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

Haochen Shi*
Stanford University
hshi74@stanford.edu

Huazhe Xu*
Stanford University
huazhexu@stanford.edu

Zhiao Huang
UC San Diego
z2huang@eng.ucsd.edu

Yunzhu Li
MIT
liyunzhu@mit.edu

Jiajun Wu
Stanford University
jiajunwu@cs.stanford.edu

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 degree 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 sensing 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 robotic interaction data, our robot can learn a dynamics model that can be used to synthesize control signals to deform elasto-plastic objects into various 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 and ability to generalize to a more complex action space, different tool shapes, and a mixture of motion modes. We also conduct comparisons between RoboCraft and untrained human subjects controlling the gripper to manipulate deformable objects in both simulation and the real world. Our learned model-based planning framework is comparable to and sometimes better than human subjects on the tested tasks. [1]

I. INTRODUCTION

Effective manipulation of deformable objects is an essential skill for robots deployed in real-world industrial and household environments. However, due to deformable objects' high degrees of freedom (DoF) and consequent challenges in state estimation and dynamics modeling, manipulating deformable objects requires significant innovations beyond the typical robotic paradigm that focuses only on rigid objects. Recent advances show promising results in manipulating clothes [38, 36, 60, 15, 63, 17] and ropes [62, 57], yet the manipulation of objects with high plasticity, such as dough or plasticine, poses a unique set of challenges and is currently underexplored [37], despite the ubiquity of such objects in household and industrial settings. In this paper, we investigate how to empower robots to model and manipulate elasto-plastic objects based on raw RGBD visual observations.
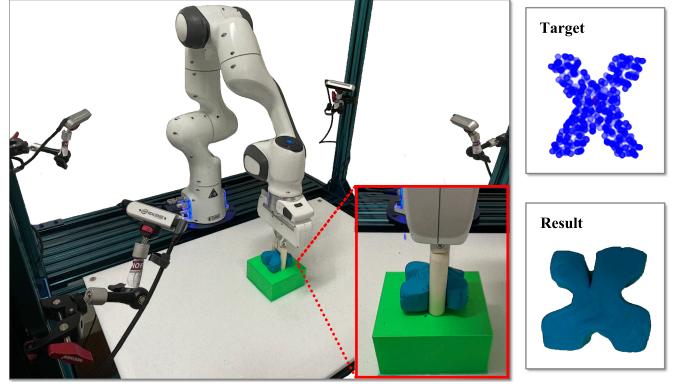
Fig. 1: **RoboCraft**. The robot uses a parallel 2-finger gripper to shape an 'X' conditioned on the target shape at the top right corner. The result is shown at the bottom right corner.

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 [52, 44, 35]. 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 [41, 27, 28, 50, 55]. They have shown great generalization results, demonstrating the benefits of explicit structured modeling. However, most of them 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, a model-based planning framework that represents elasto-plastic objects using particles, but employs distribution-based loss functions

and makes novel improvements over recently-developed GNNs to model the objects' dynamics. The learned dynamics model is then coupled with gradient-based trajectory optimization techniques to plan the robot's behaviors. The proposed approach closes the perception and control loops, which allows accurate modeling and manipulation of the elasto-plastic objects in both 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 models 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 model. Unlike prior learning-based particle dynamics works which assume temporal correspondence [41, 27, 28, 50, 55], 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 (heart, alphabetical letters, etc.) purely based on raw visual inputs. Notably, with just 10 minutes of real-world interaction training data, our learned model can make accurate predictions of the object's deformation under planned actions, modeling the interactions between the gripper and the object, as well as the elastic/plastic deformation within the object. We also asked amateur human subjects to control the robot to perform the same tasks, and our framework is on-par and sometimes more precise at achieving target shapes.

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

II. RELATED WORK

A. 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 [42], smoothed particle hydrodynamics [40], and material point methods [56] are widely used. Along with the advancement of particle-based methods, several differentiable simulators [18, 16, 51] that can propagate gradients through the model have been introduced to capture the dynamics of non-rigid bodies.

Data-driven approaches are another paradigm for learning physical dynamics [44, 34]. Recently, people have demonstrated inspiring results on learning the dynamics of deformable objects such as clothes [33], ropes [6], and fluid [25], with various representations including low-dimensional parameterized shapes [37], keypoints [29], latent vectors [23],

and neural radiance fields [30]. Inspired by prior approaches [27, 50, 55, 4], 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, from which the one-to-one temporal correspondences among particles are hard to obtain.

B. Manipulating Deformable Objects

Deformable object manipulation is a long-standing challenge in robotics. However, many existing methods focus on objects such as ropes [59, 45, 61, 57, 24, 3, 61], cables [54, 48, 53], clothes [31, 38, 36, 60, 15, 2, 1], and gauze [58]. By contrast, we investigate the less-explored manipulation of elasto-plastic objects, such as plasticine, which are only studied in limited previous works [49, 27].
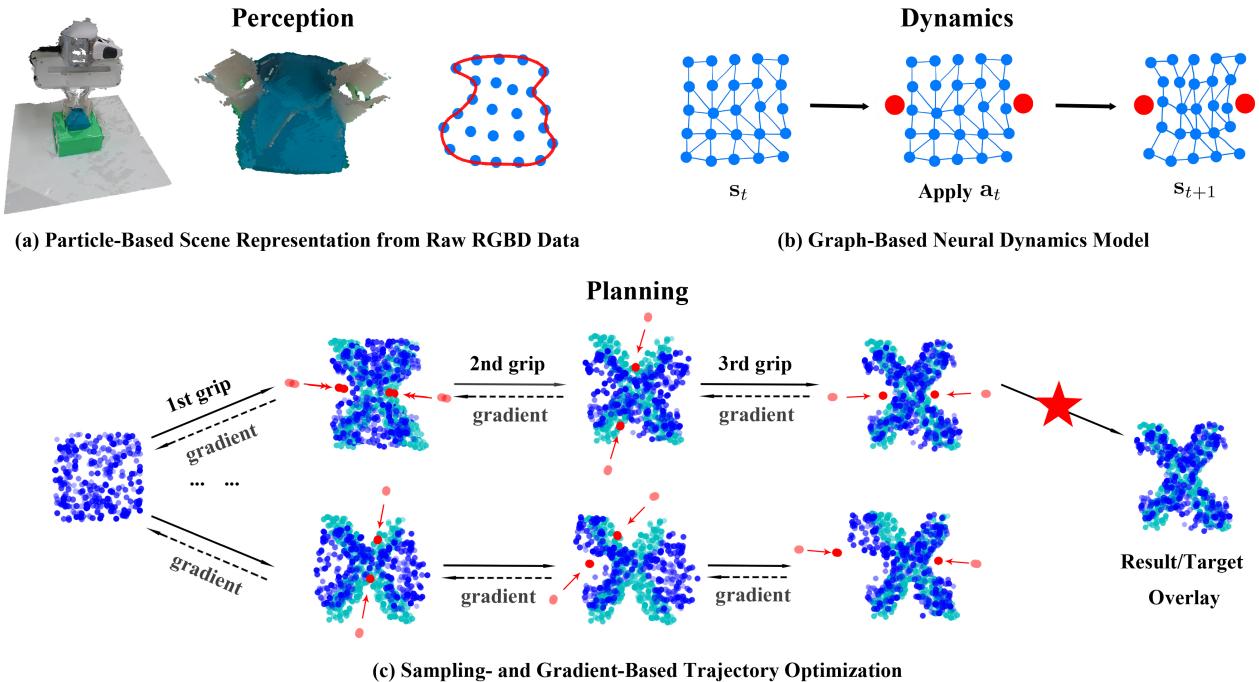
As for manipulation of deformable elasto-plastic objects (dough, plasticine, etc), prior works propose to use either model-based [8, 37, 32, 26] or adaptive methods [46, 7, 64]. As mentioned earlier, it is possible to leverage simulations as models for actual manipulation [14]. However, the challenges in state and parameter estimation make it prohibitive to accurately simulate such objects and thus hinder the transfer of policies to the real world. Another popular approach is to learn from expert demonstrations [43], which has been proved effective in shaping sand [7] and pizza dough [11]. However, obtaining demonstrations is usually expensive. Matl and Bajcsy [37] 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. By contrast, RoboCraft equips the model with a particle representation and an expressive GNN, which enable our robot to shape objects to more complex unseen shapes such as hearts or alphabetical letters. Our work is unique in the sense that we shape dough-like objects into complex shapes that contain semantic meanings.
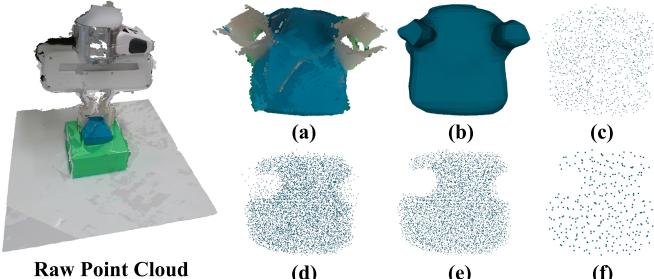
III. METHOD

