CONTACT POINTS DISCOVERY FOR SOFT-BODY MANIPULATIONS WITH DIFFERENTIABLE PHYSICS

Anonymous authors
Paper under double-blind review

ABSTRACT

Differentiable physics has recently been shown as a powerful tool for solving soft-body manipulation tasks. However, the differentiable physics solver often gets stuck when the initial contact points of the end effectors are sub-optimal or when performing multi-stage tasks that require contact point switching, which often leads to local minima. To address this challenge, we propose a contact point discovery approach (CPDeform) that guides the stand-alone differentiable physics solver to deform various soft-body plasticines. The key idea of our approach is to integrate optimal transport-based contact points discovery into the differentiable physics solver to overcome the local minima from initial contact points or contact switching. On single-stage tasks, our method can automatically find suitable initial contact points based on transport priorities. On complex multi-stage tasks, we can iteratively switch the contact points of end-effectors based on transport priorities. To evaluate the effectiveness of our method, we introduce PlasticineLab-M that extends the existing differentiable physics benchmark PlasticineLab to seven new challenging multi-stage soft-body manipulation tasks. Extensive experimental results suggest that: 1) on multi-stage tasks that are infeasible for the vanilla differentiable physics solver, our approach discovers contact points that efficiently guide the solver to completion; 2) on tasks where the vanilla solver performs sub-optimally or near-optimally, our contact point discovery method performs better than or on par with the manipulation performance obtained with handcrafted contact points. Demos are available at our project page[1].

1 INTRODUCTION

Soft body manipulation has a wide application in cooking (Bollini et al., 2013), fabric manipulation (Wu et al., 2020), healthcare (Mayer et al., 2008) and manufacturing of deformable objects (Sanchez et al., 2018). Differentiable physics has recently been shown as a powerful and effective tool for solving control problems for soft-body manipulation tasks. As demonstrated in Huang et al. (2021), given a parameterized manipulation policy, the differentiable physics solver computes the gradients of the policy parameters, enabling gradient-based optimization much more efficiently than reinforcement learning algorithms at finding optimal solutions for soft-body manipulation tasks on a diverse collection of environments.

However, the performance of the stand-alone gradient-based solver can be heavily influenced by the policy initialization. Especially, the end effectors' initial **contact points** with objects play critical roles in the optimization. Different contact points may lead to vast differences in manipulation performance due to local optima. Besides, some tasks require agents to switch contact points during the manipulation, where the local optima issue becomes a serious bottleneck for completing these multi-stage tasks. For example, as shown in Figure 2, an agent needs to control the capsule "pen" to sculpt two scribbles on the surface of a yellow plasticine cube. In order to complete the second line, the agent needs to switch contact points after drawing the first one. While the stand-alone differentiable physics solver could possibly draw the first line, it often gets stuck and struggles to draw the second one, due to the lack of gradients that push the pen to a new contact point to begin the second line. How to automatically find proper contact points for soft body manipulation tasks remains a challenge in differentiable physics.

[1] https://sites.google.com/view/cpdeformiclr2022

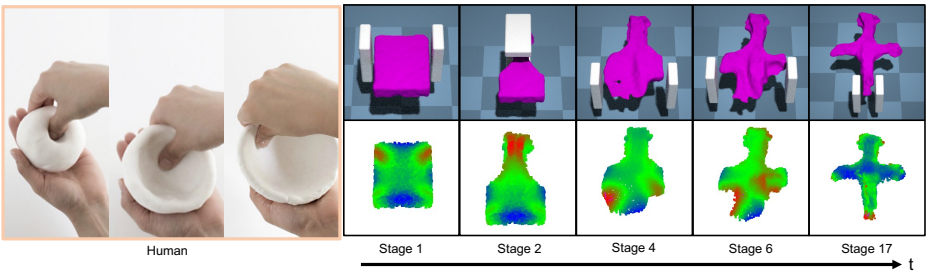


Figure 1: **Left**: the process that a human applies to transform a plasticine dough into a bowl. **Right**: we use images captured in time sequence to demonstrate how our framework reshapes a plasticine cube into a target airplane. The bottom row indicates the transport priorities found by our framework by computing the optimal transport between the current and target shapes. The top row illustrates the pose of the end-effectors selected by our framework based on the transport priorities.

In this paper, we propose a principled framework that integrates an optimal transport-based contact discovery method into differentiable physics (CPDeform) to address this important challenge. CPDeform heuristically find contact points for end effectors by using transport priorities computed from optimal transport to compare the current shape with the target shape, where soft-body manipulation is treated as a particle transportation problem. After finding contact points, CPDeform can combine the differentiable physics solver to solve soft body manipulation tasks. On single-stage tasks that do not require contact point switching, CPDeform can find suitable initial contact points to finish the task.

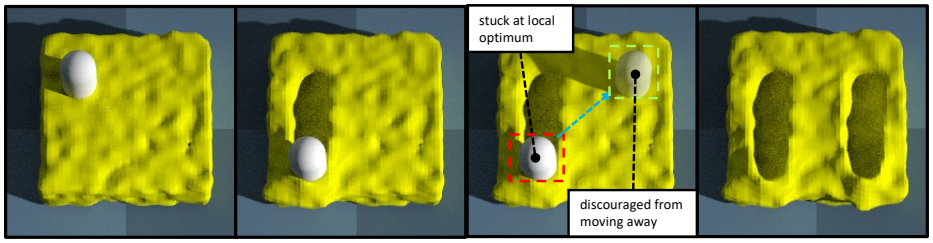
On multi-stage tasks, using an example shown in Figure 1 (right) where the goal is to reshape a plasticine cube into an airplane, CPDeform can iteratively switch the contact points of end effectors based on transport priorities. This iterative deformation process is motivated by how humans manipulate plasticine. As shown in Figure 1 (left), when humans manipulate a plasticine dough, they tend to repeatedly focus on the point of interest and modify it towards the target shape. CPDeform can mimic this process by iteratively switching contact points of interests based on transport priorities and deforming the soft bodies into the target shape with the help of the differentiable solver. By integrating contact points discovery into the differentiable physics solver, CPDeform can skip over the local minima caused by contact switching and improve the performance of the stand-alone solver.

To evaluate the effectiveness of CPDeform, we introduce PlasticineLab-M that extends the existing differentiable physics benchmark PlasticineLab (Huang et al., 2021) to seven new challenging multi-stage soft body tasks. Extensive experimental results suggest that: on single-stage tasks where the vanilla differentiable physics solver performs sub-optimally or near-optimally in PlasticineLab, we find that the backbone of CPDeform, a contact point discovery method based on optimal transport, single-handedly performs better than or on par with the manipulation performance obtained with random-selected or human-defined contact points. On multi-stage tasks that are infeasible for the vanilla gradient-based solver, we find that CPDeform performs reasonably well in practice and the iterative deformation method equipped with contact point discovery could serve as an alternative to the expensive long-horizon searching algorithm. In summary, our work makes the following contributions:

- We perform an in-depth study of local optimum issues of differentiable physics solver for initial contact points and switching contact points.
- We propose a principled framework, CPDeform, that integrates optimal transport-based contact discovery into differentiable physics.
- We find that the backbone of CPDeform, the contact point discovery method, can be directly employed by the stand-alone solver to find better initial contact points for single-stage tasks.
- On multi-stage tasks, which are infeasible for the vanilla solver, CPDeform employs a heuristic searching approach to iteratively complete the tasks.

2 Motivation

In this section, we provide an intuitive analysis of the drawback of the differentiable physics solver through motivating toy examples. We start with a brief review of how the differentiable physics solver could be employed to optimize manipulation policies. We then demonstrate how initial contact



(a) Initial Shape (b) Diff. physics optimization (c) reaching local optimum (d) Target Shape

Figure 2: (a) **Writer** to manipulate a "pen"; (b) given an initial handcrafted position of the "pen", the differentiable physics solver can control the "pen" to minimize the loss; (c) the solver gets stuck and fails to finish the second curve, due to the lack of gradient to push the "pen" towards the second line. (d) shows the target shape.

points affect the optimization performance. Finally, we take a simple but representative multi-stage task as an example and discuss why contact switching would often lead to local minima.

