Robust Manipulation of Deformable Linear Objects

Jimmy Envall and Stelian Coros

***Abstract*—Robotic manipulation is a fundamental challenge in the pursuit of automating household tasks and advancing robotic manufacturing. Manipulation planning for deformable objects is particularly challenging due to the associated large action spaces and complex dynamics. For deformable objects, some actions are sensitive to noise in the model, which can significantly degrade the accuracy of predicted trajectories.**

In this letter, we present a motion planning algorithm for arranging deformable linear objects (DLOs) into user-provided goal configurations. Using a novel problem formulation as well as a combination of sampling and gradient based optimization, the algorithm finds sequences of grasps and control inputs that are robust. We demonstrate the effectiveness of the new algorithm using numerical experiments and manipulation examples both in simulation and in real world environments.

***Index Terms*—Manipulation Planning, Task and Motion Planning, Bimanual Manipulation**

I. Introduction

ROBOTICS research is fueled by visions of scenarios in which robots complete complex manipulation tasks given only a description of the goal. Target applications exist in many industries, including manufacturing, medicine and garments where robots could route electrical cables, perform surgeries and sew shirts, but also in the household where robots could wash and fold laundry and prepare food.

Many of these tasks require manipulation of deformable or soft objects. When manipulating deformable objects, some actions are more robust than others. For example, pulling a rope from one end along a surface towards a goal is significantly more likely to succeed than pushing the rope from the opposite end. In addition, when simulating deformable objects, physical phenomena such as buckling leads to bifurcations and instabilities in the dynamics model which require special consideration in order to avoid numerical problems during simulation [1], [2], [3].

In this work, we devise an algorithm for generating robust control inputs for rearranging deformable linear objects (DLOs) into user-provided configurations. Here, a robust control is a control for which the resulting object trajectory changes only slightly in response to small perturbations in the initial configuration.

Large deviations in the trajectory due to small variations in the initial conditions indicate that the trajectory passes through states with high sensitivity, such as buckling. Such configurations cause difficulties for gradient based optimization methods since the Jacobian of the forward simulation residual can become ill-conditioned. They also reduce the likelihood of successful sim-to-real transfer, since small deviations in the initial configuration—for example due to measurement errors—can drastically alter the resulting trajectory. Robustness to small changes in the initial conditions aids in avoiding sensitive configurations in the trajectory, such as buckling, which aids sim-to-real transfer and improves optimization performance.

A robust action consists of both a grasp sequence and control that together are likely to bring the object into a desired configuration. Finding an appropriate control input requires the robot to be able to reason about and predict the behavior of an object, for example a cable that needs to be installed, after applying forces at selected points. On the other hand, selecting an appropriate grasp point hinges on the robot knowing how close to the goal configuration it can bring the object by applying forces at the selected points. The robot is thus faced with a chicken-and-egg problem where the choice of grasp points and control forces are interleaved. Our algorithm leverages GPU parallelization to quickly estimate the cost for a large number of grasp sequences. The best motion plan candidate is then further refined in a separate optimization step based on sensitivity analysis.

The algorithm cannot guarantee a buckling-free trajectory, but nevertheless provides a practical approach for manipulation planning for DLOs. We demonstrate the effectiveness of our algorithm using numerical experiments and evaluate the generated trajectories using real world robot experiments.

II. Related work

Model Based Manipulation Planning: Robotic manipulation of deformable objects presents a great challenge in robotics [4]. Manipulation planning algorithms for DLOs often leverage simplified dynamics assumptions, such as quasi-statics. In an early work, Moll and Kavraki developed a method to construct minimal energy paths between sampled gripper configurations for use in a roadmap planner [5]. Other options include checking the stability of the sampled configurations [6] or projecting them onto a stable manifold [7]. The quasi-static assumption can also be used for manipulation planning of DLOs or sheets using position-based dynamics [8], [9], or with the finite element method [10] or rod models [11] using gradient based optimization. However, the quasi-static framework assumes infinitely slow movements and makes integration of friction challenging, and therefore we instead opt for a dynamics based formulation. Dynamics based formulations have been used previously for manipulation planning e.g., by Zimmermann et al. [12].

Manuscript received: December 23, 2025; Revised March 25, 2026; Accepted April 22, 2026.

This paper was recommended for publication by Editor Júlia Borràs Sol upon evaluation of the Associate Editor and Reviewers' comments.

This work was supported by a ETH RobotX research grant funded through the ETH Zurich Foundation.

Jimmy Envall and Stelian Coros are with the Computational Robotics Lab, Department of Computer Science, ETH Zurich, Switzerland jenvall@ethz.ch; scoros@ethz.ch.

Digital Object Identifier (DOI): see top of this page.

In general, the configuration of a deformable object is the result of all actions applied to an object up to a particular point in time. Therefore, as the number of available grasp locations increase, the complexity of the search increases exponentially. In a recent work, Wang et al. presented an approach to simplify the geometry of the manipulated object [13], which efficiently reduces the search space while maintaining high fidelity. Their approach targets efficient reduction of the action space, whereas in this work the focus is on efficiently searching through available actions.

In the model-based setting, an accurate simulation is a crucial part of the pipeline. Deformable objects have been extensively studied in the literature [1], [3], [4], [14], [15], and exhibit certain characteristics that are challenging to simulate. For example, buckling arising from compression induces non-convexity in the forward simulation energy [3], which causes problems for motion planning [16]. One way to avoid non-convexity is to simplify the model by ignoring compressive energy [3], [16]. This yields an abstracted model that is well behaved also with large time steps [16]. The abstracted formulation works well for DLO arrangement tasks, but neglecting internal restorative forces neglects important physical behavior of real world objects, and the formulation is unable to generate controls that compress the object, even if buckling is absent. It is also unclear how the idea of abstraction can be generalized to other forms of indefiniteness and to higher-dimensional systems. To avoid these issues, we here use a standard formulation based on the mass-spring model without abstraction.

