Shape Control of Elastoplastic Deformable Linear Objects through Reinforcement Learning

Rita Laezza[1] and Yiannis Karayiannidis[1]

***Abstract*— Deformable object manipulation tasks have long been regarded as challenging robotic problems. However, until recently, very little work had been done on the subject, with most robotic manipulation methods being developed for rigid objects. As machine learning methods are becoming more powerful, there are new model-free strategies to explore for these objects, which are notoriously hard to model. This paper focuses on shape control problems for Deformable Linear Objects (DLOs). Despite being one of the most researched classes of DLOs in terms of geometry, no other paper has focused on materials with elastoplastic properties. Therefore, a novel shape control task, requiring permanent plastic deformation is implemented in a simulation environment. Reinforcement Learning methods are used to learn a continuous control policy. To that end, a discrete curvature measure is used as a low-dimensional state representation and as part of an intuitive reward function. Finally, three state-of-the-art actor-critic algorithms are compared on the proposed environment and successfully achieve the goal shape.**

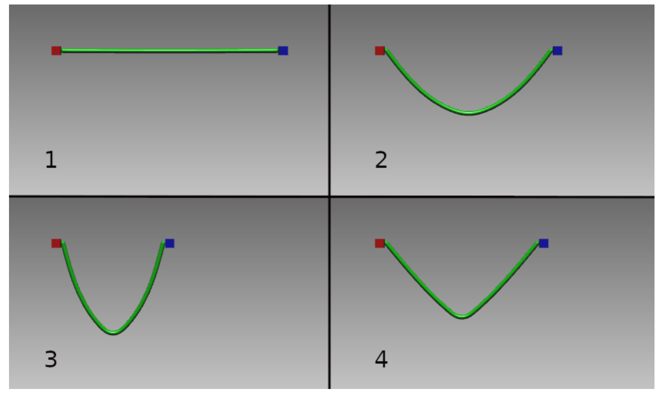

Fig. 1. Simulation of DLO with plastic properties. By bending the object inwards in 1-3, when returning the gripper to the same position as before i.e. gripper is in back in the same place as in 2, the shape of the DLO is different due to permanent plastic deformation.

I. INTRODUCTION

In recent years, there has been a growing interest in deformable object grasping and manipulation problems by the robotics community [1], [2]. This is due in part to their prevalence in many diverse applications as well as their complexity, which makes these problems challenging to solve with classical approaches [1]. Consequently, learning-based approaches are being favored as a more powerful alternative [2]. Intuitively, if a robot is to reach human-level dexterity, there may be a need for human-inspired learning. Reinforcement Learning (RL) consists of a particularly promising group of methods which seek to make robots capable of learning through experience [3]. RL has been proven successful in solving complex games, such as Go, as well as robotic control tasks [4].

Contrary to grasping and manipulation of rigid objects, which have been extensively addressed in the robotics literature, non-rigid objects have been largely overlooked [1]. Though some of the same methods can be extended to particular types of deformable objects, there are still many problems left unsolved [1]. Most notably, while manipulation of rigid objects focuses mainly on controlling their pose, when manipulating deformable objects it is often their shape which needs to be controlled [2]. Furthermore, when materials are highly deformable have with elastoplastic properties, modeling and sensing of these objects presents a difficult challenge.

Most work on deformable object manipulation has focused on specialized tasks, from applications like robotic surgery, food processing, fabric manufacturing, etc. While this is the most practical choice, since they aim to solve real-life problems, the solutions are often not general [5]. With this work we aim to lay a foundation for shape control, which could potentially be applied to a large range of problems. According to classification criteria suggested by Sanchez et al. [1], objects can be categorized based on their mechanical properties, i.e. low or high compression strength and their geometric properties, i.e. linear, planar or solid shapes. We focus on a subclass of deformable objects within each category, namely Deformable Linear Objects (DLOs) with elastoplastic properties.