We study a **Writer** task as shown in Figure 2(a). In this task, an agent needs to manipulate the capsule "pen" to initiate contact with the yellow plasticine cube, and sculpt a line scribble on the plasticine surface. The agent can move the tip of the pen along three dimensions.

To solve this task with differentiable physics, we manually initialize the end-effector "pen" near the suitable contact point that allows the "pen" to initiate contact with the plasticine. We then parameterize the desired motion trajectory of the pen as a sequence of three-dimensional actions $\{a_1, \ldots, a_T\}$ where T is the number of simulation steps. Let $\{s_t\}_{0 \leq t \leq T}$ be simulation states at different time steps which include the state of the plasticine and manipulator. The differentiable simulator starts from the initial state s_0 and completes the action sequences by repeatedly executing the forward function $s_{t+1} = \phi(s_t, a_{t+1})$ until the ending state s_T has been reached. The objective of this optimizer is to minimize the distance between the current shape and the target shape. We represent the objective as a loss function $\mathcal{L}(s_T, g)$ where g is the target shape. Since the simulation forward function ϕ is fully-differentiable, we can compute gradients $\partial\mathcal{L}/\partial a_t$ of the loss $\mathcal{L}$ with respect to action a_t, and run multiple gradient descent steps to optimize the action sequences by $a_t = a_t - \alpha\partial\mathcal{L}/\partial a_t$ where α is the learning rate. As shown in Figure 2(b), we can see that the agent succeeds at sculpting the target scribble by moving the "pen" downwards. We refer the readers to Algorithm 2 in Appendix D for more details on differentiable physics for controller optimization.

However, if we do not initialize the positions of the end effectors well, the solver would get stuck in the local minima. For example, in the task as shown in Figure 3, we illustrate the optimization outcomes for end effectors with different contact points by showing their corresponding resulting shapes. Even with an arbitrarily large number of steps T given, the gradient-based solver is unable to discover a policy that moves away from the local optimum to the a new contact point that allows for task completion. Such phenomena are commonly observed across soft body manipulation tasks using the differentiable physics solver (Huang et al., 2021). When end effectors are far away from the region of interest, it is unlikely that the gradient could push the end-effectors towards the desired region. This observation poses the question of how to efficiently find optimal contact points to place the end effectors.

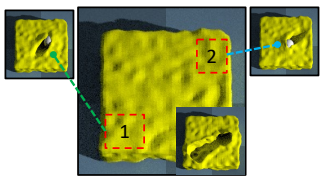


Figure 3: Writer-v1 in PlasticineLab. The bottom-right figure shows the target shape.

The local minima problem caused by inappropriate contact points becomes a more serious issue in multi-stage tasks. Taking the multi-stage writer task illustrated in Figure 2(c) as an example, the agent now needs to write an additional straight line on the plasticine surface by switching its contact point. Even with a well-initialized contact point for the first line, the solver is unable to relocate to the new region of interest for the upcoming line. We observe that, differing from the vanilla differentiable physics solver, humans tend to employ an explicit "iterative deformation" schema to complete such a task. Humans would decompose this task into two stages. In each stage, we tend to iteratively derive

the correspondence between the current shape and target shape to arrive at useful contact points from observations and then subsequently move the "pen" and write the lines. This motivates us to combine contact point discovery with iterative deformation.

3 METHOD

In this section, we introduce CPDeform, a principled framework that integrates optimal transport-based contact discovery into differentiable physics for solving challenging soft-body manipulation tasks. We first describe our contact point discovery method in relation to the transport priorities found by optimal transport in Section 3.1. In Section 3.2, we describe how transport priorities can be used to place end-effectors. Finally, in Section 3.3, we show how CPDeform integrates contact points discovery with differentiable physics and iteratively deforms the soft bodies for multi-stage tasks.

3.1 OPTIMAL TRANSPORT AND CONTACT POINT DISCOVERY

One way to consider soft-body manipulation is by treating it as a particle transportation problem. By evaluating the cost of transporting the current state particles μ to the target state particles ν, optimal transport provides a useful framework for comparing differences between any given pair of shapes, which can guide us to discover contact points. Let all particles be weighted equally in our simulator. Given a cost matrix M, optimal transport finds a transportation plan P in transportation polytope U by minimizing the transportation cost $\min_{P \in U} \langle P, M \rangle$. Casting the problem into dual form, we have $\text{OT}(\alpha, \beta) := E_\mu[f] + E_\nu[g]$ such that $\forall i, j$, Lagrange multipliers f_i, g_j satisfy $f_i + g_j \leq M_{ij}$, where α, β are the mass vectors that for the particles in μ, ν respectively. We refer the reader to Appendix B for more details on optimal transport. We focus on the Lagrange multipliers f of the source particles, which we refer to as the dual potentials. Since it represents the supports of the source measure, we interpret f as the *transport priorities* for the source particles μ.

Transport priorities are helpful for selecting contact points. Given a pair of current and target soft-body shapes, we intuitively would place the end-effectors around the region of the largest difference between the two shapes, in order to substantially modify the shapes. This observation leads us to place the end-effectors at contact points whose corresponding optimal manipulation policies can minimize the difference between the current and target shapes. However, given the computation budget, it is not easy to directly evaluate the optimality of the contact point by enumerating through a set of contact points. Thus, we propose to use optimal transport priorities to heuristically identify contact points, based on a simple rule of selecting contact points with high transport priorities. We observe that contact points with high transport priorities mostly correspond with superior optimization performances.

3.2 END EFFECTOR PLACEMENT WITH TRANSPORT PRIORITIES

In this section, we describe how we place the manipulator at advantageous locations based on optimal transport priorities. We first describe the single manipulator case, and it can be extended to multiple manipulator environments by adding a heuristic.

Direct Single Manipulator Placement. The transport priorities can be employed to directly place the single manipulator. We identify the source particle μ^* corresponding to the largest potential f^*. We then find a suitable contact point around μ^* through grid search, where we create a 3D grid of size $I \times I \times I$ centered at μ^* with each dimension evenly spaced into N_i intervals. The criterion of the grid search is as follows: We reject points whose placements of the manipulator lead to collisions with the particles. And we iterate over each grid point $x \in \mathbb{R}^3$, and place the manipulator at the point with the maximal score according to the following criterion: $\text{score}(x) = \frac{1}{N_p} \sum_{i=1}^{N_p} \frac{f_i}{d(\mu_i, x)^2 + 1}$ where $d(\mu_i, x)$ computes the signed distance between the particle μ_i to the closest point on the manipulator placed at grid point x. As shown in Figure 4 (top), the criterion leads us to discover contact points that allow the manipulator to cover high potential particles while reducing subsequent changes to the low potential particles.