A. Problem Statement

The objective of this work is to use a parallel 2-finger robot 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 $\mathbf{a}_t \in \mathcal{A}$ upon the plasticine, and the state of the plasticine transitions 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 $\Phi : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{S}$. This dynamics model takes as input environment observations $\mathbf{s}_{t-h,...,t} \in \mathcal{S}$ and actions $\mathbf{a}_{t-h,...,t} \in \mathcal{A}$ and predicts 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

Perception

(a) Particle-Based Scene Representation from Raw RGBD Data

Dynamics

$\mathbf{s}_t$ Apply $\mathbf{a}_t$ $\mathbf{s}_{t+1}$

(b) Graph-Based Neural Dynamics Model

Planning

1st grip   gradient   ... ...   gradient

2nd grip   gradient

3rd grip   gradient

gradient   gradient

Result/Target Overlay

(c) Sampling- and Gradient-Based Trajectory Optimization

Fig. 2: **Overview of RoboCraft.** (a) The perception module obtains the particle representation from RGBD cameras. The algorithm first crops out the object point cloud, then samples particles to represent the object. (b) The dynamics model predicts the object's deformation based on graph neural networks (GNN). (c) After obtaining the learned dynamics model, we apply a combination of sampling- and gradient-based trajectory optimization techniques to solve the model-predictive planning problem.

Raw Point Cloud

(a) (b) (c)
(d) (e) (f)

Fig. 3: **The particle sampling procedure.** Left: Visualized point cloud for the manipulation scene. Right: (a) Extracted point cloud for the object and the gripper. (b) Reconstructed mesh surface. (c) Sampled particles within the mesh. (d) Refinement with physics prior. (e) Refinement with shape prior. (f) GNN-ready downsampled particles.

the MPC problem measures the distance between the state of the plasticine at the last time step T and the target shape $\mathbf{g}$. The choice of cost function $\mathcal{J}$ will be discussed in section III-D. And a sequence of actions of length T can be selected by minimizing the cost function:

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

Figure 2 shows the overall framework of RoboCraft.

B. Particle Sampling from Raw RGBD Data

We aim to sample particles from the raw visual data (i.e., RGBD images) for the GNN-based dynamics model as shown in Figure 2(a). To obtain useful particles, we need to sample particles that can represent the shape of the object from heavily occluded raw inputs. We illustrate how we obtain the required particles through preprocessing, surface reconstruction, refinement with physics prior, refinement with shape prior, and postprocessing as shown in Figure 3.

1) Preprocessing: We first convert RGBD images into point clouds, using the intrinsic and extrinsic parameters of the cameras. We then use RANSAC [12] to detect the plane with the largest support in the raw point cloud (i.e., the platform on which we place the object) and remove the inliers of this plane. We also remove other noisy backgrounds based on the location of the object. The outcome is shown in Figure 3(a).

2) Surface Reconstruction: Two popular surface reconstruction algorithms are alpha shapes [9] and ball pivoting [5]. However, these algorithms are not robust to occlusion. We instead use Poisson surface reconstruction [21], 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 is shown in Figure 3(b).

3) Refinement with Physics Prior: We exploit the knowledge that when the gripper is not touching the plastic object, the shape of the object should largely remain unchanged.

Hence, after sampling the particles as in Figure 3(c), missing particles that are not reconstructed from the previous step (e.g., an artifact caused by shadow) are patched using points from the previous frame. The outcome is shown in Figure 3(d).

4) Refinement with Shape Prior: In the process of gripping, the object is occluded by itself as well as the gripper. The heaviest occlusion occurs when the gripper is pinching the object. We first perform a simple color-based filter to extract the object point cloud. However, the point cloud of the plasticine is incomplete due to occlusion, and thus reconstructing a watertight mesh from it becomes difficult. We propose leveraging shape prior to refine the point cloud. Specifically, after obtaining a smooth and watertight mesh from the full point cloud (object and gripper), we then sample particles inside the reconstructed mesh based on its signed distance function (SDF). We finally use the fingers' SDFs to remove the points inside them. The outcome is shown in Figure 3(e).

5) Post-processing: To further improve the quality of the sampled point cloud, we apply voxel down-sampling to get a uniform point distribution and remove the points that are statistical outliers. We then use the farthest point sampling method [39] to further down-sample the point cloud of the plasticine into a reasonable number of points for the GNN. The final output is shown in Figure 3(f).

C. Learning Object Dynamics via Graph Networks

We now introduce how RoboCraft constructs a particle graph and uses graph neural networks to model the dynamics of the system, as shown in Figure 2(b).

