Estimating Mass Distribution of Articulated Objects using Non-prehensile Manipulation

K. Niranjan Kumar
Georgia Institute of Technology
niranjankumar@gatech.edu

Irfan Essa
Georgia Institute of Technology
irfan@gatech.edu

Sehoon Ha
Georgia Institute of Technology
sehoonha@gatech.edu

C. Karen Liu
Stanford University
karenliu@cs.stanford.edu

Abstract

We explore the problem of estimating the mass distribution of an articulated object by an interactive robotic agent. Our method consists of two neural networks: (i) the policy network which decides how to interact with the object, and (ii) the predictor network that estimates the mass distribution given a history of observations and interactions. Using our method, we train a robotic arm to estimate the mass distribution of an object with multiple moving parts (e.g. an articulated rigid body system) by pushing it on a surface with unknown friction properties and demonstrate how our training from simulation can be transferred to real hardware by fine-tuning on a small amount of real-world. We use a UR10 robot to interact with 3D printed articulated chains with varying mass distributions and show that our method significantly outperforms the baseline system that uses random pushes.

1 Introduction

Infants engage in exploratory play to discover non-obvious physical properties of objects [2] and develop a core knowledge in long-term cognition about the physical world [17]. Can we teach a robot to engage in similar exploration with a new object in order to learn more about it? In this work, we develop an interactive agent to estimate the mass distribution of an articulated object by pushing it around on a surface. We design a simulated world, Figure 1 (Left) and a real robot setup, Figure 1 (Middle), where a robotic arm can interact with an articulated chain by pushing it at any point on any segment. While estimating mass using physical interaction has been previously studied [25, 31, 26], our problem is uniquely challenging in two aspects. First, the Euclidean distance between two equilibrium states of an articulated object is insufficient to infer the mass of each moving segment. Second, we do not expect the robot to be able to push or track the state of a moving object, a challenging task in the real world, where the end-effector can partially occlude the moving object making such information inaccessible. Instead, our method only uses the information acquired when the object comes to a complete stop (an equilibrium state). We propose a dual network architecture that consists of (i) a policy network and (ii) a predictor network working in tandem towards the goal. The predictor observes the reaction of the object to the pushes imparted, and predicts the mass distribution while the policy learns to push the object so that useful information can be extracted.

Our main contribution is to demonstrate that embodied learning of physical parameters of a system benefits from an intelligent interaction policy. Furthermore, we show that not all actions are created equal when an agent interacts with the intention to discover the physical properties of an object. By comparing to a baseline random interaction policy, we show that our agent learns to exploit the subset of informative actions to significantly improve the accuracy of predictions.

NeurIPS 2020 Workshop on Object Representations for Learning and Reasoning, Vancouver, Canada.

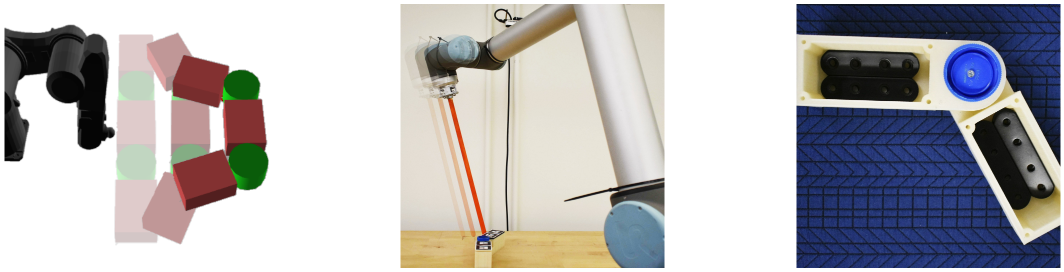

Figure 1: An example interaction in the Simulated world **(left)**. Example of a push (in two frame time-lapse) that can be executed by the robot. The chain loses contact with the end-effector (painted orange in the picture) and continues sliding on the surface briefly before coming to rest **(middle)**. We vary the effective mass of each link by adding iron braces to the mass chamber **(right)**.

