Modeling of a Bullwhip Using a NARX Network for Robot Control

Mahdiar Edraki*, Reza Sharif Razavian†‡, Mohsen Sadeghi†‡, Aleksei Krotov§ and Dagmar Sternad†‡¶‖
*Departments of Mechanical and Industrial Engineering, †Biology, ‡Electrical and Computer Engineering,
§Bioengineering, ¶Physics, and ‖Institute for Experiential Robotics, Northeastern University, Boston, MA
Email: {edraki.m, r.sharifrazavian, m.sadeghi, krotov.a, d.sternad}@northeastern.edu

***Abstract*—Modeling of deformable objects is essential for a robot to successfully interact with the complex objects present in the human world. Traditional methods developed to model and control rigid objects are not viable for robotic manipulation of flexible objects. Newly introduced techniques to model deformable objects typically require long computation times either to develop or to deploy the model during run time. This study introduced a data-driven approach using a nonlinear autoregressive neural network with exogenous inputs (NARX) to obtain an inverse model of a flexible object that estimates the appropriate inputs to the controller based on a desired system output. The efficacy of this approach is demonstrated at the example of manipulating a whip. We show that the NARX network can estimate appropriate inputs to the whip handle to generate movements of the tip of the whip along a user-defined trajectory. With training on 50 seconds of data, it achieves ∼90% accuracy in a new test condition.**

I. Introduction

As robots enter our work spaces and homes, they increasingly come in contact with a variety of complex objects that have novel dynamical behaviors. Modeling and control of deformable objects is necessary for a robot to successfully act and interact in the human environment. Over the past few decades, many algorithms have been introduced to model deformable objects using finite-element modeling (FEM) [6], deep reinforcement learning (DeepRL) [2], and physics-based modeling [4]. These techniques provide adequate models of flexible objects, but the computation times needed to develop (DeepRL models) or use the models at run time (FEM and physics-based models) are extremely long. This toll on computational resources makes them infeasible for real-time modeling of novel deformable objects in robot control. In contrast, humans can learn to manipulate objects with novel dynamics within a few iterations [3]. Therefore, it has been argued that humans may not need a physically accurate internal model to interact with objects for their movement control [1][5].

More recent approaches have been proposed that can model novel objects with shorter computation time. For example, Yan et al. [8] introduced a vision-based state estimation algorithm to model and manipulate a flexible string. Nair et al. [7] succeeded to train a robot to manipulate a rope with only few human demonstrations. While these studies made important advances in modeling deformable objects, they have been limited to cases where the object makes relatively slow movements. Quick and possibly even chaotic movements are fundamental kinematic characteristics of moving flexible objects, such as waving a flag or cracking a bullwhip.

This study proposes a data-driven approach to rapidly develop a 3-dimensional model of a deformable object. We used a nonlinear autoregressive neural network with exogenous inputs (NARX) to create an inverse model of a simulated bullwhip. The whip has 22 degrees of freedom (DOFs), with only 2 actuated DOFs at the handle. This makes the whip highly underactuated, resulting in an ill-posed inverse problem, where multiple input trajectories may yield the desired outcome.

We show that with only a small training data set, the NARX network can estimate an appropriate input trajectory to the whip handle to drive the tip of the whip close to a user-defined trajectory. Unlike deep neural networks used in DeepRL, our proposed algorithm only has 2 layers and therefore requires significantly shorter training and computation time. Hence, this approach makes it feasible to acquire a model online while controlling the complex object.

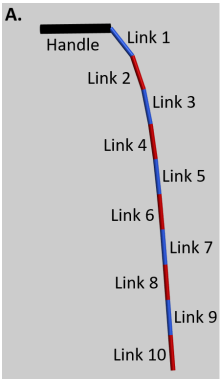
II. Method

A. Simulated Whip

A 1.8m long whip model with a total mass of 2.8kg was generated using the Simscape Multibody Toolbox in MATLAB R2019b (MathWorks, Natick, MA). As shown in Fig. 1A, the whip was modeled as 10 rigid cylindrical links, with each link-pair connected by 2 orthogonal revolute joints. To mimic the tapering of a real whip, the stiffness, damping, and mass of the system decreased linearly from the handle to the tip. Fig. 1B lists the stiffness, damping, and mass of the sections of the whip. These parameters were chosen through trial and error to elicit whip-like behavior.

The base of the whip handle was held stationary and the input to the model were the azimuth and altitude angles of

A.



