# DLO@Scale: A Large-Scale Meta Dataset for Learning Non-Rigid Object Pushing Dynamics

**Robert Gieselmann**
robgie@kth.se
KTH

**Alberta Longhini**
albertal@kth.se
KTH

**Alfredo Reichlin**
alfrei@kth.se
KTH

**Danica Kragic**
dani@kth.se
KTH

**Florian T. Pokorny**
fpokorny@kth.se
KTH

## Abstract

The ability to quickly understand our physical environment and make predictions about interacting objects is fundamental to us humans. To equip artificial agents with similar reasoning capabilities, machine learning can be used to approximate the underlying state dynamics of a system. In this regard, deep learning has gained much popularity but is relying on the availability of large-enough datasets. In this work, we present DLO@Scale, a new dataset for studying future state prediction in the context of multi-body deformable linear object pushing. It contains a large collection of 100 million simulated interactions enabling thorough statistical analysis and algorithmic benchmarks. Our data is generated using a high-fidelity physics engine which simulates complex mechanical phenomena such as elasticity, plastic deformation and friction. An important aspect is the large variation of the physical parameters making it suitable for testing meta learning algorithms. We describe DLO@Scale and present a first empirical evaluation using neural network baselines. More information and videos can be found at https://sites.google.com/view/dloscale.

## 1 Introduction

Understanding and predicting the outcome of actions is an integral part of intelligence. To enable artificial agents with similar reasoning capabilities, internal models must accurately capture the dynamics of an environment. Centuries of physics research has provided the means to describe the behavior of physical bodies using compact equations. While physics models have been proven useful in many engineering applications, they require exact knowledge of the underlying system's states and parameters. Machine learning, on the other hand, learns models given a collection of data samples.

Recently, deep neural networks have shown great potential as general purpose tools for predicting the future state of a physical systems. Several works applied Graph Neural Networks [6] to emulate the dynamics of a physical system [18, 15, 5, 20, 16, 14, 22, 19]. A deep generative model was used in [23] to predict the state of a deformable body given the forces applied on its surface. Neural network video prediction models were developed in [24, 25, 8, 4, 13, 9] which predict future image frames. Recent model-based reinforcement learning architectures use neural networks to approximate the underyling transition dynamics [12, 11, 10, 21]. While impressive results have been achieved, a general intuition from the learning perspective is still missing. Open questions are: How many data samples are needed to achieve a certain prediction accuracy? How many samples are required to adapt to another environment? Which representations and inductive biases are most suitable? Are those findings consistent with respect to the system's physical parameters?

Physical Reasoning and Inductive Biases for the Real World at NeurIPS 2021.

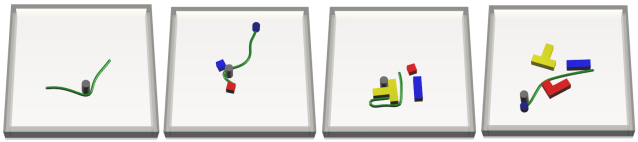

Figure 1: Illustration of pushing environments (deformable object in green and pusher in dark gray).

We present DLO@Scale, a large-scale data collection for evaluating machine learning methods in the context of non-rigid object pushing. The key motivation of our dataset is to enable rigorous large-scale investigations by means of a multi-body pushing problem. We focus on contact interactions between linear deformable and rigid objects that were generated by pushing on a planar surface. From a learning standpoint, predicting the future state of deformable objects is particularly challenging due to complicated mechanics, high-dimensional observations and vast dependency on physical parameters. Our dataset contains more than 100 million interactions simulated using the AGX Dynamics [1] physics engine. A key feature is the large variety of object parameters such as stiffness values for plastic or elastic deformation, friction coefficients, object geometries and mass distributions. This provides an opportunity to study a family of physical systems and investigate various aspects of domain adaptation.

We provide a first experimental evaluation given standard neural network baselines and discuss the dependencies between prediction error, number of training samples and the length of the prediction horizon. Interestingly, our results show significant performance differences between soft, flexible and elastoplastic deformable linear objects.

