Presenting *ReForm*, a Robot Learning Sandbox for Deformable Linear Object Manipulation

Rita Laezza[1], Robert Gieselmann[2], Florian T. Pokorny[2] and Yiannis Karayiannidis[1]

***Abstract*— Recent success of Deep Reinforcement Learning in games has sparked interest to extend such methods to robotic applications. However, since these require extensive amounts of data to be successful, it is challenging to gather enough real-world experience to make them viable in robotics. In an attempt to solve this problem, several simulation environments have been created to allow robots to learn in simulation, thus reducing the time and cost of such an endeavor. *ReForm* is our own contribution to the growing number of such resources. At the moment of its conception, there was no other resource which focused on deformable object manipulation tasks. This has recently changed with the release of SoftGym, which includes cloth, rope and liquid simulations and PlasticineLab, which is targeted at Plasticine shaping tasks. *ReForm* on the other hand, focuses exclusively on Deformable Linear Objects (DLOs), with an emphasis of realistic mechanical properties, ranging from low compression strength ropes, elastoplastic metal cables, to purely elastic rubber bands. The motivation to focus on DLOs comes from their widespread across numerous application areas, such as medical (e.g. suturing), industrial (e.g. electrical wiring) and service robotics (e.g. household cables). *ReForm*, is intended as a simulation sandbox and a tool for benchmarking manipulation of DLOs, with six distinct customizable environments. The implementation is modular and provides interfaces to change parameters such as end-effector degrees of freedom, type of observation and reward function.**

I. Introduction

Deformable Linear Objects (DLOs) are found in various day-to-day tasks such as knot tying, as well as in highly specialized applications such as medical suturing. DLOs are characterized by having one dimension significantly larger than the other two [1]. Due to deformation, the configuration space of DLOs is remarkably larger than any given rigid object, which can only translate and rotate. Compared to planar or volumetric objects, these have the lowest dimensionality, making them computationally lighter to simulate. Nevertheless, it is worth noting that DLOs can actually lead to more complex shapes than their higher-dimensional counterparts, particularly when they become entangled.

With this paper we aim to complement the work in [2] and present *ReForm*, from a practical perspective. We start by enumerating some of the related work in Section II, followed by a description of the sandbox in Section III, and concluding with a short tutorial on how to use *ReForm* for robot learning research, in Section IV.

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

[2] Division of Robotics, Perception and Learning, Department of Electrical Engineering and Computer Science, KTH Royal Institute of Technology, Sweden {robgie , fpokorny}@kth.se

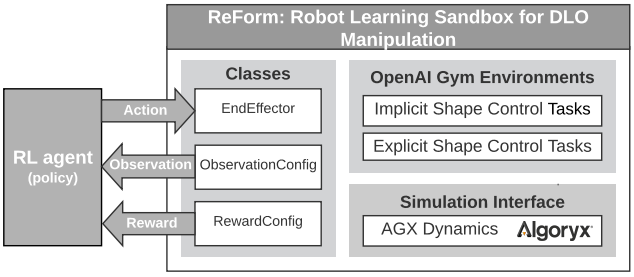

Fig. 1: ***System overview.*** *ReForm has three main modules: a simulation interface which connects the OpenAI Gym structure to the underlying AGX Dynamics simulator; implicit and explicit shape control environments which connect to that interface; a set of classes which abstract the observation, action and reward definitions from the environments.*

II. Related Work

RLBench [3] is a simulation benchmark made up of 100 manipulation tasks, of which one involves a deformable object, namely the rope-straightening task. Concept2Robot [4], is a framework for learning concepts of manipulation from visual demonstrations and language instructions, and it includes a task related to deformation, namely *(un)folding something*. AssistiveGym [5] provides six benchmark tasks for human-robot interaction, one of them consists of dressing a human with a deformable textile. Many more simulation resources for robot manipulation exclude deformation from their repertoire altogether. These include: SURREAL [6], a framework which provides a robotics benchmark with six classical manipulation tasks; Meta-World [7], a benchmark which focuses on multi-task learning in 50 scenarios; SAPIEN [8], a household simulation benchmark with manipulation of articulated objects, e.g doors; and iGibson [9], that allows interactions with the environment but is mainly focused towards indoor navigation.