Multiple Manipulator Placement With Heuristic. For multiple manipulator environments, we need to consider a heuristically defined candidate pose set $\mathcal{T}$, where each pose corresponds with the desired manipulation strategy. Taking Figure 4 (bottom) as an example, it is more advantageous to

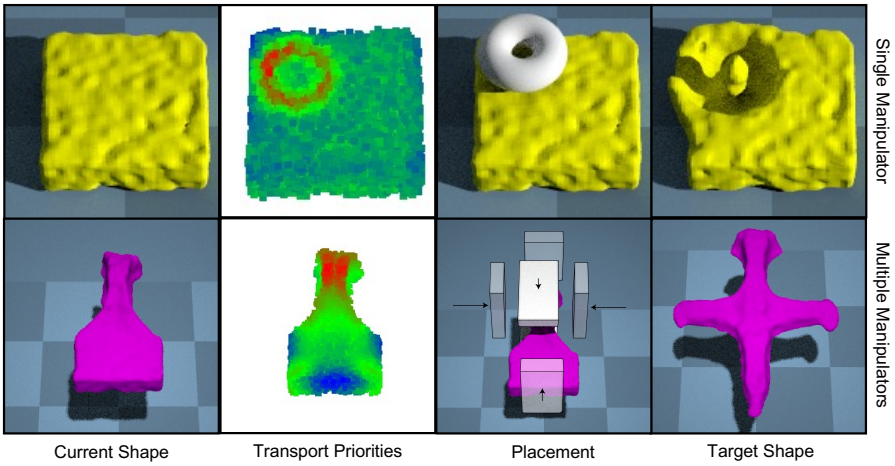


Figure 4: We demonstrate the visualizations of the placement strategies by our method in single and multiple manipulator cases respectively. We show the the current and target soft-body shapes, accompanied with the transport priorities computed by CPDeform that led to the placement of the manipulators. **Top**: Single-Manipulator Torus environment from PlasticineLab **Bottom**: Multiple-Manipulator Airplane environment from PlasticineLab-M.

use the highlighted pose than the ones drawn with low opacity. Each pose can be specified by the rotations of the manipulators as well as the direction vectors used to place the other manipulators with respect to the first manipulator. For each pose in set $\mathcal{T}$, we employ the single-manipulator placement strategy for the first manipulator. Using the first placement position as a starting point, for the remaining manipulators, we search along their pose-specific directions to find placement positions with minimal distances from the first manipulator that do not result in collisions with soft bodies. Since the differentiable physics solver is able to effectively adjust manipulator poses during optimization, in practice, the candidate pose set does not need to be exhaustively large. In practice, we use three poses to cover left-right, top-bottom, and front-back grasping poses.

3.3 Iterative Deformation for Multi-stage Soft Body Manipulation

Continuing with the focus on multi-stage soft-body manipulation tasks, we now describe how CPDeform combines the contact point discovery method with the use of differentiable physics for these tasks. For every stage, we iteratively specify the contact points and perform stage-level manipulation with the differentiable physics solver. For the multiple manipulator case, we search over contact plans corresponding to different poses, and choose the plan that achieves the lowest loss.

We present the detailed Algorithm 1 in Appendix C. We start with the initial shape S_0. Our goal is to reach the target shape G within a given number of stage n_{stage}. Each stage contains n_{step} simulation steps. As described in Section 3.2, we consider a heuristically defined candidate pose set $\mathcal{T}$. In each stage, we first compute the optimal transport between the current shape S_i and the target shape G to derive the transport priorities. Next, we select the particle p with the largest transport priority. For each pose, we search for the placement in relation to p that maximizes the aforementioned criterion.

4 Experiments

We conduct multiple experiments to test the efficacy of CPDeform on soft-body manipulation tasks to address the following questions:

- On multi-stage tasks that involve multiple contact switches, can CPDeform complete these tasks by iteratively manipulating the soft-bodies?
- How robust is the backbone of CPDeform, the contact point discovery method, when tested on single-stage tasks where it is only allowed to specify a single contact point in one shot?
- How is transport priority compared with other contact discovery methods?

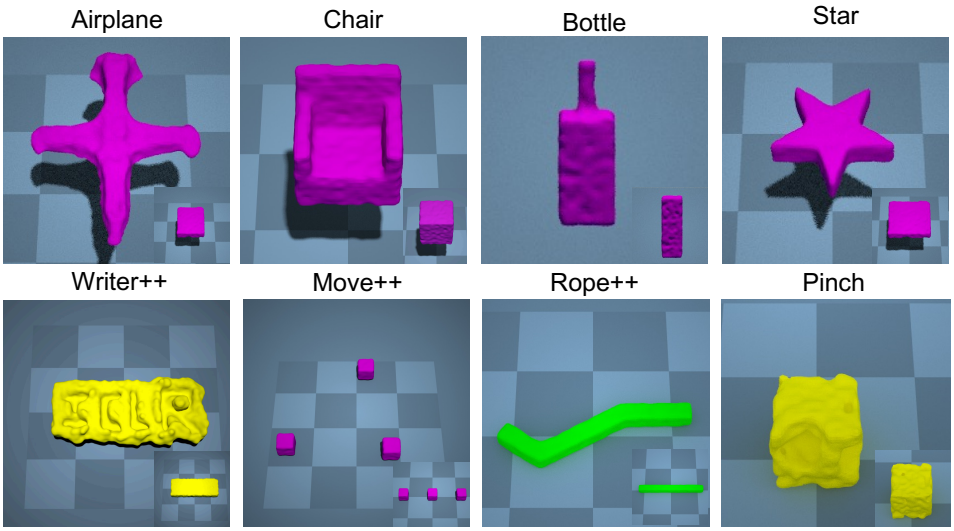


Figure 5: Task illustrations of PlasticineLab-M with target and initial shapes

4.1 DATASETS

To extensively evaluate our method, we introduce PlasticineLab-M, a dataset that extends the existing differentiable physics benchmark PlasticineLab with seven new challenging multi-stage soft-body manipulation tasks, and contains the multi-stage environment **Pinch** in PlasticineLab.

We show these eight multi-stage tasks in Figure 5. We refer the reader to Appendix A for detailed task descriptions. We also use the remaining eight tasks in PlasticineLab to evaluate our contact point discovery method for single stage tasks.

For multi-stage environments, we use the Wasserstein-1 distance approximated by Sinkhorn iteration (Cuturi, 2013) between the source and target particles to quantify the fine-grained difference between a state and the goal for all environments. As for single-stage environments from PlasticineLab, we evaluate our approach using the IoU metric for a fair comparison with PlasticineLab.

4.2 EVALUATION OF CPDEFORM ON MULTI-STAGE TASKS

We compare our approach with the following baselines: **PlasticineLab**: The vanilla gradient-based solver does not come with any contact point discovery features. It corresponds with the stand-alone differentiable physics solver described in Huang et al. (2021). **Reinforcement Learning (RL)**: We evaluate the performance of the existing RL algorithms on our tasks. We use three model-free reinforcement learning algorithms: Soft Actor-Critic (SAC) (Haarnoja et al., 2017), Policy Proximal Optimization (PPO) (Schulman et al., 2017), and TD3 (Fujimoto et al., 2018). For each stage, we optimize for 200 episodes for differentiable physics-based approaches with a learning rate 0.1. For each environment, we modestly choose a horizon of 10 or 20. We restrict the number of environment steps used for optimization under 1 million. We train each RL algorithm on each environment for 1000 episodes, with 1000 environment steps per episode, which accounts for the 1 million environment-step limit. Our reward function is of the form $\mathcal{R} = -\mathcal{R}_{shape} - \mathcal{R}_{grasp}$, where $\mathcal{R}_{shape}$ is the Wasserstein-1 distance between the source and target particles for measuring shape differences. And $\mathcal{R}_{grasp}$ encourages the manipulators to be closer to the soft bodies. For fair comparison with Pinch from PlasticineLab, we use the reward formulation described in Huang et al. (2021).

