Robot Learning, Visual Perception and Learning, Deformable object manipulation

Introduction

Effective manipulation of deformable objects is essential for robots to be deployed in real-world industrial and household environments. However, due to deformable objects' high degrees of freedom (DoFs) and consequent challenges in state estimation and dynamics modeling, manipulating deformable objects requires significant innovations beyond the typical robotic paradigm, focusing only on rigid objects. Recent advances show promising results in manipulating clothes (Miller et al. 2012; Matas et al. 2018; Wu et al. 2020; Ha and Song 2022; Yin et al. 2021; Hoque et al. 2022; Xu et al. 2022; Avigal et al. 2022; Huang et al. 2022) and ropes or cables (Yan et al. 2021; Sundaresan et al. 2020; Chi et al. 2022; Viswanath et al. 2022). Yet, the manipulation of objects with high plasticity, such as dough or plasticine, poses a unique set of challenges and is currently underexplored (Matl and Bajcsy 2021), despite the ubiquity of such objects in household and industrial settings. This paper investigates how to empower robots to model and manipulate elasto-plastic objects based on raw RGBD visual observations.

The primary challenges of manipulating deformable objects stem from their high DoFs, partial observability, and complex non-linear local interactions. Learning dynamics models directly from high-dimensional sensory data offers a promising data-driven avenue for us to perform effective planning. For example, model-based reinforcement learning (RL) algorithms have achieved great success in various planning and control tasks (Schrittwieser et al. 2020; Nagabandi et al. 2020; Manuelli et al. 2021). However, when faced with elasto-plastic objects, these prior methods may fail due to a lack of explicit exploitation of the objects' structure. Another thread of works represents deformable objects using particles and employs graph neural networks (GNNs) to model their dynamics (Mrowca et al. 2018; Li et al. 2018, 2020a; Sanchez-Gonzalez et al. 2020; Shlomi et al. 2020). They have shown great generalization results, demonstrating the benefits of explicit structured dynamics modeling. However, most works require full-state information and a particle-based simulator to provide particle-to-particle correspondence between frames. Such strong supervision is difficult to obtain from raw sensory data, limiting their use in real-world applications. Hence, the natural question to ask here is: would it be possible to model the dynamics and manipulate elasto-plastic objects in the real world solely based on RGBD visual observations without needing particle-to-particle temporal correspondence? To tackle this problem, we propose RoboCraft (Figure 1), a model-based planning framework that represents elasto-plastic objects with particles but employs distribution-based loss functions and makes novel improvements over recently-developed GNNs

[1]Computer Science Department, Stanford University, Stanford, CA, USA
[2]Computer Science & Engineering Department, University of California, San Diego, La Jolla, CA, USA
[3]Computer Science Department, University of Illinois Urbana-Champaign, Urbana, IL, USA
[*]These authors have contributed equally.
[†]Now at the Institute for Interdisciplinary Sciences, Tsinghua University, Beijing, China

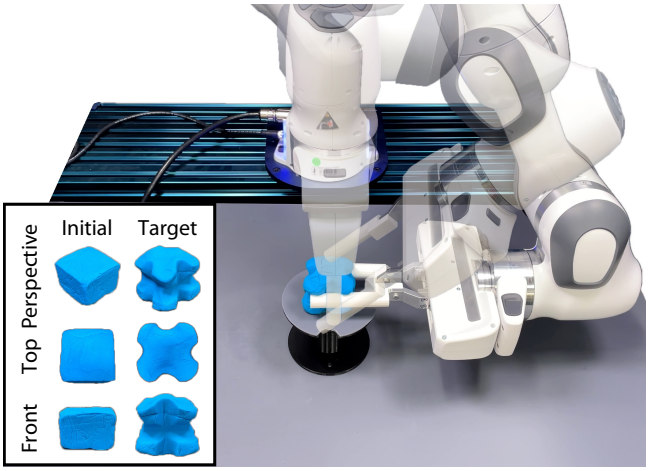

Figure 1. RoboCraft. Starting from the initial cubic shape shown in the first column at the bottom left corner, the robot uses a parallel two-finger gripper to shape a 3D 'X,' which looks like the alphabet letter 'X' from every direction, conditioned on the target shape shown in the second column.

to model the objects' dynamics. The learned dynamics model is coupled with gradient-based trajectory optimization techniques to plan the robot's behaviors.

The proposed approach closes the perception and control loop, allowing accurate modeling and manipulation of elasto-plastic objects in simulated and real-world settings. Specifically, our framework consists of (1) a perception module that constructs the particle representation of the object by sampling from the reconstructed object mesh, (2) a dynamics model that simulates the particle interactions using GNNs, and (3) a planning module that uses model predictive control (MPC) and solves the trajectory optimization problem using gradients from the learned dynamics model. Unlike prior works on learning-based particle dynamics, which assume temporal correspondence (Mrowca et al. 2018; Li et al. 2018, 2020a; Sanchez-Gonzalez et al. 2020; Shlomi et al. 2020), we train the dynamics model directly from raw visual data using loss functions that measure the distance between predicted and observed particle distributions.

In this work, we take concrete steps towards more general deformable object manipulation by enabling robots to model and manipulate elasto-plastic objects. Through extensive evaluations in both simulation and the real world, we show that our model-based planning framework allows the robot equipped with a parallel gripper to deform the plasticine into complex target shapes such as alphabetical letters and 3D objects in real life purely based on raw visual inputs. Notably, with just 10 minutes of real-world interaction training data, our learned GNN-based simulator can model the interactions between the gripper and the object and accurately predict the object's elastic/plastic deformation when applied actions.

Our paper demonstrates that the RoboCraft framework can manipulate plasticine-like material accurately with a model-based planning framework, and we hope the insights from this paper can inspire future works on more complex deformable object manipulation tasks.

Related Work

Modeling the Dynamics of Deformable Objects

Particle-based simulation is a popular family of methods to model deformable objects, wherein the dynamics are usually computed based on each particle's interaction with neighbor particles. Within this domain, position-based dynamics (Müller et al. 2007), smoothed particle hydrodynamics (Monaghan 1992), and Material Point Methods (MPM) (Sulsky et al. 1995) are widely used. In contrast to physics simulation techniques like Finite Element Methods (Reddy 2019; Zimmerman 2006; Felippa 2004), MPM eliminates the need for periodic re-meshing and state variable remapping. This makes it more suitable for handling large material deformations. Along with the advancement of particle-based methods, several differentiable simulators (Hu et al. 2019; Holl et al. 2019; Schoenholz et al. 2018) that can propagate gradients through the model have been introduced to capture the dynamics of non-rigid bodies. However, these simulation tools rely on approximate modeling approaches. They cannot fully capture the complexity of real-world physics, resulting in a large sim-to-real gap that limits their utility in real-world settings. Complex system identifications are required to bridge the gap (Le Lidec et al. 2021; Jatavallabhula et al. 2021).

Our approach avoids the sim-to-real gap by directly training a Graph Neural Network (GNN)-based simulator on robot interaction data from the real world. Our work follows the data-driven paradigm for learning physical dynamics (Nagabandi et al. 2020; Luo et al. 2018). Recently, people have demonstrated inspiring results in learning the dynamics of deformable objects such as clothes (Lin et al. 2022c), ropes (Chang and Padır 2020), and fluid (Leggard et al. 2021), with various representations including low-dimensional parameterized shapes (Matl and Bajcsy 2021), keypoints (Li et al. 2020b), latent vectors (Kurutach et al. 2018), and neural radiance fields (Li et al. 2022b). Inspired by prior approaches (Li et al. 2018; Sanchez-Gonzalez et al. 2020; Shlomi et al. 2020; Battaglia et al. 2016), we choose to use a GNN-based method due to its expressiveness in modeling the structure of an object, with very few assumptions on the underlying governing equations. Unlike prior methods that assume perfect perception and access to ground truth particle positions, our proposed method trains dynamics models directly from raw visual inputs. By doing so, our method gains the advantage of directly learning real-world dynamics, while approaches relying on the accuracy of physics simulators still suffer from the sim-to-real gap.

Manipulating Deformable Objects

Deformable object manipulation is a long-standing challenge in robotics. However, many existing methods focus on objects such as ropes (Wang et al. 2019; Nair et al. 2017; Yan et al. 2020; Sundaresan et al. 2020; Lee et al. 2021; Antonova et al. 2022; Yan et al. 2020), cables (She et al. 2021; Sánchez et al. 2020; Seita et al. 2021), clothes (Lin et al. 2021; Miller et al. 2012; Matas et al. 2018; Wu et al. 2020; Ha and Song 2022; Antonova et al. 2021b,a), and gauze (Thananjeyan et al. 2017). By contrast, we investigate the less-explored manipulation of elasto-plastic objects, such as plasticine, which are only studied in a limited number of

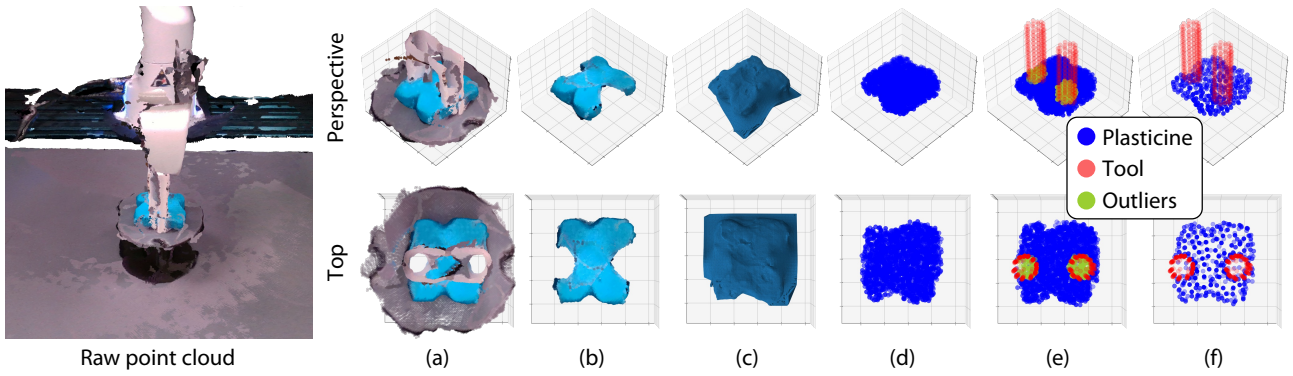

Figure 2. Perception Module of RoboCraft. We use four calibrated Intel RealSense D415 cameras to capture a complete point cloud of the robot's workspace. From the raw point cloud, our algorithm (a) crops to the region where the plasticine and the tool reside, (b) extracts the point cloud of the plasticine by color segmentation, (c) reconstructs a watertight mesh around the point cloud with Poisson surface reconstruction method, (d) uses SDF (Signed Distance Function) of the watertight mesh to sample points inside the mesh, (e) removes points lying within the SDF of the tools, (f) does alpha-shape surface reconstruction and uniformly samples 300 points on the reconstructed surface with Poisson disk sampling method.

works (Sanchez et al. 2018; Li et al. 2018; Lin et al. 2022b). Also, unlike existing studies on elasto-plastic materials that only consider the simple deforming task or tasks in the simulator, our work targets a more challenging task in the real world: pinching the elasto-plastic object into complex 3D shapes.