## 2 A new large-scale dataset of non-rigid pushing interactions

In the following, we present DLO@Scale, a new dataset for state prediction in the setting of multi-body and deformable object pushing. In particular, we study the problem of estimating the state $x_{t+n}$ of a physical system given its current state $x_t$ and a sequence of actions $a_{t:t+n-1}$. In this regard, $t$ determines the time index and $n$ the prediction horizon.

### 2.1 Simulation and data collection

We simulate our pushing environments using the AGX Dynamics [1] high-fidelity physics engine allowing us to model complicated mechanical phenomena such as contact friction, elasticity and plastic deformation. The data was generated over the course of several days by running multiple parallel simulation instances on a high-performance computing cluster at the National Supercomputer Centre in Sweden (NSC) [2]. It contains samples for a large number of domains where each one corresponds to one particular configuration of physical object parameters. Fig. 2 visualizes several example pushing domains. We utilize a box-shaped planar workspace which is surrounded by walls. A new domain is created by sampling from a predefined set of parameters such as type of shapes, mass densities, density distribution, friction, etc (Sec. 2.2). Then, a deformable linear object is randomly initialized inside the box. Several rigid objects are randomly positioned close to the center of the surface. The state of the system is changed by moving a cylindrical pusher on the horizontal plane. We employ a standard PID controller to update the position of the pusher given desired planar displacements $a_t \in \mathbb{R}^2$. Actions $a_t$ are sampled randomly but biased towards the direction of closest deformable object segment. We found this strategy to produce sufficiently rich and non-trivial contact scenarios. After $n_P$ interactions, the trajectory terminates and the simulation is reset.

Our data contains temporal sequences of the poses of all bodies for each interaction step. The simulation runs in headless mode, hence we do not explicitly store images. Yet, it is possible to render scenes post-hoc given the recorded object poses (App. 5.1.1). This allows for easy change of colors, textures, lighting or the camera viewpoints without rerunning the simulation.

### 2.2 Environment and object parameters

One of the key aspects of DLO@Scale is the large variety of physical parameters resulting in different domains. We distinguish between three types of parametes *Environment*, *Deformable Object* and *Rigid Object*:

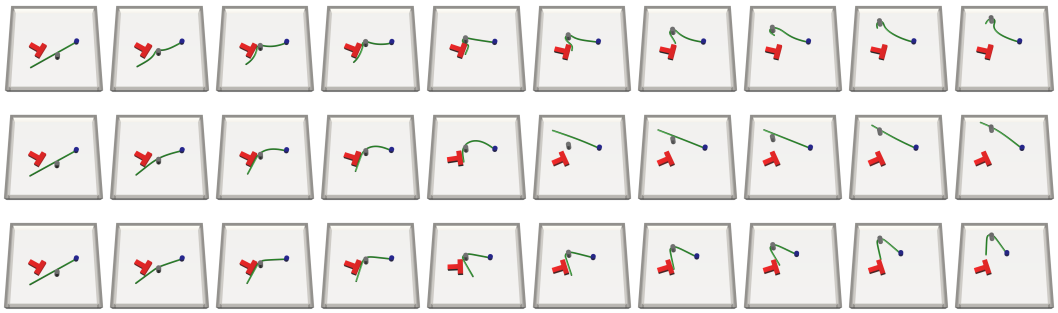

Figure 2: Influence of the material type (top row: soft, middle row: flexible, last row: elastoplastic).

### Environment

- **Surface friction coefficient:** Dimensionless value between 0.2 (low friction) and 0.8 (high friction).
- **Number of rigid objects:** Each domain contains between 0 and 3 rigid objects using the shapes in Fig. 5.

### Deformable Object

- **Length**: Resting length of the object. Integer values between 10mm and 16mm.
- **Density**: Mass density in $\frac{kg}{m^3}$. Ranges between $700\,\frac{kg}{m^3}$ and $1400\,\frac{kg}{m^3}$
- **Material type**: We distinguish between *soft*, *flexible*, *elastoplastic* materials. The *soft* material is characterized by a low stiffness and behaves similar to a rope. The *flexible* type has a higher stiffness and typically snaps back to its original shape if applied forces are removed. The *elastoplastic* material exhibits elastic and plastic properties, i.e non-reversible deformations. Fig. 2 illustrates the effect of different material types given identical pusher trajectories.
- **Attachment:** We enable the possibility to attach a cylindrical body to one end of the deformable object. The cylindrical object is either *movable* or *fixed*, the latter meaning that it's rigidly attached to the surface.