For robust planning, Jankowski et al. presented a stochastic formulation for non-prehensile planar manipulation based on Gaussian beliefs that explicitly models the uncertainty in the dynamics [17]. Their formulation allows the variance of the object configuration to be predicted and explicitly included in the optimization objective. In our work we instead optimize over a collection of perturbed systems. This idea is reminiscent of domain randomization that is often used in data driven approaches [18], [19], [20]. Similar ideas have also been used to find swing-up trajectories for double-link pendulums with uncertain design parameters [21].

Data-driven approaches: Data driven approaches promise robustness to uncertain dynamics and behavior that is hard to model. Chi et al. proposed an efficient method that adapts control input from task residuals for highly dynamic actions [22], but it assumes the action is repeatable. For rope rearrangement tasks, Nair et al. show that rope dynamics can be learned in a self-supervised fashion [23]. The learned model can then be used to mimic an expert manipulating a rope. Salhotra et al. showed that expert guidance can be useful in guiding reinforcement learning exploration towards directions that are likely to be successful [24]. Seita et al. propose to use transporter networks for learning pick-and-place actions to rearrange different deformable objects in a goal-conditioned fashion [25]. These approaches ([23], [24], [25]) assume that expert demonstrations are available, which can be expensive to obtain in practice.

Luo et al. propose a hierarchical approach for cable routing based on imitation learning where a high-level policy reasons whether to retry a failed attempt or continue to the next step of the task [26]. They highlight the importance of robustness to failures and ability to retry a failed step in order to compensate for imperfect low-level controllers. The focus of our work is complementary and instead makes the low-level planner more robust and thus more likely to succeed.

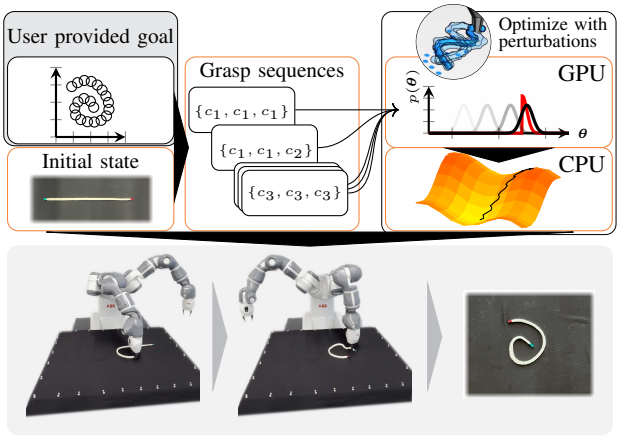


Fig. 1: Given a target configuration, our algorithm prunes grasp sequences using sampling based optimization on the GPU and refines the trajectory of a selected sequence using gradient based optimization before execution on a robot.

In general, data-driven approaches struggle with the intricate dynamics of deformable objects, and their ability to generalize to new, unseen data is still uncertain.

Contact planning: Contact planning appears in many contexts in robotics. One approach is to incorporate the contacts directly into the optimization problem, which enables contact sequence discovery both for locomotion [27], [28], [29] and rigid object manipulation tasks [30], [31]. Our formulation instead relies on fixed grasp sequences which avoids the challenges associated with complementarity constraints.

Finding grasp points can be seen as a special case of a more general *task and motion planning* problem. Standard work in this field includes Logic Geometric Programming (LGP) [32], [33] which combines symbolic search with nonlinear optimization and also supports dynamic interactions [34]. However, as Garrett et al. pointed out, TAMP with deformable objects is an open problem [35]. In our approach, we leverage parallel computation which enables fast identification of a promising grasp sequence from an exhaustive list of candidate sequences.

III. Overview

Our method is based on an optimal control formulation. The optimal control input for moving a DLO into a target configuration consists of a sequence of grasp points and a control that is applied at the grasps. The cost captures how close to the target configuration the object can be brought using the given grasp sequence.

The formulation is based on a geometric description of the object. We use the mass spring model, which constitutes a balanced choice between computational performance and

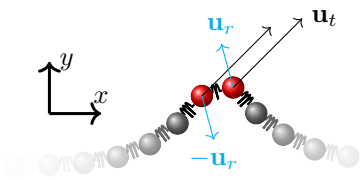


Fig. 2: A grasp always involves two adjacent point masses. The control is three dimensional and can jointly translate the masses in the plane ($\mathbf{u}_t$), or rotate them around the midpoint of the spring that connects them ($\mathbf{u}_r$).

physical accuracy [4], however, the formulation is not restricted to this particular choice. We denote the configuration of a deformable linear object object at time t by $\mathbf{x}^t \in \mathbb{R}^{2N_m}$ where N_m is the number of mass points in the model. The goal is to bring the DLO into a user specified target configuration, given an initial configuration. Actuation happens in the plane such that the DLO is dragged along a surface. Robot joint angles are solved from the resulting trajectory using kinematic trajectory optimization.

At a given time, the control is applied on a specific location on the DLO. The grasp point may change throughout the trajectory, but the sequence remains fixed throughout the optimization. The control consists of a two-dimensional translational component and a rotational component as shown in Fig. 2, and involves two neighboring mass points. To reduce the dimensionality of the problem, the control is parametrized by three piecewise cubic splines and is written as $\mathbf{u}(\boldsymbol{\theta}, t) : \mathbb{R}^{4\cdot 3H} \times [0, T] \to \mathbb{R}^3$, where H is the number of subsequent splines in the trajectory and $\boldsymbol{\theta}$ contains all spline parameters. Henceforth, we use $\mathbf{u}^t$ as a shorthand for the interpolated control at time t. With "grasp" we refer to a single grasp, move, and subsequent release of the DLO.

To efficiently search a large number of grasp points we leverage a sampling based planning algorithm known in the literature as *model predictive path integral control* or MPPI. MPPI is designed to solve control-affine stochastic problems with quadratic control costs [36], [37]. MPPI has become popular for online motion planning due to its flexibility, making it suitable for applications such as autonomous driving [38] and legged locomotion [39].