1) Graph Building: 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, $\mathbf{o}_{i,t} = \langle \mathbf{x}_{i,t}, \mathbf{c}_{i,t}^o \rangle$, where $\mathbf{x}_{i,t}$ is the position of particle i at time t, and $\mathbf{c}_{i,t}^o$ denotes the corresponding attributes (e.g., mass, radius). The edges $\mathcal{E}_t$ between the vertices are computed dynamically over time from their spatial relationship. We connect all the neighbors within a predefined distance. For relations between particles (e.g., a pair of particles with an edge), we use $\mathbf{e}_k = \langle u_k, v_k, \mathbf{c}_k^e \rangle$, 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).

2) Model Training: The goal of the GNN is to infer the system dynamics and predict the future from the current graph as $\hat{\mathbf{s}}_{t+1} = \Phi(\mathbf{s}_t, \mathbf{a}_t)$. 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 instantaneous propagation of forces.

D. 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.

Earth mover's distance (EMD). Consider $\mathcal{O}_1, \mathcal{O}_2 \subseteq \mathbb{R}^3$. The EMD between them is defined as

$$\mathcal{L}_{\text{EMD}}(\mathcal{O}_1, \mathcal{O}_2) = \min_{\mu:\mathcal{O}_1 \to \mathcal{O}_2} \sum_{\mathbf{x} \in \mathcal{O}_1} \|\mathbf{x} - \mu(\mathbf{x})\|_2,$$

where $\mu : \mathcal{O}_1 \to \mathcal{O}_2$ is a bijection [47]. 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. Intuitively, in this work, EMD matches distributions while preventing outliers in the point cloud by the bijection definition.

Chamfer distance (CD). The CD between $\mathcal{O}_1, \mathcal{O}_2 \subseteq \mathbb{R}^3$ is

$$\mathcal{L}_{\text{CD}}(\mathcal{O}_1, \mathcal{O}_2) = \sum_{\mathbf{x} \in \mathcal{O}_1} \min_{\mathbf{y} \in \mathcal{O}_2} \|\mathbf{x} - \mathbf{y}\|_2^2 + \sum_{\mathbf{y} \in \mathcal{O}_2} \min_{\mathbf{x} \in \mathcal{O}_1} \|\mathbf{x} - \mathbf{y}\|_2^2.$$

In the strict sense, we slightly abuse the term Chamfer distance because it does not satisfy the triangle inequality [10]. Intuitively, 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: $\mathcal{L}(\mathcal{O}_1, \mathcal{O}_2) = w_1 \mathcal{L}_{\text{EMD}}(\mathcal{O}_1, \mathcal{O}_2) + w_2 \mathcal{L}_{\text{CD}}(\mathcal{O}_1, \mathcal{O}_2)$. Based on experimental results, we find the optimal hyper-parameters are $w_1 = 0.9, w_2 = 0.1$.

E. Model Predictive Control

In this section, we introduce a new model predictive control (MPC) algorithm for controlling the actuators to manipulate the plasticine as shown in Figure 2(c).

1) Action Space: We simplify the action space of each grip into a parameterized space: $\{x, y, z, r_z, d\}$. In this simplified space, $\{x, y, z\}$ are bounded locations indicating the center of the grip, i.e. the midpoint of the line segment connecting the centers of mass of the gripper's two fingers. r_z is the robot gripper's rotation about the z axis (i.e., the vertical axis). The rotations about the x and y axes are always 0. d represents the minimal distance between gripper fingers during this pinch. The robot gripper pinches the plasticine at a constant speed.

2) 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 2(c). We
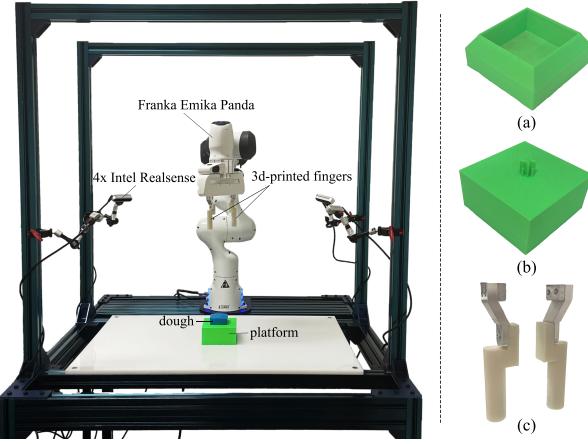
Fig. 4: **Robot Setup.** Left: overview of the robot setup; Right: 3D-printed tools including (a) the mold for resetting, (b) the supporting platform, and (c) the parallel 2-finger gripper.

first do random shooting in the simplified action space and compute the costs using the GNN-based dynamics model. We then apply the limited-memory BFGS [13] algorithm on the lowest-cost trajectories to optimize the actions with gradients, where the loss is the same as our dynamics model's training loss.

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.

While each video in the training dataset includes three grips, we may require more or fewer grips when aiming at unseen shapes. To address this issue, we perform MPC with a fixed look-ahead horizon and execute one grip at a time. The robot performs additional grips until the loss is small enough or the maximum number of grips is reached.

IV. EXPERIMENTAL SETUP

A. Physical Setup

In Figure 4, we show the general setup in the simulator and real world. For real-world manipulation, we use a Franka Emika Panda robot arm with 7 DoF and Franka's parallel jaw gripper. We substitute the original grippers with a pair of 3D-printed cylindrical parallel gripper fingers (Figure 4(c)). Four RealSense D415 RGBD cameras are fixed at four locations surrounding the plasticine to capture the RGBD images at 30Hz and 848×480 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. The initial shape of the plasticine is roughly a 6cm×6cm×2.5cm cuboid. In the coordinate frame where the robot base is at the origin, the initial position of the center of mass of the plasticine is at (43.00cm, 0.00cm, 7.25cm) on a 3D-printed platform, with a central protruding plastic rod to fix the object (Figure 4(b)). At the beginning of each episode, we stuff the plasticine into a 3D printed mold in the shape of a cuboid cavity (Figure 4(a)) to reset the shape of the object. We will release STL files for all the 3D-printed parts.

B. Data Collection

We collect 6,000 frames (10 mins) of training data, including 50 episodes with a horizon of 120 frames. For each episode, three grips are applied to the plasticine. At the beginning of each episode, we manually shape the plasticine into a cuboid with a mold and plastic wrap. We knead the manipulated object before molding it to avoid wrinkles on the plasticine. The data collection behavior policy randomly selects parameters in the action space consisting of rotation angles, translations, and gripper fingertip distances. During each episode, we save the generated partial point clouds from each of the RGBD cameras and the robot joint poses from the robot controller. The data is only saved when the gripper is on the same plane as the object to optimize memory usage.