We show the quantitative results in Table 3 and the qualitative results in Figure 6. We find that our approach is capable of finishing these complex tasks, and significantly outperforms the baselines. We find that with the discovered contact points, our approach is able to iteratively build and refine the nose, tail, and wings of the **Airplane**. In **Chair**, we find that our approach guides the solver to first create the general seat, then refine the arm rest and back of the chair. In **Bottle**, our approach first pushes down the top of the plasticine cube to create the neck, before refining the sides of the bottle. For **Move++**, our approach is able to complete the transportation tasks of the three cubes by selecting the most advantageous object to transfer at each stage. In **Rope++**, our approach first moves the rope to form the general shape, before refining the ends of the rope. In **Writer++**, our approach is capable

Env	Airplane	Chair	Bottle	Star
SAC	0.0437 ± 0.0006	0.0319 ± 0.0003	0.0383 ± 0.0020	0.0382 ± 0.0012
PPO	0.0507 ± 0.0082	0.0329 ± 0.0004	0.0394 ± 0.0002	0.0352 ± 0.0003
TD3	0.0555 ± 0.0043	0.0364 ± 0.0033	0.0443 ± 0.0001	0.0403 ± 0.0047
PlasticineLab	0.0324 ± 0.0023	0.0218 ± 0.0002	0.0283 ± 0.0007	0.0274 ± 0.0003
CPDeform	$\mathbf{0.0073 \pm 0.0001}$	$\mathbf{0.0072 \pm 0.0001}$	$\mathbf{0.0093 \pm 0.0001}$	$\mathbf{0.0198 \pm 0.0004}$
Env	Move++	Rope++	Writer++	Pinch
SAC	0.1660 ± 0.0185	0.0333 ± 0.0038	0.0149 ± 0.0002	0.0204 ± 0.0002
PPO	0.1936 ± 0.0508	0.0387 ± 0.0082	0.0164 ± 0.0001	0.0189 ± 0.0002
TD3	0.2164 ± 0.0057	0.0396 ± 0.0023	0.0153 ± 0.0083	0.0202 ± 0.0003
PlasticineLab	0.1895 ± 0.0001	0.0075 ± 0.0004	0.0122 ± 0.0001	0.0094 ± 0.0016
CPDeform	$\mathbf{0.0127 \pm 0.0030}$	$\mathbf{0.0052 \pm 0.0004}$	$\mathbf{0.0067 \pm 0.0001}$	$\mathbf{0.0081 \pm 0.0001}$

Table 1: The averaged Wasserstein-1 distance and the standard deviations of each method.

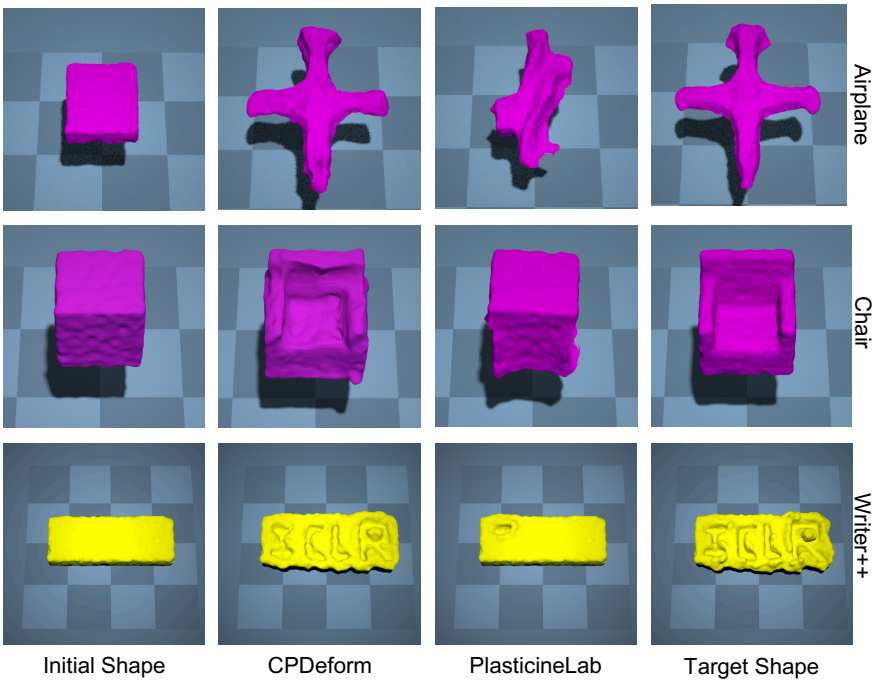


Figure 6: Qualitative results of CPDeform and PlasticineLab on multi-stage task environments.

to iteratively guide the solver to print the "ICLR" letters on the plasticine cube. Comparatively, we find that the stand-alone differentiable physics solver fails completely on the multi-stage tasks, as it lacks the necessary exploration power to overcome the local minima. Take **Move++** as an example, the vanilla solver is unable to move away from the cube it first initiates contact with after a stage. Similar to PlasticineLab (Huang et al., 2021), we also observe that the RL approaches in general perform worse than the vanilla gradient-based method.

4.3 Evaluation of Contact Point Discovery on Single Stage Tasks

To further demonstrate the effectiveness of our approach, we compare the one-shot contact points discovered by the backbone of CPDeform with the human-defined contact points on single-stage tasks from PlasticineLab. Table 2 lists the normalized incremental IoU scores, together with the standard deviations of all approaches.

Env	Move	Tri. Move	Torus	Rope
PlasticineLab	$\mathbf{0.90 \pm 0.12}$	0.35 ± 0.20	0.77 ± 0.39	0.59 ± 0.13
CPDeform	0.82 ± 0.12	$\mathbf{0.63 \pm 0.15}$	$\mathbf{0.99 \pm 0.01}$	$\mathbf{0.73 \pm 0.12}$
Env	RollingPin	Chopsticks	Assembly	Table
PlasticineLab	$\mathbf{0.93 \pm 0.04}$	$\mathbf{0.88 \pm 0.08}$	$\mathbf{0.90 \pm 0.10}$	0.01 ± 0.01
CPDeform	0.89 ± 0.06	0.34 ± 0.19	0.82 ± 0.13	$\mathbf{0.74 \pm 0.23}$

Table 2: We compare our contact point discovery backbone with human-defined contact points from PlasticineLab on single-stage tasks. We show the averaged normalized incremental IoU scores and the standard deviations of each method.

From Table 2 we can see that on most tasks, CPDeform performs better than or on par with the manipulation performance obtained from the human-defined initial end-effector positions from PlasticineLab. In **Table**, the agent needs to push one of the four legs of the plasticine. The initial position of the agent in PlasticineLab does not directly establish contact between the end effectors and the plasticine. The stand-alone differentiable physics solver from PlasticineLab with contact loss is inadequate for guiding the agent to find the correct "leg" to push. Whereas in CPDeform, the optimal priorities is capable of identifying the "leg" of interest, resulting in a suitable placement of the manipulator that significantly improves task completion. However, we recognize that CPDeform is not a panacea. We notice that CPDeform struggles on **Chopsticks** due to the limitation of transport priorities for capturing the dramatic topological change. Specifically, we find that the transport priorities cannot discover the transport plan that preserves the continuity of the topology, as discussed in Feydy (2020). How to find a continuous correspondence to map a source shape to target remains to be an interesting future direction to explore.

4.4 Comparison with other contact discovery method

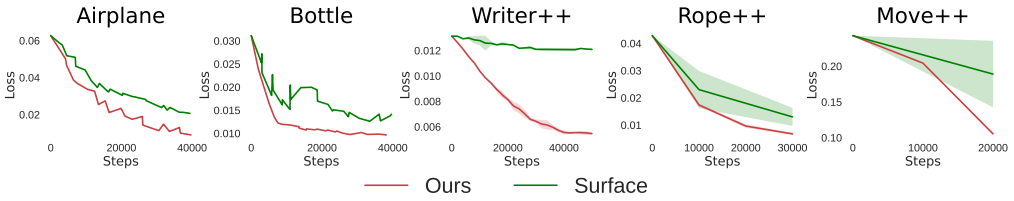


Figure 7: Ablation Learning Curves

In this section, we perform an ablation study to verify if transport priorities help find accurate solutions efficiently. We compare CPDeform with random sampling contact points on the surface of the soft bodies, while maintaining all other settings and hyperparameters, including the number of stages and candidate pose sets. Figure 7 shows the comparison of randomly sampled contact points with CPDeform. We observe that our method in CPDeform outperforms the surface-point sampling strategy by a large margin in terms of both accuracy and efficiency, proving the effectiveness of the transport-priority-based contact point discovery backbone employed by CPDeform.