DLOs are an appealing choice for their relative geometric simplicity without loss of manipulation complexity. Within this class, we found that objects with elastoplastic properties have been largely overlooked, with most of the literature focusing on purely elastic DLOs or ropes [1]. This large class of objects, includes metal wires and cables found in numerous applications [5]. To that end, we have implemented a simulation environment with a set of shape control problems for DLOs which sustain plastic deformation. These present a particularly difficult class of objects since they exhibit non-linear behavior, suffering irreversible changes in their mechanical properties, as illustrated in Figure 1. Our interest is not on the modeling accuracy of the simulation, but on the

[1] Division of Systems and Control, Department of Electrical Engineering, Chalmers University of Technology, Sweden {laezza , yiannis}@chalmers.se

Workshop on Robotic Manipulation of Deformable Objects (ROMADO)
IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) 2020

ability to learn how to manipulate these objects in a model-free fashion using actor-critic methods. Specifically, Deep Deterministic Policy Gradient (DDPG) [6], Twin Delayed DDPG (TD3) [7] and Soft Actor-Critic (SAC) [8].

Finally, summarized below are our main contributions within the field of deformable object manipulation:

i. An RL formulation of a shape control problem for elastoplastic DLOs using curvature state representation.
ii. A new simulation environment using a multi-physics engine with specialized DLO models.
iii. Comparison of state-of-the-art actor-critic algorithms for our simulation environment.

II. RELATED WORK

To date, ropes or rope-like objects are the most researched group of DLOs in robotic manipulation. Common problems involving ropes include knot tying, untangling, threading and reaching goal-configurations on a flat surface [1]. While all of these present interesting challenges, only the latter represents a true shape control problem. For the rest, what matters is not the final geometric deformation, but the configuration of the DLO, relative to itself, or other objects. Recent research on shape control problems with DLOs seems to focus on the state-estimation problem. Zhu et al. [5] used Fourier series to model the DLO shape and successfully controlled cables with low compression strength into desired deformations, using a dual-armed robot. Similarly, Yan et al. [9] used self-supervised learning to learn the state of ropes in an analogous scenario, but with a single-arm approach, thus requiring re-grasping.

While to the best of our knowledge there has been no work where large plastic deformations were considered for DLOs, Cherubini et al. [10] addressed the manipulation of plastic materials, using kinetic sand as a test case. To solve this task, they relied on human demonstrations, further providing a dataset for comparative studies and benchmarking.

Most robotic tasks related to DLOs usually require both sensing and manipulation. In the aforementioned works, sensing of the deformable objects relied at least partially on vision measurements. This is a natural choice since force-torque readings are not sufficient to identify the state of a deformable object. From image data, there have been different approaches to represent and estimate a DLO's state, including node-graphs, Frenet coordinate frames, Kirchoff elastic rods, etc. [1], [11].

Using deep learning techniques has opened up the possibility to learn directly from the high-dimensional raw data. This can be used in end-to-end strategies, where robot joint velocities are obtained directly from pixels. An example of such an approach was the work from Matas et al. [12], which produced promising results in cloth-manipulation using a state-of-the-art RL algorithm. Their work also proved successful in sim-to-real transfer. However, they used a variation of Deep Deterministic Policy Gradient, named DDPGfD which was seeded by employing LfD. Conversely, Wu et al. [13] proposed to solve pick-and-place tasks without demonstrations.

III. METHODS

Although our aim is to implement robot learning in real-life scenarios, it can be a laborious and time consuming endeavor with RL algorithms. This is especially true when learning from scratch and using model-free methods, which are notoriously sample inefficient [8]. Due to their need for extensive experience before convergence, it is common-practice to tackle problems first in simulation. With this in mind, we have defined a DLO shape control problem in a virtual environment, to evaluate the potential of these methods for deformable object manipulation.

A. Problem Statement

We propose a shape control problem of an elastoplastic DLO, on a vertical plane. While scenarios involving ropes or flexible cables typically have the object lying on a surface, here we consider two grippers holding a DLO in free space. Therefore, the limitation to a planar motion is induced solely by the gripper configuration, and not by any other object such as a table top.

From this configuration, we can formulate a variety of control tasks with increasing degrees of difficulty. For instance, both the linear and angular velocities of the gripper, together with the compliance of the grip (i.e. how much the grasped object can rotate around the gripping point) can be controlled. However, in this work a single control input is considered, namely the linear velocity of the gripper along a fixed axis, aligned with the DLO. This implies that the gripper is constrained to move along a straight line. Furthermore, to produce smoother deformations, the DLO is grasped with a passive compliant grip.

The proposed task is to control a single gripping point on the DLO in order to reach a goal shape of the entire object. This can be viewed as controlling an under-actuated continuum robot, where only the first joint is actuated. To generate the goal shape, a sinusoidal trajectory is used, so that plastic deformation occurs as seen in Figure 1. In a way, this resembles the classical RL problem of an under-powered car in a valley, climbing a mountain. Both problems require moving in a direction which is farther from the desired goal, in order to be successful.