For our purposes, MPPI is interesting because of its capability to parallelize, making it suitable for parallel computing architectures such as GPUs, and because of its probabilistic nature and comparatively weak requirements. MPPI requires a control affine formulation but does not rely on gradient information and therefore does not suffer from numerical issues caused by buckling in the same way as gradient based methods. MPPI is usually employed in a receding horizon fashion, but in this work we keep the horizon fixed.

However, being a sampling-based algorithm, MPPI solutions are associated with large variance depending on how the sampling parameters, specifically the variance, are chosen [36], [39]. We use a fixed sampling variance together with a refinement step using gradient-based optimization, which enables efficient local search. Gradient-based optimization is efficient for trajectory optimization and has previously shown promise in DLO manipulation tasks [16]. Hybrid sampling and gradient-based motion planning have previously been explored for manipulation planning [40] and for aerial drones [41]. A schematic of our algorithm is shown in Fig. 1 and outlined in Algorithm 1.

Algorithm 1: Combined MPPI and gradient based optimization for motion planning over different grasp sequences.

Input: S: Set of available grasp sequences
Output: X^*, U^*: State and control trajectories
$s_{\text{min}} \leftarrow \varnothing$, $c_{\text{min}} \leftarrow \infty$ `// best sequence and cost`
for s *in* S `// for each sequence`
do
  $X_{\cdot,s}, U_s, c \leftarrow mppi(\mathbf{x}_{t_0}, s)$ `// coarse planning`
  if $c \leq c_{min}$ **then**
    $c_{\text{min}} \leftarrow c$, $s_{\text{min}} \leftarrow s$ `// store best sequence`
$(X^*, U^*) \leftarrow newton(X_{s_{\text{min}}}, U_{s_{\text{min}}})$ `// refine control`
return $(X^*, \boldsymbol{\theta}^*)$

IV. Method

We write the motion planning problem as an optimal control problem. Let $X^t = \begin{bmatrix}(\mathbf{v}_u^t)^T & (\mathbf{x}_u^t)^T & (\mathbf{x}_c^t)^T\end{bmatrix}$ where $\mathbf{v}_u^t$ and $\mathbf{x}_u^t$ denote the velocities and positions of the unactuated and $\mathbf{x}_c^t$ the position of the actuated mass points respectively. The optimal control problem is then

$$\begin{aligned} \min_{U} \quad & \mathcal{J}(X, U) \\ \text{s.t.} \quad & X^{t+1} = f(X^t, \mathbf{u}^t) \; \forall \, t \in \{1, ..., T-1\} \\ & \mathbf{x}^0 = \mathbf{x}^{\text{initial}}. \end{aligned} \tag{1}$$

where X and U denote the stacked state and control vectors.

A. Dynamics

The dynamics in Eq. (1) are described by a second order ODE that establishes a relationship between mechanical forces, such as stress and friction, and mass point positions. An option for solving the ODE is the optimization based implicit Euler method where a pseudo-potential $\phi(X^{t+1}, X^t)$ is constructed such that the equations of motion hold whenever $\nabla\phi(X^{t+1}, X^t) = \mathbf{0}$. The implicit Euler method is stable also for long time steps. However, solving the forward simulation using this method includes solving linear systems and line search. Line search is difficult to implement efficiently on parallel architectures such as GPUs due to divergent control flow. Even if trajectory refinement happens on a CPU, the problem formulation must be consistent in order for the MPPI output to be an informative initial guess for the gradient based optimizer.

Therefore, we instead opt for the symplectic Euler integrator [42]. This integrator is better suited for parallel evaluation on a GPU, with the drawback that the time step length must be limited for the simulation to remain stable. Further numerical

considerations related to the integrator will be addressed in the following section. We write the dynamics as follows.

$$X^{t+1} = f(X^t, \mathbf{u}^t) = \begin{bmatrix} \left(f_s(\mathbf{x}_u^t, \mathbf{x}_c^t) + f_\mu(\mathbf{v}_u^t) + f_\alpha(\mathbf{x}_u^t, \mathbf{x}_c^t)\right)/m \\ \mathbf{v}_u^{t+1} \\ \mathbf{u}^t \end{bmatrix} \Delta t$$

where f_s denotes the stacked Hookean spring forces, f_μ the friction and f_α the bending stiffness. Previous work used viscous damping as a proxy for friction [16]. However, viscous damping lacks the step response characteristic of Coulomb friction [43]. Therefore we use a nonlinear function that better models the step response:

$$f_\mu(\mathbf{v}_u^t) = \left[-\mu\left(\langle \mathbf{v}_{u,p}^t, \mathbf{v}_{u,p}^t \rangle + d^2\right)^{-\frac{1}{2}} \mathbf{v}_{u,p}^t\right]_{p=1}^{N_m}$$

where μ is the friction coefficient and d is a coefficient that adjusts the slope of the friction response. For the bending stiffness f_α, we follow the formulation of Bergou et al. [14] where the bending stiffness is denoted by α.

B. Robust optimization

Previous work on tabletop manipulation of DLOs showed how buckling causes problems for gradient-based optimization methods [16], and also how the problem can be solved by using a mass-spring model that abstracts away compressive stress. In the symplectic integrator, buckling manifests itself as high sensitivity, or even rank deficiency in the Jacobian of the dynamics.This is not only a numerical artifact, but also a real-world phenomenon - a long, slender piece of foam that is compressed from the ends will buckle differently depending on only small changes in the compressing forces and initial conditions. When close to buckling, the system is highly sensitive to perturbations, and real-world objects are unlikely to buckle in the same way as the simulation model at all times. In addition, buckling causes numerical issues for gradient based optimization methods [16].