5 Related Work

Soft Body Manipulation Soft body manipulation has a long history in robotics across multiple fields, such as fabric manipulation (Liang et al., 2019; Wu et al., 2020; Ha & Song, 2021), rope controlling (Yan et al., 2020; Wu et al., 2020), and food preparation (Bollini et al., 2013; Heiden et al., 2019). In this work, we study the elastoplastic materials in PlasticineLab (Huang et al., 2021) due to their wider applications than the pure elastic material model, leading to more general and realistic modeling of real-world soft bodies. The high degrees of freedom of soft bodies (Hu et al., 2018; Essahbi et al., 2012) limit the application of motion planning methods (Kuffner & LaValle, 2000; Kavraki et al., 1996) and most works focus on linear models like rope (Saha & Isto, 2006; 2007; Wakamatsu et al., 2006) or planar models like cloth (McConachie et al., 2020). Control methods such as Hirai & Wada (2000); Wada et al. (2001); Smolen & Patriciu (2009) bypass the expensive global

planning by approximating local models and manipulating soft bodies with local controllers, which sacrifices the ability to plan for the multi-stage tasks that we study in this work.

Recent development of Deep RL (Mnih et al., 2013; Schulman et al., 2017; Haarnoja et al., 2017; Fujimoto et al., 2018) methods has enabled a unified approach to learn both perception module and manipulation policy in an end-to-end way (Lin et al., 2020; Wu et al., 2020; Nair et al., 2017; Wang et al., 2019). However, policy learning suffers from exploration issue and usually require a huge amount of data or additional expert demonstrations to support imitation learning (Wu et al., 2020; Lee et al., 2015; Matas et al., 2018; Seita et al., 2019). Another popular approach is to use neural networks to approximate soft body dynamics (Li et al., 2018; Lin et al., 2021; Hoque et al., 2021) and solve control problems with model predictive control (Rubinstein, 1999) or gradient-based optimization. While these methods gain the generalizability by employing a shared local dynamic model, they need a way to use the model for planning. Similar to our iterative contact-and-deform approach, methods like Lin et al. (2020); Seita et al. (2019); Li et al. (2018) apply a pick-and-place action space to manipulate soft bodies like cloth and rope. While this action space enables efficient exploration and simplifies learning tasks, it poses a restriction for the type of end effectors, making it unsuitable for the fine-grained manipulation of soft bodies such as plasticine. Comparatively, CPDeform leverages the differentiable physics solver to complete more sophisticated manipulation tasks.

Differentiable Physics for Trajectory Optimization Our method uses the differentiable simulator developed in PlasticineLab (Huang et al., 2021) for trajectory optimization. Encouraged by the success of gradient descent in neural network learning, differentiable physics with analytic physics models (Geilinger et al., 2020; Degrave et al., 2016; de Avila Belbute-Peres et al., 2018; Carpentier & Mansard, 2018; Giftthaler et al., 2017; Heiden et al., 2019; 2020; 2021; Toussaint et al., 2018; Hu et al., 2019; 2020; Qiao et al., 2020; Murthy et al., 2020; Millard et al., 2020; Werling et al., 2021; Du et al., 2021) has gained increasing popularity. As mentioned in Sec. 2, applying gradient-optimization with an inappropriate contact point would get stuck in the local minima, especially in tasks where multiple contact switches are needed. Previous research has explored various ways to handle contacts for rigid bodies. Contact-invariant optimization (Mordatch et al., 2012) imposes variables as soft representation of objects' contact relationship, and in (Posa et al., 2014; Sleiman et al., 2019), contacts are handled implicitly within analytical models. However, due to the nonlinear and contact-rich nature of manipulation tasks, we often have to combine search and optimization to solve mixed-integer program (Han & Tedrake, 2020) or logic-geometric program (Toussaint et al., 2018) problems, which are inefficient given the complexity of 3D soft bodies. We explore a complementary direction of previous approaches. By integrating visual cues into differentiable physics, we are able to skip many local minima and boost the performance in various soft body manipulation tasks.

Grasping and Rigid Body Manipulation Our contact point discovery approach shares a similar spirit with grasp pose detection in rigid body manipulation. Determining contact point or grasp pose for different end effectors is one of the everlasting topics in rigid body manipulation (Bohg et al., 2014; Miller & Allen, 2004; Dang & Allen, 2012; Qin et al., 2020; Mahler et al., 2017). By analyzing 3D geometry (Sahbani et al., 2012), one can find antipodal grasps that satisfy force closure (Nguyen, 1988; Chen & Burdick, 1993), and grasp objects without running simulation. Similarly, our contact point discovery method tries to find contact points through geometrical analysis, which could be viewed as a step towards extending geometric analysis for more general soft body manipulation tasks.

6 Conclusion and Future Work

In this paper, we propose a novel framework, CPDeform, that integrates optimal transport-based contact discovery into differentiable physics. Our framework requires a properly defined Wasserstein distance on the object's representation. The choice of material types does not affect our optimal transport heuristic because it relies on shape information only. We find that the contact point discovery method, can be directly employed by the differentiable physics solver and performs on par with or better than human-defined initial contact points in single-stage tasks. On multi-stage tasks that are infeasible for the vanilla solver, CPDeform employs a heuristic searching approach to iteratively solve the tasks. Our work demonstrates the importance of contact point in policy learning with differentiable physics and the advantage of geometric-analysis methods as a heuristic. Interesting avenues for future work include generalizing the discovery of useful contact points through learning methods for a diverse set of shapes, and applying a similar contact point discovery principle to dexterous rigid body manipulation, or combining it with other planning approaches.

REFERENCES

Jeannette Bohg, Antonio Morales, Tamim Asfour, and Danica Kragic. Data-driven grasp synthesis—a survey. *IEEE Transactions on Robotics*, 30(2):289–309, 2014. doi: 10.1109/TRO.2013.2289018. 9

Mario Bollini, Stefanie Tellex, Tyler Thompson, Nicholas Roy, and Daniela Rus. Interpreting and executing recipes with a cooking robot. In *Experimental Robotics*, pp. 481–495. Springer, 2013. 1, 8

Justin Carpentier and Nicolas Mansard. Analytical derivatives of rigid body dynamics algorithms. In *Robotics: Science and systems (RSS 2018)*, 2018. 9

I-Ming Chen and J.W. Burdick. Finding antipodal point grasps on irregularly shaped objects. *IEEE Transactions on Robotics and Automation*, 9(4):507–512, 1993. doi: 10.1109/70.246063. 9

Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger (eds.), *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013. URL https://proceedings.neurips.cc/paper/2013/file/af21d0c97db2e27e13572cbf59eb343d-Paper.pdf. 6

Hao Dang and Peter K. Allen. Semantic grasping: Planning robotic grasps functionally suitable for an object manipulation task. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1311–1317, 2012. doi: 10.1109/IROS.2012.6385563. 9

Filipe de Avila Belbute-Peres, Kevin Smith, Kelsey Allen, Josh Tenenbaum, and J Zico Kolter. End-to-end differentiable physics for learning and control. In *Advances in Neural Information Processing Systems*, pp. 7178–7189, 2018. 9

Jonas Degrave, Michiel Hermans, Joni Dambre, et al. A differentiable physics engine for deep learning in robotics. *arXiv preprint arXiv:1611.01652*, 2016. 9

Tao Du, Kui Wu, Pingchuan Ma, Sebastien Wah, Andrew Spielberg, Daniela Rus, and Wojciech Matusik. Diffpd: Differentiable projective dynamics with contact. *arXiv preprint arXiv:2101.05917*, 2021. 9

Nabil Essahbi, Belhassen Chedli Bouzgarrou, and Grigore Gogu. Soft material modeling for robotic manipulation. In *Applied Mechanics and Materials*, volume 162, pp. 184–193. Trans Tech Publ, 2012. 8

