Deep Recurrent Models for Nonlinear Model Predictive Control in Deformable Manipulation Tasks

James A. Preiss*, David Millard*, Tao Yao*, Gaurav S. Sukhatme†

***Abstract*— We propose a method for robotic control of deformable objects using a learned nonlinear dynamics model. After collecting a dataset of trajectories from the real system, we train a recurrent neural network (RNN) to approximate its input-output behavior with a latent state-space model. The RNN internal state is low-dimensional enough to enable realtime nonlinear control methods. We demonstrate a closed-loop control scheme with the RNN model using a standard nonlinear state observer and model-predictive controller. We apply our method to track a highly dynamic trajectory with a point on the deformable object, in real time and on real hardware. Our experiments show that the RNN model captures the true system's dynamics and can be used to track trajectories outside the training distribution. In an ablation study, we find that the full method improves tracking accuracy compared to an open-loop version without the state observer.**

I. Introduction and Related Work

Manipulating deformable objects represents a challenging area of robotics. In contrast to typical objects consisting of a single rigid body, deformable objects often admit limited control authority and have dynamics that are difficult to predict. At the same time, many objects of human interest are deformable. Safe, reliable robotic manipulation of these objects is critical for capable general-purpose robots [1], [2].

Physics-based models of deformable objects have been studied extensively in science and engineering contexts [3]. Recent work with Finite Element Method (FEM) modeling addresses dynamic control of deformable objects and soft robots with offline trajectory optimization [4]–[9]. FEM is appealing as it admits extensive theoretical analysis [10], [11]. For real-world problems, it is possible to estimate the parameters of FEM meshes in a principled way from exteroceptive sensors common in robotics [12]. Alternative discretizations include the material point method [13], [14], (extended) position based dynamics [15], and meshless shape matching [16].

Deformable objects are complex to model, and even high-resolution FEM approaches have significant inductive biases. In this light, purely data-driven methods are an appealing alternative to physics-based models. Machine learning methods have been explored in deformable manipulation [17], for purely kinematic trajectory tracking [18], for cable-driven soft robot actuators [19], for control of pneumatic deformable mechanisms [20], and for structures actuated by shape-memory alloys [21]. Other data-driven approaches studied in soft robot control include proper orthogonal decompositions [22].

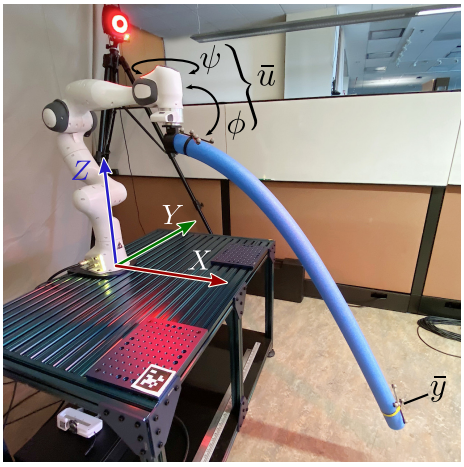

Fig. 1. Physical testbed for our method. Trajectories are tracked by a Vicon motion capture marker attached to the end of a pool noodle, rigidly held by a Franka Emika Panda robot. Pitch/yaw inputs $\bar{u} = (\phi, \psi)$, tracking point measurement $\bar{y}$, and coordinate axes (X, Y, Z) shown. (The fiducial marker visible in the image is not used.)

We propose a data-driven method for modeling and trajectory-tracking control of a deformable object at a speed where the passive dynamics are the dominant behavior of the system. Our method models the dynamics with a long short-term memory (LSTM) recurrent neural network (RNN) [23], [24], trained in a standard sequence modeling setup using input-output trajectory data from a real physical system. The internal state of the learned RNN is not physically meaningful, but the RNN still forms a discrete-time dynamical system that is compatible with standard methods in state-space nonlinear control. We apply model-predictive control [25] and extended Kalman filtering methods [26] to track the trajectory using the RNN model. We find that closed-loop control with our method reduces trajectory tracking error compared to an open loop trajectory optimizer.

