Goal-Conditioned Model Simplification for Deformable Object Manipulation

Shengyin Wang*, Matteo Leonetti†, Mehmet Dogar*
* School of Computing, University of Leeds, Leeds, United Kingdom
Email: scswan@leeds.ac.uk, m.r.dogar@leeds.ac.uk
† Department of Informatics, King's College London, London, United Kingdom
Email: matteo.leonetti@kcl.ac.uk

Abstract—**Deformable object manipulation has been a challenge for a long time in robotics due to its high-dimensional configuration space and complex dynamics. In this work, we explore the idea of goal-conditioned model simplification which has a great potential to improve motion planning, perception, and policy learning. Two workflows are proposed for objects that can be approximated by lines (1D linear model) and surfaces (2D flat model) respectively to find the minimal number of key points which are used to reduce the action search space for motion planners. Representative tests on ropes and cloth are conducted which demonstrate the effectiveness of the proposed method.**

I. Introduction & Related Work

Due to the ubiquitous existence of deformable objects and their extensive applications [1], [2], there has been a great deal of work dedicated to the development of deformable-object manipulation methods. Detailed survey of deformable modelling, planning, control, and learning can be found by Zhu [3], Yin [4], Bhagat [5], and Arriola-Rios [6].

The intrinsic high-dimensional configuration space of deformable objects make both perception and planning complex problems, especially for traditional search or optimization-based methods [7]–[9]. To circumvent this problem, earlier work focused on predefined features such as key points [10], contours [11], or wrinkles [12]. However, defining such features requires expert knowledge about the deformable object. Learning based methods are also employed to extract a compact representation of the deformable object directly from raw sensor inputs [13], and achieve faster planning, however the latent space is not explainable or intelligible to humans. Although researchers proposed various schemes to alleviate the challenges posed by high-dimensional configuration spaces, they are mostly designed for a particular task [11] or require large amount of training data [14].

Another challenge for deformable-object manipulation is that the dynamics of the object is much more complex than rigid objects. Unlike rigid objects, it is difficult to derive an explicit model for deformable objects to describe their dynamical behaviors due to high-dimensionality of the configuration space, material characteristics, and contacts with environments [13]. Several simulators have been developed based on mass-spring systems, particle based dynamics, or finite element models [6], to support deformable object manipulation, including softgym [15], mujoco [16], and pybullet [17]. Although these simulators achieve similar deformable behaviors to the real deformable object, planning motions or learning policies [18]–[20] based on high fidelity simulators is still time consuming due to a vast number of rollouts. To reduce the time cost, learning based methods [21] are employed to learn the dynamic model of the deformable objects either in pixel space [22] or latent space [13] before rolling out for planning. However, the interpretability of these models is poor, and the number of samples required for training a neural network model is still forbiddingly high. Power et al. [23] use simple models to improve learning efficiency for controlling complex systems. Nonetheless, only a few tasks are performed and choosing the simple model manually is non-trivial.

As pointed out by Lin et al. [15], a compact representation that captures key features relevant to the goal could make planing faster and learning more accurate. For example, when flattening cloth, four corners are enough for both representing the state and achieving the task, whereas for side folding tasks, additional two middle points are indispensable for specifying the goal state and can give intuitive information for action selection. Thus, our key idea to circumvent these problems is designing a goal-conditioned model simplification scheme to automatically extract key features that are most relevant to perform the task while keeping the representation as concise as possible. Centering on this idea, we developed line-fitting and mesh simplification methods for linear and flat deformable objects separately. Once the key points are acquired, we use it to simplify the action space for a motion planer, which shows better performance and faster convergence speed for all the given deformable manipulation tasks on both ropes and cloth.

II. Problem Formulation

Considering a deformable object modelled by a mass-spring system, in which a grid of mass points are subjected to gravity, Hookean spring force, and contact forces, the full state of the object when stabilized can be described by the position of all mass points. Suppose there are N_o mass points in the object model, each of which is deemed as a particle indexed by $i, (i = 1, ..., N_o)$ with $p_i = [x_i, y_i, z_i]$ representing the position of the i^{th} particle. Then, the configuration at time t can be defined as $\boldsymbol{\xi_t} := [p_i \mid \forall i \in \{1, ..., N_o\}]$.