J Feydy. *Geometric data analysis, beyond convolutions*. PhD thesis, PhD thesis, Université Paris-Saclay, 2020. 8

Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning*, pp. 1582–1591, 2018. 6, 9

Moritz Geilinger, David Hahn, Jonas Zehnder, Moritz Bächer, Bernhard Thomaszewski, and Stelian Coros. Add: Analytically differentiable dynamics for multi-body systems with frictional contact. *arXiv preprint arXiv:2007.00987*, 2020. 9

Markus Giftthaler, Michael Neunert, Markus Stäuble, Marco Frigerio, Claudio Semini, and Jonas Buchli. Automatic differentiation of rigid body dynamics for optimal control and estimation. *Advanced Robotics*, 31(22):1225–1237, 2017. 9

Huy Ha and Shuran Song. Flingbot: The unreasonable effectiveness of dynamic manipulation for cloth unfolding. *arXiv preprint arXiv:2105.03655*, 2021. 8

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. 2017. 6, 9

Weiqiao Han and Russ Tedrake. Local trajectory stabilization for dexterous manipulation via piecewise affine approximations. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 8884–8891. IEEE, 2020. 9

Eric Heiden, David Millard, Hejia Zhang, and Gaurav S Sukhatme. Interactive differentiable simulation. *arXiv preprint arXiv:1905.10706*, 2019. 8, 9

Eric Heiden, David Millard, Erwin Coumans, and Gaurav S Sukhatme. Augmenting differentiable simulators with neural networks to close the sim2real gap. *arXiv preprint arXiv:2007.06045*, 2020. 9

Eric Heiden, Miles Macklin, Yashraj Narang, Dieter Fox, Animesh Garg, and Fabio Ramos. Disect: A differentiable simulation engine for autonomous robotic cutting. *arXiv preprint arXiv:2105.12244*, 2021. 9

Shinichi Hirai and Takahiro Wada. Indirect simultaneous positioning of deformable objects with multi-pinching fingers based on an uncertain model. *Robotica*, 18(1):3–11, 2000. 8

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. *arXiv preprint arXiv:2102.09754*, 2021. 9

Yuanming Hu, Yu Fang, Ziheng Ge, Ziyin Qu, Yixin Zhu, Andre Pradhana, and Chenfanfu Jiang. A moving least squares material point method with displacement discontinuity and two-way rigid body coupling. *ACM Transactions on Graphics (TOG)*, 37(4):1–14, 2018. 8

Yuanming Hu, Jiancheng Liu, Andrew Spielberg, Joshua B Tenenbaum, William T Freeman, Jiajun Wu, Daniela Rus, and Wojciech Matusik. Chainqueen: A real-time differentiable physical simulator for soft robotics. In *2019 International Conference on Robotics and Automation (ICRA)*, pp. 6265–6271. IEEE, 2019. 9

Yuanming Hu, Luke Anderson, Tzu-Mao Li, Qi Sun, Nathan Carr, Jonathan Ragan-Kelley, and Frédo Durand. Difftaichi: Differentiable programming for physical simulation. *ICLR*, 2020. 9

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. *arXiv preprint arXiv:2104.03311*, 2021. 1, 2, 3, 6, 7, 8, 9

Lydia E Kavraki, Petr Svestka, J-C Latombe, and Mark H Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE transactions on Robotics and Automation*, 12(4):566–580, 1996. 8

James J Kuffner and Steven M LaValle. Rrt-connect: An efficient approach to single-query path planning. In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, volume 2, pp. 995–1001. IEEE, 2000. 8

Alex X Lee, Abhishek Gupta, Henry Lu, Sergey Levine, and Pieter Abbeel. Learning from multiple demonstrations using trajectory-aware non-rigid registration with applications to deformable object manipulation. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5265–5272. IEEE, 2015. 9

Yunzhu Li, Jiajun Wu, Russ Tedrake, Joshua B Tenenbaum, and Antonio Torralba. Learning particle dynamics for manipulating rigid bodies, deformable objects, and fluids. *arXiv preprint arXiv:1810.01566*, 2018. 9

Junbang Liang, Ming C Lin, and Vladlen Koltun. Differentiable cloth simulation for inverse problems. *Advances in Neural Information Processing Systems*, 2019. 8

Xingyu Lin, Yufei Wang, Jake Olkin, and David Held. Softgym: Benchmarking deep reinforcement learning for deformable object manipulation. *arXiv preprint arXiv:2011.07215*, 2020. 9

Xingyu Lin, Yufei Wang, and David Held. Learning visible connectivity dynamics for cloth smoothing. *arXiv preprint arXiv:2105.10389*, 2021. 9

Jeffrey Mahler, Jacky Liang, Sherdil Niyaz, Michael Laskey, Richard Doan, Xinyu Liu, Juan Aparicio Ojea, and Ken Goldberg. Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics. *CoRR*, abs/1703.09312, 2017. URL http://arxiv.org/abs/1703.09312. 9

Jan Matas, Stephen James, and Andrew J Davison. Sim-to-real reinforcement learning for deformable object manipulation. *arXiv preprint arXiv:1806.07851*, 2018. 9

Hermann Mayer, Faustino Gomez, Daan Wierstra, Istvan Nagy, Alois Knoll, and Jürgen Schmidhuber. A system for robotic heart surgery that learns to tie knots using recurrent neural networks. *Advanced Robotics*, 22(13-14):1521–1537, 2008. 1

Dale McConachie, Andrew Dobson, Mengyao Ruan, and Dmitry Berenson. Manipulating deformable objects by interleaving prediction, planning, and control. *The International Journal of Robotics Research*, 39(8):957–982, 2020. 8

David Millard, Eric Heiden, Shubham Agrawal, and Gaurav S Sukhatme. Automatic differentiation and continuous sensitivity analysis of rigid body dynamics. *arXiv preprint arXiv:2001.08539*, 2020. 9

A.T. Miller and P.K. Allen. Graspit! a versatile simulator for robotic grasping. *IEEE Robotics Automation Magazine*, 11(4):110–122, 2004. doi: 10.1109/MRA.2004.1371616. 9

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013. 9

Igor Mordatch, Emanuel Todorov, and Zoran Popović. Discovery of complex behaviors through contact-invariant optimization. *ACM Transactions on Graphics (TOG)*, 31(4):1–8, 2012. 9

J Krishna Murthy, Miles Macklin, Florian Golemo, Vikram Voleti, Linda Petrini, Martin Weiss, Breandan Considine, Jérôme Parent-Lévesque, Kevin Xie, Kenny Erleben, et al. gradsim: Differentiable simulation for system identification and visuomotor control. In *International Conference on Learning Representations*, 2020. 9

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)*, pp. 2146–2153. IEEE, 2017. 9

Van-Duc Nguyen. Constructing force- closure grasps. *The International Journal of Robotics Research*, 7(3):3–16, 1988. doi: 10.1177/027836498800700301. URL https://doi.org/10.1177/027836498800700301. 9

Michael Posa, Cecilia Cantu, and Russ Tedrake. A direct method for trajectory optimization of rigid bodies through contact. *The International Journal of Robotics Research*, 33(1):69–81, 2014. 9

Yi-Ling Qiao, Junbang Liang, Vladlen Koltun, and Ming C Lin. Scalable differentiable physics for learning and control. *arXiv preprint arXiv:2007.02168*, 2020. 9

Yuzhe Qin, Rui Chen, Hao Zhu, Meng Song, Jing Xu, and Hao Su. S4g: Amodal single-view single-shot se(3) grasp detection in cluttered scenes. In Leslie Pack Kaelbling, Danica Kragic, and Komei Sugiura (eds.), *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pp. 53–65. PMLR, 30 Oct–01 Nov 2020. URL https://proceedings.mlr.press/v100/qin20a.html. 9

Reuven Rubinstein. The cross-entropy method for combinatorial and continuous optimization. *Methodology and computing in applied probability*, 1(2):127–190, 1999. 9