2 Related work

Humans utilize an intuitive physics model to reason about the world, and predict its change in the near future [14, 10]. Building such a physical model has traditionally been approached as a system identification problem [18], which has the potential to significantly improve the control policies operated in the real world. Recent advances in deep learning methods advocate a new approach towards predicting the future state of the world directly from the recent observation history [3, 6, 19, 28, 1, 24, 8, 16]. While directly predicting the next state is a powerful tool, some applications focus on identifying specific physical parameters useful for control algorithm analysis, or developing environment-conditioned control policies [30, 32, 27, 21, 22, 5, 12, 13]. Our work, like those discussed above, falls under the wide umbrella of Interactive Perception [4]. While previous approaches that deal with articulated objects infer the kinematic structure, our work focuses on the more difficult problem of estimating the mass distribution.

Predicting the mass of an object has been of particular interest in the subarea of robotic manipulation [29, 25, 31]. A robotic arm was used to generate random pushes for estimating the mass and friction coefficients of rigid objects in [26]. Although this can work for simple rigid objects, we show that using a random probing policy typically results in poor estimation of the mass distribution in case of articulated objects with multiple moving parts. Instead, we train a policy that strategically pushes (exclusively non-prehensile manipulation) the object to extract maximum information for estimating its mass distribution, without any grasping (prehensile manipulation) or auxiliary objects as in [26].

3 Method

We introduce a method to enable a robot to estimate the mass distribution of an articulated object by strategically pushing it on a flat surface. A conventional approach to this problem is to conduct system identification such that the observed data is consistent with the equations of motion. However, since we only observe the object and apply the actions at equilibrium states, we cannot utilize dynamic equations to infer masses in the absence of derivative information (e.g. velocity, acceleration). Further, effective system identification also requires analysis on persistent excitation, continuous forces/torques applied to generate continuous motion, to ensure that the observations span a wide range of system behaviors. Although the object in our problem is a passive multibody system, the mass parameters can only be uniquely identified by a subset of observation and action trajectories (see Supplementary file). This problem is further exacerbated in practical scenarios where the robot is only expected to interact with the object a few times.

Consider an articulated chain with n rigid bodies connected by revolute joints. Let $\boldsymbol{q}$ be the observed configuration of the object, and $\boldsymbol{a}$ represent the push (magnitude and direction) the robot exerts on it. Starting from the initial configuration $\boldsymbol{q}_0$, the robot applies a push $\boldsymbol{a}_0$ and observes the object after it comes to a complete stop, at $\boldsymbol{q}_1$. After repeating this process K times, the robot infers the mass distribution $\boldsymbol{m} \in \mathbf{R}^n$ of the object, where each element of $\boldsymbol{m}$ indicates the mass of a rigid body normalized by the total mass. We predict the mass distribution instead of the actual mass, so that our method can be agnostic to the surface friction. With this problem setting, we learn a predictor f_μ using supervised learning. The predictor takes as input the belief from the previous

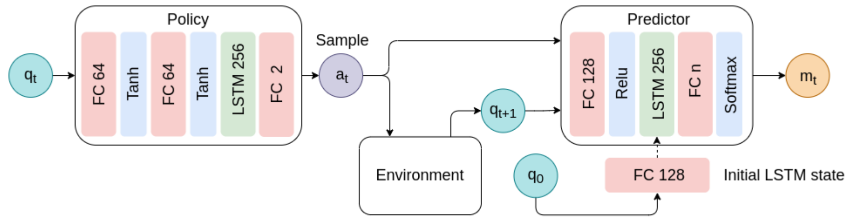

Figure 2: Architecture of our proposed method with Predictor and Policy networks

step $\boldsymbol{h}_t = f_h(\boldsymbol{q}_0, \boldsymbol{a}_0, \cdots, \boldsymbol{a}_{t-1}, \boldsymbol{q}_t)$, the current action $\boldsymbol{a}_t$, and the next observation $\boldsymbol{q}_{t+1}$, to predict the mass distribution $\boldsymbol{m}_t$ at the current step t. We use a Long Short-Term Memory (LSTM) network (Figure 2) parameterized by $\boldsymbol{\mu}$ to model the predictor and define a loss function as follows:

$$\mathcal{L}(\mu) = \frac{1}{N}\frac{1}{K}\sum_{i=1}^{N}\sum_{t=0}^{K}\|\boldsymbol{m}^i - f_\mu(\boldsymbol{h}_t, \boldsymbol{a}_t^i, \boldsymbol{q}_{t+1}^i)\|_2, \tag{1}$$

where superscript i is the index of the N training samples. Minimizing Equation 1 results in a predictor network that achieves high accuracy when the input trajectory contains sufficient information to identify the mass distribution, i.e. , there is a unique mass distribution consistent with the observations induced by given actions. However, as discussed earlier, some input trajectories cannot uniquely identify the mass distribution due to the singularity in the dynamic system, and sparse observations and actions. As a result, the predictor performs poorly on predicting the mass distributions for those regions in the input space.

Given a model that predicts mass distribution from trajectories, what should those trajectories be to get an accurate estimate? We model this problem as a Partially Observed Markov Decision Process (POMDP) where the underlying state $\boldsymbol{s_t}$ of the system is defined to be a combination of the pose of the object $\boldsymbol{q_t}$, actual system parameters $\bar{\boldsymbol{m}}$ and predictor's belief $\boldsymbol{h_t}$ of the actual system parameters. The input that is available to the agent, however, is just the pose of the object $\boldsymbol{q_t}$ and not the complete state. Following [23] we build our policy with an LSTM that can model sufficient statistics $\hat{\boldsymbol{h}}_{\boldsymbol{t}}$ of past inputs. The objective of our policy is to produce trajectories that can be efficiently reasoned by the predictor to give an accurate estimate of the mass distribution. To this end, we design a reward function that encourages actions that help the predictor achieve a lower error:

$$r(\boldsymbol{s}_t, \boldsymbol{a}_t, \boldsymbol{s}_{t+1}) = 1 - \beta\|\bar{\boldsymbol{m}} - f_\mu(\boldsymbol{h}_t, \boldsymbol{a}_t, \boldsymbol{q}_{t+1})\|_1, \tag{2}$$

where $\bar{\boldsymbol{m}}$ is the ground truth mass distribution and β is a multiplicative constant. We design the reward function to be proportional to the negative of the error in mass distribution estimated by the predictor network, and scale its output to be between $[-1, 1]$ with β. Intuitively, learning a policy can be seen as adapting the input data distribution to operate on low error regions of the predictor. We use a policy gradient method, Proximal Policy Optimization [20] to solve for the policy $\pi_\theta(\boldsymbol{a}|\hat{\boldsymbol{h}}_{\boldsymbol{t}})$, where π is represented as a LSTM neural network parameterized by θ with hidden state $\hat{\boldsymbol{h}}_{\boldsymbol{t}}$ that evolves through time with every new $\boldsymbol{q}_t$. In practice, we find that alternating between training the predictor and the policy a few times results in predicting the mass distribution more accurately by gradually refining the training distribution for the predictor from a uniform distribution to a task-relevant one, induced by the policy.