To address these issues, we propose optimizing over collection of systems in which the initial condition and properties of the system have been perturbed. Informally, we seek one control that yields as good a performance as possible for multiple variations of the system. When the system is in a sensitive configuration, such as buckling, small changes in the initial conditions can significantly impact the behavior of the system. A robust control, as per the notion in this paper, implies that the final configuration of the system is affected only slightly by changes in initial conditions. As a result, sensitive configurations where the behavior of the physical system is unpredictable are avoided. Extending Eq. (1) with this idea yields

$$\begin{aligned} \min_U \quad & \mathbb{E}_\xi[\mathcal{J}(X, U)] \\ \text{s.t.} \quad & X^{t+1} = f(\xi, X^t, \mathbf{u}^t) \quad \forall\, t \in \{1, \dots, T-1\} \\ & \mathbf{x}^0 = \mathbf{x}^{\text{initial}} + \xi \end{aligned}$$

where the initial and rest state of the system has been perturbed by the random variables ξ. We estimate the objective value by sampling $N_\xi - 1$ new initial configurations:

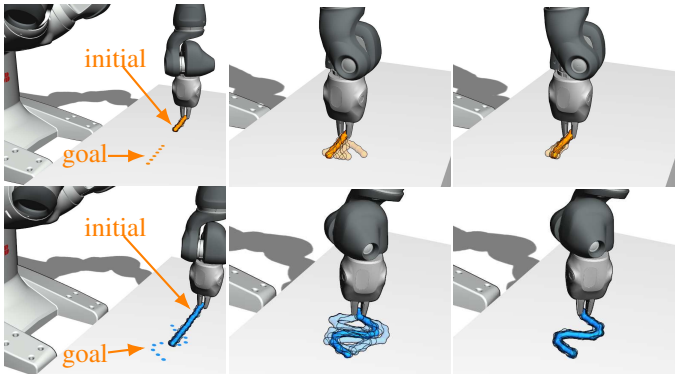


Fig. 3: A rearrangement task where the goal is to move the DLO to the dotted target configuration. Perturbed versions of the system are rendered in a transparent color. **Left**: Initial state. **Center**: Final state when optimizing without the (transparent) perturbed systems. **Right**: Final state after including the perturbed (transparent) systems in the optimization.

$$\begin{aligned} \min_U \quad & \sum_{i=1}^{N_\xi} \frac{1}{N_\xi} \mathcal{J}(X_i, U) \\ \text{s.t.} \quad & X_i^{t+1} = f(\xi_i, X_i^t, \mathbf{u}^t) \quad \forall \quad i \in \{1, ..., N_\xi\}, \\ & \qquad\qquad\qquad\quad t \in \{0, ..., T-1\} \\ & \mathbf{x}_i^0 = \mathbf{x}^{\text{initial}} + \xi_i. \end{aligned} \tag{2}$$

Perturbed initial states are constructed by sampling an angular offset for every edge in the DLO model from the normal distribution. The resting curvature is set such that the perturbed state is the minimal energy state of the model.

Figure 3 illustrates how optimizing over a set of systems impacts the robustness of the trajectory; when optimizing over a single system, the control leads to a large variance in the final configuration when applied to systems with slightly perturbed initial states or rest curvatures. Meanwhile, optimizing over sets of systems leads to smaller variance.

C. Objective

We here write the objective in two forms that are applicable in the MPPI context as well as with gradient based optimization. We write the terminal cost as

$$j_f(X) = w_{\text{goal}}(\mathbf{x}^{t_f} - \tilde{\mathbf{x}})^T(\mathbf{x}^{t_f} - \tilde{\mathbf{x}}) \tag{3}$$

where $\tilde{\mathbf{x}}$ denotes the target configuration and w_{goal} a weight parameter. The running cost consists of the regularization terms

$$j_{\text{vel}}(X, t) = w_{\text{vel}}(\mathbf{v}^t)^T\mathbf{v}^t,$$

$$j_{\text{acc}}(X, t) = w_{\text{acc}}(\mathbf{v}^t - \mathbf{v}^{t-1})^T(\mathbf{v}^t - \mathbf{v}^{t-1}),$$

$$j_{\mathbf{u}}(U, t) = w_{\mathbf{u}}(\mathbf{u}^t)^T\mathbf{u}^t.$$

In addition, we use the bending prevention term from [16] to prevent large bends between consecutive elements, since, in

our experience, large bends often lead to local minima where the DLO is entangled:

$$j_{\text{fold}}(X,t) = w_{\text{fold}} \sum_{s=1}^{n_s-1} -\left[\min\left((\mathbf{d}_s^t)^T \mathbf{d}_{s+1}^t, 0\right)\right]^3$$

where n_s is the number of springs in the DLO and $\mathbf{d}_s^t$ the unit vector in the direction of the spring s at timestep t.

The cost-to-go function can then be written as

$$C(X,U,t) = j_f(X) + \sum_{\tau=t}^{T}\big(j_{\text{vel}}(X,\tau) + j_{\text{acc}}(X,\tau) + j_{\mathbf{u}}(U,\tau) + j_{\text{fold}}(X,\tau)\big). \quad (4)$$

D. Sampling Based Optimization

MPPI is derived starting from the stochastic Hamilton-Jacobi-Bellman (HJB) equation of the continuous optimal control problem [36], [37]. After converting to a linear PDE, the HJB equation can be converted into a path integral. The gradient of this path integral with respect to the state of the system yields an expression for the optimal control.

In its discrete form, MPPI provides an iterative update law that enables estimation of the optimal control through sampling by weighing control perturbations based on how well they achieve the desired behavior. For a control problem, the update law is approximated using a limited number K of control samples for a cost-to-go function R [36], [37]:

$$\mathbf{u}^{t*} = \mathbf{u}^t + \frac{\sum_{k=1}^{K} \exp(-1/\lambda R(X, U+\delta U_k^t, t))\delta \mathbf{u}_k^t}{\sum_{k=1}^{K} \exp(-1/\lambda R(X, U^t+\delta U_k^t, t))}.$$

For simplicity, we have written the update law with respect to $\mathbf{u}^t$ instead of the spline parameters (cf. Section III).

1) Robust MPPI: The solution found by MPPI is only informative for the gradient based optimizer if both methods operate on the exact same problem. Otherwise, a grasp sequence selected from a set of candidates using MPPI might not be a good choice any longer after applying gradient based optimization. Therefore, the robustness mechanism described in Section IV-B must be included in the MPPI formulation. The cost-to-go is then computed as the sum of the cost of all perturbed systems with a given control, more precisely

$$R(X,U,t) = \frac{1}{N_\xi}\sum_{i=1}^{N_\xi} C(X_i,U,t).$$

E. Control Refinement

Both the sampling based optimizer (cf. Section IV-D) and the subsequent gradient based optimizer (this section) operate on the exact same problem in order to ensure that the initial, coarse control estimate is a useful initial guess for the subsequent trajectory refinement using gradient based optimization. This includes the robustness mechanism detailed in Section IV-B.