To simplify the planning problem, we assume the robot pickers can pick any given particle in the object. The available actions at each time t move a particle with the picker by a certain distance along a target direction. To let the planner allow a picker not to pick any particle, $None$ is added for grasping nothing, thus the action at time t can be given as:

$$a_t^{(k)} = \begin{cases} \langle p_i, \delta x_t, \delta y_t, \delta z_t \rangle \\ None \end{cases} \qquad (1)$$
$$a_t = \left[a_t^{(1)}, a_t^{(2)} \right]$$

where the superscript k denotes the k^{th} picker, in this work we assume two pickers $k \in \{1, 2\}$. We constrain the movement of the picker complying to the quasi-static assumption: $|\delta x_t| <= \Delta x$, $|\delta y_t| <= \Delta y$, $|\delta z_t| <= \Delta z$; Δx, Δy, and Δz are motion limits along each axis in Cartesian space. Thus, the original action space has size $(2 \times N_o \times R^3 + 2)$.

The planning problem is to find a sequence of N_a actions, $\tau = a_1, a_2, ..., a_{N_a}$, that minimizes a goal-conditioned cost function $c_g(\tau)$. For instance, the goal can be a triangle shape for rope folding and a diagonally folded shape for cloth folding (last column in Fig. 5). Admissible action sequences are those that do not cause the object to stretch excessively. Formally, the optimization problem solved by the planner is:

$$\begin{aligned} &min \quad && c_g(\tau) \\ &s.t. \quad && C(a_t) < 0, \ \forall t \in \{1, ..., N_a\}, \end{aligned} \qquad (2)$$

where $C(a_t)$ is the stretching constraint for the action at step t, as defined in the softgym simulator in practice.

III. Methodology

To solve the planning problem defined in Equation 2 we used Cross Entropy Method (CEM). However, due to the high dimensional action space, CEM requires a large number of rollouts. To reduce the complexity of the planning problem, we simplify the model by reducing the number of graspable points based on the goal configuration.

A. Model Simplification

Given the goal configuration of the deformable object, we aim to find the smallest accurate model that contains the most important and relevant key particles to achieve the task. In most cases, the number of key particles, N_s, is much smaller than the original number, i.e. $N_s \ll N_o$, making the action space of the planner much smaller. To extract such particles from the given deformable object, we used two different methods, depending on whether the object can be effectively approximated by a one or two-dimensional entity. In the former case we used piece-wise line-fitting, while in the latter case we used a mesh-simplification method. It is worth noting that manipulation still happens in 3D space while objects can be approximated by a set of lines or surfaces.

1) Line Fitting for Ropes: We start with two key points, $N_s = 2$, the positions of which can be anywhere in the space, and iteratively increase the number of key points (see Fig. 1) until we have a good fit to the goal configuration. At each iteration, (i.e. for each value of N_s), we solve a Quadratic

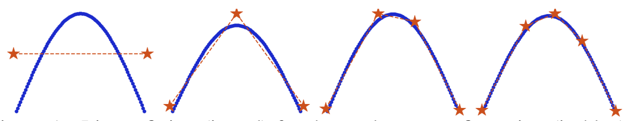
Figure 1. Linear fitting (in red) for the goal rope configuration (in blue).

Programming (QP) problem, to find the optimal piece-wise line fit, which minimizes the error:

$$E = \sum\nolimits_{n=1}^{N_o} (p_i - f_{j=0}^{N_s}(\frac{i}{N_o})])^2, \qquad (3)$$

where $f_{j=0}^{N_s}(\cdot)$ denotes the piece-wise line equation interpolated from the given key points $q_j, j \in \{1, ..., N_s\}$, with $f_{j=0}^{N_s}(0)$ representing the first key point, $f_{j=0}^{N_s}(1)$ as the last key point, and $f_{j=0}^{N_s}(i/N_o)$ representing the i^{th} point on the fitted line corresponding to the i^{th} particle on the original rope. For example, in Fig. 1, the blue dots represent the original particles, the red stars represent the simplified key particles, and the interpolated particle q_j can be extracted from the red dotted line. The iterative process stops at the smallest number of key points, N_s, for which the error E is below a threshold τ_l. For example in Fig. 1, our process stops at the rightmost iteration with $N_s = 5$.

2) Mesh Simplification for Cloth: For objects that can be approximated by surfaces, we rely on a widely used mesh processing toolkit called Meshlab [24], and use its built-in module named Quadric Edge Collapse Decimation [25] to simplify the meshes to a given number of triangles (starting from two triangles with four key points). To measure the error of the simplified mesh with respect to the original one, (i.e. in place of the error E above), we use the Hausdorff distance [26]. Similarly to the rope simplification, an iterative process stops at the smallest number of triangles whose error is below a threshold τ_s.