4 Experiments

We evaluate our method on a simulated KUKA robotic arm using, DART [15], as well as on a real UR10 robot and compare our method to a predictor trained with trajectories from a random interaction policy. The predictor network consists of fully connected and LSTM layers as shown in Figure 2. For the initial training of the predictor, we collect $32k$ and $64k$ episodes for the two-link and three-link chains respectively, by running a random policy that outputs a uniformly sampled action. This trained predictor is used as a part of the reward function for the policy training step. The policy is trained using Proximal Policy Optimization (PPO) [20, 11] with an entropy loss term to promote exploration. During this step the weights of the predictor network are frozen and only the policy weights are trained to maximize the expected reward and in-turn minimize prediction error. In

	2-Link $[0.1, 1.0]$	2-Link $[1.0, 10]$	2-Link $[0.1, 10]$	3-Link $[0.1, 1.0]$	3-Link $[1.0, 10]$	3-Link $[0.1, 10]$
RP	14.9	17.1	26.2	29.09	31.05	54.78
TP(Ours)	**6.07**	**8.26**	**12.03**	**12.02**	**15.99**	**23.93**

Table 1: Comparison of the percentage error of our method to the baseline.

our experiments, we converge after six meta-iterations of policy training and predictor fine-tuning. We call this the Trained Policy Predictor (TP) model. We also train a Random Policy Predictor (RP) with $224k$ episodes for two-link and $448k$ episodes for three-link chain and use it as the baseline to compare our method against. The dataset size is chosen such that both TP predictor (at the end of the $6th$ meta iteration) and RP predictor are trained on the same amount of data. RP predictor models are similar to [26] where the action taken is independent of the configuration of the object.

We evaluate our model in our simulated world with a KUKA KR 5 Sixx R650 Robotic arm in DART with articulated chains of rigid cuboid segments connected by revolute joints. We experiment with 2-link and 3-link chains, with the link masses uniformly sampled from three mass ranges (values reported here in kg): $[0.1, 1]$, $[1, 10]$ and $[0.1, 10]$. A wider range is more challenging because the set of informative pushing forces needs to be more diverse. When the robot uses a large force on a light object, it might rapidly rotate over many cycles with self-collisions, which is lost in the training set because the observation only records the equilibrium state, rather than the entire trajectory. We also randomize the friction coefficient from a range of $[0.5, 1]$ and the length of each segment from $[0.1, 0.15]$ meters. We add Gaussian noise to the observations and actions to ensure that the networks trained in simulation can be reliable transferred to the real world. We find that having a policy that strategically pushes with the intention of minimizing prediction error outperforms the random policy across all cases as seen in Table 1. This observation falls in line with evidence from neuroscience community on the importance of active inference in learning a model of the world [9, 4].

To evaluate our method in the real world we choose a UR10 robot which has a significantly larger workspace than the KUKA robot and 3D printed two-link chains with a joint range of $[-\pi/2, \pi/2]$. We first use the uniform policy to collect 30 episodes each (150 in total) of the robot interacting with 5 different mass distributions. The objects are visually identical and can be one of the following weight pairs: $[0.16, 0.16]$, $[0.064, 0.192]$, $[0.064, 0.256]$, $[0.192, 0.064]$ and $[0.256, 0.064]$. We simulate the robot in PyBullet [7] and train the networks in simulation on the mass range $[0.01, 0.3]$. The features learnt by the predictor are then fed to a small two-layer fully connected network $[64 - Relu - 32 - Relu - 5]$, trained to classify the real objects into one of the 5 classes by using a cross-entropy loss. We find that the Random Policy Predictor reaches an accuracy of 40.74%, with random chance being 20%. We then collect another 150 episodes with our learned policy and train the classification network using the features from the predictor of the last meta iteration. Our method outperforms the baseline by a large margin achieving an accuracy of 81.4%. We notice that the learned policy tries to push the object such that there is movement about the revolute joint so that inference about individual links is possible. Intuitively this makes sense, as in cases where the joint is locked down, the object behaves like a rigid object and it becomes hard to decouple properties about individual links in the presence of real-world sensing and actuation noise. Although we need significant amount of training data in simulation, the real world experiments show that it is possible to fine-tune the models trained in simulation with reasonable amount of real world data.