C. Tasks and Benchmarks

The main challenges in deformable object manipulation are 1) the high DoF of the object, unlike the other deformable objects, and 2) a mixture of rigid and non-rigid motion modes. To highlight these challenges, we propose a series of tasks of shaping fixed and movable elasto-plastic objects, which require rich local operations in addition to global pose matching. With the final goal of real-world manipulation in mind, we first set up experiments in simulation [19] with two capsule-shaped fingers and cuboid-shaped plasticine. The following is a brief overview of the tasks:

Shaping w/ Rigid Motion: Grip a freely movable deformable object into the target shape. Gripper rotation and translation are allowed only in the plane parallel to the tabletop.
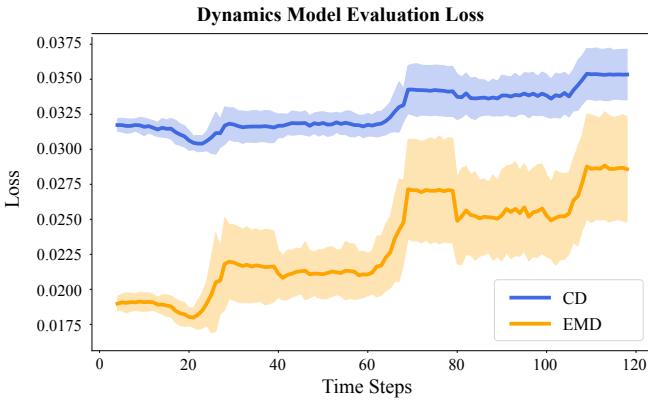
Shaping w/ 7-DoF gripper: Grip a fixed deformable object into the target shape. There is no constraint of the parallel plane for the gripper - it can rotate freely in the 3D space.

Tool Selection: Grip a fixed deformable object. For each grip, the agent can choose between two tools of different sizes.
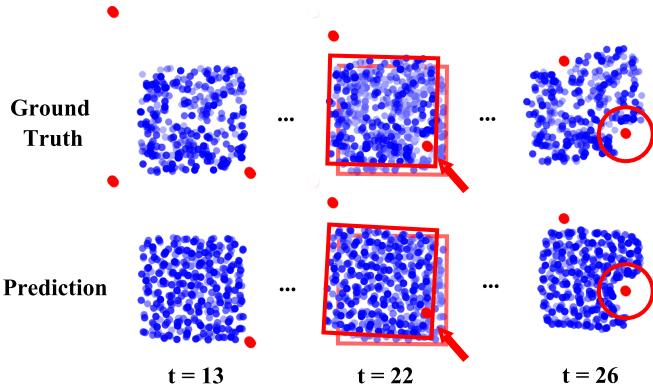
Along with these tasks, we propose a benchmark for deformable object shaping in both simulation and the real world. We use 26 alphabetical letters and five other shapes as our target shapes in the simulator. For the real robot, we use seven letters and five other shapes. Note that for letters with holes such as 'B' or 'P', we only focus on the contour because the current tools are not suitable for carving out the hollow portions. We believe that the proposed shapes span a large spectrum of possible shapes and can serve as a benchmark for future research works.

D. Evaluation Metrics

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

Fig. 5: **Quantitative results on dynamics prediction.** The figure shows the CD and EMD between the predicted particles and the ground truth particles over time. The shaded area is one standard deviation.

Fig. 6: **Rigid motion and non-rigid motion over time frames.** The first row is the ground truth and the second row is the predicted motion from the dynamics model.

metrics. In addition, we use Hausdorff distance (HD) [20] to measure whether every point of either set is close to a point of the other set.

V. EXPERIMENTAL RESULTS

In this section, we evaluate the proposed method for various complex deformable elasto-plastic object manipulation tasks. We compare the proposed method with a set of baseline algorithms and untrained human subjects. We also carefully investigate various design decisions to determine the important factors for future researchers' reference. Lastly, we apply the lessons learned from simulation experiments to enable a robot to manipulate elasto-plastic objects in the real world.
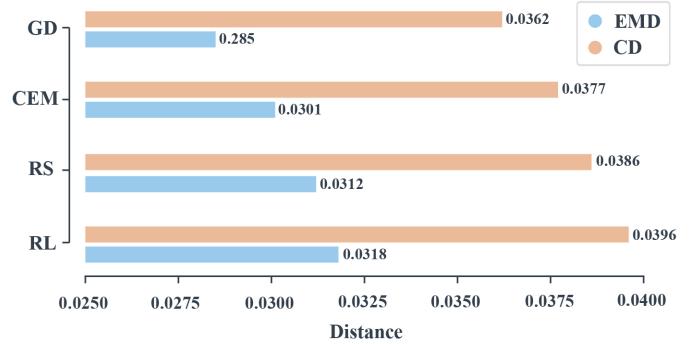
A. Results in Simulation

1) 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 incomplete point cloud and extracts points from

TABLE I: Sampling results averaged over 120 frames in Chamfer distance (CD) and Earth mover's distance (EMD).

Methods	CD↓	EMD↓
Patch-Based	0.0384 ± 0.003	0.0317 ± 0.005
Crop-Based	$\mathbf{0.0374} \pm 0.001$	$\mathbf{0.0308} \pm 0.002$

TABLE II: The mean and standard deviation of the dynamics model's performance over 12,000 frames when trained with different loss functions.

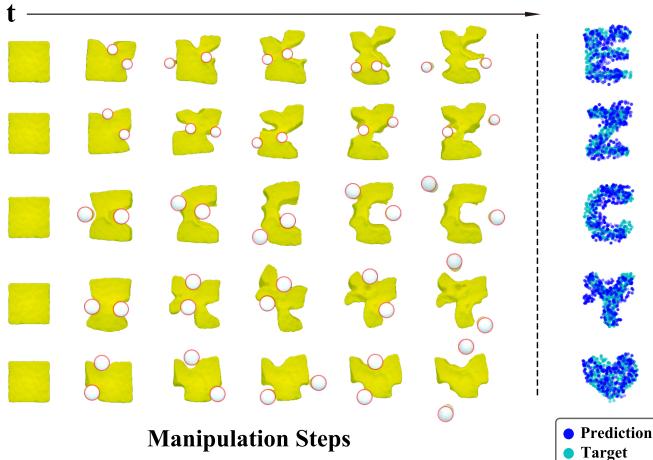| Loss Types	CD↓	EMD↓	HD ↓
CD Loss	0.0344 ± 0.003	0.0267 ± 0.006	0.0851 ± 0.018
EMD Loss	0.0328 ± 0.002	$\mathbf{0.0230} \pm 0.004$	0.1068 ± 0.011
Mixed Loss	$\mathbf{0.0327} \pm 0.002$	0.0231 ± 0.004	$\mathbf{0.0790} \pm 0.016$

Fig. 7: **Comparison of different trajectory optimization methods.** We benchmarked 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.

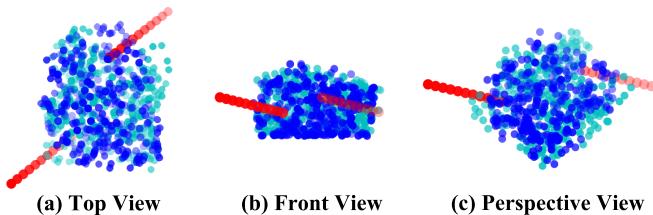the gripper point cloud within the reconstructed convex hull to patch the point cloud of plasticine.