Finally, it is important to note that grasping is not a part of this work. For our simulations, a perfect grasp is assumed, by reducing the interaction between robot and DLO to a single control point on the object. This implies there is no relative displacement between the controlled point and the robot gripper, e.g. by slippage.

B. Modeling and Simulation

When choosing a multi-physics engine, there are many factors to consider, such as accuracy, speed and development

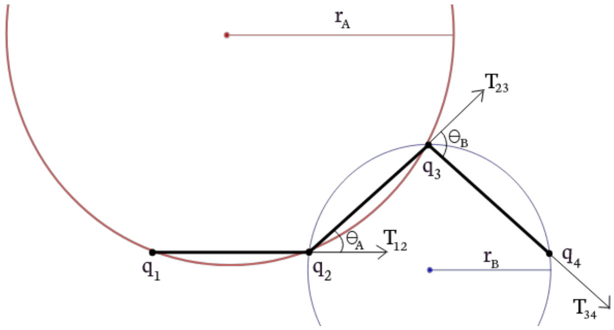

Fig. 2. Discrete curve described by points q_i, tangent vectors T_{ij} and angles θ_a. Two osculating circles are illustrated to show the inverse relationship between radius r and curvature κ.

time. The robotics and classical control environments available in Gym [14] were implemented using MuJoCo (Multi-Joint dynamics with Contact) [15]. This proprietary software seems to be the Reinforcement Learning community's predominant choice. We opted for AGX Dynamics, another commercial engine which provides specialized models for different classes of DLOs [16], [17], and great documentation thus enabling shorter development times.

To simulate our problem we approximate the gripping points on the DLO as being attached on each extremity to a rigid object, by a `Hinge` constraint. This removes 5 Degrees of Freedom (DoF) from the relative motion between the object and the DLO, only allowing for movement around the hinge center. Further, varying the constraint compliance makes rotation around its axis more or less stiff, which is analogous to having more or less friction between the DLO and the gripper. We use `Prismatic` constraints, between each rigid object and the initial position of the other rigid body, when the DLO is undeformed. This constrains the motion of the controlled gripper to translation along one dimension. Thus, the gripping points of the DLO move along 2 DoF, with only one of them actuated.

Finally, we have modeled the state of a DLO by its curvature. This assumes that there is an object tracking algorithm in place, to obtain an approximation of the DLO's state in Euclidean space, as a set of discrete points, $q = (x, y, z) \in \mathbb{R}^3$. However, in simulation, the exact coordinates can be used to compute the discrete curvature.

For any discrete curve with $N \geq 3$ points, it is possible to compute its curvature through the circumscribed osculating circle, as shown in Figure 2. Since the curvature of a circle with radius $r > 0$, is defined as $\kappa = 1/r$, the curvature can be computed as:

$$\kappa_a = \frac{2}{l} \tan \frac{\theta_a}{2} \tag{1}$$

where l is the segment length, and θ is the angle between tangent vectors of two consecutive segments. It is assumed that all segments have equal length, and that $\theta_a \in [0, \frac{\pi}{2}]$ with $a = 1, \ldots, N-1$. Thus, for each pair of adjacent points, the tangent vector can be obtained:

$$T_{ij} = \overrightarrow{q_i q_j} = (x_j - x_i, y_j - y_i, z_j - z_i),\ j := i + 1 \tag{2}$$

for all q_i with $i = 1, \ldots, N$. Further, for each pair of consecutive tangent vectors, the angle can be computed,

$$\theta_a = \arccos\left(\frac{T_{ij} \times T_{jk}}{||T_{ij}||||T_{jk}||}\right), \quad k := j + 1 \tag{3}$$

which is enough to approximate the discrete curvature κ_a between each segment, as in equation (1), since l is defined in the simulation [18].

C. Reinforcement Learning

In RL, control problems are framed as Markov Decision Processes (MDPs). We consider an infinite-horizon discounted MDP, defined as a tuple $(\mathcal{S}, \mathcal{A}, p, r, \gamma)$, where γ is the discount factor and $\mathcal{S}$ and $\mathcal{A}$ are continuous state and action spaces, respectively. In real-life problems this MDP is unknown since the probability density function, $p : \mathcal{S} \times \mathcal{S} \times \mathcal{A} \to [0, \infty)$, depends on an environment which cannot be accurately modeled. This function represents the probability of transitioning to state s_{t+1}, given the current state s_t and action a_t, with $s_t, s_{t+1} \in \mathcal{S}$ and $a_t \in \mathcal{A}$. Further, in practical applications, the reward function $r : \mathcal{S} \times \mathcal{A} \to [r_{min}, r_{max}]$, is defined based on the desired task. To provide a measure of expected success, the return at time t is defined as the sum of discounted future rewards: $R_t = \sum_{i=t}^{+\infty} \gamma^{i-t} r(s_i, a_i)$.