5 Discussion

We present an agent that learns to predict the mass distribution of articulated chains by strategically selecting few pushes that extract maximum information. We test our hypothesis by pushing 2-link and 3-link articulated chains with a robotic arm. Our experiments show that our approach consistently outperforms the baseline method and that our model learned in simulation can be transferred to the real world with a small amount of data for fine-tuning. One drawback of our method is that the action space is tightly coupled to the geometry of the object we are trying to manipulate. This requires us to train a separate network for every object type. However, interaction strategies learnt by interacting with one object often transfer to similar object types. In future, we want to design architectures that can be applied across a family of object types by potentially leveraging an image based observation space.

Acknowledgments and Disclosure of Funding

We would like to thank Dr. Matthew Gombolay for letting us use the UR10 robot and Harish Haresamudram for proofreading the paper.

References

[1] Anurag Ajay, Jiajun Wu, Nima Fazeli, Maria Bauza, Leslie P. Kaelbling, Joshua B. Tenenbaum, and Alberto Rodriguez. Augmenting physical simulators with stochastic neural networks: Case study of planar pushing and bouncing. *2018 IEEE/RSJ IROS*, Oct 2018.

[2] Dare A. Baldwin, Ellen M. Markman, and Riikka L. Melartin. Infants' ability to draw inferences about nonobvious object properties: Evidence from exploratory play. *Child Development*, 64(3):711–728, 1993.

[3] Peter Battaglia, Razvan Pascanu, Matthew Lai, Danilo Jimenez Rezende, et al. Interaction networks for learning about objects, relations and physics. In *NIPS*, pages 4509–4517, 2016.

[4] Jeannette Bohg, Karol Hausman, Bharath Sankaran, Oliver Brock, Danica Kragic, Stefan Schaal, and Gaurav S Sukhatme. Interactive perception: Leveraging action in perception and perception in action. *IEEE Transactions on Robotics*, 33(6):1273–1291, 2017.

[5] O. Brock, J. Trinkle, and F. Ramos. *Learning to Manipulate Articulated Objects in Unstructured Environments Using a Grounded Relational Representation*. MITP, 2009.

[6] Michael B Chang, Tomer Ullman, Antonio Torralba, and Joshua B Tenenbaum. A compositional object-based approach to learning physical dynamics. *arXiv preprint arXiv:1612.00341*, 2016.

[7] Erwin Coumans and Yunfei Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning. `http://pybullet.org`, 2016–2019.

[8] Sebastien Ehrhardt, Aron Monszpart, Niloy J Mitra, and Andrea Vedaldi. Taking visual motion prediction to new heightfields. *Computer Vision and Image Understanding*, 181:14–25, 2019.

[9] Karl Friston, Thomas FitzGerald, Francesco Rigoli, Philipp Schwartenbeck, and Giovanni Pezzulo. Active inference: a process theory. *Neural computation*, 29(1):1–49, 2017.

[10] Jessica B Hamrick, Peter W Battaglia, Thomas L Griffiths, and Joshua B Tenenbaum. Inferring mass in complex scenes by mental simulation. *Cognition*, 157:61–76, 2016.

[11] Ashley Hill, Antonin Raffin, Maximilian Ernestus, Rene Traore, Prafulla Dhariwal, Christopher Hesse, Oleg Klimov, Alex Nichol, Matthias Plappert, Alec Radford, John Schulman, Szymon Sidor, and Yuhuai Wu. Stable baselines, 2018.

[12] Dov Katz, Moslem Kazemi, J. Andrew Bagnell, and Anthony Stentz. Interactive segmentation, tracking, and kinematic modeling of unknown 3d articulated objects. *2013 IEEE ICRA*, pages 5003–5010, 2013.

[13] Dov Katz, Andreas Orthey, and Oliver Brock. Interactive perception of articulated objects. In *Experimental Robotics*, pages 301–315. Springer, 2014.

[14] James R Kubricht, Keith J Holyoak, and Hongjing Lu. Intuitive physics: Current research and controversies. *Trends in cognitive sciences*, 21(10):749–759, 2017.