The gradient and the Hessian of Eq. (1) can be computed using sensitivity analysis and the sensitivity matrix $\mathbf{S} := \frac{dX}{dU}$ by writing the trajectory as a function of the control input.

TABLE I: Fixed parameters used for experiments.

MPPI parameters		Sim. parameters		Objective weights	
Samples	128	d	1×10^{-1}	w_{goal}	1×10^{2}
Iterations	30	Δt	1×10^{-3} s	$w_{\mathbf{u}}$	1
σ	1×10^{-4}	N_ξ	8	$w_{\mathbf{v}}$	1×10^{-1}
λ	1×10^{-1}			$w_{\mathbf{a}}$	1×10^{-4}
				j_{fold}	1×10^{2}

The stacked residual of the dynamics is $R = X - F(X,U)$, and assuming $R = \mathbf{0}$ everywhere implies that $\frac{d}{dU}R = 0$

$$\frac{dR}{\partial U} = \frac{\partial R}{\partial X}\frac{dX}{dU} + \frac{\partial R}{\partial U} = \mathbf{0},$$

from where $\mathbf{S}$ can be solved as

$$\mathbf{S} = -\left(\frac{\partial R}{\partial X}\right)^{-1}\frac{\partial R}{\partial U}. \quad (5)$$

The gradient and an approximate Hessian [44] of Eq. (1) can then be computed as

$$\frac{d\mathcal{J}(X(U),U)}{dU} = \frac{\partial \mathcal{J}}{\partial X}\mathbf{S} + \frac{\partial \mathcal{J}}{\partial U}$$

$$\mathbf{H} = \mathbf{S}^T\frac{\partial^2 \mathcal{J}}{\partial X^2}\mathbf{S} + 2\mathbf{S}^T\frac{\partial^2 \mathcal{J}}{\partial X \partial U} + \frac{\partial^2 \mathcal{J}}{\partial U^2},$$

which can be directly extended to Eq. (2) in order to find solutions that are robust.

V. Results

We perform experiments to demonstrate the performance, versatility and robustness according of the control trajectories generated by our formulation.

A. Robustness

We show how the robust formulation detailed in Section IV-B contributes to generating robust trajectories experimentally. We solve two tasks (cf. Fig. 3) with $N_\xi = 8$ and $N_\xi = 1$ (baseline), and compare the outcome after applying the generated controls to 32 perturbed systems that were not considered in the optimization. We study the solution generated by gradient based optimization starting from a zero guess, without the use of MPPI for warm starting the optimization. We also compare with the abstracted model from prior work [16]. Our implementation of the abstracted model from [16] includes the optimization based implicit Euler integrator, which we run with $\Delta t = 0.05$.

For the straight line (cf. Fig. 3) we use a bending stiffness of $\alpha = 1.4 \times 10^{-4}$ and a friction coefficient of $\mu = 0.5$. For the curve target we use $\alpha = 2 \times 10^{-4}$ and $\mu = 0.2$ respectively. Perturbations are generated according to Section IV-B with a standard derivation of 0.2. We record the average terminal cost $\bar{j}_f(X) = 1/N_\xi \sum_i j_f(X_i)$ and standard deviation obtained after applying the optimized control to 32 new perturbed systems in Table II. Our method yields more robust controls compared to the baseline. The abstracted model yields solutions with slightly smaller standard deviation. However, for the line example, which is perfectly symmetric, a perturbation of the initial state was necessary in order to

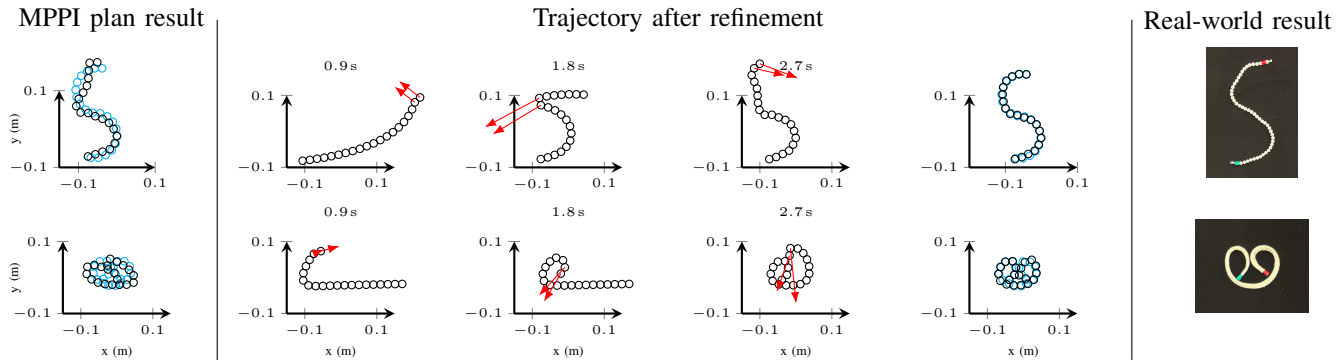


Fig. 4: **Left**: The final configuration achieved by the best ranked grasp sequence during coarse search. **Center**: Trajectory and final configuration after refining the selected grasp sequence. **Right**: Physical result.

TABLE II: Comparing the robustness of different solutions.

	Baseline		Abstracted model [16]		Ours	
	$\bar{j}_f(X)$	Avg. σ	$\bar{j}_f(X)$	Avg. σ	$\bar{j}_f(X)$	Avg. σ
Line	1.84	0.0251	0.0186*	0.00218*	0.0181	0.00391
Curve	1.29	0.0250	0.713	0.00184	0.0956	0.00723

* Required perturbing the initial state for braking symmetry.