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

2) Dynamics Model Learning: We train the GNN model as explained in Section III-C. Specifically, we first construct the graph by connecting edges between two vertices with the proximity threshold $d = 0.05$. For each edge and vertex, we encode them with 3-layer MLPs with hidden and output layers each of size 150. The propagator consists of one fully connected layer with an output layer size of 150. The motion predictor is another 3-layer MLP 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 [22], a batch size of 4, and a learning rate of 1e-4.

We then evaluate GNN-based dynamics models trained with different loss functions. In Table II, we find that although using

Fig. 8: **Manipulation result in the simulation.** On the left are the manipulation steps. On the right are the result and its overlay with the target point cloud. The cyan point cloud is the target, blue the result.

Fig. 9: **The three views of result and target point cloud overlay with a free-moving gripper.** The blue dots are the point cloud of the plasticine, the cyan ones are the target point cloud, and the red dots represent the two fingers of the gripper.

TABLE III: Results for tool selection averaged over all the target shapes.

Methods	CD↓	EMD↓
w/ Tool Selection	**0.0408** ± 0.005	**0.0337** ± 0.007
w/o Tool Selection	0.0409 ± 0.005	0.0345 ± 0.007

Fig. 10: **Dynamics prediction of the plasticine in the real world.** The first row shows the sampled particles from the actual visual observation. The second row is the open-loop prediction of the particle distributions using our learned dynamics from just 10 minutes of real-world interaction data. Red dots represent the gripper's location.
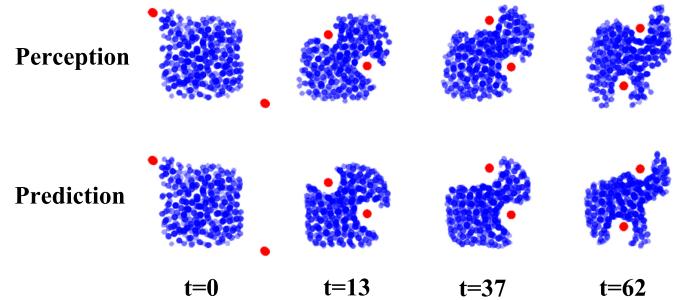
CD, EMD, or a combination of CD and EMD has similar performance in terms of CD and EMD losses, the Hausdorff distance is significantly improved with the combined loss. In Figure 5, we plot the CD and the EMD against the time horizon. We find that although the distance inevitably becomes larger as the model predicts farther into the future, it remains in a reasonable range. This result confirms that it is possible to use these models for manipulation. In Figure 6, we visualize the predicted particles when rigid motions exist in the sequence. We find that our model is able to predict particles under both rigid motion and non-rigid motion.

3) Manipulation Results: We also evaluate the manipulation results of RoboCraft through ablation studies.

Trajectory optimization methods. 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, for the reinforcement learning baseline, we use the same action space as in other planning methods. The state-space consists of the position of all the particles and also the gripper. The reward function is computed from the difference in CD or EMD after each grip. For training, we use a discount factor of 0.99 and a learning rate of 0.001 with the Adam optimizer. 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 1e6 and the target smooth coefficient is 0.005. As shown in Figure 7, 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-time BFGS* optimizer provides the best results in practice.

In Figure 8, we visualize the procedure of manipulating the object towards the target shape using a gradient-based method. We find that 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.

Complex action space. We evaluate our method when all DoFs are actuated for the robot gripper, namely *Shaping with 7 DOF gripper*. In Figure 9, we show the top view, front view, and perspective view of the overlaid result and the target point cloud. We find that the proposed method can learn to rotate the gripper in order to achieve the target shape. This experiment shows the potential of the proposed method to scale to more complex action spaces.

Tool selection. We evaluate our method when two grippers with different sizes can be selected for each grip. In Table III, we find that when more tool choices are provided, the performance can be further improved.

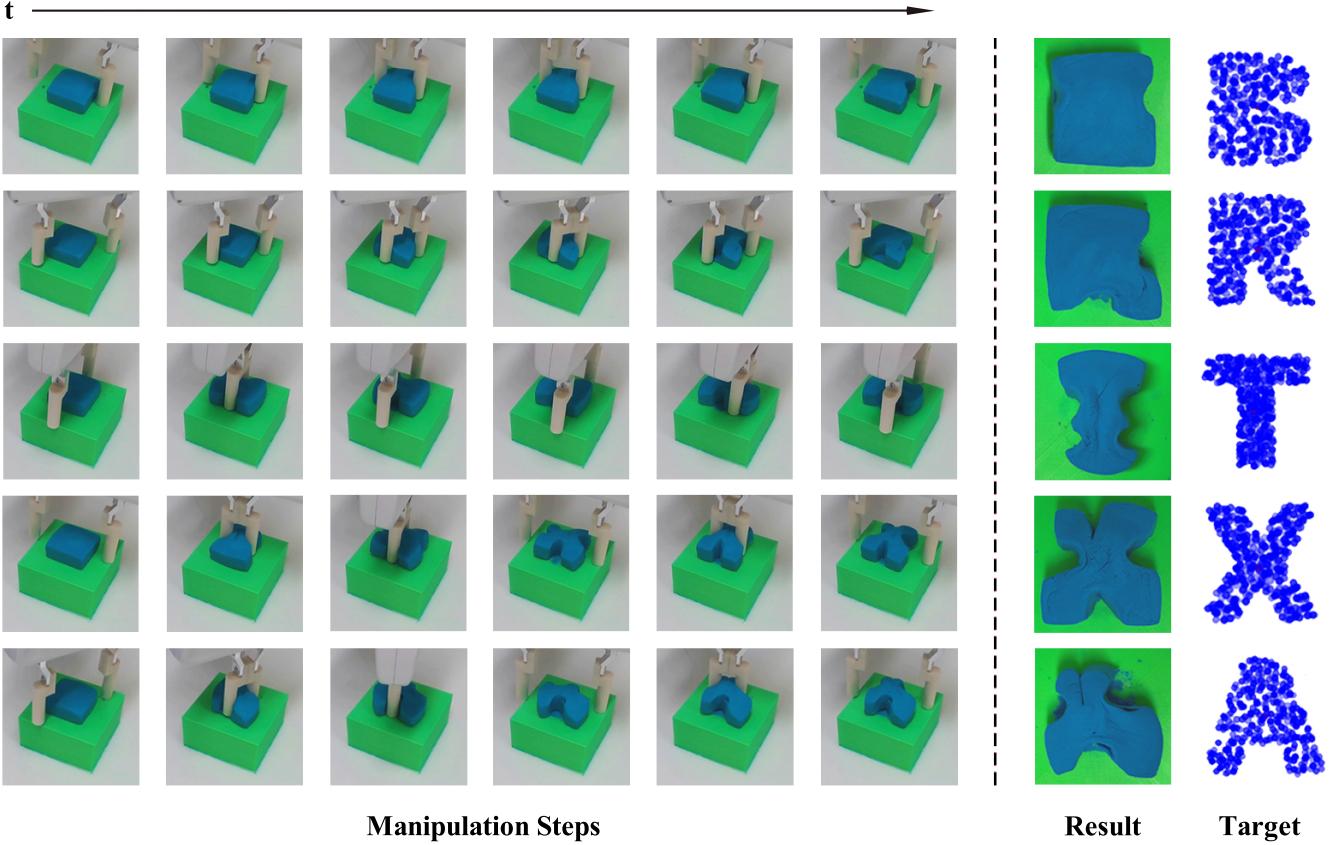**Manipulation Steps**   **Result**   **Target**

Fig. 11: **Manipulation results on a real robot.** On the left are the shaping steps. On the right are the results and corresponding target point clouds. We want to emphasize that 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.