Upon identifying a need for simulation environments which address deformable objects in depth, the idea for *ReForm* emerged. Similar observations also led to concurrent works SoftGym [10] and Elastica [11]. The former is a recently developed benchmark for manipulating soft objects in simulation. It contains control tasks for cloth-like objects, ropes and fluid. This is a great tool for researchers and shows a lot of potential. However, SoftGym does not include objects such as metal wires which have both elastic and plastic properties. Furthermore, due to the fact that Nvidia Flex [12] is the backend simulator, there is a hardware requirement of a D3D11 capable graphics card [13]. Further, the system has only been tested on Linux systems, given its dependency on PyFlex [14]. Elastica, on the other hand, is a simulation environment for soft rods which was designed to simulate

*TABLE I: For each environment it is possible to define RL parameters related to observations, actions and rewards, using the classes defined below. The table summarizes the key properties and methods which are relevant for each class. These classes allow modification of RL parameters without changing the underlying simulation. Control details such as DoFs can be set using the **EndEffector** class. Multiple types of observations are provided out of the box, and can be accessed through the **ObservationConfig** class. RL rewards can be set using the **RewardConfig** class.*

ObservationConfig			**EndEffector**		**RewardConfig**	
Vision	DLO	End-effector	Settings	Constraints	Dense	Sparse
• RGB • depth	• segment positions • segment rotations • discrete curvature • discrete torsion	• position • rotation • force/torque • linear velocity • angular velocity	• controllable • observable • max velocity • max acceleration	• end-effector DoF • compute forces enabled • velocity control • grip compliance control	**reward_function** method returns scalar reward value	**success_condition** checks if goal is reached and returns Boolean

soft robotic actuators, rather than to model shape control problems with DLOs. More recently, PlasticineLab [15] was published, providing a soft-body manipulation benchmark with differentiable physics. Plasticine is a great representation learn manipulation of volumetric objects with plastic properties.

III. Presenting *ReForm*

ReForm includes six robotic manipulation tasks for DLOs. These are integrated with OpenAI's Gym [16] and provide a modular interface so that tasks can be easily modified and created. All environments are designed with continuous control of Cartesian manipulators, where the active Degrees of Freedom (DoFs) may also be changed. Several types of observations are implemented, including vision and force-torque measurements. *ReForm* was designed to be flexible in terms of RL settings (i.e. observations, actions and reward), enabling users to quickly set up new experiments with custom parameters. Table I shows the main settings which can be quickly accessed without changing the underlying simulation environment.

ReForm uses AGX Dynamics [17] as the backend physics engine. DLOs are modeled as lumped elements using the specialized **Cable** class for which properties, such as Young modulus and Poisson's ratio, can be defined along stretch, twist and bend directions [18], [19]. Elastic **Cable** objects can also easily be assigned plasticity properties by defining a yield point. To the best of our knowledge, this is currently not possible in other physics engines e.g. Mujoco [20], SOFA [21] or Flex [12]. In order to create new environments, AGX Dynamics provides a simple Python library. Section IV, addresses how this can be done in more detail.

In our work [2], we categorize different tasks into explicit and implicit shape control. As the name indicates, in **explicit shape control** tasks the goal is to achieve an explicit shape of the object, e.g. bending a wire into a paperclip. On the other hand, **implicit shape control** tasks have a more abstract goal e.g. tying a rope into a knot, which requires deformation but the exact shape is not important. This categorization is particularly useful in the context of RL, given that a key step in formulating the problem is to define a reward function. It is our belief that setting a fixed reward, as is done in benchmark papers, would lead to blind comparison of cumulative reward results potentially loosing sight of the true goal of each task. For this reason, *ReForm* offers the possibility to change the reward function and benchmarking should be done by looking at qualitative results. In the following sections we give a short overview of the six environments and their key challenges.

A. Explicit Shape Control

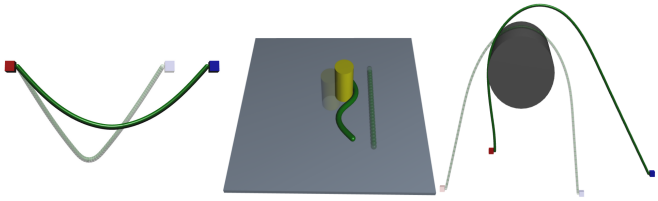

*Fig. 2: (left to right) **BendWire**, **PushRope** and **BendWireObstacle**. Note that it is possible to render the desired shapes in each environment, if such a vision-based approach is to be implemented.*

This type of problems can be expressed by some distance measure between the desired shape and the current state. However, the type of measure will depend on the goal of the task. For example, if one wants to bend a wire into a paperclip and does not care how this is located or oriented in space, the reward should not be a simple Euclidean distance, as this will incorrectly encode the desired goal.

BendWire: a metal wire is attached to two grippers with hinge constraints. The goal is to deform the wire into a desired shape. A key challenge is that the wire is stiff and exhibits elastoplastic properties leading to irreversible plastic deformations. DLOs of this kind are found throughout manufacturing as well as in medical applications, such as in dental braces. A similar problem was tackled by [22].

PushRope: a soft rope is located on a planar surface. A controllable pusher is used to bring the rope into a goal shape. The rope exhibits little stiffness and deforms immediately after contact. The friction between the rope and surface is an interesting feature of this manipulation task. This is a common benchmark task which has been studied in numerous papers, e.g. [3], [23].