As for manipulation of deformable elasto-plastic objects (dough, plasticine, etc.), prior works propose to use either model-based (Cretu et al. 2011; Matl and Bajcsy 2021; Lin et al. 2022a; Li et al. 2022a) or adaptive methods (Navarro-Alarcon et al. 2016; Cherubini et al. 2020; Yoshimoto et al. 2011). As mentioned earlier, it is possible to leverage simulations as models for actual manipulation (Ganapathi et al. 2020). However, the challenges in state and parameter estimation make it prohibitive to simulate such objects accurately and thus hinder the transfer of policies to the real world. Another popular approach is to learn from expert demonstrations (Nadon et al. 2018), which has been proven effective in shaping sand (Cherubini et al. 2020) and pizza dough (Figueroa et al. 2016). However, obtaining demonstrations is usually expensive and sometimes prohibitive. Matl and Bajcsy (2021) propose to use soft robotic end-effectors to make a dough a sphere or rope-like shape with low-dimension representations such as bounding boxes. However, such low-dimensional representations are insufficient to fully capture the state of elasto-plastic objects, which possess infinite degrees of freedom. By contrast, RoboCraft equips the model with a particle representation and an expressive GNN-based dynamics model, which can model complex deformations in particle movement. The dynamics model enables our robot to shape objects into more complex unseen shapes, such as alphabetical letters and 3D objects. Our work is unique because we shape dough-like objects into complex shapes with semantic meanings.

Method

Problem Statement

The objective of this work is to let the robot use a parallel two-finger gripper to shape an elasto-plastic object to match a target shape $\mathbf{g}$. Specifically, we focus on using a sequence of pinching actions $\mathbf{a}_{0,...,T-1} \in \mathcal{A}$, given an observation of the initial state $\mathbf{s}_0$ of the plasticine. At time step t, the robot applies action $\mathbf{a}_t \in \mathcal{A}$ on the plasticine, and the state of the plasticine transition from $\mathbf{s}_t$ to $\mathbf{s}_{t+1}$ in response.

To predict the complex dynamics of the deformable plasticine, we propose to use a Graph Neural Network (GNN) Φ to learn the transition function $\Phi : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$. This dynamics model takes as input environment observations $\mathbf{s}_{t-h,...,t} \in \mathcal{S}$ and actions $\mathbf{a}_{t-h,...,t} \in \mathcal{A}$ to predict a future observation $\hat{\mathbf{s}}_{t+1}$, where h is the length of history and t is the current time step.

With the learned dynamics model in hand, we can naturally formulate the manipulation task as a model predictive control (MPC) problem. The cost function $\mathcal{J}$ of the MPC problem measures the distance between the state of the plasticine at the last time step T and the target shape $\mathbf{g}$. A sequence of actions of length T can be selected by minimizing the cost function:

$$(\mathbf{a}_0, ..., \mathbf{a}_{T-1}) = \underset{\mathbf{a}_{0,...,T-1} \in \mathcal{A}}{\arg\min}\, \mathcal{J}(\Phi(\mathbf{s}_0, (\mathbf{a}_0, ..., \mathbf{a}_{T-1})), \mathbf{g}) \tag{1}$$

Particle Sampling from Raw RGBD Data

Since the raw visual RGBD data are dense and noisy, we need to downsample the particles to get sparse and clean point cloud data to train the GNN-based dynamics model. Sometimes, certain parts of the object of interest are heavily occluded either by the robot hand or by other parts of the object. Therefore, it is challenging to obtain useful particles that can represent the object's shape from such data. To overcome these difficulties, our method leverages a combination of sampling algorithms and reasonable priors about the interactions between the gripper and the plasticine. We illustrate how RoboCraft addresses the issue through six steps in Figure 2.

We begin by converting RGBD images captured by four calibrated depth cameras into point clouds with the intrinsic and extrinsic parameters of the cameras. We merge the four partial point clouds into a complete point cloud, as shown on the left of Figure 2. Then, we crop out the region of

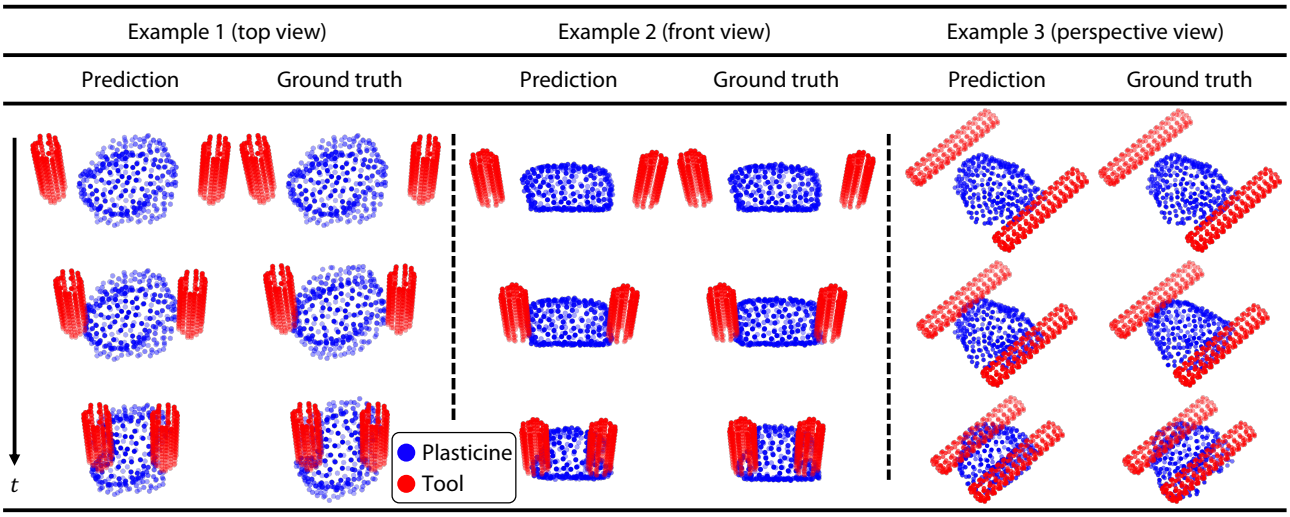

Figure 3. Dynamics Module of RoboCraft. Our GNN-based dynamics model accurately predicts the change of the plasticine's state in a long-horizon gripping task. We show three examples from the top, front, and perspective views. We compare the prediction of our dynamics model with the ground truth acquired from the perception module to show our model's accuracy. The blue dots represent the plasticine particles, and the red dots represent the tool particles.

interest based on the object's location. The outcome is shown in Figure 2 (a). After that, we use color segmentation in the HSV (hue, saturation, and value) space to extract the point cloud of the plasticine as shown in Figure 2 (b).

Two popular surface reconstruction algorithms are alpha shapes (Edelsbrunner and Mücke 1994) and ball pivoting (Bernardini et al. 1999). However, these algorithms are not robust to occlusion. We instead use Poisson surface reconstruction (Kazhdan et al. 2006), a method that can smooth the occluded surface by solving a regularized optimization problem. To prepare for Poisson surface reconstruction, we estimate the normal for each point in the point cloud by calculating the principal axis of its adjacent points. The outcome of Poisson surface reconstruction is not necessarily watertight, so we use the MeshFix algorithm (Attene 2010) to obtain the result in Figure 2 (c). Then in Figure 2 (d), we use the reconstructed mesh's Signed Distance Function (SDF) to sample points inside it randomly.

In the gripping process, heavy occlusions occur when the gripper pinches the object. To capture the full state of the dough, we need to remove the noisy points penetrating through the tool. We reconstruct the pose of the robot gripper at each time frame from the robot's end effector pose (recorded during data collection) and the ground truth tool mesh and then remove points based on the SDF of the tool mesh. We can access the tool meshes because they are designed in CAD software and 3D printed. The tool particles and the penetrating outliers of the plasticine particles are shown in Figure 2 (e).

Next, we apply alpha-shape surface reconstruction and the Poisson disk sampling (Yuksel 2015) method to uniformly sample 300 points on the reconstructed surface as the representation of the plasticine. To represent the parallel two-finger gripper, we uniformly sample an array of particles on the surface of their ground truth meshes to reflect their geometric features. A concatenation of the two sets of particles is our final scene representation at each time frame, as shown in Figure 2 (f).

We also use the continuity of the video data to inject simple geometric heuristics for better consistency between frames. We exploit the knowledge that when the gripper is not touching the plasticine, the object's shape should remain unchanged. Therefore, we reuse the sampling result from the previous frame in those cases. We also subsample the original videos to ensure that each video in the dataset has the same number of frames (12 in practice) for ease of training.

We recognize that our perception system relies on some prior knowledge of the environment. First, we assume that a single object of interest exists in the environment without distracting objects. Second, we assume the object features a color distinct enough for straightforward segmentation from the background. Lastly, we consider that occlusions are mostly caused by the robot's hand and gripper and self-occlusions from the concavities of the dough, with no other sources of occlusions present. Our perception system may fail if these assumptions are not met in the environment, or the object's complex geometry leads to too many self-occlusions.

Learning Object Dynamics via Graph Networks

We now introduce how RoboCraft constructs a particle graph and uses Graph Neural Networks (GNNs) to model the system's dynamics, as shown in Figure 3. With the approximation power of GNNs, only 10 minutes of real-world data are needed to train the dynamics model to accurately predict the rigid and non-rigid motions of the plasticine.

In a graph formed by states $\mathbf{s}_t = (\mathcal{O}_t, \mathcal{E}_t)$, the vertices $\mathcal{O}_t$ of the graph are the particles $\mathbf{o}_{i,t}$ of the object. Specifically, $\mathbf{o}_{i,t} = \langle \mathbf{x}_{i,t}, \mathbf{c}^o_{i,t} \rangle$, where $\mathbf{x}_{i,t}$ is the position of particle i at time t, and $\mathbf{c}^o_{i,t}$ denotes the corresponding attributes, including group information (i.e., which object the particle belongs to, either the dough or the rods in our setup) and the offset to the object center (i.e., the position in the local object frame). We select $|\mathcal{O}_t| = 300$ to balance efficiency and effectiveness well.