B. Learning Real-World Manipulation of Deformable Objects

We now present an evaluation of our method on a real robot. We first evaluate the quality of the dynamics model with only 10 minutes of real-world data. In Figure 10, we visualize the comparison between the output from the trained model and the sampled data. Our method is able to model the dynamics even with limited training data. We then show that the proposed method is able to manipulate the plasticine to shapes that are unseen in the training data. Example trajectories of the robot manipulating the plasticine are shown in Figure 11. The method successfully identifies the asymmetry in the target shape 'B' by putting the finger closer to one side of the plasticine at the beginning of the grip. For more complex shapes such as the letter 'A', the method also seems to creatively discover a solution that roughly achieves the target shape. These results illustrate that, although the task is very challenging, our method is able to perform well with a small amount of training data.
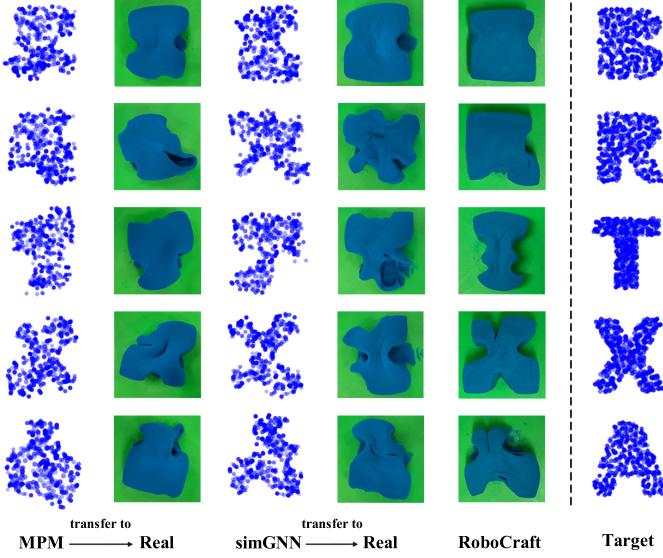
C. Comparing with Other Dynamics Models in the Real World

RoboCraft uses a GNN-based dynamics model learned from real-world data. We compare its efficacy with two other widely adopted options: 1) a phy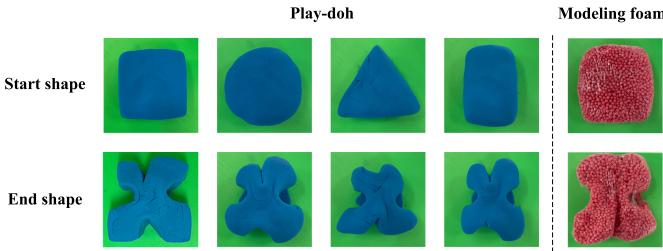sics simulator based on Material Point Method (MPM) with manually selected parameters and 2) a GNN learned in the simulator with simulation data. We follow previous work [19] to create the MPM dynamics model. As shown in Figure 12, RoboCraft outperforms both baselines. We attribute this to the fact that RoboCraft learns from real-world data while other models suffer from the large domain gap between simulation and reality.

D. 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 13, 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. In the right side of Figure 13, we test whether the dynamics model of RoboCraft which is trained on plasticine can be used to manipulate modeling foam. We find that it can also roughly make an 'X' shape without retraining. The results illustrate the potential of RoboCraft to generalize to various unseen scenarios.

Fig. 12: **RoboCraft outperforms methods that use Material Point Method (MPM) or GNN-based model trained in simulation (simGNN) as the dynamics model.** The first two columns show the outcomes of transferring control signals from MPM models into the real world. The third and fourth columns show the outcomes of transferring the simGNN model into the real world. We list the outcomes of RoboCraft as well as the target shapes for comparison.
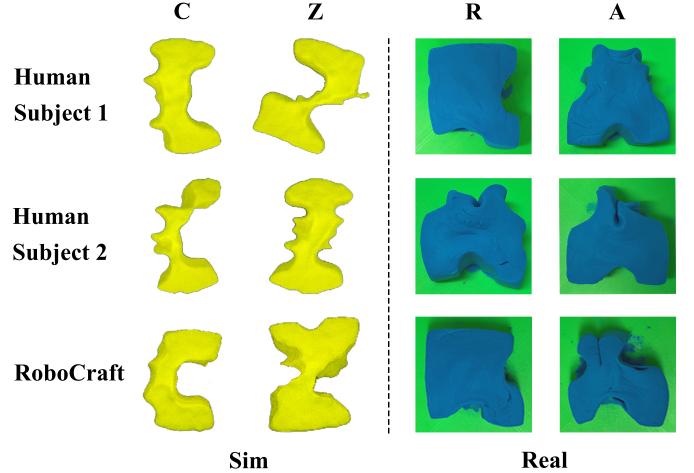

Fig. 13: **RoboCraft generalizes to different initial shapes and material types.** Left: RoboCraft is applied on 3 different initial shapes including circle, triangle, and rectangle. Right: RoboCraft is applied on modeling foam.

E. Results on Comparing with Human Performance

We invited four amateur humans to perform the same task with the robot gripper in both simulation and real-world settings. While humans are not trained to manipulate plasticine, they usually have strong intuitive understandings of the dynamics of plasticine. Each user was asked to shape three pieces of plasticine in each domain. For the simulation experiments, we provided a successful manipulation video as an example for the users and allowed two trials for each shape since the dynamics in the simulation were also new to each human. The human subjects controlled the gripper using the keyboard to shape the plasticine to 'C', 'E', 'Y', 'Z', and heart. In the real world, we allowed the users to play with the plasticine for one minute before resetting it to the starting

TABLE IV: Results of human subjects and the robot in the simulator. Numbers are averaged over all the tested shapes.