Mitul Saha and Pekka Isto. Motion planning for robotic manipulation of deformable linear objects. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pp. 2478–2484. IEEE, 2006. 8

Mitul Saha and Pekka Isto. Manipulation planning for deformable linear objects. *IEEE Transactions on Robotics*, 23(6):1141–1150, 2007. 8

A. Sahbani, S. El-Khoury, and P. Bidaud. An overview of 3d object grasp synthesis algorithms. *Robotics and Autonomous Systems*, 60(3):326–336, 2012. ISSN 0921-8890. doi: https://doi.org/10.1016/j.robot.2011.07.016. URL https://www.sciencedirect.com/science/article/pii/S0921889011001485. Autonomous Grasping. 9

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

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. 6, 9

Daniel Seita, Aditya Ganapathi, Ryan Hoque, Minho Hwang, Edward Cen, Ajay Kumar Tanwani, Ashwin Balakrishna, Brijen Thananjeyan, Jeffrey Ichnowski, Nawid Jamali, et al. Deep imitation learning of sequential fabric smoothing policies. *arXiv preprint arXiv:1910.04854*, 2, 2019. 9

Jean-Pierre Sleiman, Jan Carius, Ruben Grandia, Martin Wermelinger, and Marco Hutter. Contact-implicit trajectory optimization for dynamic object manipulation. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6814–6821. IEEE, 2019. 9

Jerzy Smolen and Alexandru Patriciu. Deformation planning for robotic soft tissue manipulation. In *2009 Second International Conferences on Advances in Computer-Human Interactions*, pp. 199–204. IEEE, 2009. 8

Marc A Toussaint, Kelsey Rebecca Allen, Kevin A Smith, and Joshua B Tenenbaum. Differentiable physics and stable modes for tool-use and manipulation planning. 2018. 9

Takahiro Wada, Shinichi Hirai, Sadao Kawamura, and Norimasa Kamiji. Robust manipulation of deformable objects by a simple pid feedback. In *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No. 01CH37164)*, volume 1, pp. 85–90. IEEE, 2001. 8

Hidefumi Wakamatsu, Eiji Arai, and Shinichi Hirai. Knotting/unknotting manipulation of deformable linear objects. *The International Journal of Robotics Research*, 25(4):371–395, 2006. 8

Angelina Wang, Thanard Kurutach, Kara Liu, Pieter Abbeel, and Aviv Tamar. Learning robotic manipulation through visual planning and acting. *arXiv preprint arXiv:1905.04411*, 2019. 9

Keenon Werling, Dalton Omens, Jeongseok Lee, Ioannis Exarchos, and C Karen Liu. Fast and feature-complete differentiable physics for articulated rigid bodies with contact. *arXiv preprint arXiv:2103.16021*, 2021. 9

Yilin Wu, Wilson Yan, Thanard Kurutach, Lerrel Pinto, and Pieter Abbeel. Learning to manipulate deformable objects without demonstrations, 2020. 1, 8, 9

Mengyuan Yan, Yilin Zhu, Ning Jin, and Jeannette Bohg. Self-supervised learning of state estimation for manipulating deformable linear objects. *IEEE Robotics and Automation Letters*, 5(2):2372–2379, 2020. 8

A ENVIRONMENT DETAILS

- **Airplane** The agent needs to manipulate a pair of "spatulas" (represented as round boxes) to sculpt the nose, wing, and tail on the plasticine cube to match the target airplane shape.
- **Chair** With a pair of "spatulas", the agent needs to build the seat, back, and arm rest from the plasticine cube to reach the target chair shape.
- **Bottle** With a pair of "spatulas", the agent needs to build the neck and body from the plasticine cube to reach the target bottle shape.
- **Star** The agent needs manipulate a pair of "spatulas" to build the tips of the star from the plasticine cube to reach the target bottle shape.
- **Move++** The agent needs to use a single pair of sphere manipulators to separately transport three plasticine cubes to fulfill the three destinations of demand.
- **Rope++** The agent needs to use a single pair of sphere manipulators to reshape the plasticine rope to reach the target polyline shape.
- **Writer++** The agent manipulates a "pen" (represented using a vertical capsule), to print the letters "ICLR" on the plasticine cube.
- **Pinch** In this task, the agent manipulates one rigid sphere to create dents specified by the target shape on the plasticine box.

B MORE DETAILS ABOUT OPTIMAL TRANSPORT

Optimal transport is useful for comparing measures μ and ν in a Lagrangian framework, by accounting for the cost of transporting one measure to another. Here, we consider the first measure μ as the soft-body particles in the source state and ν as the soft-body particles in the target state that we wish to achieve through manipulation, where both input measures are on the space $\mathbb{R}^{N_p \times 3}$. Let $\alpha, \beta \in \mathbb{R}^{N_p}$ be the mass vectors that for the particles in μ, ν respectively. Since all particles are weighted equally in our simulator, $\alpha = \beta = \mathbf{1}$. Let $U(\alpha, \beta)$ be the polytope that contains all transport plans, which are non-negative matrices whose rows and columns sum to α, β respectively. We have $U(\alpha, \beta) := \{P \in \mathbb{R}_+^{N_p \times N_p} \mid P\mathbf{1} = \alpha, P^T\mathbf{1} = \beta\}$. Given a cost function $C : \mathbb{R}^3 \times \mathbb{R}^3 \mapsto [0, \infty]$, we create the cost matrix $M \in \mathbb{R}^{N_p \times N_p}$, where each element $M_{i,j} = C(\mu_i, \nu_j)$ measures the cost of transporting the particle μ_i to ν_j. Now, the cost of mapping μ to ν can be quantified as $\langle P, M \rangle$. We define the primal form of the *optimal transport* (OT) problem as $\text{OT}(\alpha, \beta) := \min_{P \in U(\alpha, \beta)} \langle P, M \rangle$.

C PSEUDOCODE OF CPDEFORM

Algorithm 1 CPDeform

Input: Current shape S_0 with N particles, goal shape G, number of stages n_{stage}, number of steps per stage n_{step}, candidate pose set $\mathcal{T}$

```
for 0 ≤ i < n_stage do
    Compute optimal transport priorities {α_i}_{i≤N} between S_i and goal shape G
    Find point p with largest priorities α_p
    for pose t ∈ T do
        Place manipulators around point p with largest heuristic value without collisions
        Solve trajectories with differentiable physics to generate plan p_t and compute its shape matching loss c_t
    end for
    Execute the plan p_t with the minimal loss c_t
end for
```

D CONTROLLER OPTIMIZATION WITH DIFFERENTIABLE PHYSICS

Consider that the motion trajectory is parameterized as a sequence of three-dimensional actions $\{a_1, \ldots, a_T\}$, where T is the number of simulation steps. Let $\{s_t\}_{0 \le t \le T}$ be simulation states at

different time steps which include the state of the plasticine and manipulator. The differentiable simulator starts from the initial state s_0 and completes the action sequences by repeatedly executing the forward function $s_{t+1} = \phi(s_t, a_{t+1})$ until the ending state s_T has been reached. The objective of the optimizer is to minimize the distance between the final shape and the target shape. We represent the objective as a loss function $\mathcal{L}(s_T, g)$ where g is the target shape. Since the simulation forward function ϕ is fully-differentiable, we can compute gradients $\partial\mathcal{L}/\partial a_t$ of the loss $\mathcal{L}$ with respect to each action a_t, and run gradient descent steps to optimize the action sequences by $a_t = a_t - \alpha\partial\mathcal{L}/\partial a_t$ where α is the learning rate. We repeat this optimization loop for N iterations. The overall process is reflected by Algorithm 2.