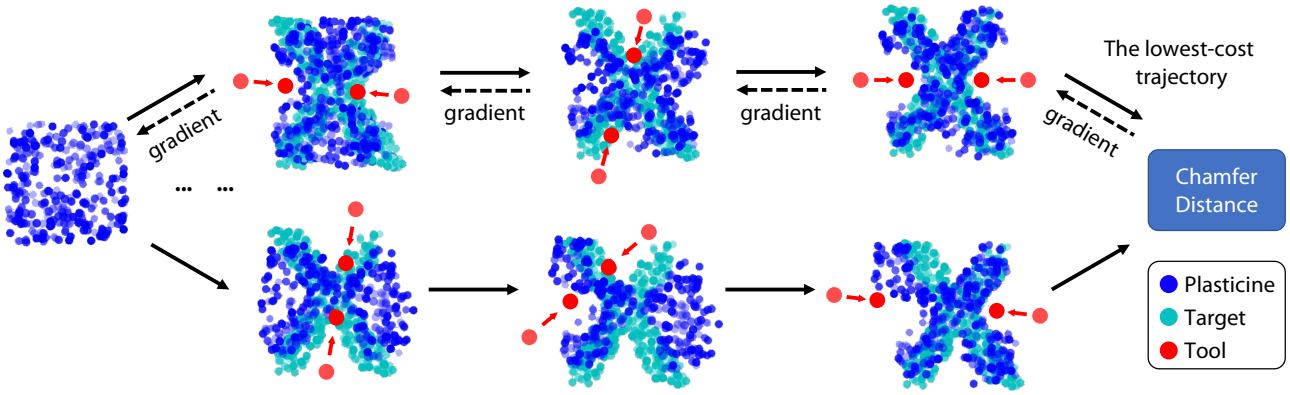

Figure 4. Planning Module of RoboCraft. We use a combination of sampling- and gradient-based trajectory optimization techniques to solve the planning problem. We first do grid sampling in the simplified action space and then forward the trained GNN-based dynamics model with the initial state of the plasticine and the sampled actions as input. After we obtain the final state of the plasticine after applying the actions, we compute the Chamfer distance between them and the target state. Next, we apply gradient-based trajectory optimization on the lowest-cost trajectory to improve the solution further. The red dots and arrows represent the motion of the parallel two-finger gripper.

The edges $\mathcal{E}_t$ between the vertices are computed dynamically from their spatial relationships over time. We connect all the neighbors within a predefined neighborhood radius. We use different neighborhood radiuses for the edges between plasticine particles and the edges between plasticine particles and tool particles.

We use $\mathbf{e}_k = \langle u_k, v_k, \mathbf{c}_k^e \rangle$ to represent relations between particles (e.g., a pair of particles with an edge), where $1 \leq u_k, v_k \leq |\mathcal{O}_t|$ are the receiver particle index and sender particle index respectively, k is the edge index, and $\mathbf{c}_k^e$ is the type of relationship (e.g., object internal relation or gripper-to-object relation).

The goal of the GNN is to infer the system dynamics and predict the future from the current graph as $\hat{\mathbf{s}}_{t+1} = \Phi(\mathbf{s}_t, \mathbf{a}_t)$. $\mathbf{a}_t$ is represented as the ground-truth point cloud of the two-finger gripper in the world frame, sampled using the tool's ground-truth mesh and transformed with the robot's end-effector pose. Additionally, delta motions at each point of the tool point cloud are appended to these nodes' feature vectors as an approximation of the particle velocities.

We use f_O^{enc} and f_E^{enc} to encode the object features and the relation features respectively as follows:

$$h_{i,t}^o = f_O^{\text{enc}}(\mathbf{o}_{i,t}), \tag{2}$$

$$h_{k,t}^e = f_E^{\text{enc}}(\mathbf{o}_{u_k,t}, \mathbf{o}_{v_k,t}, \mathbf{c}_k^e). \tag{3}$$

We then use an object function f_O^{dec} and a relation function f_E^{dec} to model the dynamics. The future state at time $t+1$ is predicted as

$$b_{k,t} = f_E^{\text{dec}}(h_{k,t}^e)_{k=1,\cdots,|\mathcal{E}_t|}, \tag{4}$$

$$\hat{\mathbf{o}}_{i,t+1} = f_O^{\text{dec}}(h_{i,t}^o, \sum_{k \in \mathcal{N}_i} b_{k,t})_{i=1,\cdots,|\mathcal{O}_t|}, \tag{5}$$

where $\mathcal{N}_i$ is a set of relations with particle i as the receiver. During training, we also use multi-step message passing to handle the instantaneous propagation of forces. We localize the particle locations in the object frame by subtracting the object's center of mass from the particle locations in the world frame. In this way, the GNN predicts the movement of dough in an object-centric way and is naturally translation-invariant.

The motion of the plasticine mostly comprises non-rigid motion. But when one finger of the gripper touches the plasticine before the other comes into contact, the plasticine may have some rigid motion. Therefore, we incorporate two multi-layer perceptrons (MLPs) to independently predict the rigid and non-rigid motions, which are combined to give the overall motion predicted by the GNN. We clamp the predicted motion to a reasonable range during training to regularize and speed up training. Since we use distribution-based loss functions (i.e., Chamfer Distance and Earth Mover's Distance) for 3D point clouds, such regularization techniques can ensure that the model initializes within a reasonable parameter range.

Loss Functions

Since our training data comes from sampled point cloud data, there is no one-to-one correspondence among the points of each frame as required by particle-wise losses. We explore two loss functions to measure the similarity between the distributions of point cloud data.

Consider $\mathcal{O}_t, \widehat{\mathcal{O}}_t \subseteq \mathbb{R}^3$. The Earth Mover's Distance (EMD) between them is defined as

$$\mathcal{L}_{\text{EMD}}(\mathcal{O}_t, \widehat{\mathcal{O}}_t) = \min_{\mu:\mathcal{O}_t \to \widehat{\mathcal{O}}_t} \sum_{\mathbf{x} \in \mathcal{O}_t} \|\mathbf{x} - \mu(\mathbf{x})\|_2, \tag{6}$$

where $\mu : \mathcal{O}_t \to \widehat{\mathcal{O}}_t$ is a bijection (Rubner et al. 1998). The EMD solves an assignment problem. For all but a zero-measure subset of point set pairs, the optimal bijection μ is unique and invariant under the infinitesimal movement of the points. We implement EMD by solving a linear sum assignment problem (Crouse 2016) and computing the Euclidean distance between the corresponding particle positions after finding the match. Intuitively, EMD matches distributions while preventing outliers in the point cloud by the bijection definition.

The Chamfer Distance (CD) between $\mathcal{O}_t, \widehat{\mathcal{O}}_t \subseteq \mathbb{R}^3$ is

$$\mathcal{L}_{\text{CD}}(\mathcal{O}_t, \widehat{\mathcal{O}}_t) = \sum_{\mathbf{x} \in \mathcal{O}_t} \min_{\mathbf{y} \in \widehat{\mathcal{O}}_t} \|\mathbf{x} - \mathbf{y}\|_2^2 + \sum_{\mathbf{y} \in \widehat{\mathcal{O}}_t} \min_{\mathbf{x} \in \mathcal{O}_t} \|\mathbf{x} - \mathbf{y}\|_2^2. \tag{7}$$

Here, we slightly abuse Chamfer Distance because it does not satisfy the triangle inequality (Fan et al. 2017). Intuitively, our definition of CD finds the nearest neighbor for a point in the other set and sums the squared distances.

Our loss function is a weighted sum of the two distance functions mentioned above:

$$\mathcal{L}(\mathcal{O}_t, \widehat{\mathcal{O}}_t) = w \cdot \mathcal{L}_{\text{CD}}(\mathcal{O}_t, \widehat{\mathcal{O}}_t) + (1 - w) \cdot \mathcal{L}_{\text{EMD}}(\mathcal{O}_t, \widehat{\mathcal{O}}_t), \tag{8}$$

where w is a weight factor of CD. Empirically, we find the optimal value for our task is $w = 0.3$.

To improve the accuracy of long-horizon predictions during inference time, we train the model to predict multiple time steps forward from the current state. The loss is calculated as the sum of the distances between the predictions and the corresponding ground-truth states. Thus, our training loss is defined by

$$\mathcal{L}_{\text{train}} = \sum_{i=0}^{s} \mathcal{L}(\mathcal{O}_{t+i}, \widehat{\mathcal{O}}_{t+i}), \tag{9}$$

where the dynamics model takes $\widehat{\mathcal{O}}_{t+i-1}$ as the input to predict $\widehat{\mathcal{O}}_{t+i}$ when $i > 0$. We select $s = 3$ as the optimal value based on empirical experiments.

Action Space

We simplify the action space of each grip into a parameterized space in an object-centric way: $\{r, \theta_z, \phi, \theta_x, d\}$, as shown in Figure 5 (b). In this simplified action space, $\{r, \theta_z, \phi\}$ represent the center of the grip in the spherical coordinate system, i.e., the midpoint of the line segment connecting the centers of mass of the gripper's two fingers. This is a relative position, assuming the coordinate system's origin lies at the object's center. θ_x is the robot gripper's rotation about the x-axis (the coordinate system in the robot frame is illustrated in Figure 5). d represents the minimal distance between gripper fingers during this pinch. The robot gripper pinches the plasticine at a constant speed. During planning time, lower and upper bounds for each action parameter are provided to constrain the sampled actions into a valid range.

We leverage a few priors to simplify the action space. First, since the actions with non-zero rotations around the y-axis are usually invalid and likely to collide with the object stand, we prune the action space by assuming that the end effector's rotations around the y-axis are always 0. To further remove invalid actions in the space, we assume that the line segment connecting the centers of mass of the gripper's two fingers is always on the same plane and perpendicular to the line segment connecting the center of the object and the center of the grip. Therefore, we effectively reduce the 7-DoF action space (6-DoF end-effector pose and the gripping width) into a 5-DoF space without losing much flexibility.

We recognize that such simplified action spaces are not desirable in general manipulation settings. Our insight is that for most tasks, especially tasks where the robot uses a tool, there is redundant and undesired space in the full 6-DoF action space. In most cases, humans also adhere to specific rules and follow a constrained action space when using a specific tool.

Goal-Conditioned MPC

We denote $\mathbf{g}$ as the target shape of the plasticine and $\mathbf{a}_{0:T-1}$ as the action sequence sampled from the simplified action space, where T is the time horizon. We denote the resulting trajectory after applying the control inputs as $\mathbf{s}_{0:T}$. The task is to determine the actions that minimize the distance between the actual outcome and the specified goal $\mathcal{J}(\mathbf{s}_T, \mathbf{g}) \triangleq \mathcal{L}(\mathcal{O}_T, \mathcal{O}_\mathbf{g})$.