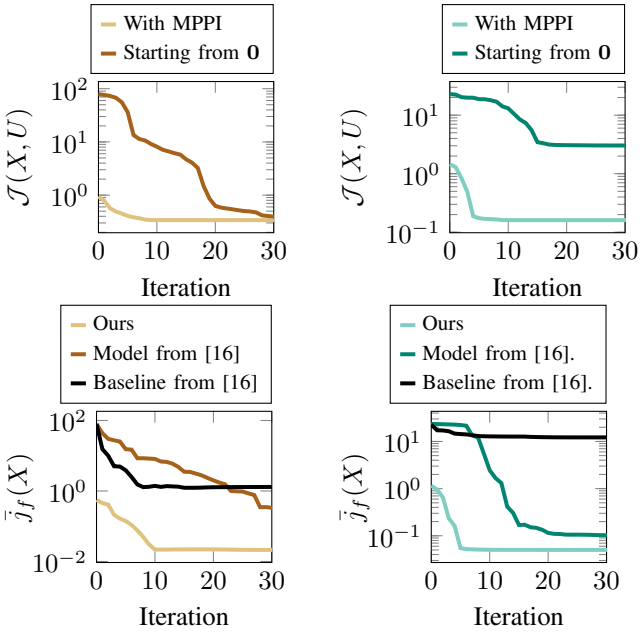
break the symmetry to enable the optimizer to make progress. For these experiments, we used a termination criterion of a gradient 2-norm of 10^{-4}. For the straight-line experiment, the gradient-based optimizer was unable to reach the termination criterion for the baseline case. Consequently, we report the controls obtained at the point where the optimizer ceased to make meaningful progress.

B. Grasp Sequence Search

We examine the runtime behavior of our algorithm. We define four potential grasp points distributed evenly along the DLO, including the ends points. A grasp sequence consists of three subsequent grasps where the grasp duration is fixed to 1 s. This yields $4^3 = 64$ different potential grasp sequences. The DLO model consists of 20 subsequently connected mass points with a rest distance of 0.02 m between each pair. Fixed parameters of MPPI, simulation and objective weights are listed in Table I. The MPPI component of the algorithm is implemented in SYCL using AdaptiveCpp 24.10.0 [45].

Figure 4 shows the output of various stages of the algorithm for shapes resembling the letter S and a pretzel. We measure a runtime of 82.4 s and 104.9 s for the S and the pretzel respectively. This includes coarse planning of all 64 grasp point sequences for approximately 14 s, and trajectory refinement until a gradient 2-norm of 10^{-3}. For comparison, finding the control for the same grasp sequence using gradient based optimization starting from a zero initial guess takes 186.7 s for the S-shape. For the pretzel shape, finding a control solely using gradient based optimization did not succeed in our experiments as the gradient based optimizer would fail to make progress. The runtime was measured on an AMD Ryzen 7900X equipped desktop computer featuring an NVIDIA RTX 4090 GPU accelerator.

Figure 5 shows how warm starting the gradient based optimization with the MPPI solution accelerates convergence



(a) "S"-shape (b) "Pretzel"-shape

Fig. 5: The cost of the MPPI trajectory is further improved using gradient based optimization. Warm starting gradient based optimization from the MPPI solution yields faster convergence. Compared to previous methods, our method shows faster convergence towards the target shape.

compared to a cold start from a zero control. Performing coarse planning using MPPI thus has several benefits. Firstly, thanks to the parallelizability, it allows us to choose a grasp sequence efficiently. Secondly, it allows us to find a control that is likely (though not guaranteed) to be stable and free from buckling, and thirdly, the refinement of the trajectory corresponding to the selected grasp sequence becomes more efficient thanks to the initial guess that MPPI provides. Our method also shows improved performance over a prior formulation (and baseline including compression prevention term) from Envall et al. [16]. For the pretzel shape, the baseline method of [16] ended up in an unfortunate local minimum that yielded a poor target match.

As part of the experiments, we execute the trajectories on

an ABB YuMi robot. The initial DLO configuration and the location of the grasp points throughout the experiments are estimated using an Intel RealSense D435i combined depth and RGB camera.

We determine the bending stiffness and friction coefficient by conducting a preliminary robot experiment and then iteratively refine the model parameters until they align with the real world behavior. For the S-shape we used a pearl band and a bending stiffness and friction coefficient of $\alpha = 1 \times 10^{-5}$ and $\mu = 0.3$ and for the pretzel a cotton rope and $\alpha = 1 \times 10^{-6}$ and $\mu = 0.5$.

C. Stiff Materials

Finally, we demonstrate how our model enables manipulation of DLOs with large bending stiffness in a bimanual setting where the DLO is grasped from the ends. We use a thick rubber rope modeled with a bending stiffness of $\alpha = 10^{-2}$ and a steel wire with $\alpha = 1.2 \times 10^{-2}$ and a friction coefficient of $\mu = 0.5$ in both cases. The obtained trajectories and real-world results are shown in Fig. 6. Both trajectories feature controls that compress the DLO, which prior methods do not allow [16].

VI. Discussion, Limitations and Conclusion

Our experiments show that our new formulation is useful for efficiently selecting a contact sequence among a number of options and that the combined optimization formulation and simulation model enables robust manipulation planning for a wide range of materials.

The MPPI algorithm enables coarse evaluation and ranking of a large number of contact sequences in parallel, with the drawback of often leaving the simulated DLO far from the desired target configuration. Although optimizing over several perturbed systems is likely to lead to a local minimum that is free from buckling, no guarantees exist, and the subsequent gradient based optimization might still fail. In practice we encountered few such cases, but we saw several cases where gradient based optimization from a zero initial guess without MPPI warm starting would fail due to incorrect gradients, one of them being the pretzel shape demonstrated in Section V. This shows that combining sampling and gradient based optimization is a worthwhile direction for manipulation planning of deformable linear objects.

The two optimization methods used in our algorithm operate on the exact same problem in order to ensure that the initial, coarse control estimate is a useful initial guess for the subsequent trajectory refinement using gradient based optimization. However, since the sampling based planner operates directly in the space of the DLO, we might obtain trajectories that are impossible (or near impossible) to execute on a robot due to kinematic constraints. This issue could be mitigated in part by restricting the actuation domain in the problem formulation using barrier terms in the objective.