II. Problem Setting and Preliminaries

We consider a robot manipulating a deformable object such that a particular point on the object tracks a given trajectory. We use the following notation: The special Euclidean group of rigid transformations in three-dimensional space is denoted by $SE(3)$. The notation $A \succeq 0$ (resp. $A \succ 0$) indicates that the matrix A is positive semidefinite (resp.

All authors are affiliated with the University of Southern California.

* Equal contribution. {japreiss,dmillard,taoyao}@usc.edu

† G.S. Sukhatme holds concurrent appointments as a Professor at USC and as an Amazon Scholar. This paper describes work performed at USC and is not associated with Amazon.

This work was supported in part by a NASA Space Technology Research Fellowship, grant number 80NSSC19K1182.

An expanded version of this work has been accepted to the 2022 IEEE Conference on Robotics and Automation.

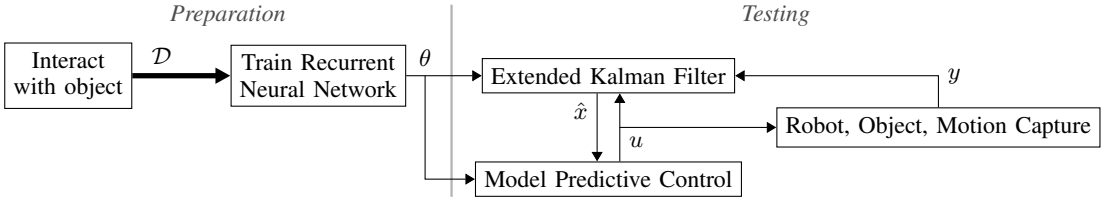

Fig. 2. Diagram of our system. A small dataset of control inputs and tracked marker locations is obtained from the real system. A recurrent neural network (RNN) model is trained to predict input/output behavior with a latent state. The RNN forms the nonlinear dynamics model for an extended Kalman filter (EKF) and model-predictive controller (MPC) to track a dynamic input trajectory with the deformable object.

definite). We assume that the robot has already rigidly grasped the object, that we can accurately measure specific points on the object, and that the robot does not know the specific target trajectory during its data collection stage.

Our control policy interacts with the system in discrete time steps of fixed length Δ_t. Our control task is specified by a discrete-time signal of K goal positions $z[1], \ldots, z[K] \in \mathbb{R}^3$ for the tracked point and the cost function

$$J = \textstyle\sum_{k=1}^{K} \|z[k] - \bar{y}[k]\|_W^2, \tag{1}$$

where the weighting matrix $W \succeq 0$ encodes a (potentially) non-isotropic tracking objective.

III. Methods

A. Data collection

We begin by collecting a training dataset $\mathcal{D}$ containing N input-output trajectories from the real system. We denote by $\bar{u}[j,k]$ and $\bar{y}[j,k]$ the input and output from the k^{th} time step of the j^{th} trajectory in $\mathcal{D}$. Before starting each trajectory, we apply the identity input and allow the system to settle for several seconds (10 in our experiments) to ensure that the system returns to a state very close to the rest state $\bar{x}_0$. We then apply random sinusoidal pitch and yaw inputs. with frequency ν sampled log-uniformly between $1/8$ Hz and 3 Hz. The maximum angular acceleration of a sinusoid is given by $2\pi A\nu^2$, where A is a uniformly sampled amplitude.

B. RNN dynamics model

We represent the system state as the hidden state of a learned RNN. The RNN is a generic function approximation scheme parameterized by a real-valued vector θ, and consists of a discrete-time dynamics model

$$x[k+1] = f_\theta(x[k], u[k]), \quad y[k] = h_\theta(x[k]), \tag{2}$$

where $x \in \mathbb{R}^n$ is the internal state, u is the input, y is the output, and f_θ and h_θ are the dynamics and measurement functions respectively. Both f_θ and h_θ are differentiable with respect to their arguments and the parameter θ.

The RNN is trained on the dataset $\mathcal{D}$ to minimize a regression loss on the input-output map over complete sequences:

$$\underset{\theta}{\text{minimize}} \quad \sum_{j=1}^{N} \sum_{k=1}^{K_j} \|\bar{y}[j,k] - h_\theta(x[j,k])\|_2^2 \tag{3}$$

where K_j is the length of the j^{th} trajectory. The fixed initial state of $\mathbf{0}$ is justified in our setting because each trajectory in $\mathcal{D}$ begins at the rest state $\bar{x}_0$. The optimization problem (3) is solved with stochastic gradient descent (SGD) using backpropagation.