B.

Link	Length (m)	Mass (kg)	Stiffness (Nm)	Damping (Nm/s)
Handle	0.30	-	-	-
1	0.15	0.50	3.00	1.00
2	0.15	0.42	2.50	0.84
3	0.15	0.38	2.25	0.76
4	0.15	0.34	2.00	0.68
5	0.15	0.30	1.75	0.60
6	0.15	0.26	1.50	0.52
7	0.15	0.22	1.25	0.44
8	0.15	0.18	1.00	0.36
9	0.15	0.14	0.75	0.28
10	0.15	0.10	0.50	0.20

Fig. 1. Simulated whip in MATLAB Simscape environment. A: Whip shape when the handle is held horizontally. B: Mechanical properties of the whip.

the whip handle. Based on the user inputs to the handle, the Simscape's physics engine simulated the behavior of the whip and the 3D positions of the 10 links of the whip were sampled at 100 Hz.

B. NARX Network Structure

The NARX network was implemented using the Deep Learning Toolbox from MATLAB R2019b (MathWorks, Natick, MA). It is a 2-layer neural network consisting of a hidden layer and an output layer as shown in Fig. 2. The hidden layer used a symmetric sigmoid activation function, while the output layer used a linear activation function. The Bayesian regularization backpropagation was implemented with the function (`trainbr`) as the optimization algorithm to update the weights and biases of the network.

The NARX network consisted of 5 neurons, 3-step input delays, and a 1-step feedback delay. The hidden layer takes in the known desired inputs, $\mathbf{u}(t)$, with user-defined input delays, ID, as well as previously estimated values of the desired outputs, $\widehat{\mathbf{y}}(t)$, with user-defined feedback delays, FD, to estimate future values of the outputs.

C. Training and Testing of the NARX Network

The 22 DOF of the whip included 2 actuated DOFs: the handle's azimuth and altitude angle. The azimuth trajectory for both the training and testing stages was a sine function with 0.26 Hz frequency and 25 deg amplitude. The altitude trajectory was a sine function with 0.46 Hz frequency and a 20 deg amplitude for the training data and a 40 deg amplitude for the testing data set. These input parameters were chosen through trial and error as they prevented the whip from folding onto itself; folding caused the simulator fail to converge. The training and testing data sets consisted of 50 s, and 2 s of whip data, respectively.

Fig. 3 shows the flow chart used to develop and evaluate the inverse model of the whip. The pre-defined handle inputs for the training and testing data sets were used to generate the behavior of the whip using Simscape Multibody. In the training stage, the NARX model used the training data to estimate the handle inputs that drove the whip tip along a desired trajectory. In the testing stage, the trained NARX model estimated the appropriate handle trajectories for the whip tip to closely follow the desired testing data set trajectory.

In the evaluation stage, the estimated handle trajectories were passed to Simscape to simulate the response of the whip. The accuracy of the model was evaluated by calculating the Euclidean difference between the 3D tip trajectory from the simulation and the test data, normalized over the maximum tip displacement. Note, only the tip of whip and not the links was used for evaluation.

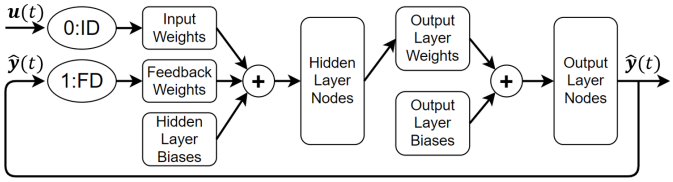


Fig. 2. NARX network architecture.

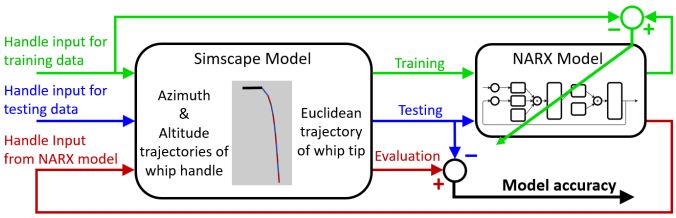


Fig. 3. Flow chart used to estimate the appropriate handle angle inputs to manipulate the whip along the user-defined trajectory.