We use gradient-based trajectory optimization to obtain the trajectory with the lowest cost, as shown in Figure 4. We first do grid sampling in the simplified action space and compute the costs using the GNN-based dynamics model. We then apply the limited-memory BFGS (Fletcher 2013) algorithm on the lowest-cost trajectories to optimize the actions using gradients with Chamfer distance as the loss function. We select limited-memory BFGS to stabilize the gradient descent process.

At the beginning of closed-loop control, the cameras capture the complete point cloud of the workspace and pass the data to our perception module to get a clean and sparse point cloud of the dough. The robot takes as input the current configuration of the dough to plan the actions and manipulate the dough. After that, the robot lifts its hand to avoid occluding the camera views, and then visual feedback is collected to plan the next move. In the standard form of MPC, the controller can use intermediate states as feedback to correct future actions. As a trade-off in this work, acquiring intermediate states with particle sampling would improve the effectiveness but reduce the time efficiency. Hence, we provide intermediate states from cameras only after each grip, ignoring the visual feedback within the duration of a grip.

Another issue is that we may require more or fewer grips when aiming at unseen shapes. We perform MPC with a fixed look-ahead horizon of two grips to address it. After we plan the actions, the robot will execute one grip and collect visual feedback to plan the following actions. The robot performs additional grips until the loss is small enough or the maximum number of grips (six in practice) is reached.

Experimental Setup

Physical Setup

We show the robot setup in the real world in Figure 5. We use a Franka Emika Panda robot arm with 7 DoFs and Franka's parallel jaw gripper for real-world manipulation. We substitute the original fingers with a pair of 3D-printed cylindrical parallel gripper fingers as shown in Figure 5(c). Four RealSense D415 RGBD cameras are fixed at four locations surrounding the plasticine to capture the RGBD images at 30Hz and 1280×720 resolution. The four cameras are calibrated to get the relative positions with respect to each other and the robot base to reconstruct the object geometry. We use a blue Play-Doh modeling compound as the deformable object. Due to the kinematic limits of the robot, we designed a rotating object stand as the platform to place the plasticine, as shown in Figure 5 (d). In this way, the robot can hold the cavities on two sides of the stand to rotate the stand along with the plasticine before gripping. This is

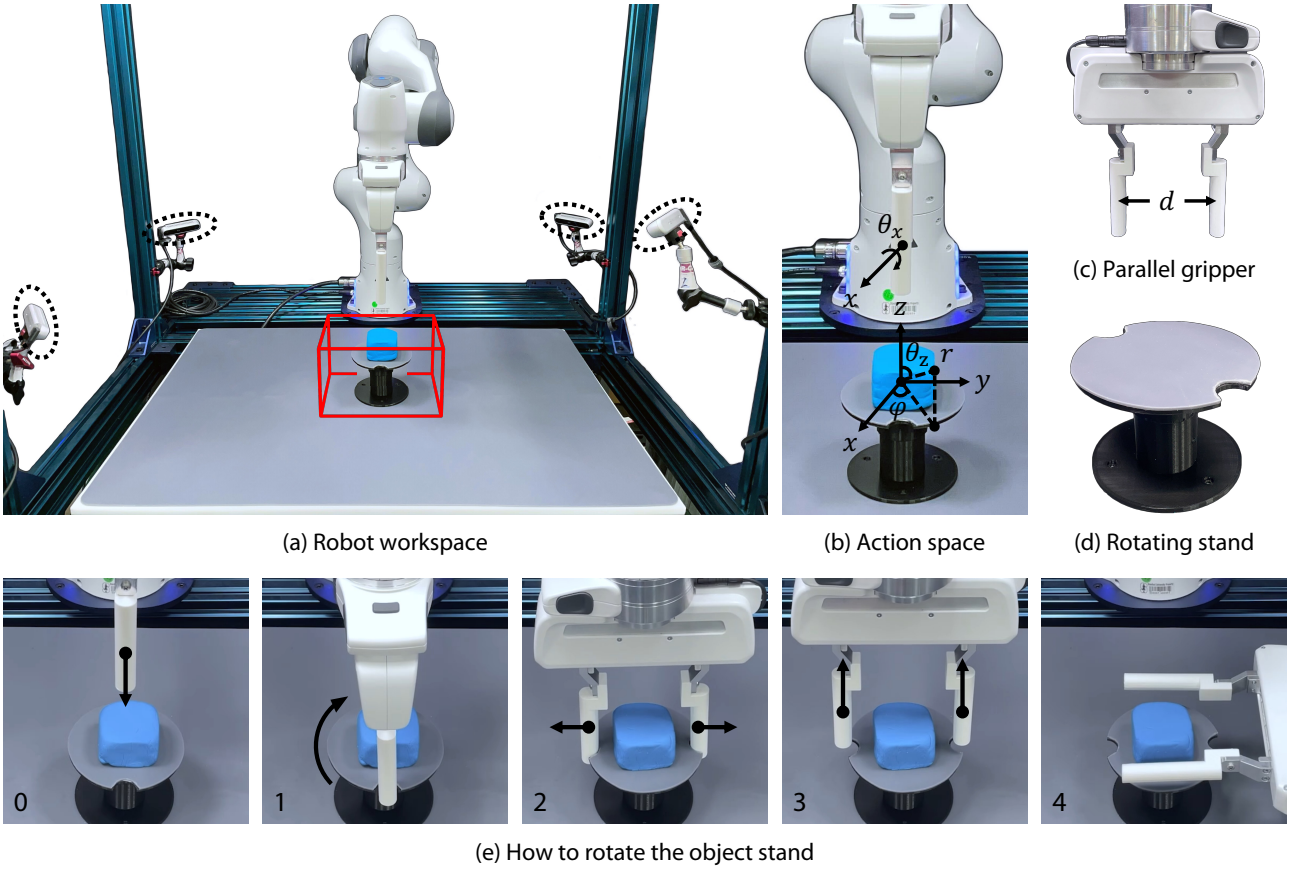

Figure 5. Robot Setup. (a) An overview of the robot workspace. The dashed black circles show the four RGBD Intel RealSense D415 cameras mounted at the four corners of the robot table. The red cubic contour denotes the robot's manipulation area. (b) We illustrate the xyz coordinate system in the robot frame and the simplified action space of the gripping task. (c) The 3D-printed parallel two-finger gripper that the robot uses to pinch the plasticine. (d) Since some end effector poses are close to the robot's kinematic limits, we designed a rotating object stand to place the plasticine so that the robot can rotate the stand instead of rotating its hand on the z-axis. (e) The robot can rotate the object stand through three steps: (1) insert the gripper into the cavities on two sides of the gripper; (2) rotate the stand along with the plasticine; (3) open two fingers to release the stand. Then, the robot can start gripping.

equivalent to rotating the robot's end effector about the z-axis and effectively avoids approaching the robot's kinematic limits. After rotating the object stand, the robot only needs to rotate its hand on the x-axis before gripping.

Data Collection

We collect 6,000 frames (10 mins) of training data, including 40 episodes with a horizon of 150 frames. For each episode, five grips are applied to the plasticine. At the beginning of each episode, we shape the plasticine into a random bulky shape. The data collection behavior policy randomly selects parameters in the action space. During each episode, we save the partial point clouds from the four RGBD cameras and the robot joint poses from the robot controller. The data is only saved when the centers of the gripper's two fingers are on the same plane as the center of the object to optimize memory usage.

Tasks Specifications

With the final goal of real-world manipulation in mind, we first set up experiments in simulation (Huang et al. 2020) with two capsule-shaped fingers and a cuboid-shaped plasticine. We chose 26 alphabetical letters and five other shapes as our target shapes in the simulator.

In the real world, we use four letters and four 3D shapes. Note that for letters with holes such as 'R' or 'A,' we only focus on the contour because the current tools are not suitable for carving out the holes in these letters. We believe that the proposed shapes span a large spectrum of possible shapes and can serve as a benchmark for future research works.

We quantitatively evaluate each step of our whole framework, including the particle sampling, the dynamics model training, and the manipulation results with Chamfer distance (CD) and Earth mover's distance (EMD) as our main metrics.

Experimental Results

In this section, we evaluate RoboCraft for various complex deformable elasto-plastic object manipulation tasks in simulation and the real world. We compare the proposed method with a set of baseline algorithms.

Sampling Results

We first compare the performance of the proposed sampling method and the patch-based baseline. The patch-based method extracts the partial point cloud of the plasticine based on color. It then reconstructs a convex hull around the

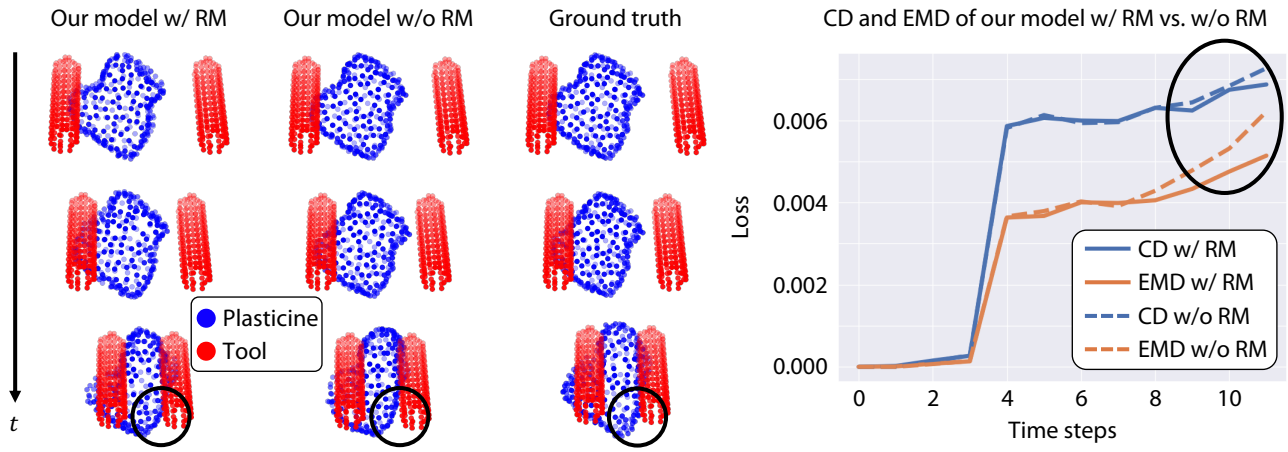

Figure 6. Rigid Motion. On the left is a qualitative comparison between our model with rigid and non-rigid motion predictors and our model with only the non-rigid motion predictor. The first two columns are the predictions of the two variations of our dynamics models, and the third column shows the ground truth trajectory acquired by the perception module. On the right is a quantitative illustration of the CD and EMD between the predicted and target states for these two models over time. The black circles highlight the discrepancy in accuracy between the two dynamics models. Although the non-rigid motion predictor can sometimes translate rigid motion into plausible non-rigid motion, we observe that a separate rigid motion predictor allows our model to make more accurate predictions.