C. Model-predictive control with reduced-order model

We use the RNN model in a model-predictive control (MPC) framework to optimize the trajectory-tracking objective (1) in an online manner. We solve the short-horizon optimal control problem

$$\begin{aligned} \underset{\substack{\bar{u}[k],\ldots,\\ \bar{u}[k+H-1]}}{\text{minimize}} \quad & \sum_{i=1}^{H} \|z[k+i] - h_{\theta^*}(x[k+i])\|_W^2 + R(u) \\ \text{subject to} \quad & x[k] = \hat{x}[k], \\ & x[k+i+1] = f_{\theta^*}(x[k+i], \bar{u}[k+i]) \ \forall i \end{aligned} \tag{4}$$

where $H \ll K$ is the MPC horizon and R is a quadratic regularization function to smooth the control signal and to return to the rest state absent other goals. Solving (4) yields a sequence of inputs $\bar{u}^*[k], \ldots, \bar{u}^*[k+H-1]$ optimized to track the next H steps of the full goal trajectory. Following the standard moving horizon architecture, we apply only the first input $\bar{u}^*[k]$ from the solution to the real system. Then, at time step $k+1$, we solve a new instance of (4). We obtain an approximate solution with a few steps of gradient descent with momentum, reusing the momentum between MPC problems.

D. Estimating the RNN state

In Section III-C, we assumed the availability of a RNN state $\hat{x}[k]$ that is consistent with previous inputs and outputs from the real-world system. Because the RNN model is not perfect, the true outputs $\bar{y}[1], \ldots, \bar{y}[k-1]$ obtained from the real-world system may diverge from the outputs $h_{\theta^*}(x[1]), \ldots, h_{\theta^*}(x[k-1])$ predicted by applying the RNN model in an open-loop manner.

Instead we observe that, although the RNN state has no direct physical meaning, its hidden state can be estimated from the input-output history using standard techniques from estimation theory. In particular, we apply an extended Kalman filter (EKF) to the RNN model. The EKF maintains a Gaussian-distributed belief over the RNN state with mean μ and covariance Σ. We use the mean of the belief distribution as the initial state for the MPC problem (4), $\hat{x}[k] = \mu[k]$.

TABLE I

VALUES OF USER-CHOSEN CONSTANTS IN OUR EXPERIMENTS.

Q	EKF process covariance	$10^{-6}I$
R	EKF measurement covariance	$10^{-2}I$
H	MPC horizon	25
α	MPC smoothness weight	1.0
β	MPC homing weight	1e-1
—	MPC gradient descent rate	4e-1
—	MPC gradient descent steps	5
—	LSTM layers	1
n	Reduced state dimension	200
—	LSTM SGD steps	1e4
—	LSTM SGD learning rate	1e-3
—	LSTM SGD batch size	10
N	# trajs. in dataset	100

E. Implementation

For the RNN reduced-order dynamics model f_θ, we use the long short-term memory (LSTM) architecture [23]. Numerical values of the architectural and training hyperparameters are listed in Table I. We train the RNN in PyTorch [27]. We also solve the model-predictive control problems and implement the EKF in PyTorch. We run the MPC control loop at $40\,\mathrm{Hz}$.

IV. EXPERIMENTS

In all experiments, our deformable body is a thin cylinder of uniform closed-cell polyethylene foam, commonly known as a pool noodle, with length $1.5\,\mathrm{m}$ and diameter $6.5\,\mathrm{cm}$, rigidly attach to a Franka Emika Panda robot end effector. To track the object, we attach a rigid assembly of motion capture markers and track its full pose with a Vicon motion capture system. Our experimental setup is shown in Figure 1.

A. MPC tracking

We apply our method to track several test trajectories which attempt to expose the controller's performance with regard to resonant dynamics. The goal trajectories $z[1], \ldots, z[K] \in \mathbb{R}^3$ are specified by the user. As described in eq. (1), our tracking cost $W = \mathrm{diag}(0, 1, 1)$ is non-isotropic. to focus only on tracking in the Y- and Z-axes.