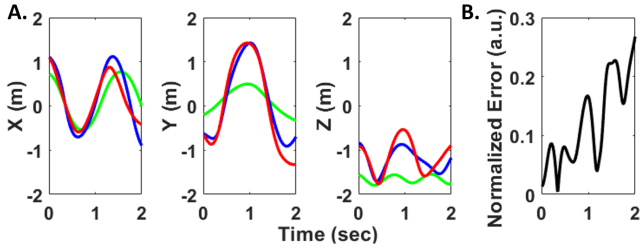


Fig. 4. A: 2-second Euclidean trajectory of the whip tip simulated using handle inputs from the training data set (green), testing data set (blue), and the NARX model estimate (red). B: Normalized Euclidean error of the whip tip trajectory obtained from the NARX model.

III. Results & Discussion

Fig. 4A shows a 2-second sample of the simulated Euclidean trajectory of the whip tip from the training (green) and the testing (blue) data, as well as the simulation result of the estimated NARX-model handle inputs (red line). Fig. 4B shows the normalized Euclidean distance between the tip of the whip and the NARX-model estimate compared to the desired trajectory from the testing data set. The mean normalized Euclidean error using the NARX model was 12%. Given a user-defined trajectory, the model output had an average error of 0.12 m for every 1 m movement of the tip of the whip.

Fig. 4A shows an example of the nonlinear behavior of the whip. Even though the handle input trajectory of the training and testing data sets were very similar, the resulting behavior of the whip was significantly different. This could explain the increase in the Euclidean error observed in Fig. 4B as time progresses. Small differences in the input can result in different behaviors of the whip over time, demonstrating the challenge of inverse modeling of deformable objects. Despite these significant difficulties, our proposed NARX modeling approach made it possible to obtain a solution to the ill-posed inverse model problem with nearly 88% accuracy.

The additional noteworthy benefit of this approach is its extremely short computation time to arrive at this solution. The NARX model took approximately 5 minutes to train on an Intel Core i7-8700K CPU, and the trained model could estimate new whip handle inputs in less than 1 s. These fast computations make it possible for robots to develop and modify models of deformable objects almost in real time while they interact with new and unfamiliar objects.

Future work will go beyond the simplified handle movements and demonstrate the efficacy of our approach in the task of hitting a target with the whip. This challenging task involves extremely fast hand actions generating complex whip movement patterns that can further differentiate our approach from existing techniques.

REFERENCES

[1] Salah Bazzi and Dagmar Sternad. Human control of complex objects: towards more dexterous robots. *Advanced Robotics*, 34(17):1137–1155, 2020. URL https://doi.org/10.1080/01691864.2020.1777198.

[2] Alexander Clegg, Wenhao Yu, Jie Tan, C. Karen Liu, and Greg Turk. Learning to Dress: Synthesizing Human Dressing Motion via Deep Reinforcement Learning. *ACM Trans. Graph.*, 37(6), December 2018. ISSN 0730-0301. doi: 10.1145/3272127.3275048.

[3] A. M. Gordon, G. Westling, K. J. Cole, and R. S. Johansson. Memory representations underlying motor commands used during manipulation of common and novel objects. *Journal of Neurophysiology*, 69(6):1789–1796, 1993. doi: 10.1152/jn.1993.69.6.1789.

[4] A.m. Howard and G.a. Bekey. Intelligent learning for deformable object manipulation. *Proceedings 1999 IEEE International Symposium on Computational Intelligence in Robotics and Automation. CIRA99 (Cat. No.99EX375)*. doi: 10.1109/cira.1999.809935.

[5] Aleksei Dmitrievich Krotov. Human control of a flexible object. 2020. doi: 10.17760/d20385583.

[6] Miao Liao, Qing Zhang, Huamin Wang, Ruigang Yang, and Minglun Gong. Modeling deformable objects from a single depth camera. In *2009 IEEE 12th International Conference on Computer Vision*, pages 167–174, 2009. doi: 10.1109/ICCV.2009.5459161.

[7] Ashvin Nair, Dian Chen, Pulkit Agrawal, Phillip Isola, Pieter Abbeel, Jitendra Malik, and Sergey Levine. Combining self-supervised learning and imitation for vision-based rope manipulation. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2146–2153, 2017. doi: 10.1109/ICRA.2017.7989247.

[8] Mengyuan Yan, Yilin Zhu, Ning Jin, and Jeannette Bohg. Self-Supervised Learning of State Estimation for Manipulating Deformable Linear Objects. *IEEE Robotics and Automation Letters*, 5(2):2372–2379, 2020. doi: 10.1109/LRA.2020.2969931.