BendWireObstacle: this environment is similar to **BendWire** but includes a cylindrical obstacle in the workspace. A key challenge is to adequately interact with the obstacle to achieve the desired deformation. The work by Zhu et al. [24] covered such a scenario for cable routing.

B. Implicit Shape Control

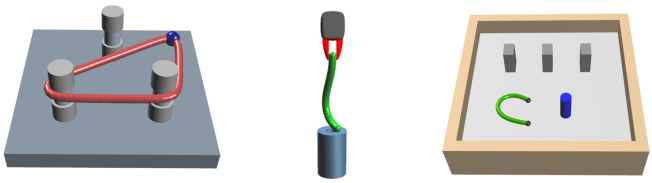

Fig. 3: (left to right) ***RubberBand****,* ***PegInHole*** *and* ***CableClosing****. Note that the gripper in the PegInHole is for visualization purposes only, and the underlying control is the same as for the other environments.*

While explicit shape control tasks can be expressed by a distance measure, the same is typically not true for implicit shape control tasks. Defining rewards is therefore not trivial and is likely to require a trial-and-error approach. For such tasks, it is common to add rewards for intermediate goals.

RubberBand: a purely elastic circular rubber band[1] is connected to a gripper by a ball joint. The task is to wrap the rubber band around three poles. This environment incorporates complicated contact mechanics between the DLO and the obstacles. This type of task bridges the gap to industrial applications, such as [25].

PegInHole: a soft peg is rigidly attached to a gripper, on one end. The task is to insert the peg into a hollow rigid object. Note that [26] presented the converse problem in which a rigid peg is inserted into a hollow deformable object.

CableClosing: a cable is controlled on both ends by planar actuators. The task is to make the cable fully enclose a goal object, while avoiding several other obstacles. This task is quite unique, although it can be compared with the work by [27], where objects in the environment were used to tie different types of knots.

IV. Using *ReForm*

In order to use *ReForm*, AGX Dynamics [17] must be installed. Though this is a proprietary software, there are no specific GPU requirements and all major operating systems are supported. As mentioned in the previous section, *ReForm* is organized so that RL parameters can be easily modified. However, it is also possible to modify the underlying environment properties. To do that, library files under `sims` need to be altered, which can be found in the directory tree:

```
gym-agx
├── envs
│   ├── assets
│   ├── explicit
│   └── implicit
├── rl
├── sims
├── ...
└── __init__.py
```

[1]A rubber band is technically not a DLO but also not a planar object. It can be seen as a DLO connected with itself.

In this directory, there is a file for each simulation environment. Key properties are defined by constants found at the top of these files, but structural changes (e.g. type of constraints) require changing more code. Note that only physical details (e.g. material properties) are defined here, with rendering being taken care of in the environment files under `envs`. This is useful because the simulation can be executed completely without graphics, if vision data is not used for the RL loop. After changing the simulation script under `sims`, this must be executed to generate a new simulation file which will be stored under `assets`.

To extend *ReForm*, and create new environments, a new script needs to be included under the `sims` directory for each addition, and the following steps executed: *i.* add objects to scene, *ii.* define material and inter-material properties, *iii.* attach constraints to control the DLO, *iv.* generate simulation file and store it in `assets`. Once this is done, a new environment file needs to be created either under the `implicit` or `explicit` directory. This file defines which simulation file is loaded from `assets`, and also includes the default **ObservationConfig**, **EndEffector** and **RewardConfig** objects which need to be defined to be able to interact with the simulation. Each new environment must also be registered in `__init__.py`.

The key difference between implementing implicit or explicit shape control environments is that the former inherit from Gym's **Env** class, while the latter inherit from the multi-goal oriented **GoalEnv** class. This distinction is important since when learning policies which can generalize to multiple target shapes, it is necessary to have the desired shape as part of the observation. This is enforced by the **GoalEnv** class, first introduced in [28], by requiring the observation space to contain: **observation**, **desired_goal**, and **achieved_goal**. For implicit shape control tasks, there is typically a fixed goal, although one could also modify targets categorically, e.g. closing a cable around one of several target objects.

V. Conclusion

In this paper we have presented *ReForm* [2], a robot learning sandbox for DLO manipulation. We gave a short overview of how it is organized and what are its key features. Further, we described the six manipulation tasks which are currently available, and discussed some of their key challenges. A short description of how to modify and add new environments was also given. For a more in depth coverage of the sandbox, as well as some preliminary RL results, refer to [2]. To read more about the library and how to use it, refer to [29].

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] R. Laezza, R. Gieselmann, F. T. Pokorny, and Y. Karayiannidis, "Reform: A robot learning sandbox for deformable linear object manipulation," *2021 IEEE International Conference on Robotics and Automation (ICRA)*, May 2021.