We show the results from three trajectories in Figure 3. The first two trajectories are a circle of diameter $0.6\,\mathrm{m}$ and a figure-8 Lissajous curve of width $1\,\mathrm{m}$ and height $0.4\,\mathrm{m}$. To expose open-loop control deviations, the circle and the vertical axis of the Lissajous curve are set to the system's resonant frequency $0.8\,\mathrm{Hz}$ as determined experimentally. The

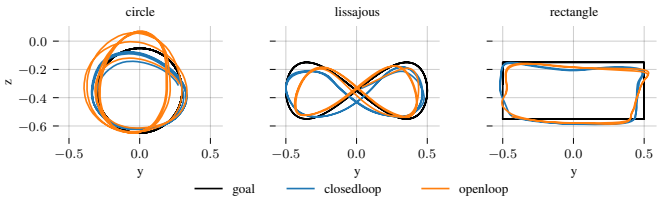

Fig. 3. Two-dimensional projections of paths traced by pool noodle free end in MPC tracking experiments.

TABLE II

MPC TRACKING ERRORS

shape	kind	max error (cm)	mean error (cm)
circle	closedloop	12.59	5.59
	openloop	29.15	11.61
lissajous	closedloop	9.19	4.43
	openloop	14.85	4.77
rectangle	closedloop	14.60	6.34
	openloop	14.62	6.01

third trajectory is a rectangle with constant linear velocity throughout.

To show that the EKF observer provides meaningful feedback to the controller (the "closed-loop" setup), we compare it to a setup that assumes the predicted feedforward state, i.e. the value yielded by applying the RNN dynamics f_{θ^*} to the full sequence of past inputs, is always correct (the "open-loop" setup). The results of this comparison are visualized in Figure 3, and the tracking errors are compared numerically in Table II.

For the circle trajectory, we observe significantly improved performance in both mean and maximum error from the closed-loop setup. Over time, the open-loop solution drifts towards stronger resonance in the vertical axis and weaker resonance in the horizontal axis. In contrast, the error of the closed-loop solution does not grow over time, indicating that our EKF setup is able to compensate for model error. For the Lissajous trajectory, the closed-loop setup yields a minor improvement in mean tracking error but a significant improvement in maximum error. The rectangle trajectory shows little difference between the open- and closed-loop approaches, but it is noteworthy that both approaches track the rectangle reasonably, even though it is physically infeasible and not similar to the training data. This result suggests that the LSTM model behaves reasonably for sequence inputs that are dissimilar to those in the training data.

V. CONCLUSION

We have described and demonstrated a system to manipulate a deformable object such that a particular point on the object tracks a fast trajectory. Our approach is completely data-driven, and requires a fixed initial data-collection phase, without further exploratory actions. We model our dynamical system as an LSTM recurrent neural network and design a closed-loop nonlinear MPC controller, with an EKF state observer. We validate our model on a real hardware setup with a robot manipulator holding a foam pool noodle, measured by a motion capture system. Our experiments show that closing the loop with the EKF observer improves tracking performance compared to an open loop control scheme for several of the test trajectories.

In future work, we aim to improve tracking accuracy by investigating other nonlinear state estimation methods and MPC implementations.

REFERENCES

[1] 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.

[2] J. Zhu, A. Cherubini, C. Dune, D. Navarro-Alarcon, F. Alambeigi, D. Berenson, F. Ficuciello, K. Harada, X. Li, J. Pan, and W. Yuan, "Challenges and outlook in robotic manipulation of deformable objects," *CoRR*, vol. abs/2105.01767, 2021.

[3] G. A. Holzapfel, "Nonlinear solid mechanics: A continuum approach for engineering science," *Meccanica*, vol. 37, no. 4, pp. 489–490, Jul. 2002.

[4] 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, Apr. 2021, conference Name: IEEE Robotics and Automation Letters.

[5] J. M. Bern, P. Banzet, R. Poranne, and S. Coros, "Trajectory optimization for cable-driven soft robot locomotion." in *Robotics: Science and Systems*, vol. 1, 2019, issue: 3.