Policy gradient methods update parameters ϕ, by taking the gradient of the expected return. Note that the learned policy can be both deterministic or stochastic. DDPG, TD3 and SAC all learn a deterministic policy, modeled as a Neural Network. Therefore, policy parameters can be updated using the deterministic policy gradient theorem [19]:

$$\nabla_\phi J(\phi) = \mathbb{E}_{s \sim p_\pi}\left[\nabla_a Q^\pi(s, a)|_{a=\pi(s)} \nabla_\phi \pi_\phi(s)\right] \tag{4}$$

where $Q^\pi(s, a) = \mathbb{E}_{s_i \sim p_\pi, a_i \sim \pi}[R_t | s, a]$, is the expected return when taking action a in state s, and following policy π after that.

To evaluate the success of these methods on the problem formulated in III-A, we need to define the states s, actions a and the reward r. The state can be defined as a one-dimensional vector of the discrete curvature, concatenated with the forces and torques measured on the controlled gripper, $s = [\boldsymbol{\kappa}, \boldsymbol{F}, \boldsymbol{\tau}]$, with $\boldsymbol{\kappa} = [\kappa_a, \ldots, \kappa_{N-1}]$ and $\boldsymbol{F}, \boldsymbol{\tau} \in \mathbb{R}^3$. The actions provided by the learned policy are clipped such that $a \in [-1, 1]$. Finally, the reward function is defined in equation (5), based on the L2-norm of the distance between the current curvature, $\boldsymbol{\kappa}_t$ and the goal curvature, $\bar{\boldsymbol{\kappa}}$: $L_t(\boldsymbol{\kappa}_t) = ||\boldsymbol{\kappa}_t - \bar{\boldsymbol{\kappa}}||$. Reward values are also clipped such that $r \in [-1.5, 1.5]$. In addition, the threshold value depends on the complexity of the goal shape, and thus needed to be tuned. For our experiments it is set to 0.06.

$$r_t = \begin{cases} -L_t(\boldsymbol{\kappa}_t), & \text{if } L_t(\boldsymbol{\kappa}_t) \geq \text{threshold} \\ r_{\max} - |a_i|, & \text{otherwise} \end{cases} \tag{5}$$

where the second term is meant to encourage the RL agent to maintain zero velocity, once the desired shape is achieved.

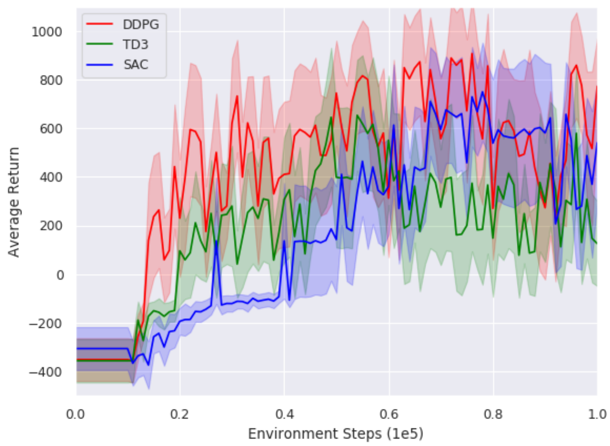

Fig. 3. Training curves for DDPG, TD3 and SAC algorithms. The shaded region represents half a standard deviation of the average evaluation over 5 trials. First 10^4 steps without updates (`min_steps_learn`).

IV. EXPERIMENTAL RESULTS

For our experiments, the open source rlpyt [20] codebase is used, which provides modular implementations of major RL algorithms. All default hyper-parameters provided by this library were used to obtain our results, since they corresponded to the values from the original papers.

To build a realistic simulation the DLO is modeled as an Aluminum wire, which is 10cm long and has a radius of 1mm. Being a lumped-element model, the resolution of `Cable` needs to be defined, and is arbitrarily set to 1000 segments per meter. Furthermore, a maximum velocity is enforced by limiting the force range of the controlled `motor` to $[-2.5, 2.5]$N. The simulation time-step was set to 0.02s, while actions were applied every second step leading to a control frequency of 25Hz.