[15] Jeongseok Lee, Michael X. Grey, Sehoon Ha, Tobias Kunz, Sumit Jain, Yuting Ye, Siddhartha S. Srinivasa, Mike Stilman, and C. Karen Liu. DART: Dynamic animation and robotics toolkit. *The Journal of Open Source Software*, 3(22):500, Feb 2018.

[16] Damian Mrowca, Chengxu Zhuang, Elias Wang, Nick Haber, Li F Fei-Fei, Josh Tenenbaum, and Daniel L Yamins. Flexible neural representation for physics prediction. In *Advances in Neural Information Processing Systems*, pages 8799–8810, 2018.

[17] Paul Muentener, Elise Herrig, and Laura Schulz. The efficiency of infants' exploratory play is related to longer-term cognitive development. *Frontiers in Psychology*, 9:635, 2018.

[18] Duy Nguyen-Tuong and Jan Peters. Model learning for robot control: a survey. *Cognitive processing*, 12(4):319–340, 2011.

[19] Senthil Purushwalkam, Abhinav Gupta, Danny Kaufman, and Bryan Russell. Bounce and learn: Modeling scene dynamics with real-world bounces. In *ICLR*, 2019.

[20] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[21] Jurgen Sturm, Vijay Pradeep, Cyrill Stachniss, Christian Plagemann, Kurt Konolige, and Wolfram Burgard. Learning kinematic models for articulated objects. In *IJCAI*, 2009.

[22] Abhishek Venkataraman, Brent Griffin, and Jason J Corso. Learning kinematic descriptions using spare: Simulated and physical articulated extendable dataset. *arXiv preprint arXiv:1803.11147*, 2018.

[23] Daan Wierstra, Alexander Foerster, Jan Peters, and Juergen Schmidhuber. Solving deep memory pomdps with recurrent policy gradients. In *International Conference on Artificial Neural Networks*, pages 697–706. Springer, 2007.

[24] Jiajun Wu, Erika Lu, Pushmeet Kohli, Bill Freeman, and Josh Tenenbaum. Learning to see physics via visual de-animation. In *Advances in Neural Information Processing Systems*, pages 153–164, 2017.

[25] Jiajun Wu, Ilker Yildirim, Joseph J Lim, Bill Freeman, and Josh Tenenbaum. Galileo: Perceiving physical object properties by integrating a physics engine with deep learning. In *Advances in Neural Information Processing Systems 28*, pages 127–135. 2015.

[26] Zhenjia Xu, Jiajun Wu, Andy Zeng, Joshua B Tenenbaum, and Shuran Song. Densephysnet: Learning dense physical object representations via multi-step dynamic interactions. In *RSS*, 2019.

[27] Jiachen Yang, Brenden Petersen, Hongyuan Zha, and Daniel Faissol. Single episode policy transfer in reinforcement learning. *arXiv preprint arXiv:1910.07719*, 2019.

[28] Tian Ye, Xiaolong Wang, James Davidson, and Abhinav Gupta. Interpretable intuitive physics model. In *ECCV*, 2018.

[29] Yong Yu, T. Arima, and S. Tsujio. Estimation of object inertia parameters on robot pushing operation. In *Proceedings of the 2005 IEEE ICRA*, pages 1657–1662, April 2005.

[30] Wenhao Yu, Jie Tan, C Karen Liu, and Greg Turk. Preparing for the unknown: Learning a universal policy with online system identification. *arXiv preprint arXiv:1702.02453*, 2017.

[31] David Zheng, Vinson Luo, Jiajun Wu, and Joshua B Tenenbaum. Unsupervised learning of latent physical properties using perception-prediction networks. *arXiv preprint arXiv:1807.09244*, 2018.

[32] Wenxuan Zhou, Lerrel Pinto, and Abhinav Gupta. Environment probing interaction policies. In *ICLR*, 2019.