[3] S. James, Z. Ma, D. R. Arrojo, and A. J. Davison, "Rlbench: The robot learning benchmark & learning environment," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3019–3026, 2020.

[4] L. Shao, T. Migimatsu, Q. Zhang, K. Yang, and J. Bohg, "Concept2robot: Learning manipulation concepts from instructions and human demonstrations," *Robotics: Science and Systems*, 2020.

[5] Z. Erickson, V. Gangaram, A. Kapusta, C. K. Liu, and C. C. Kemp, "Assistive gym: A physics simulation framework for assistive robotics," *IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

[6] L. Fan, Y. Zhu, J. Zhu, Z. Liu, O. Zeng, A. Gupta, J. Creus-Costa, S. Savarese, and L. Fei-Fei, "Surreal: Open-source reinforcement learning framework and robot manipulation benchmark," in *Conference on Robot Learning*, pp. 767–782, 2018.

[7] T. Yu, D. Quillen, Z. He, R. Julian, K. Hausman, C. Finn, and S. Levine, "Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning," in *Conference on Robot Learning*, pp. 1094–1100, 2020.

[8] F. Xiang, Y. Qin, K. Mo, Y. Xia, H. Zhu, F. Liu, M. Liu, H. Jiang, Y. Yuan, H. Wang, L. Yi, A. X. Chang, L. J. Guibas, and H. Su, "Sapien: A simulated part-based interactive environment," *ArXiv*, vol. abs/2003.08515, 2020.

[9] F. Xia, W. B. Shen, C. Li, P. Kasimbeg, M. E. Tchapmi, A. Toshev, R. Martín-Martín, and S. Savarese, "Interactive gibson benchmark: A benchmark for interactive navigation in cluttered environments," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 713–720, 2020.

[10] X. Lin, Y. Wang, J. Olkin, and D. Held, "Softgym: Benchmarking deep reinforcement learning for deformable object manipulation," *arXiv preprint arXiv:2011.07215*, 2020.

[11] N. Naughton, J. Sun, A. Tekinalp, G. Chowdhary, and M. Gazzola, "Elastica: A compliant mechanics environment for soft robotic control," 2020.

[12] M. Macklin, M. Müller, N. Chentanez, and T.-Y. Kim, "Unified particle physics for real-time applications," *ACM Transactions on Graphics (TOG)*, vol. 33, no. 4, pp. 1–12, 2014.

[13] "Nvidia flex." `https://docs.nvidia.com/gameworks/content/gameworkslibrary/physx/flex/index.html`.

[14] "Pyflex." `https://github.com/YunzhuLi/PyFleX`.

[15] Z. Huang, Y. Hu, T. Du, S. Zhou, H. Su, J. B. Tenenbaum, and C. Gan, "Plasticinelab: A soft-body manipulation benchmark with differentiable physics," *arXiv preprint arXiv:2104.03311*, 2021.

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

[17] "Agx dynamics." `https://www.algoryx.se/agx-dynamics/`.

[18] 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.

[19] H. Lang, J. Linn, and M. Arnold, "Multi-body dynamics simulation of geometrically exact Cosserat rods," *Multibody System Dynamics*, vol. 25, no. 3, pp. 285–312, 2011.

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

[21] F. Faure, C. Duriez, H. Delingette, J. Allard, B. Gilles, S. Marchesseau, H. Talbot, H. Courtecuisse, G. Bousquet, I. Peterlik, *et al.*, "Sofa: A multi-model framework for interactive physical simulation," in *Soft tissue biomechanical modeling for computer assisted surgery*, pp. 283–321, Springer, 2012.

[22] 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.

[23] X. Lin, H. S. Baweja, and D. Held, "Reinforcement learning without ground-truth state," *arXiv preprint arXiv:1905.07866*, 2019.

[24] J. Zhu, B. Navarro, R. Passama, P. Fraisse, A. Crosnier, and A. Cherubini, "Robotic manipulation planning for shaping deformable linear objects withenvironmental contacts," *IEEE Robotics and Automation Letters*, vol. 5, no. 1, pp. 16–23, 2019.

[25] M. Murase, K. Yamazaki, and T. Matsubara, "Kullback leibler control approach to rubber band manipulation," in *2017 IEEE/SICE International Symposium on System Integration (SII)*, pp. 680–685, IEEE, 2017.

[26] J. Luo, E. Solowjow, C. Wen, J. A. Ojea, and A. M. Agogino, "Deep reinforcement learning for robotic assembly of mixed deformable and rigid objects," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2062–2069, IEEE, 2018.

[27] M. Saha and P. 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.

[28] 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.

[29] "Gym agx." `https://github.com/ritalaezza/gym-agx`.