### Rigid object

- **Shape:** Rigid objects are implemented as composite bodies consisting of simple box-shaped geometries. An overview of all possible shapes is given in App. 5.1.2.
- **Density distribution:** The composite structure of rigid objects allows us to specify mass density parameters independently for each sub-geometry. Densities take values between $1200\,\frac{kg}{m^3}$ and $8000\,\frac{kg}{m^3}$ roughly corresponding to the weight of plastic materials respectively steel.
- **Friction:** Dimensionless value between 0.2 (low friction) and 0.8 (high friction).

### 2.3 Dataset partition

Our dataset is divided into the subsets **single-domain-large**, **single-domain-medium** and **meta-learning**. The **single-domain-large** set consists of 18 domains each associated with $10^6$ training pushes. The parameter variations are limited to the *material type*, *attachment* and *number of rigid objects*. Our idea is to enable rigorous statistical analysis by providing a large number of samples for few selected domains. **single-domain-medium** contains $10^5$ pushes for 27 different domains and considers different *length* of the deformable object. The **meta-learning** set comprises 10000 pushes for 6300 domains using the variation of all parameters in Sec. 2.2. We further improve diversity by sampling elasticity and plasticity properties for *soft*, *flexible*, *elastoplastic* within predefined ranges.

Real-world systems often introduce uncertainty due to partial observability or imperfect measurements. To resemble those challenges in simulation, we provide noise-augmented actions and separate test sets to compare generative model predictions against distributions over future states.

## 3 Experiments

We ran a set of initial experiments evaluating standard neural network baselines on our data. The top row in Fig. 3 presents the mean prediction errors for different train set sizes and prediction horizons averaged over all 18 domains in the **single-domain-large** set. As shown, our results confirm the

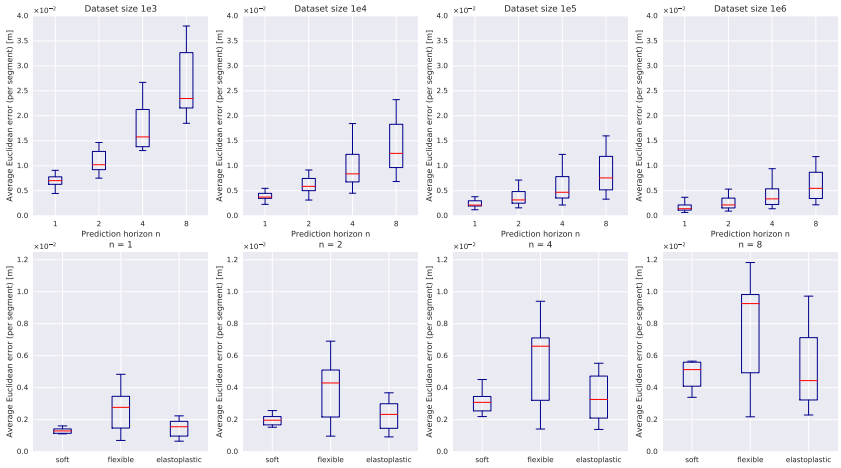


Figure 3: Top row: Dependencies between prediction error, prediction horizon and dataset size (**single-domain-large**). Bottom row: Prediction errors grouped by deformable object material and prediction horizon.

intuition that the difficulty of the task increases with the time horizon. Increasing the number of train samples from $10^5$ to $10^6$ still reduces the overall approximation error even in the single-step setting. The bottom row in Fig. 3 presents the results for a train set of size $10^6$ grouped by the type of deformable object material. Interestingly, we observe lower accuracy and higher variance for the *flexible* material type. One reason might be the difficulty in predicting abrupt state changes introduced by the snapping behavior of flexible rods.

The **meta-learning** set provides a variety of different dynamics influenced by unobservable properties such as the material types or the friction coefficients. We provide initial results for a single model trained on data from different domains and compare it to a gradient-based meta learning method [7]. For this preliminary study, we do not consider additional rigid objects and focus on the prediction of the deformable object.