Methods	CD↓	EMD↓
Human Subjects	0.0655 ± 0.025	0.0661 ± 0.023
RoboCraft (ours)	$\mathbf{0.0359} \pm 0.007$	$\mathbf{0.0340} \pm 0.005$

Fig. 14: **Shaping results by amateur humans and RoboCraft.** Results in the first two rows are from human subjects. The results in the third row are from the robot. The left two columns are results in the simulation. The right two columns are the results in the real world.

shape for the experiment. After that, human users controlled the robot arm and gripper manually in the guiding mode to shape the plasticine to 'A', 'B', 'R', 'T', and 'X'. In Table IV, we show the average distance over all the target shapes from four users, in comparison with that of our agent. Empirical evidence suggests that the proposed tasks are challenging for both manipulation algorithms and untrained human subjects alike. We also find that RoboCraft is comparable to or stronger than amateur humans on the tested tasks. One observation is that RoboCraft outperforms humans in the distance metrics. However, the visualized human results are recognizable even when the distances are high, suggesting that better evaluation metrics are desired. In Figure 14, we show the outcome from both human users and the robot for comparison.

VI. 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, with the control algorithm outperforming amateur human users. Furthermore, we explore the possibility of applying the proposed method to harder tasks with complex action spaces, mixtures of motion modes, and multiple tools.

References

[1] Rika Antonova, Peiyang Shi, Hang Yin, Zehang Weng, and Danica Kragic Jensfelt. Dynamic environments with deformable objects. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, 2021.

[2] Rika Antonova, Anastasia Varava, Peiyang Shi, J Frederico Carvalho, and Danica Kragic. Sequential topological representations for predictive models of deformable objects. In *Learning for Dynamics and Control*, pages 348–360. PMLR, 2021.

[3] Rika Antonova, Jingyun Yang, Priya Sundaresan, Dieter Fox, Fabio Ramos, and Jeannette Bohg. A bayesian treatment of real-to-sim for deformable object manipulation. *IEEE Robotics and Automation Letters*, 7(3):5819–5826, 2022.

[4] Peter Battaglia, Razvan Pascanu, Matthew Lai, Danilo Jimenez Rezende, et al. Interaction networks for learning about objects, relations and physics. *Advances in Neural Information Processing Systems (NeurIPS)*, 29, 2016.

[5] Fausto Bernardini, Joshua Mittleman, Holly Rushmeier, Claudio Silva, and Gabriel Taubin. The ball-pivoting algorithm for surface reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, 5(4):349–359, 1999.

[6] Peng Chang and Taşkın Padır. Model-based manipulation of linear flexible objects with visual curvature feedback. In *IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, pages 1406–1412. IEEE, 2020.

[7] Andrea Cherubini, Valerio Ortenzi, Akansel Cosgun, Robert Lee, and Peter Corke. Model-free vision-based shaping of deformable plastic materials. *The International Journal of Robotics Research*, 39(14):1739–1759, 2020.

[8] Ana-Maria Cretu, Pierre Payeur, and Emil M Petriu. 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, 2011.

[9] Herbert Edelsbrunner and Ernst P Mücke. Three-dimensional alpha shapes. *ACM Transactions on Graphics (TOG)*, 13(1):43–72, 1994.

[10] Haoqiang Fan, Hao Su, and Leonidas J Guibas. 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)*, pages 605–613, 2017.

[11] Nadia Figueroa, Ana Lucia Pais Ureche, and Aude Billard. Learning complex sequential tasks from demonstration: A pizza dough rolling case study. In *2016 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 611–612. Ieee, 2016.

[12] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[13] Roger Fletcher. *Practical methods of optimization*. John Wiley & Sons, 2013.

[14] Aditya Ganapathi, Priya Sundaresan, Brijen Thananjeyan, Ashwin Balakrishna, Daniel Seita, Jennifer Grannen, Minho Hwang, Ryan Hoque, Joseph E Gonzalez, Nawid Jamali, et al. Learning to smooth and fold real fabric using dense object descriptors trained on synthetic color images. *arXiv preprint arXiv:2003.12698*, 2020.

[15] Huy Ha and Shuran Song. Flingbot: The unreasonable effectiveness of dynamic manipulation for cloth unfolding. In *Conference on Robot Learning (CoRL)*, pages 24–33. PMLR, 2022.

[16] Philipp Holl, Nils Thuerey, and Vladlen Koltun. Learning to control pdes with differentiable physics. In *International Conference on Learning Representations (ICLR)*, 2019.

[17] Ryan Hoque, Daniel Seita, Ashwin Balakrishna, Aditya Ganapathi, Ajay Kumar Tanwani, Nawid Jamali, Katsu Yamane, Soshi Iba, and Ken Goldberg. Visuospatial foresight for physical sequential fabric manipulation. *Autonomous Robots*, 46(1):175–199, 2022.

[18] Yuanming Hu, Luke Anderson, Tzu-Mao Li, Qi Sun, Nathan Carr, Jonathan Ragan-Kelley, and Fredo Durand. Difftaichi: Differentiable programming for physical simulation. In *International Conference on Learning Representations (ICLR)*, 2019.

[19] Zhiao Huang, Yuanming Hu, Tao Du, Siyuan Zhou, Hao Su, Joshua B Tenenbaum, and Chuang Gan. Plasticinelab: A soft-body manipulation benchmark with differentiable physics. In *International Conference on Learning Representations (ICLR)*, 2020.

[20] Daniel P Huttenlocher, Gregory A. Klanderman, and William J Rucklidge. Comparing images using the hausdorff distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 15(9):850–863, 1993.

[21] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics Symposium on Geometry Processing*, volume 7, 2006.

[22] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representation (ICLR)*, 2015.

[23] Thanard Kurutach, Aviv Tamar, Ge Yang, Stuart J Russell, and Pieter Abbeel. Learning plannable representa-

tions with causal infogan. *Advances in Neural Information Processing Systems (NeurIPS)*, 31, 2018.

[24] Robert Lee, Masashi Hamaya, Takayuki Murooka, Yoshihisa Ijiri, and Peter Corke. 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, 2021.

[25] Christian Møldrup Leggard, Thomas Schranz, Gerald Schweiger, Ján Drgoňa, Basak Falay, Cláudio Gomes, Alexandros Iosifidis, Mahdi Abkar, and Peter Gorm Larsen. Constructing neural network-based models for simulating dynamical systems. *ACM Computing Surveys*, 1(1), 2021.

[26] Sizhe Li, Zhiao Huang, Tao Du, Hao Su, Joshua Tenenbaum, and Chuang Gan. Contact points discovery for soft-body manipulations with differentiable physics. In *International Conference on Learning Representations (ICLR)*, 2022.

[27] Yunzhu Li, Jiajun Wu, Russ Tedrake, Joshua B Tenenbaum, and Antonio Torralba. Learning particle dynamics for manipulating rigid bodies, deformable objects, and fluids. In *International Conference on Learning Representations (ICLR)*, 2018.

[28] Yunzhu Li, Toru Lin, Kexin Yi, Daniel Bear, Daniel Yamins, Jiajun Wu, Joshua Tenenbaum, and Antonio Torralba. Visual grounding of learned physical models. In *International Conference on Machine Learning (ICML)*, pages 5927–5936. PMLR, 2020.