For each algorithm, 5 trials are performed and the results averaged, as seen in Figure 3. From the obtained results, all algorithms seem to converge to similar average return levels, although DDPG is the fastest to reach higher values. Since all hyper-parameters were left as in the original papers, a thorough hyper-parameter search needs to be done to reach more conclusive results. What is relevant here is that all methods seem to reach a solution, with comparable success. The learned policy reaches the desired threshold and then stops, sustaining zero velocity until the end of the episode. All algorithms reach similar results with the maximum return across trials and iterations, more specifically 1172 by DDPG, 1167 by TD3 and 1165 by SAC.

V. CONCLUDING REMARKS

In this work we have presented a new shape control problem for elastoplastic DLOs. We proposed a state representation based on curvature, which is independent from the object position, thus making it better suited for shape control tasks. To test this representation, a specialized simulation environment was implemented. Three different RL algorithms were used to evaluate the potential of our state representation. Results show that all actor-critic methods learn a velocity trajectory which achieves the desired shape, within the established threshold. This indicates that our approach may be extended to more complex problems.

REFERENCES

[1] J. Sanchez, J.-A. Corrales, B.-C. Bouzgarrou, and Y. Mezouar, "Robotic manipulation and sensing of deformable objects in domestic and industrial applications: a survey," *The International Journal of Robotics Research*, vol. 37, no. 7, pp. 688–716, 2018.

[2] F. F. Khalil and P. Payeur, "Dexterous robotic manipulation of deformable objects with multi-sensory feedback-a review," in *Robot Manipulators Trends and Development*, IntechOpen, 2010.

[3] J. Kober, J. A. Bagnell, and J. Peters, "Reinforcement learning in robotics: A survey," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1238–1274, 2013.

[4] M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, O. P. Abbeel, and W. Zaremba, "Hindsight experience replay," in *Advances in neural information processing systems*, pp. 5048–5058, 2017.

[5] J. Zhu, B. Navarro, P. Fraisse, A. Crosnier, and A. Cherubini, "Dual-arm robotic manipulation of flexible cables," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 479–484, IEEE, 2018.

[6] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[7] S. Fujimoto, H. Van Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," *arXiv preprint arXiv:1802.09477*, 2018.

[8] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," *arXiv preprint arXiv:1801.01290*, 2018.

[9] M. Yan, Y. Zhu, N. Jin, and J. Bohg, "Self-supervised learning of state estimation for manipulating deformable linear objects," *IEEE Robotics and Automation Letters*, 2020.

[10] A. Cherubini, J. Leitner, V. Ortenzi, and P. Corke, "Towards vision-based manipulation of plastic materials," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 485–490, IEEE, 2018.

[11] H. Wakamatsu and S. Hirai, "Static modeling of linear object deformation based on differential geometry," *The International Journal of Robotics Research*, vol. 23, no. 3, pp. 293–311, 2004.

[12] J. Matas, S. James, and A. J. Davison, "Sim-to-real reinforcement learning for deformable object manipulation," *arXiv preprint arXiv:1806.07851*, 2018.

[13] Y. Wu, W. Yan, T. Kurutach, L. Pinto, and P. Abbeel, "Learning to manipulate deformable objects without demonstrations," *arXiv preprint arXiv:1910.13439*, 2019.

[14] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," 2016.

[15] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033, 2012.

[16] M. Servin and C. Lacoursière, "Rigid body cable for virtual environments," *IEEE Transactions on Visualization and Computer Graphics*, vol. 14, no. 4, pp. 783–796, 2008.

[17] M. Servin, C. Lacoursiere, and N. Melin, "Interactive simulation of elastic deformable materials," in *SIGRAD 2006. The Annual SIGRAD Conference; Special Theme: Computer Games*, Linköping University Electronic Press, 2006.

[18] D. Carroll, E. Hankins, E. Köse, and I. Sterling, "A survey of the differential geometry of discrete curves," *arXiv preprint arXiv:1311.5862*, 2013.

[19] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," in *ICML*, 2014.

[20] A. Stooke and P. Abbeel, "rlpyt: A research code base for deep reinforcement learning in pytorch," *arXiv preprint arXiv:1909.01500*, 2019.