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

[7] Y. Li, T. Du, K. Wu, J. Xu, and W. Matusik, "DiffCloth: Differentiable cloth simulation with dry frictional contact," *arXiv preprint arXiv:2106.05306*, 2021.

[8] Y.-L. Qiao, J. Liang, V. Koltun, and M. C. Lin, "Scalable differentiable physics for learning and control," *arXiv preprint arXiv:2007.02168*, 2020.

[9] E. Heiden, M. Macklin, Y. Narang, D. Fox, A. Garg, and F. Ramos, "DiSECt: A differentiable simulation engine for autonomous robotic cutting," *arXiv preprint arXiv:2105.12244*, 2021.

[10] J. Barbič and J. Popović, "Real-time control of physically based simulations using gentle forces," *ACM transactions on graphics (TOG)*, vol. 27, no. 5, pp. 1–10, 2008.

[11] M. Thieffry, A. Kruszewski, C. Duriez, and T.-M. Guerra, "Control design for soft robots based on reduced-order model," *IEEE Robotics and Automation Letters*, vol. 4, no. 1, pp. 25–32, 2018, publisher: IEEE.

[12] D. Hahn, P. Banzet, J. M. Bern, and S. Coros, "Real2sim: Visco-elastic parameter estimation from dynamic motion," *ACM Transactions on Graphics (TOG)*, vol. 38, no. 6, pp. 1–13, 2019, publisher: ACM New York, NY, USA.

[13] D. Sulsky, Z. Chen, and H. L. Schreyer, "A particle method for history-dependent materials," *Computer Methods in Applied Mechanics and Engineering*, vol. 118, no. 1, pp. 179–196, Sep. 1994.

[14] Y. Hu, J. Liu, A. Spielberg, J. B. Tenenbaum, W. T. Freeman, J. Wu, D. Rus, and W. Matusik, "ChainQueen: A real-time differentiable physical simulator for soft robotics," *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, 2019.

[15] M. Macklin, M. Müller, and N. Chentanez, "XPBD: position-based simulation of compliant constrained dynamics," in *Proceedings of the 9th International Conference on Motion in Games*, 2016, pp. 49–54.

[16] M. Muller, B. Heidelberger, M. Teschner, and M. Gross, "Meshless deformations based on shape matching," p. 8.

[17] N. M. Mirza, "Machine learning and soft robotics," in *2020 21st International Arab Conference on Information Technology (ACIT)*, Nov. 2020, pp. 1–5.

[18] J. M. Bern, Y. Schnider, P. Banzet, N. Kumar, and S. Coros, "Soft robot control with a learned differentiable model," in *2020 3rd IEEE International Conference on Soft Robotics (RoboSoft)*, May 2020, pp. 417–423.

[19] D. Bruder, B. Gillespie, C. D. Remy, and R. Vasudevan, "Modeling and control of soft robots using the koopman operator and model predictive control," *arXiv:1902.02827 [cs]*, Jul. 2019, arXiv: 1902.02827.

[20] M. T. Gillespie, C. M. Best, E. C. Townsend, D. Wingate, and M. D. Killpack, "Learning nonlinear dynamic models of soft robots for model predictive control with neural networks," in *2018 IEEE International Conference on Soft Robotics (RoboSoft)*, Apr. 2018, pp. 39–45.

[21] A. P. Sabelhaus and C. Majidi, "Gaussian process dynamics models for soft robots with shape memory actuators," in *2021 IEEE 4th International Conference on Soft Robotics (RoboSoft)*, Apr. 2021, pp. 191–198.

[22] S. Tonkens, J. Lorenzetti, and M. Pavone, "Soft robot optimal control via reduced order finite element models," *arXiv preprint arXiv:2011.02092*, 2020.

[23] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997, publisher: MIT Press.

[24] Z. C. Lipton, "A critical review of recurrent neural networks for sequence learning," *CoRR*, vol. abs/1506.00019, 2015.

[25] F. Allgöwer and A. Zheng, *Nonlinear model predictive control*. Birkhäuser, 2012, vol. 26.

[26] R. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960, publisher: Citeseer.

[27] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "PyTorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d. Alché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035.