[29] Yunzhu Li, Antonio Torralba, Anima Anandkumar, Dieter Fox, and Animesh Garg. Causal discovery in physical systems from videos. *Advances in Neural Information Processing Systems (NeurIPS)*, 33:9180–9192, 2020.

[30] Yunzhu Li, Shuang Li, Vincent Sitzmann, Pulkit Agrawal, and Antonio Torralba. 3d neural scene representations for visuomotor control. In *Conference on Robot Learning (CoRL)*, pages 112–123. PMLR, 2022.

[31] Xingyu Lin, Yufei Wang, Jake Olkin, and David Held. Softgym: Benchmarking deep reinforcement learning for deformable object manipulation. In *Conference on Robot Learning (CoRL)*, pages 432–448. PMLR, 2021.

[32] Xingyu Lin, Zhiao Huang, Yunzhu Li, Joshua Tenenbaum, David Held, and Chuang Gan. Diffskill: Skill abstraction from differentiable physics for deformable object manipulations with tools. In *International Conference on Learning Representations (ICLR)*, 2022.

[33] Xingyu Lin, Yufei Wang, Zixuan Huang, and David Held. Learning visible connectivity dynamics for cloth smoothing. In *Conference on Robot Learning (CoRL)*, pages 256–266. PMLR, 2022.

[34] Yuping Luo, Huazhe Xu, Yuanzhi Li, Yuandong Tian, Trevor Darrell, and Tengyu Ma. Algorithmic framework for model-based deep reinforcement learning with theoretical guarantees. In *International Conference on Learning Representations (ICLR)*, 2018.

[35] Lucas Manuelli, Yunzhu Li, Pete Florence, and Russ Tedrake. Keypoints into the future: Self-supervised correspondence in model-based reinforcement learning. In *Conference on Robot Learning (CoRL)*, pages 693–710. PMLR, 2021.

[36] Jan Matas, Stephen James, and Andrew J Davison. Sim-to-real reinforcement learning for deformable object manipulation. In *Conference on Robot Learning (CoRL)*, pages 734–743. PMLR, 2018.

[37] Carolyn Matl and Ruzena Bajcsy. Deformable elastoplastic object shaping using an elastic hand and model-based reinforcement learning. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3955–3962. IEEE, 2021.

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

[39] Carsten Moenning and Neil A Dodgson. Fast marching farthest point sampling. Technical report, University of Cambridge, Computer Laboratory, 2003.

[40] Joe J Monaghan. Smoothed particle hydrodynamics. *Annual Review of Astronomy and Astrophysics*, 30(1):543–574, 1992.

[41] Damian Mrowca, Chengxu Zhuang, Elias Wang, Nick Haber, Li F Fei-Fei, Josh Tenenbaum, and Daniel L Yamins. Flexible neural representation for physics prediction. *Advances in Neural Information Processing Systems (NeurIPS)*, 31, 2018.

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

[43] Félix Nadon, Angel J Valencia, and Pierre Payeur. Multi-modal sensing and robotic manipulation of non-rigid objects: A survey. *Robotics*, 7(4):74, 2018.

[44] Anusha Nagabandi, Kurt Konolige, Sergey Levine, and Vikash Kumar. Deep dynamics models for learning dexterous manipulation. In *Conference on Robot Learning (CoRL)*, pages 1101–1112. PMLR, 2020.

[45] Ashvin Nair, Dian Chen, Pulkit Agrawal, Phillip Isola, Pieter Abbeel, Jitendra Malik, and Sergey Levine. Combining self-supervised learning and imitation for vision-based rope manipulation. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2146–2153. IEEE, 2017.

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

[47] Yossi Rubner, Carlo Tomasi, and Leonidas J Guibas. A metric for distributions with applications to image databases. In *Sixth International Conference on Computer Vision (IEEE Cat. No. 98CH36271)*, pages 59–66. IEEE, 1998.

[48] Daniel Sánchez, Weiwei Wan, and Kensuke Harada.

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

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

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

[51] Samuel S Schoenholz, Ekin D Cubuk, and MD Jax. End-to-end differentiable, hardware accelerated, molecular dynamics in pure python. *arXiv preprint arXiv:1912.04232*, 2018.

[52] Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020.

[53] Daniel Seita, Pete Florence, Jonathan Tompson, Erwin Coumans, Vikas Sindhwani, Ken Goldberg, and Andy Zeng. Learning to rearrange deformable cables, fabrics, and bags with goal-conditioned transporter networks. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4568–4575. IEEE, 2021.

[54] Yu She, Shaoxiong Wang, Siyuan Dong, Neha Sunil, Alberto Rodriguez, and Edward Adelson. Cable manipulation with a tactile-reactive gripper. *The International Journal of Robotics Research (IJRR)*, 40(12-14):1385–1401, 2021.

[55] Jonathan Shlomi, Peter Battaglia, and Jean-Roch Vlimant. Graph neural networks in particle physics. *Machine Learning: Science and Technology*, 2(2):021001, 2020.

[56] Deborah Sulsky, Shi-Jian Zhou, and Howard L Schreyer. Application of a particle-in-cell method to solid mechanics. *Computer Physics Communications*, 87(1-2):236–252, 1995.

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

[58] Brijen Thananjeyan, Animesh Garg, Sanjay Krishnan, Carolyn Chen, Lauren Miller, and Ken Goldberg. 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)*, pages 2371–2378. IEEE, 2017.

[59] Angelina Wang, Thanard Kurutach, Kara Liu, Pieter Abbeel, and Aviv Tamar. Learning robotic manipulation through visual planning and acting. In *Robotics: Science and Systems (RSS)*, 2019.

[60] Yilin Wu, Wilson Yan, Thanard Kurutach, Lerrel Pinto, and Pieter Abbeel. Learning to Manipulate Deformable Objects without Demonstrations. In *Robotics: Science and Systems (RSS)*, July 2020. doi: 10.15607/RSS.2020. XVI.065.

[61] Mengyuan Yan, Yilin Zhu, Ning Jin, and Jeannette Bohg. Self-supervised learning of state estimation for manipulating deformable linear objects. *IEEE Robotics and Automation Letters (RA-L)*, 5(2):2372–2379, 2020.

[62] Wilson Yan, Ashwin Vangipuram, Pieter Abbeel, and Lerrel Pinto. Learning predictive representations for deformable objects using contrastive estimation. In *Conference on Robot Learning (CoRL)*, pages 564–574. PMLR, 2021.

[63] Hang Yin, Anastasia Varava, and Danica Kragic. Modeling, learning, perception, and control methods for deformable object manipulation. *Science Robotics*, 6(54), 2021.

[64] Kayo Yoshimoto, Mitsuru Higashimori, Kenjiro Tadakuma, and Makoto Kaneko. Active outline shaping of a rheological object based on plastic deformation distribution. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1386–1391. IEEE, 2011.