Our implementation uses the symplectic Euler integrator as it maps well to GPU hardware. However, simulating deformable objects with this method requires comparatively small time steps, which leads to large memory consumption. The large memory consumption limits the trajectory length and the number of grasps that can be simulated before GPU memory is exhausted. More advanced solution techniques could potentially enable a longer time step length and reduce the memory usage.

Our formulation is limited to two-dimensional manipulation in the plane. However, we hope that the ideas presented here can be extended into a more general formulation, enabling sheet and textile manipulation in three dimensions. In a longer perspective, we hope that this will provide improved convenience and efficiency in domestic and industrial environments.

VII. Acknowledgments

The authors thank Yijiang Huang, Valentin N. Hartmann, and the anonymous reviewers for their valuable insights, suggestions, and feedback, which significantly improved the clarity and quality of this paper.

References

[1] E. Riks, "The application of newton's method to the problem of elastic stability," *Journal of Applied Mechanics*, vol. 39, no. 4, pp. 1060–1065, 12 1972.

[2] R. Seydel, *Practical bifurcation and stability analysis*. Springer New York, NY, 2009, vol. 5.

[3] K.-J. Choi and H.-S. Ko, "Stable but responsive cloth," *ACM Transactions on Graphics (ToG)*, vol. 21, no. 3, p. 604–611, July 2002.

[4] H. Yin, A. Varava, and D. Kragic, "Modeling, learning, perception, and control methods for deformable object manipulation," *Science Robotics*, vol. 6, no. 54, 2021, eabd8803.

[5] M. Moll and L. E. Kavraki, "Path planning for deformable linear objects," *IEEE Transactions on Robotics*, vol. 22, no. 4, pp. 625–636, 2006.

[6] T. Bretl and Z. McCarthy, "Quasi-static manipulation of a kirchhoff elastic rod based on a geometric analysis of equilibrium configurations," *The International Journal of Robotics Research*, vol. 33, no. 1, pp. 48–68, 2014.

[7] M. Yu, K. Lv, C. Wang, M. Tomizuka, and X. Li, "A coarse-to-fine framework for dual-arm manipulation of deformable linear objects with whole-body obstacle avoidance," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 10 153–10 159.

[8] F. Liu, E. Su, J. Lu, M. Li, and M. C. Yip, "Robotic manipulation of deformable rope-like objects using differentiable compliant position-based dynamics," *IEEE Robotics and Automation Letters*, vol. 8, no. 7, pp. 3964–3971, 2023.

[9] Y. Zhang, F. Liu, X. Liang, and M. Yip, "Achieving autonomous cloth manipulation with optimal control via differentiable physics-aware regularization and safety constraints," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 9931–9938.

[10] S. Duenser, J. M. Bern, R. Poranne, and S. Coros, "Interactive robotic manipulation of elastic objects," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 3476–3481.

[11] S. Duenser, R. Poranne, B. Thomaszewski, and S. Coros, "Robocut: Hot-wire cutting with robot-controlled flexible rods," *ACM Transactions on Graphics (ToG)*, vol. 39, no. 4, 2020, article 98, 15 pages.

[12] S. Zimmermann, R. Poranne, and S. Coros, "Dynamic manipulation of deformable objects with implicit integration," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 4209–4216, 2021.

[13] S. Wang, M. Leonetti, and M. Dogar, "Goal-conditioned model simplification for 1-d and 2-d deformable object manipulation," *IEEE Transactions on Robotics*, vol. 41, pp. 4023–4040, 2025.

[14] M. Bergou, M. Wardetzky, S. Robinson, B. Audoly, and E. Grinspun, "Discrete elastic rods," *ACM Transactions on Graphics*, vol. 27, no. 3, p. 1–12, 2008.

[15] J. Teran, E. Sifakis, G. Irving, and R. Fedkiw, "Robust quasistatic finite elements and flesh simulation," in *Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation*, 2005, pp. 181–190.

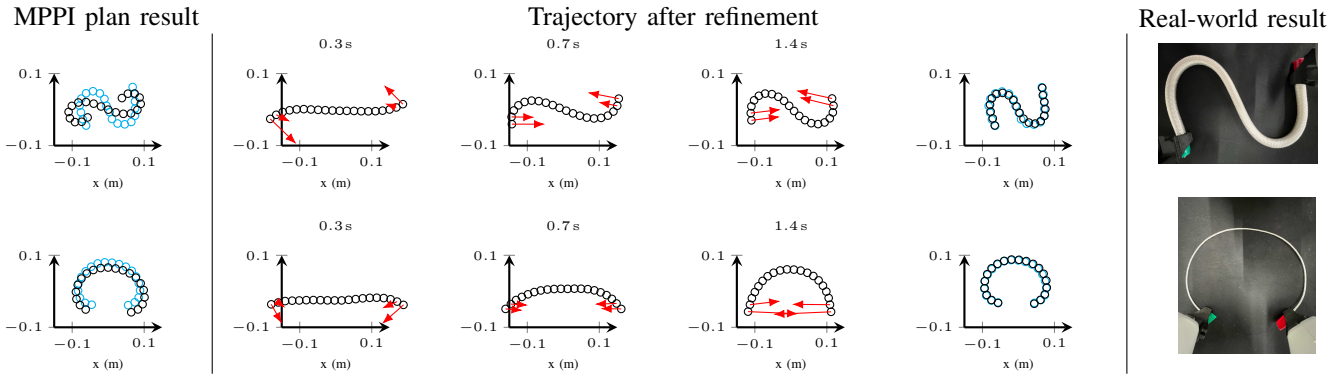


Fig. 6: Our formulation works in a bimanual setting for DLOs with large bending stiffness. **Left**: The final configuration of the MPPI generated trajectory. **Center**: Trajectory and final configuration after refining. **Right**: Physical result.