Table 1. We show the sampling results averaged over 120 frames in Chamfer distance (CD) and Earth mover's distance (EMD). These two methods aim to address the occlusion issue by the robot gripper in the raw point cloud, as shown on the left of Figure 2. The idea of the crop-based sampling method is first to sample more particles than needed and then remove the invalid ones. For example, it removes the plasticine particles penetrating the tools by the SDF of the tool meshes. By contrast, the patched-based sampling method samples point clouds on the tool meshes' surface to patch the plasticine's incomplete surface point cloud.

Methods	CD↓	EMD↓
Crop-Based (Ours)	**0.0374** $\pm$ 0.0001	**0.0308** $\pm$ 0.0002
Patch-Based	0.0384 $\pm$ 0.0003	0.0317 $\pm$ 0.0005

Table 2. We show the mean and standard deviation of the dynamics model's performance over ten unseen test trajectories when trained with different loss functions. The metrics are Chamfer Distance and Earth Mover's Distance to the ground truth.

Loss function	CD↓	EMD↓
Mixed Loss (Ours)	**0.0070** $\pm$ 0.0005	**0.0048** $\pm$ 0.0004
CD Loss	0.0072 $\pm$ 0.0007	0.0052 $\pm$ 0.0007
EMD Loss	0.0081 $\pm$ 0.0010	0.0055 $\pm$ 0.0011

incomplete point cloud and extracts points from the gripper point cloud within the reconstructed convex hull to patch the point cloud of plasticine.

In Table 1, we compute the average distance between the sampled particles and the ground-truth particles provided by the simulator. Our method achieves lower losses for CD and EMD and outperforms the patch-based baseline. These results resonate with the intuition that additional shape priors of the gripper can significantly improve the sampling quality in occluded scenes.

Dynamics Model Learning

We construct the graph by connecting edges between two vertices within the neighborhood radius to learn the dynamics model. When the neighborhood radius increases, more edges are connected in the graph, so the GNN becomes more powerful at modeling the interactions between particles, at the cost of requiring more computation. Based on this trade-off, we choose the neighbor radiuses of plasticine particles and tool particles to be 0.025. For each edge and each vertex, we encode it with a 3-layer MLP with hidden and output layers each of size 150. The propagator comprises one fully connected layer with an output layer size of 150. The rigid and non-rigid motion predictors are another two 3-layer MLPs with a hidden layer size of 150. All of the neural networks use ReLU activations. The models are trained for 100 epochs with the Adam optimizer (Kingma and Ba 2015), a batch size of 4, and a learning rate of 10^{-4}.

We then evaluate GNN-based dynamics models trained with different loss functions. In Table 2, we run an ablation on the value of w, which is the weight of the CD in the mixed loss function. When $w = 0$, the training loss is vanilla EMD; when $w = 1$, the training loss is vanilla CD. We observe that the model has a lower EMD loss during inference time using a mixed loss function with $w = 0.3$ compared to using vanilla EMD to train, demonstrating the benefits of using a mixed loss function to train the GNN-based dynamics model.

Our dynamics model can predict a combination of rigid and non-rigid motions. In Figure 6, we visualize the predicted particles when rigid motions exist in the ground truth trajectory. Since there is a misalignment between the center of the grip and the center of the plasticine, the left finger, which touches the plasticine before the right one, will push the plasticine to the right side for a small distance. In such scenarios, our model has more accurate predictions than our model with the rigid motion predictor turned off.

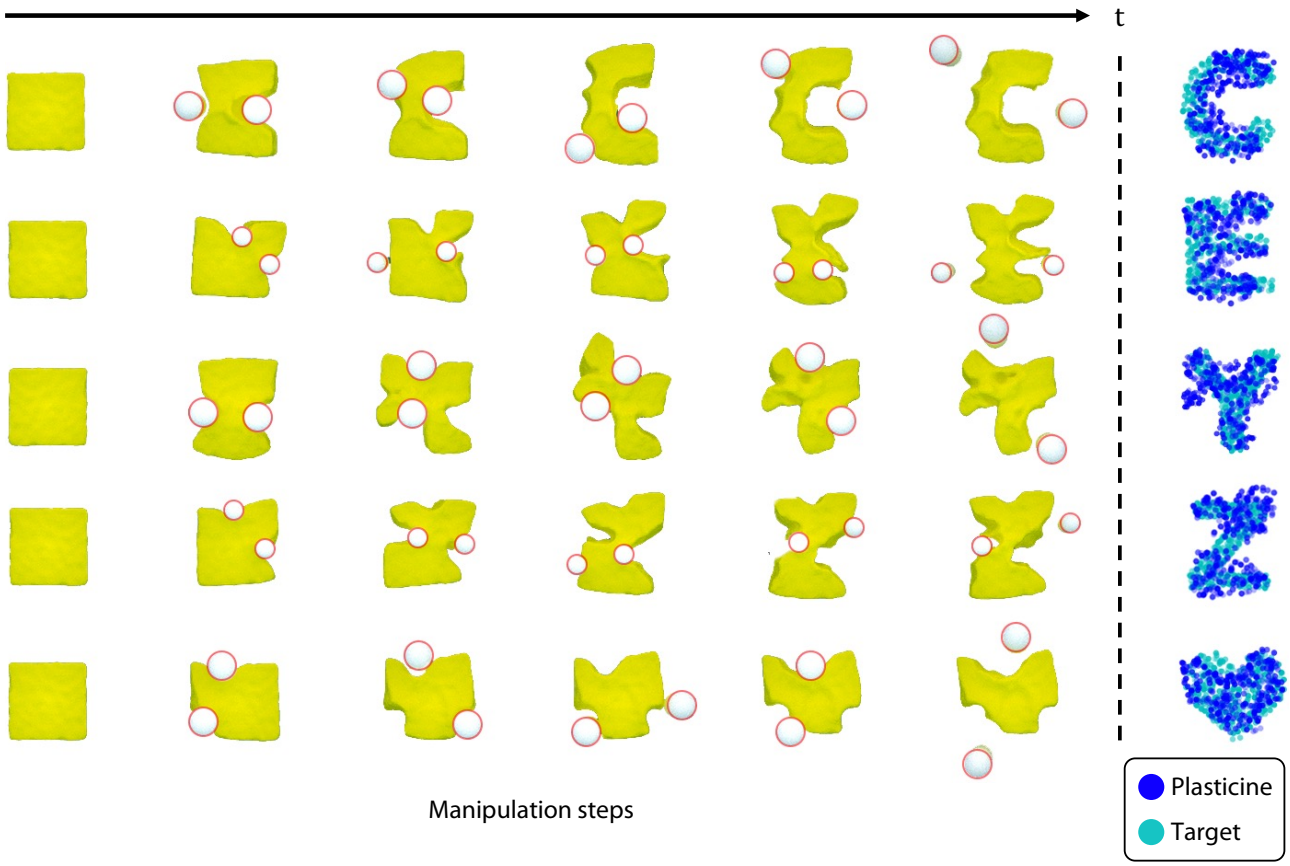

Figure 7. Manipulation Results in the Simulation. On the left are the manipulation steps of the alphabet letter 'C,' 'E,' 'Y,' 'Z,' and a heart in the simulator. The results and their overlay with the target point cloud are on the right. The cyan point clouds are the targets, and the blue point clouds are the results. These targets all take five grips with 15 steps per grip to accomplish, demonstrating RoboCraft's ability to solve long-horizon planning tasks.

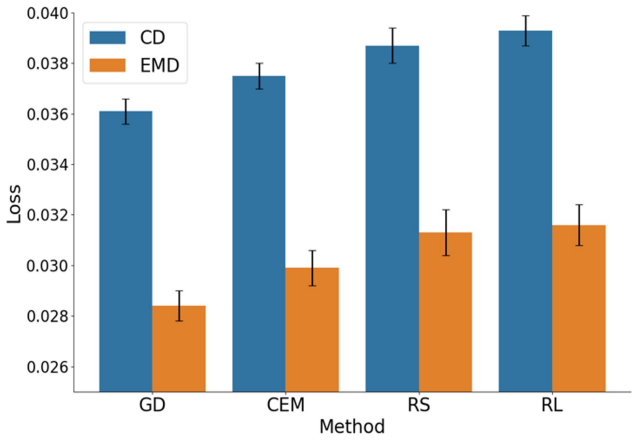

Figure 8. Comparison of Different Trajectory Optimization Methods. We benchmark four different optimization algorithms, including gradient descent (GD), cross-entropy Method (CEM), random shooting (RS), and reinforcement learning (RL). CD and EMD are computed on the target shape 'A.' Lower is better. We show that the gradient-based trajectory optimization method based on limited-memory BFGS outperforms other methods. We run each method using different random seeds ten times and report the means and standard deviations. This shows that our approach is statistically superior to alternative methods and robust to initial conditions since the standard deviation is small.

Manipulation Results in Simulation

In Figure 7, we visualize manipulating the object towards the target shapes using a gradient-based method. The agent can handle various challenges, such as small grooves in the letter 'E' and asymmetry in the letter 'Z.' This demonstrates that the method can leverage the GNN-based dynamics model for effective manipulation under the MPC framework. More visualized results can be found in the video material.

Table 3. We show the results for tool selection averaged over all the target shapes. Two grippers with different sizes can be selected for each grip. The results show that our method has the potential to generalize to tools of different geometries and sizes and can benefit from a more flexible choice of tools.