B. CEM based Planning

CEM [27] is a well established optimization algorithm, which has been applied to address plenty of manipulation problems including deformable objects. To demonstrate the effectiveness of the proposed model simplification methods, CEM-Original (all points on the model are graspable) and CEM-Simple (only extracted key points are graspable) are implemented on a set of tasks for rope and cloth manipulation.

IV. Experiments

To evaluate the effectiveness of the proposed method, a set of experiments are conducted to illustrate how the simplified model can be used to improve the performance of motion planning for deformable objects covering both ropes and cloth. CEM-Original is implemented as a benchmark method to compare the performance of CEM-Simple. The simulator used in this paper is based on softgym [15] which is built on top of PyFlex [28] provided by Nvidia. All tests are conducted on a workstation with Intel Xeon(R) CPU E5-2665 @2.40GHz, 32 GB of RAM, and Nvidia Quadro K4000 GPU.

A. Tasks

1) Rope Straightening [15]: This task refers manipulating a rope from a randomized initial configuration to a slack and straightened goal configuration. The cost function is the accumulation of the absolute difference between the current distance of the endpoints and the original length of the rope.

2) Rope Folding [15]: For the task of Rope Folding, the two pickers need to deform the rope into a specific shape, a triangle in this test, from an initially straightened state. The cost is also an accumulative quantity at each step, which is computed by the bipartite matching distance between the current positions and the goal positions of all particles.

3) Cloth Side Folding [15]: Cloth Side Folding task aims to fold a flattened cloth into half sideways. The cost function is the accumulative step costs composed by two parts: one is the distance of the corresponding particles between the two halves; another is the displacement of the still half from its initial state which penalizes the cloth being dragged away.

4) Cloth Diagonal Folding [29]: This task is very similar to the Cloth Side Folding task, with the only difference of using the diagonal halves instead of sideways halves. The structure of the cost function is also same by replacing sideways correspondence with diagonal matching relations.

B. Simulation Results

1) Model Simplification: Model simplification costs and the simplified models are shown in Fig. 2 and 3 respectively. As the results show, only the two end points are necessary for the rope straightening task, while another middle point is required for the folding task. Mesh simplification for cloth is much more complex than rope simplification: a simplified mesh with only six points and six triangles can be used to approximate a complex mesh when folded in half sideways; meanwhile, a piece of diagonally folded cloth can be approximated using four points and four triangles. In this way, the points that are most relevant to complete the task are extracted to facilitate motion planning.

2) CEM based Motion Planning: Based on the extracted key points, CEM-Simple is implemented to demonstrate its effectiveness compared to CEM-Original. For rope manipulation, each task is tested for ten times, while only one round of optimization is conducted for cloth manipulation using the built-in CEM method in softgym due to the time cost. Planning costs per iteration during optimization are shown in Fig. 4, where CEM-Simple achieves lower cost and faster convergence than CEM-Original for all tasks. As the results show, for rope manipulation tasks, CEM-Original achieves similar cost to CEM-Simple because the dimensionality difference between them is much smaller than those for cloth manipulation tasks where CEM-Original cannot find a feasible plan while CEM-Simple succeeds with the same number of simulation steps. Snapshots for rope folding and cloth diagonal folding tasks are shown in Fig. 5, which validate that the proposed method can find more effective motion plans for rope manipulation, and achieve complex cloth folding tasks whereas using the original action space failed.

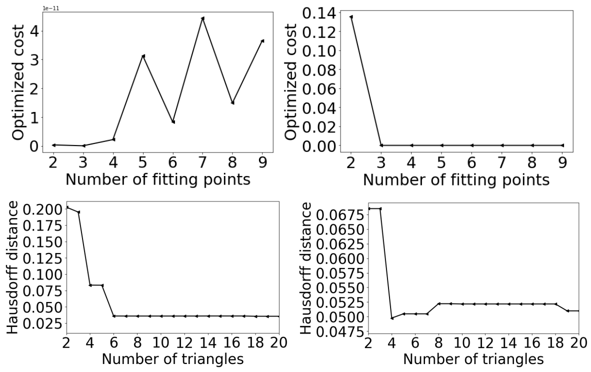

Figure 2. Model simplification costs (first row: rope straightening & folding; second row: cloth side & diagonal folding)