[16] J. Envall, B. Thomaszewski, and S. Coros, "Understanding the impact of modeling abstractions on motion planning for deformable linear objects," in *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2025, pp. 21 741–21 748.

[17] J. Jankowski, L. Brudermüller, N. Hawes, and S. Calinon, "Robust pushing: Exploiting quasi-static belief dynamics and contact-informed optimization," *International Journal of Robotics Research (IJRR)*, vol. 44, no. 12, pp. 1959–1980, 2025.

[18] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Sim-to-real transfer of robotic control with dynamics randomization," in *2018 IEEE international conference on robotics and automation (ICRA)*, 2018, pp. 3803–3810.

[19] A. Loquercio, E. Kaufmann, R. Ranftl, A. Dosovitskiy, V. Koltun, and D. Scaramuzza, "Deep drone racing: From simulation to reality with domain randomization," *IEEE Transactions on Robotics*, vol. 36, no. 1, pp. 1–14, 2019.

[20] T. Miki, J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning robust perceptive locomotion for quadrupedal robots in the wild," *Science robotics*, vol. 7, no. 62, 2022, eabk2822.

[21] M. McNaughton, "Castro: robust nonlinear trajectory optimization using multiple models," in *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2007, pp. 177–182.

[22] C. Chi, B. Burchfiel, E. Cousineau, S. Feng, and S. Song, "Iterative residual policy: for goal-conditioned dynamic manipulation of deformable objects," *The International Journal of Robotics Research*, vol. 43, no. 4, pp. 389–404, 2024.

[23] A. Nair, D. Chen, P. Agrawal, P. Isola, P. Abbeel, J. Malik, and S. Levine, "Combining self-supervised learning and imitation for vision-based rope manipulation," in *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2017, pp. 2146–2153.

[24] G. Salhotra, I.-C. A. Liu, M. Dominguez-Kuhne, and G. S. Sukhatme, "Learning deformable object manipulation from expert demonstrations," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 8775–8782, 2022.

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

[26] J. Luo, C. Xu, X. Geng, G. Feng, K. Fang, L. Tan, S. Schaal, and S. Levine, "Multistage cable routing through hierarchical imitation learning," *IEEE Transactions on Robotics*, vol. 40, pp. 1476–1491, 2024.

[27] J. Carius, R. Ranftl, V. Koltun, and M. Hutter, "Trajectory optimization with implicit hard contacts," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3316–3323, 2018.

[28] I. Chatzinikolaidis, Y. You, and Z. Li, "Contact-implicit trajectory optimization using an analytically solvable contact model for locomotion on variable ground," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6357–6364, 2020.

[29] I. Mordatch, E. Todorov, and Z. Popović, "Discovery of complex behaviors through contact-invariant optimization," *ACM Transactions on Graphics (ToG)*, vol. 31, no. 4, 2012, article 43, 8 pages.

[30] V. Kurtz, A. Castro, A. Özgün Önol, and H. Lin, "Inverse dynamics trajectory optimization for contact-implicit model predictive control," *The International Journal of Robotics Research (IJRR)*, vol. 45, no. 1, pp. 23–40, 2026.

[31] T. Pang, H. T. Suh, L. Yang, and R. Tedrake, "Global planning for contact-rich manipulation via local smoothing of quasi-dynamic contact models," *IEEE Transactions on Robotics*, vol. 39, no. 6, pp. 4691–4711, 2023.

[32] M. Toussaint, "Logic-geometric programming: An optimization-based approach to combined task and motion planning," in *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI)*, 2015, pp. 1930–1936.

[33] M. Toussaint and M. Lopes, "Multi-bound tree search for logic-geometric programming in cooperative manipulation domains," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 4044–4051.

[34] M. Toussaint, K. Allen, K. Smith, and J. Tenenbaum, "Differentiable physics and stable modes for tool-use and manipulation planning," in *Proceedings of Robotics: Science and Systems*, Pittsburgh, Pennsylvania, June 2018.

[35] C. R. Garrett, R. Chitnis, R. Holladay, B. Kim, T. Silver, L. P. Kaelbling, and T. Lozano-Pérez, "Integrated task and motion planning," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 4, pp. 265–293, 2021.

[36] G. Williams, A. Aldrich, and E. A. Theodorou, "Model predictive path integral control: From theory to parallel computation," *Journal of Guidance, Control, and Dynamics*, vol. 40, no. 2, pp. 344–357, 2017.

[37] M. Kazim, J. Hong, M.-G. Kim, and K.-K. K. Kim, "Recent advances in path integral control for trajectory optimization: An overview in theoretical and algorithmic perspectives," *Annual Reviews in Control*, vol. 57, 2024, 100931.

[38] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, and E. A. Theodorou, "Aggressive driving with model predictive path integral control," in *2016 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2016, pp. 1433–1440.

[39] H. Xue, C. Pan, Z. Yi, G. Qu, and G. Shi, "Full-order sampling-based mpc for torque-level locomotion control via diffusion-style annealing," in *2025 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2025, pp. 4974–4981.

[40] F. Rozzi, L. Roveda, and K. Haninger, "Combining sampling- and gradient-based planning for contact-rich manipulation," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2024, pp. 9901–9907.

[41] L. Campos-Macías, D. Gómez-Gutiérrez, R. Aldana-López, R. de la Guardia, and J. I. Parra-Vilchis, "A hybrid method for online trajectory planning of mobile robots in cluttered environments," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 935–942, 2017.

[42] D. F. Griffiths and D. J. Higham, *Numerical methods for ordinary differential equations: initial value problems*. Springer London, 2010.

[43] P. Wriggers, *Computational Contact Mechanics*, 2nd ed. Springer Berlin, Heidelberg, 2006.

[44] S. Zimmermann, R. Poranne, and S. Coros, "Optimal control via second order sensitivity analysis," *arXiv preprint arXiv:1905.08534*, 2019.

[45] A. Alpay and V. Heuveline, "One pass to bind them: The first single-pass sycl compiler with unified code representation across backends," in *Proceedings of the 2023 International Workshop on OpenCL*, 2023, pp. 1–12, article 7.