Methods	CD↓	EMD↓
w/ tool selection	**0.0408** $\pm$ 0.005	**0.0337** $\pm$ 0.007
w/o tool selection	0.0409 $\pm$ 0.005	0.0345 $\pm$ 0.007

We compare different trajectory optimization methods, including random shooting (RS), gradient-based planning (GD), gradient-free planning (CEM), and reinforcement learning (model-based Soft Actor-Critic). Specifically, we use the same action space as in other planning methods for the reinforcement learning baseline. The state space consists of the particles' position and the gripper. The reward function is computed from the difference in CD after each grip. We use a discount factor of 0.99 and a learning rate of 0.001 with the Adam optimizer for training. We use 2-layer MLPs with 256 hidden units and ReLU activation for both the policy and critic models. We initially collected 50 episodes of warm-up data before training. The replay buffer size is 10^6, and the

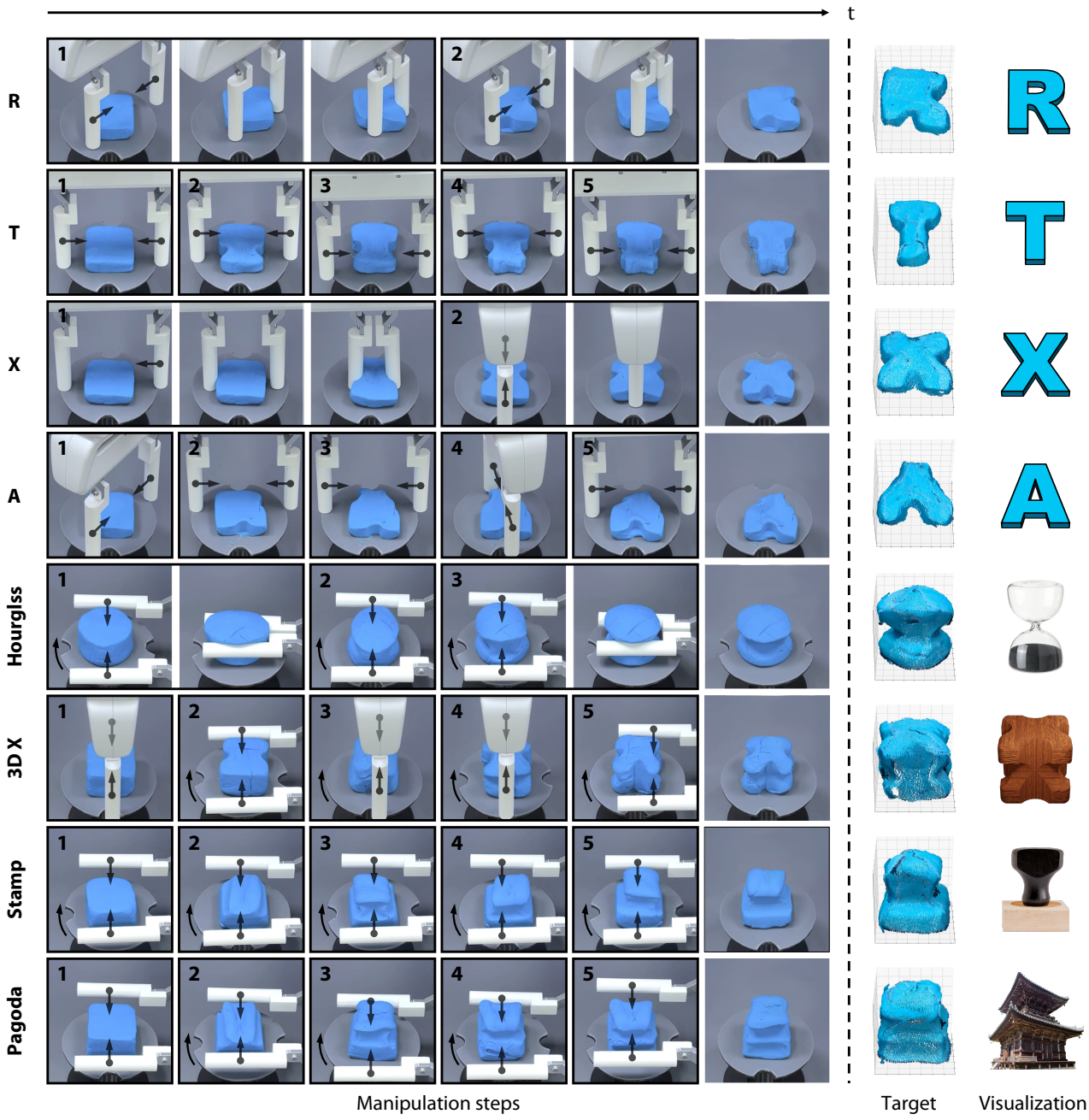

Figure 9. Manipulation Results on a Real Robot. On the left are the manipulation steps of the alphabet letters 'R,' 'T,' 'X,' 'A,' an hourglass, a 3D 'X,' a stamp, and a pagoda. We use black arrows to illustrate the rotation of the object stand and the motion of the robot gripper. The numbers at the top left corner of the images denote the i-th grip, and images that belong to the same grip are placed inside the same black contour. The sixth column shows the final result. On the right are the corresponding target point clouds acquired from expert human demonstrations using the same robot gripper to pinch the target. The last column shows the targets' visualizations in the real world. Note that these images do not supervise our proposed method but merely illustrate the targets we attempt to achieve.

target smooth coefficient is 0.005. As shown in Figure 8, we find that gradient-based optimization with the learned model outperforms all other methods. We attribute this to the strong optimization power of gradient-based optimization in the simplified action space. We note that the limited-memory BFGS optimizer provides the best results in practice.

We also evaluate our method when two grippers with different sizes can be selected for each grip. The robot iterates over the two grippers and selects the one that yields a smaller Chamfer distance to the target. In Table 3, the performance can be further improved when more tool choices are provided. This implies that the geometry of the parallel two-finger gripper is potentially a limiting factor of our method to accomplish more fine-grained targets.

Manipulation Results in the real world

We now present an evaluation of our method on a real robot with only 10 minutes of real-world robot interaction data to train the dynamics model. We show that RoboCraft can manipulate the plasticine to unseen shapes in the training data. Example trajectories of the robot manipulating the plasticine are shown in Figure 9. To regularize our

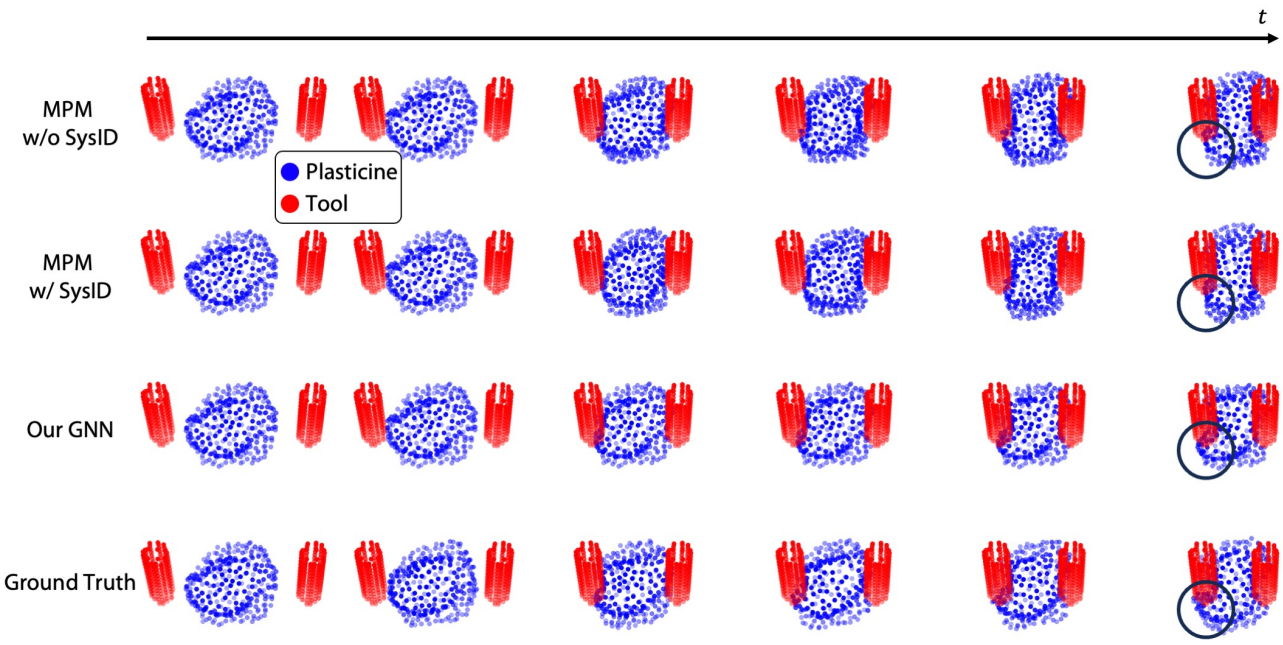

Figure 10. Qualitative Comparison with an MPM simulator on Dynamics Prediction. Row 1-3 are the dynamics predictions of our method compared with baselines on the same pinching action. Row 4 is the ground truth point cloud trajectory. Our learned GNN-based dynamics model predicts the dynamics more accurately than an MPM simulator without and with system identification. The black circles highlight the discrepancy in accuracy between these dynamics models.

manipulation results in the real world, we 3D printed two molds to shape a cuboid (the dimension is 6cm, 6cm, 4cm along the x, y, z axes respectively) or a cylinder (the radius is 6.77cm and the height is 4cm) as initial shapes before each experiment.

RoboCraft successfully identifies the cavities in the target shape 'R' and grips the correct position. To shape the long tail of the letter 'T' from a cube, the robot rotates its hand to a horizontal position and grips six times until the distance between the state of the plasticine is close enough to the target. The robot accomplishes the goal of the letter 'X' with just a horizontal grip followed by a vertical grip. For more complex shapes such as the letter 'A,' the method also seems to discover a solution that achieves the target shape creatively. These results illustrate that, although the task is very challenging, our method can still perform well with a small amount of training data.

We also tested our method on a set of more complicated target shapes, which resemble 3D objects in real life. The first task is to shape an hourglass from a cylinder. After two grips, the robot approximately achieves the target but decides to pinch again to refine the result. The next target is a 3D 'X,' which looks like an 'X' from any direction. The robot accurately predicts how the plasticine will deform and elegantly accomplishes the task. The stamp-shaped target is a more challenging one. It takes the robot six steps to accomplish the goal. Finally, we give the robot a pagoda as its target. The robot astutely identifies the cavities in the target and pinches the plasticine at the correct positions. The complete process can be found in the video material.

Our model is learned purely from offline data collected via random interactions (10 minutes); thus, the target shapes have never been seen during training. Yet, our pipeline can still achieve these targets with reasonable accuracy.

Comparison with Other Dynamics Models in the Real World

RoboCraft uses a GNN-based dynamics model learned from real-world data. We conducted two experiments to evaluate the efficacy of our Graph Neural Network (GNN) compared to other widely used approaches. In the first experiment, our GNN's dynamic predictions are compared to a Material Point Method (MPM)-based physics simulator with hand-selected parameters and parameters optimized through extensive system identification. We establish the MPM dynamics model following prior work (Huang et al. 2020). Figure 10 demonstrates that our GNN offers more accurate predictions than the MPM simulator with and without system identification.