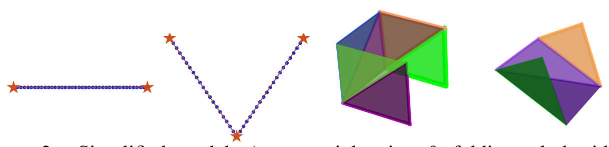
Figure 3. Simplified models (rope straightening & folding; cloth side & diagonal folding)

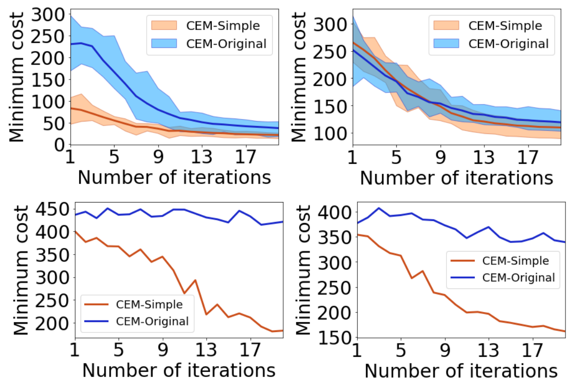

Figure 4. Motion planning cost (first row: rope straightening & folding; second row: cloth side & diagonal folding)

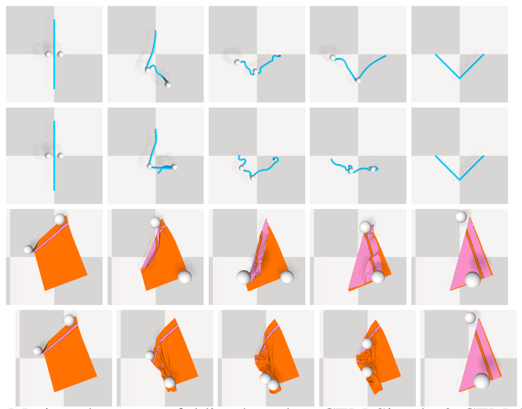
Figure 5. Motion plans: rope folding based on CEM-Simple & CEM-Original; cloth diagonal folding based on CEM-Simple & CEM-Original (first column as initial state, the last two columns as final state and goal state respectively; rope straightening and cloth side folding are omitted due to limited space)

V. Conclusion

In this paper, we proposed model simplification methods for ropes and cloth, and established a deformable object manipulation scheme based on goal conditioned model simplification which shows better performance in several tasks. Currently, we are doing research on hierarchical planning to address the case where only the final goal is not enough to achieve the task and intermediate sub-goals are required. Future work will focus on model simplification for learning and planning.

References

[1] D. Seita, N. Jamali, M. Laskey, A. K. Tanwani, R. Berenstein, P. Baskaran, S. Iba, J. Canny, and K. Goldberg, "Deep transfer learning of pick points on fabric for robot bed-making," in *The International Symposium of Robotics Research*. Springer, 2019, pp. 275–290.

[2] X. Li, X. Su, and Y.-H. Liu, "Vision-based robotic manipulation of flexible pcbs," *IEEE/ASME Transactions on Mechatronics*, vol. 23, no. 6, pp. 2739–2749, 2018.

[3] J. Zhu, A. Cherubini, C. Dune, D. Navarro-Alarcon, F. Alambeigi, D. Berenson, F. Ficuciello, K. Harada, X. Li, J. Pan *et al.*, "Challenges and outlook in robotic manipulation of deformable objects," *arXiv preprint arXiv:2105.01767*, 2021.

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

[5] S. Bhagat, H. Banerjee, Z. T. Ho Tse, and H. Ren, "Deep reinforcement learning for soft, flexible robots: Brief review with impending challenges," *Robotics*, vol. 8, no. 1, p. 4, 2019.

[6] V. E. Arriola-Rios, P. Guler, F. Ficuciello, D. Kragic, B. Siciliano, and J. L. Wyatt, "Modeling of deformable objects for robotic manipulation: A tutorial and review," *Frontiers in Robotics and AI*, vol. 7, p. 82, 2020.

[7] P. Jiménez, "Survey on model-based manipulation planning of deformable objects," *Robotics and computer-integrated manufacturing*, vol. 28, no. 2, pp. 154–163, 2012.

[8] A. Doumanoglou, A. Kargakos, T.-K. Kim, and S. Malassiotis, "Autonomous active recognition and unfolding of clothes using random decision forests and probabilistic planning," in *2014 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2014, pp. 987–993.