Algorithm 2 Differentiable Physics Solver for Controller Optimization

Input: Target shape g, action sequence $\{a_1, \ldots, a_T\}$, initial state s_0, differentiable forward simulation function ϕ, number of iterations N, learning rate α

```
for 0 ≤ i < N do
    for 0 ≤ j < T do
        s_{t+1} = φ(s_t, a_{t+1})
    end for
    Compute shape difference loss using L(s_T, g)
    Update the action sequence by a_t = a_t − α∂L/∂a_t
end for
```

E Additional Baselines

We compare CPDeform with two additional baselines, named Multiple-Restarts (Multi-Re) and Bayesian-Optimization (Bayes-Op).

Investigation on multiple restarts For each stage, we randomly sample a collection of 15 contact plans. We then use differentiable physic solver to optimize the action sequence. The execution of the solver for each plan corresponds with a single restart or an "initial guess." We use the contact plan that achieves the lowest loss for manipulation.

Investigation on Bayesian optimization For each stage, we use Bayesian optimization with Gaussian process to optimize the contact plan. We employ a black box function that takes a contact plan as its input and outputs the loss achieved by the differentiable physics solver. We then perform Bayesian optimization for 15 iterations to optimize the contact plan for the given stage.

Env	Airplane	Chair	Bottle	Star	Move++	Rope++	Writer++	Pinch
Multi-Re	0.0122	0.0098	0.0112	0.0221	0.1209	0.0055	0.0076	0.0094
Bayes-Op	0.0369	0.0103	0.0136	0.0212	0.1234	0.0067	0.0113	0.0092
PlasticineLab	0.0324	0.0218	0.0283	0.0274	0.1895	0.0075	0.0122	0.0094
CPDeform	**0.0073**	**0.0072**	**0.0093**	**0.0198**	**0.0127**	**0.0052**	**0.0067**	**0.0081**

Table 3: The averaged Wasserstein-1 distance of each method.

In general, we observe that Multiple-Restarts and Bayesian-Optimization perform better than PlasticineLab, or the standalone differentiable physics solver. CPDeform outperforms these two additinoal baselines. Additionally, because both Multiple-Restarts and Bayesian-Optimization rely heavily on trials and errors to find the suitable contact points, as illustrated in Figure 8, they are much more computationally expensive and less efficient than CPDeform.

F Runtime

We show the runtime of CPDeform, PlasticineLab, Bayesian-Optimization, and Multiple-Restarts in Figure 8. RL approaches are not drawn because they cannot complete the manipulation tasks, and their loss curves are close to the starting loss. We draw the wall-time in seconds on the x-axis and the corresponding Wasserstein-1 distance loss values on the y-axis. Each data point represents the lowest

loss achieved for that method at the end of a stage. Bayesian-Optimization and Multiple-Restarts spends more time than the CPDeform and PlasticineLab on each stage, due to the number of Bayesian optimization iterations and the number of restarts performed respectively. We observe that CPDeform is more efficient than other approaches.

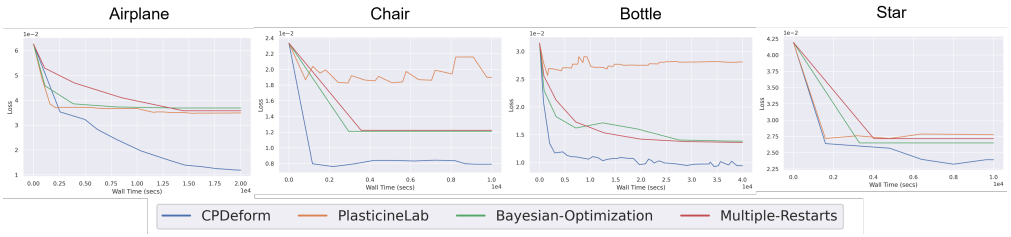


Figure 8: Runtime of each method. We draw the wall-time in seconds on the x-axis and the corresponding Wasserstein-1 distance loss values on the y-axis. Each data point represents the lowest loss achieved for that method at the end of a stage.

G Limitation and Future Works

In this section, we discuss several implicit assumptions we made when constructing tasks. These limitations are not common in our tasks and could be addressed by including additional techniques; we did not take those factors into account in this work. We believe they constitute several interesting future directions to explore.

Density Assumption Low or non-uniform density could potentially impact the optimization steps due to varying optimal transport mapping. For instance, the optimal transport loss might not be able to capture areas of the target shape whose particle distributions are extremely sparse (e.g., a few particles) and is likely unable to guide the differentiable physics solver to task completion. In such a case, having low or non-uniform density might pose a limit on our method. As a solution, we have uniformly sampled particles inside the target shapes. By uniformly sampling particles inside a given 3D shape of mesh representation, we create its corresponding target template that the differentiable physics solver can use for the manipulation tasks. We think that considerations of target shapes (e.g., a cuboid) with varying internal configurations (e.g., non-uniform density) is an interesting future direction.

Volume Assumption We assume that the volumes of the initial and target shapes are similar. Since our tasks use manipulators that are tailored to compressing instead of expanding the soft materials, the initial shape cannot be significantly smaller than the target shape. Conversely, if the initial shape is significantly larger than the target shape, it will require a lot of effort to compress the initial shape into the target shape. As a solution, we can meet this condition by cutting and assembling the initial plasticines.

Topology Assumption We assume that the initial and target shapes do not have tremendous topological differences while requiring shape transformation that involves complex motion planning. When this condition is false, the greedy algorithm that finds the best contact points at the current stage could be limited. For example, consider a task where the initial shape is a straight rope, and the target shape is a rope with knots. In order to complete the task, the rope needs to be folded and wrapped, which might lead to a temporary increment of shape-matching loss for a few stages. The greedy approach is limited because it might be unwilling to accept the temporary increment of the loss. It would potentially arrive at a solution whose general shape is matched with the target rope by omitting the detailed knots. As a possible solution, if we incorporate RL-based algorithms into our approach, the ability to plan globally could allow the differentiable physics solver to overcome the temporary increment of the loss at an intermediate stage and act for the best completion of the task at the final stage.

H INVESTIGATION ON THE OPTIMALITY OF CPDEFORM

We use **Writer** to study the optimality of our optimal-transport-based heuristic. Holding the height of the manipulator at a fixed value, as illustrated in the leftmost figure in Figure 9, we create a 2D grid that covers the top surface of the plasticine cube. The grid is of size 20×20 and its points are equidistant from each other on each row or column. We then record the loss achieved by executing the differentiable physics solver at each grid point (i.e., contact point). Having collected a grid of loss values, we use interpolation to draw the 2D loss landscape, as illustrated in Figure 9. By exhaustively searching through each grid point, we observe that the higher transport priorities indicated by our heuristic tend to correlate with lower loss values, and vice versa.

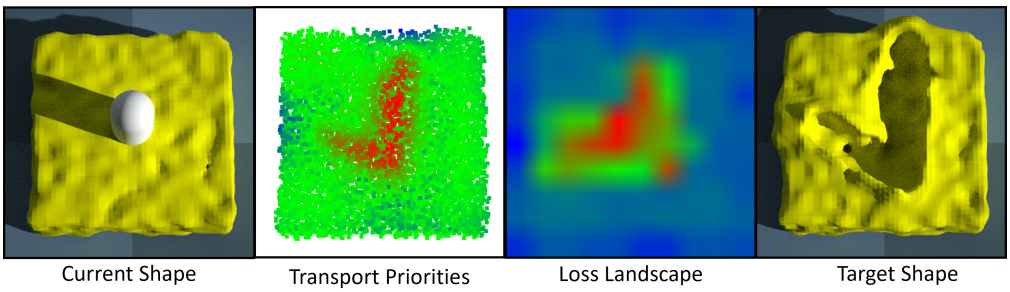


Figure 9: We demonstrate the visualizations of the transport priorities and the loss landscape using **Writer**. For the loss landscape, the color of each pixel corresponds with the loss achieved by applying the differentiable physics solver near the corresponding simulator coordinate, where the height of the manipulator is set as a fixed value (illustrated in the current shape). Here, warmer colors indicate smaller loss values and colder colors indicate larger loss values.