For the identification process, we employ our perception module to map real-world observations into a full state as the initial state in the MPM simulator. The MPM simulator takes the actions as input and predicts how the dough will deform. We use the same objective function as the training loss for our GNN – a hybrid loss of Chamfer Distance and Earth Mover's Distance. The objective function is computed between the predicted final state and the ground truth final state in the trajectory. Based on this objective function, we apply Bayesian Optimization (Victoria and Maragatham 2021) to determine the optimal parameters for the physics simulator, focusing on three key physics parameters: yield stress, Young's modulus E, and the Poisson ratio ν (Landau et al. 1986). The boundaries for these parameters are set as follows: yield stress $\in (1e2, 1e4)$, $E \in (1e3, 1e4)$, and $\nu \in (0.0, 0.5)$. We begin with ten initial steps of random exploration and run the optimization for 100 iterations. The average optimized parameters across ten test trajectories were yield stress $= 988.11$, $E = 5262.12$, and $\nu = 0.33$. Qualitative results of one trajectory are presented

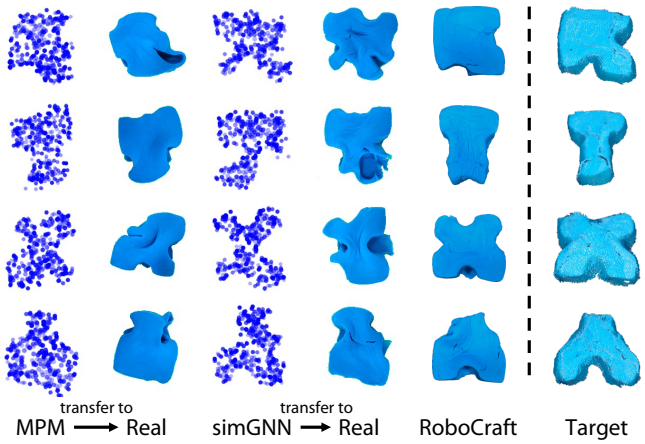

Figure 11. Comparison with Other Dynamics Models in the Real World. The first two columns show the outcomes of transferring control signals from Material Point Method (MPM) models into the real world. The third and fourth columns show the outcomes of transferring the GNN-based model trained in simulation (simGNN) into the real world. We compare the outcomes of RoboCraft and the target point clouds.

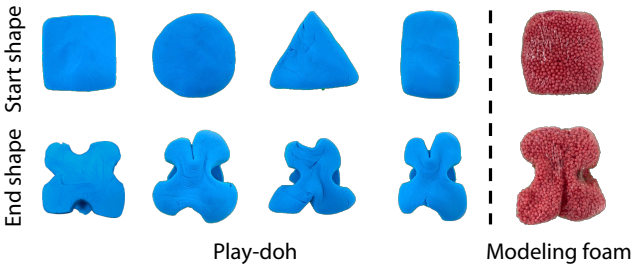

Figure 12. Generalizing to Different Initial Shapes and Material Types. On the left is applying our method on 4 different initial shapes, including square, circle, triangle, and rectangle. On the right are results from applying the dynamics model trained on play-doh to modeling foam without retraining. This shows our method's ability to generalize.

Table 4. To compare our GNN with MPM simulators with and without system identification, we report the results for dynamics prediction averaged over ten unseen test trajectories. The metrics are Chamfer Distance and Earth Mover's Distance to the ground truth.

Methods	CD↓	EMD↓
Our GNN	**0.0070** ± 0.0005	**0.0048** ± 0.0004
MPM w/o SysID	0.0082 ± 0.0005	0.0071 ± 0.0003
MPM w/ SysID	0.0077 ± 0.0006	0.0058 ± 0.0004

in Figure 10, and the Chamfer Distance and Earth Mover's Distance to the ground truth are reported in Table 4. Since our dynamics model is directly trained on real-world data, it does not suffer from the sim-to-real domain gap and outperforms physics-based simulators in this long-horizon dynamics prediction task.

The second experiment assesses the real-world manipulation performance of our method against two baselines: (1) an MPM-based physics simulator and (2) a GNN trained with simulation data. As shown in Figure 11, RoboCraft outperforms both baselines. We attribute this to the fact that the downstream manipulation performance highly depends on the dynamics model's prediction accuracy for model-based approaches. As previously shown in the first experiment, RoboCraft learns from real-world data while others suffer from the large domain gap between simulation and reality.

Generalizing to Novel Initial Shapes and Material Types

We test the generalization ability of RoboCraft by applying the framework to different initial object shapes and material types (modeling foam). As shown on the left of Figure 12, we find that RoboCraft can manipulate the objects with a circle, triangle, or rectangle as their initial shapes into our target shape, 'X.' We also display the result of the original square shape in this figure for comparison. On the right side of Figure 12, we test whether the dynamics model of RoboCraft trained on plasticine can be used to manipulate modeling foam. It can also roughly make an 'X' shape without any retraining. The results illustrate the potential of RoboCraft to generalize to various unseen scenarios.

Conclusion

We have proposed the first model-based framework that manipulates elasto-plastic objects to complex unseen target shapes both in simulation and on a real robot, with only 10 minutes of robot exploration time. Real robot experiments demonstrate the high efficacy of the proposed system.

The comparison in Figure 10 and Figure 11 shows that our data-driven graph neural networks are more accurate than traditional MPM-based simulators in the real world. An efficient model usually indicates its high practicality in real-world applications. However, a model-based planning method is slow when deployed online because the planning optimization can be time-consuming, which is usually unacceptable in dynamic real-world environments. Some possible solutions are to learn an effective sampling distribution to prune the search space or directly learn a policy under the supervision of the dynamics model.

The best loss function to use during planning also remains to be determined. We use the Chamfer distance in this work, which pays equal attention to each particle in the point cloud but sometimes fails to capture the essential features of the target. One future direction is to leverage pre-trained neural networks to extract holistic features from the point clouds to construct the loss function for planning.

Another limitation of RoboCraft is that the target shapes it could possibly achieve are highly constrained by the geometry of the parallel two-finger gripper. One potential way to resolve this issue is to add a tool-switching module and let the robot learn the most appropriate tool to use at different stages.

Building a robot capable of complex deformable object manipulation tasks requires systematic improvement in many aspects of robotics research. RoboCraft attacks the core problems of soft body manipulation and presents a welcoming avenue for future researchers to investigate.

Acknowledgement

We thank Samuel Clarke for helpful discussions, hardware support, and proofreading the paper. This work is in part supported by the Stanford Institute for Human-Centered AI (HAI), the Samsung Global Research Outreach (GRO) Program, the Toyota Research Institute (TRI), and Amazon, Autodesk, Salesforce, and Bosch.

References

Antonova R, Shi P, Yin H, Weng Z and Jensfelt DK (2021a) Dynamic environments with deformable objects. In: *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*.

Antonova R, Varava A, Shi P, Carvalho JF and Kragic D (2021b) Sequential topological representations for predictive models of deformable objects. In: *Learning for Dynamics and Control*. PMLR, pp. 348–360.

Antonova R, Yang J, Sundaresan P, Fox D, Ramos F and Bohg J (2022) A bayesian treatment of real-to-sim for deformable object manipulation. *IEEE Robotics and Automation Letters* 7(3): 5819–5826.

Attene M (2010) A lightweight approach to repairing digitized polygon meshes. *The visual computer* 26(11): 1393–1406.

Avigal Y, Berscheid L, Asfour T, Kröger T and Goldberg K (2022) Speedfolding: Learning efficient bimanual folding of garments. In: *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 1–8.

Battaglia P, Pascanu R, Lai M, Jimenez Rezende D et al. (2016) Interaction networks for learning about objects, relations and physics. *Advances in Neural Information Processing Systems (NeurIPS)* 29.

Bernardini F, Mittleman J, Rushmeier H, Silva C and Taubin G (1999) The ball-pivoting algorithm for surface reconstruction. *IEEE Transactions on Visualization and Computer Graphics* 5(4): 349–359.

Chang P and Padır T (2020) Model-based manipulation of linear flexible objects with visual curvature feedback. In: *IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*. IEEE, pp. 1406–1412.

Cherubini A, Ortenzi V, Cosgun A, Lee R and Corke P (2020) Model-free vision-based shaping of deformable plastic materials. *The International Journal of Robotics Research* 39(14): 1739–1759.

Chi C, Burchfiel B, Cousineau E, Feng S and Song S (2022) Iterative Residual Policy for Goal-Conditioned Dynamic Manipulation of Deformable Objects. In: *Proceedings of Robotics: Science and Systems*. New York City, NY, USA. DOI:10.15607/RSS.2022.XVIII.016.

Cretu AM, Payeur P and Petriu EM (2011) Soft object deformation monitoring and learning for model-based robotic hand manipulation. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 42(3): 740–753.

Crouse DF (2016) On implementing 2d rectangular assignment algorithms. *IEEE Transactions on Aerospace and Electronic Systems* 52(4): 1679–1696.

Edelsbrunner H and Mücke EP (1994) Three-dimensional alpha shapes. *ACM Transactions on Graphics (TOG)* 13(1): 43–72.

Fan H, Su H and Guibas LJ (2017) A point set generation network for 3d object reconstruction from a single image. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. pp. 605–613.

Felippa CA (2004) Introduction to finite element methods. *University of Colorado* 885.

Figueroa N, Ureche ALP and Billard A (2016) Learning complex sequential tasks from demonstration: A pizza dough rolling case study. In: *2016 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. Ieee, pp. 611–612.

Fletcher R (2013) *Practical methods of optimization*. John Wiley & Sons.

Ganapathi A, Sundaresan P, Thananjeyan B, Balakrishna A, Seita D, Grannen J, Hwang M, Hoque R, Gonzalez JE, Jamali N et al. (2020) Learning to smooth and fold real fabric using dense object descriptors trained on synthetic color images. *arXiv preprint arXiv:2003.12698* .

Ha H and Song S (2022) Flingbot: The unreasonable effectiveness of dynamic manipulation for cloth unfolding. In: *Conference on Robot Learning (CoRL)*. PMLR, pp. 24–33.

Holl P, Thuerey N and Koltun V (2019) Learning to control pdes with differentiable physics. In: *International Conference on Learning Representations (ICLR)*.

Hoque R, Seita D, Balakrishna A, Ganapathi A, Tanwani AK, Jamali N, Yamane K, Iba S and Goldberg K (2022) Visuospatial foresight for physical sequential fabric manipulation. *Autonomous Robots* 46(1): 175–199.

Hu Y, Anderson L, Li TM, Sun Q, Carr N, Ragan-Kelley J and Durand F (2019) Difftaichi: Differentiable programming for physical simulation. In: *International Conference on Learning Representations (ICLR)*.

Huang Z, Hu Y, Du T, Zhou S, Su H, Tenenbaum JB and Gan C (2020) Plasticinelab: A soft-body manipulation benchmark with differentiable physics. In: *International Conference on Learning Representations (ICLR)*.

Huang Z, Lin X and Held D (2022) Mesh-based dynamics model with occlusion reasoning for cloth manipulation. In: *Robotics: Science and Systems (RSS)*.

Jatavallabhula KM, Macklin M, Golemo F, Voleti V, Petrini L, Weiss M, Considine B, Parent-Lévesque J, Xie K, Erleben K et al. (2021) gradsim: Differentiable simulation for system identification and visuomotor control. *arXiv preprint arXiv:2104.02646* .