[9] Y. Li, Y. Yue, D. Xu, E. Grinspun, and P. K. Allen, "Folding deformable objects using predictive simulation and trajectory optimization," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 6000–6006.

[10] J. Maitin-Shepard, M. Cusumano-Towner, J. Lei, and P. Abbeel, "Cloth grasp point detection based on multiple-view geometric cues with application to robotic towel folding," in *2010 IEEE International Conference on Robotics and Automation*. IEEE, 2010, pp. 2308–2315.

[11] S. Miller, J. Van Den Berg, M. Fritz, T. Darrell, K. Goldberg, and P. Abbeel, "A geometric approach to robotic laundry folding," *The International Journal of Robotics Research*, vol. 31, no. 2, pp. 249–267, 2012.

[12] L. Sun, G. Aragon-Camarasa, P. Cockshott, S. Rogers, and J. P. Siebert, "A heuristic-based approach for flattening wrinkled clothes," in *Conference Towards Autonomous Robotic Systems*. Springer, 2013, pp. 148–160.

[13] W. Yan, A. Vangipuram, P. Abbeel, and L. Pinto, "Learning predictive representations for deformable objects using contrastive estimation," *arXiv preprint arXiv:2003.05436*, 2020.

[14] P. Zhou, J. Zhu, S. Huo, and D. Navarro-Alarcon, "Lasesom: A latent and semantic representation framework for soft object manipulation," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 5381–5388, 2021.

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

[16] 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*. IEEE, 2012, pp. 5026–5033.

[17] C. Erwin and B. Yunfei, "Pybullet a python module for physics simulation for games," *PyBullet*, 2016.

[18] D. Seita, A. Ganapathi, R. Hoque, M. Hwang, E. Cen, A. K. Tanwani, A. Balakrishna, B. Thananjeyan, J. Ichnowski, N. Jamali *et al.*, "Deep imitation learning of sequential fabric smoothing from an algorithmic supervisor," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 9651–9658.

[19] J. Matas, S. James, and A. J. Davison, "Sim-to-real reinforcement learning for deformable object manipulation," in *Conference on Robot Learning*. PMLR, 2018, pp. 734–743.

[20] R. Jangir, G. Alenya, and C. Torras, "Dynamic cloth manipulation with deep reinforcement learning," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 4630–4636.

[21] X. Ma, D. Hsu, and W. S. Lee, "Learning latent graph dynamics for deformable object manipulation," *arXiv preprint arXiv:2104.12149*, 2021.

[22] R. Hoque, D. Seita, A. Balakrishna, A. Ganapathi, A. K. Tanwani, N. Jamali, K. Yamane, S. Iba, and K. Goldberg, "Visuospatial foresight for multi-step, multi-task fabric manipulation," *arXiv preprint arXiv:2003.09044*, 2020.

[23] T. Power and D. Berenson, "Keep it simple: Data-efficient learning for controlling complex systems with simple models," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1184–1191, 2021.

[24] P. Cignoni, M. Callieri, M. Corsini, M. Dellepiane, F. Ganovelli, G. Ranzuglia *et al.*, "Meshlab: an open-source mesh processing tool." in *Eurographics Italian chapter conference*, vol. 2008. Salerno, Italy, 2008, pp. 129–136.

[25] M. Garland and P. S. Heckbert, "Surface simplification using quadric error metrics," in *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, 1997, pp. 209–216.

[26] N. Aspert, D. Santa-Cruz, and T. Ebrahimi, "Mesh: Measuring errors between surfaces using the hausdorff distance," in *Proceedings. IEEE international conference on multimedia and expo*, vol. 1. IEEE, 2002, pp. 705–708.

[27] R. Rubinstein, "The cross-entropy method for combinatorial and continuous optimization," *Methodology and computing in applied probability*, vol. 1, no. 2, pp. 127–190, 1999.

[28] Y. Li, J. Wu, R. Tedrake, J. B. Tenenbaum, and A. Torralba, "Learning particle dynamics for manipulating rigid bodies, deformable objects, and fluids," *arXiv preprint arXiv:1810.01566*, 2018.

[29] R. Hoque, D. Seita, A. Balakrishna, A. Ganapathi, A. K. Tanwani, N. Jamali, K. Yamane, S. Iba, and K. Goldberg, "Visuospatial foresight for physical sequential fabric manipulation," *Autonomous Robots*, vol. 46, no. 1, pp. 175–199, 2022.