Kazhdan M, Bolitho M and Hoppe H (2006) Poisson surface reconstruction. In: *Proceedings of the fourth Eurographics Symposium on Geometry Processing*, volume 7.

Kingma DP and Ba J (2015) Adam: A method for stochastic optimization. In: *International Conference on Learning Representation (ICLR)*.

Kurutach T, Tamar A, Yang G, Russell SJ and Abbeel P (2018) Learning plannable representations with causal infogan. *Advances in Neural Information Processing Systems (NeurIPS)* 31.

Landau LD, Lifshitz EM, Kosevich AM and Pitaevskii LP (1986) *Theory of elasticity: volume 7*, volume 7. Elsevier.

Le Lidec Q, Kalevatykh I, Laptev I, Schmid C and Carpentier J (2021) Differentiable simulation for physical system identification. *IEEE Robotics and Automation Letters* 6(2): 3413–3420.

Lee R, Hamaya M, Murooka T, Ijiri Y and Corke P (2021) Sample-efficient learning of deformable linear object manipulation in

the real world through self-supervision. *IEEE Robotics and Automation Letters (RA-L)* 7(1): 573–580.

Leggard CM, Schranz T, Schweiger G, Drgoňa J, Falay B, Gomes C, Iosifidis A, Abkar M and Larsen PG (2021) Constructing neural network-based models for simulating dynamical systems. *ACM Computing Surveys* 1(1).

Li S, Huang Z, Du T, Su H, Tenenbaum J and Gan C (2022a) Contact points discovery for soft-body manipulations with differentiable physics. In: *International Conference on Learning Representations (ICLR)*.

Li Y, Li S, Sitzmann V, Agrawal P and Torralba A (2022b) 3d neural scene representations for visuomotor control. In: *Conference on Robot Learning (CoRL)*. PMLR, pp. 112–123.

Li Y, Lin T, Yi K, Bear D, Yamins D, Wu J, Tenenbaum J and Torralba A (2020a) Visual grounding of learned physical models. In: *International Conference on Machine Learning (ICML)*. PMLR, pp. 5927–5936.

Li Y, Torralba A, Anandkumar A, Fox D and Garg A (2020b) Causal discovery in physical systems from videos. *Advances in Neural Information Processing Systems (NeurIPS)* 33: 9180–9192.

Li Y, Wu J, Tedrake R, Tenenbaum JB and Torralba A (2018) Learning particle dynamics for manipulating rigid bodies, deformable objects, and fluids. In: *International Conference on Learning Representations (ICLR)*.

Lin X, Huang Z, Li Y, Tenenbaum J, Held D and Gan C (2022a) Diffskill: Skill abstraction from differentiable physics for deformable object manipulations with tools. In: *International Conference on Learning Representations (ICLR)*.

Lin X, Qi C, Zhang Y, Huang Z, Fragkiadaki K, Li Y, Gan C and Held D (2022b) Planning with spatial-temporal abstraction from point clouds for deformable object manipulation. In: *6th Annual Conference on Robot Learning*.

Lin X, Wang Y, Huang Z and Held D (2022c) Learning visible connectivity dynamics for cloth smoothing. In: *Conference on Robot Learning (CoRL)*. PMLR, pp. 256–266.

Lin X, Wang Y, Olkin J and Held D (2021) Softgym: Benchmarking deep reinforcement learning for deformable object manipulation. In: *Conference on Robot Learning (CoRL)*. PMLR, pp. 432–448.

Luo Y, Xu H, Li Y, Tian Y, Darrell T and Ma T (2018) Algorithmic framework for model-based deep reinforcement learning with theoretical guarantees. In: *International Conference on Learning Representations (ICLR)*.

Manuelli L, Li Y, Florence P and Tedrake R (2021) Keypoints into the future: Self-supervised correspondence in model-based reinforcement learning. In: *Conference on Robot Learning (CoRL)*. PMLR, pp. 693–710.

Matas J, James S and Davison AJ (2018) Sim-to-real reinforcement learning for deformable object manipulation. In: *Conference on Robot Learning (CoRL)*. PMLR, pp. 734–743.

Matl C and Bajcsy R (2021) Deformable elasto-plastic object shaping using an elastic hand and model-based reinforcement learning. In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 3955–3962.

Miller S, Van Den Berg J, Fritz M, Darrell T, Goldberg K and Abbeel P (2012) A geometric approach to robotic laundry folding. *The International Journal of Robotics Research (IJRR)* 31(2): 249–267.

Monaghan JJ (1992) Smoothed particle hydrodynamics. *Annual Review of Astronomy and Astrophysics* 30(1): 543–574.

Mrowca D, Zhuang C, Wang E, Haber N, Fei-Fei LF, Tenenbaum J and Yamins DL (2018) Flexible neural representation for physics prediction. *Advances in Neural Information Processing Systems (NeurIPS)* 31.

Müller M, Heidelberger B, Hennix M and Ratcliff J (2007) Position based dynamics. *Journal of Visual Communication and Image Representation* 18(2): 109–118.

Nadon F, Valencia AJ and Payeur P (2018) Multi-modal sensing and robotic manipulation of non-rigid objects: A survey. *Robotics* 7(4): 74.

Nagabandi A, Konolige K, Levine S and Kumar V (2020) Deep dynamics models for learning dexterous manipulation. In: *Conference on Robot Learning (CoRL)*. PMLR, pp. 1101–1112.

Nair A, Chen D, Agrawal P, Isola P, Abbeel P, Malik J and Levine S (2017) Combining self-supervised learning and imitation for vision-based rope manipulation. In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 2146–2153.

Navarro-Alarcon D, Yip HM, Wang Z, Liu YH, Zhong F, Zhang T and Li P (2016) Automatic 3-d manipulation of soft objects by robotic arms with an adaptive deformation model. *IEEE Transactions on Robotics (T-RO)* 32(2): 429–441.

Reddy JN (2019) *Introduction to the finite element method*. McGraw-Hill Education.

Rubner Y, Tomasi C and Guibas LJ (1998) A metric for distributions with applications to image databases. In: *Sixth International Conference on Computer Vision (IEEE Cat. No. 98CH36271)*. IEEE, pp. 59–66.

Sánchez D, Wan W and Harada K (2020) Tethered tool manipulation planning with cable maneuvering. *IEEE Robotics and Automation Letters (RA-L)* 5(2): 2777–2784.

Sanchez J, Corrales JA, Bouzgarrou BC and Mezouar Y (2018) Robotic manipulation and sensing of deformable objects in domestic and industrial applications: a survey. *The International Journal of Robotics Research (IJRR)* 37(7): 688–716.

Sanchez-Gonzalez A, Godwin J, Pfaff T, Ying R, Leskovec J and Battaglia P (2020) Learning to simulate complex physics with graph networks. In: *International Conference on Machine Learning (ICML)*. PMLR, pp. 8459–8468.

Schoenholz SS, Cubuk ED and Jax M (2018) End-to-end differentiable, hardware accelerated, molecular dynamics in pure python. *arXiv preprint arXiv:1912.04232* .

Schrittwieser J, Antonoglou I, Hubert T, Simonyan K, Sifre L, Schmitt S, Guez A, Lockhart E, Hassabis D, Graepel T et al. (2020) Mastering atari, go, chess and shogi by planning with a learned model. *Nature* 588(7839): 604–609.

Seita D, Florence P, Tompson J, Coumans E, Sindhwani V, Goldberg K and Zeng A (2021) Learning to rearrange deformable cables, fabrics, and bags with goal-conditioned transporter networks. In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 4568–4575.

She Y, Wang S, Dong S, Sunil N, Rodriguez A and Adelson E (2021) Cable manipulation with a tactile-reactive gripper. *The International Journal of Robotics Research (IJRR)* 40(12-14): 1385–1401.

Shlomi J, Battaglia P and Vlimant JR (2020) Graph neural networks in particle physics. *Machine Learning: Science and Technology* 2(2): 021001.

Sulsky D, Zhou SJ and Schreyer HL (1995) Application of a particle-in-cell method to solid mechanics. *Computer Physics Communications* 87(1-2): 236–252.

Sundaresan P, Grannen J, Thananjeyan B, Balakrishna A, Laskey M, Stone K, Gonzalez JE and Goldberg K (2020) Learning rope manipulation policies using dense object descriptors trained on synthetic depth data. In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 9411–9418.

Thananjeyan B, Garg A, Krishnan S, Chen C, Miller L and Goldberg K (2017) Multilateral surgical pattern cutting in 2d orthotropic gauze with deep reinforcement learning policies for tensioning. In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 2371–2378.

Victoria AH and Maragatham G (2021) Automatic tuning of hyperparameters using bayesian optimization. *Evolving Systems* 12: 217–223.

Viswanath V, Shivakumar K, Kerr J, Thananjeyan B, Novoseller E, Ichnowski J, Escontrela A, Laskey M, Gonzalez J and Goldberg K (2022) Autonomously Untangling Long Cables. In: *Proceedings of Robotics: Science and Systems*. New York City, NY, USA. DOI:10.15607/RSS.2022.XVIII.034.

Wang A, Kurutach T, Liu K, Abbeel P and Tamar A (2019) Learning robotic manipulation through visual planning and acting. In: *Robotics: Science and Systems (RSS)*.

Wu Y, Yan W, Kurutach T, Pinto L and Abbeel P (2020) Learning to Manipulate Deformable Objects without Demonstrations. In: *Robotics: Science and Systems (RSS)*. DOI:10.15607/RSS.2020.XVI.065.

Xu Z, Chi C, Burchfiel B, Cousineau E, Feng S and Song S (2022) DextAIRity: Deformable Manipulation Can be a Breeze. In: *Proceedings of Robotics: Science and Systems*. New York City, NY, USA. DOI:10.15607/RSS.2022.XVIII.017.

Yan M, Zhu Y, Jin N and Bohg J (2020) Self-supervised learning of state estimation for manipulating deformable linear objects. *IEEE Robotics and Automation Letters (RA-L)* 5(2): 2372–2379.

Yan W, Vangipuram A, Abbeel P and Pinto L (2021) Learning predictive representations for deformable objects using contrastive estimation. In: *Conference on Robot Learning (CoRL)*. PMLR, pp. 564–574.

Yin H, Varava A and Kragic D (2021) Modeling, learning, perception, and control methods for deformable object manipulation. *Science Robotics* 6(54).

Yoshimoto K, Higashimori M, Tadakuma K and Kaneko M (2011) Active outline shaping of a rheological object based on plastic deformation distribution. In: *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, pp. 1386–1391.

Yuksel C (2015) Sample elimination for generating poisson disk sample sets. In: *Computer Graphics Forum*, volume 34. Wiley Online Library, pp. 25–32.

Zimmerman WB (2006) *Multiphysics modeling with finite element methods*, volume 18. World Scientific Publishing Company.