Table 1 shows a comparison of a standard *multi-task learning* and *meta learning* baseline. The wide range of available data allows to thoroughly evaluate both methods in terms of prediction accuracy. We analyzed the effect of the number of trajectories on the prediction accuracy and the advantages of including an adaptation phase to gain knowledge about unobservable properties. Our results suggest that *meta learning* achieves considerably better performances due to its adaptation phase. Moreover, the performance increases considerably when using multiple trajectories for adaptation. This finding suggests the complexity of the state dynamics in the data.

| Algorithm | **Multi-Task** | | **MAML** | | | | | |
|---|---|---|---|---|---|---|---|---|
| Trajectories | 100 | 500 | 100 | | | 500 | | |
| Adaptation | - | | K = 1 | K = 10 | K = 50 | K = 1 | K = 10 | K = 50 |
| N = 5 | 1.54 | 1.45 | 1.17 | 1.08 | 1.54 | 1.45 | 1.17 | 1.08 |
| N = 50 | 1.85 | 1.56 | 1.30 | 1.21 | 1.85 | 1.56 | 1.30 | 1.21 |

Table 1: MSE in [m]*$10^{-4}$ of the predicted deformable object position (4 steps ahead). N denotes the meta batch size.

## 4 Conclusion

We introduced DLO@Scale, a dataset for large-scale evaluations of physics prediction models in the setting of deformable object pushing. The data was generated using a high-fidelity physics simulator to provide realism of physical interactions. Our intention is foster research at the intersection of physical reasoning and machine learning. In future work, we seek to extend our experimental evaluation by studying different state representations and adding real-world recordings which enable the investigation of sim-to-real adaptation.

## Acknowledgments

This work was supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

The computations were enabled by resources provided by the Swedish National Infrastructure for Computing (SNIC) at the National Supercomputer Centre in Sweden (NSC) partially funded by the Swedish Research Council through grant agreement no. 2018-05973.

This work was also supported by the H2020 CANOPIES project under grant agreement 101016906 and by the European Research Council (BIRD).

## References

[1] AGX Dynamics - Real-time multi-body simulation. `http://www.algoryx.se/agx-dynamics/`, 2021.

[2] "National Supercomputer Centre in Sweden (NSC)". `https://www.nsc.liu.se/`, 2021.

[3] Pyrender. `https://github.com/mmatl/pyrender`, 2021.

[4] Mohammad Babaeizadeh, Chelsea Finn, Dumitru Erhan, Roy H. Campbell, and Sergey Levine. Stochastic variational video prediction. In *International Conference on Learning Representations*, 2018.

[5] Peter Battaglia, Razvan Pascanu, Matthew Lai, Danilo Jimenez Rezende, and koray kavukcuoglu. Interaction networks for learning about objects, relations and physics. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

[6] Peter W. Battaglia, Jessica B. Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinícius Flores Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, Çaglar Gülçehre, H. Francis Song, Andrew J. Ballard, Justin Gilmer, George E. Dahl, Ashish Vaswani, Kelsey R. Allen, Charles Nash, Victoria Langston, Chris Dyer, Nicolas Heess, Daan Wierstra, Pushmeet Kohli, Matthew Botvinick, Oriol Vinyals, Yujia Li, and Razvan Pascanu. Relational inductive biases, deep learning, and graph networks. *CoRR*, abs/1806.01261, 2018.

[7] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, pages 1126–1135. PMLR, 2017.

[8] Chelsea Finn and Sergey Levine. Deep visual foresight for planning robot motion. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2786–2793, 2017.

[9] Jean-Yves Franceschi, Edouard Delasalles, Mickael Chen, Sylvain Lamprier, and Patrick Gallinari. Stochastic latent residual video prediction. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 3233–3246. PMLR, 13–18 Jul 2020.

[10] David Ha and Jürgen Schmidhuber. Recurrent world models facilitate policy evolution. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[11] Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, 2020.

[12] Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2555–2565. PMLR, 09–15 Jun 2019.

[13] Alex X. Lee, Richard Zhang, Frederik Ebert, Pieter Abbeel, Chelsea Finn, and Sergey Levine. Stochastic adversarial video prediction. *arXiv preprint arXiv:1804.01523*, 2018.

[14] Yunzhu Li, Antonio Torralba, Anima Anandkumar, Dieter Fox, and Animesh Garg. Causal discovery in physical systems from videos. volume 33, pages 9180–9192. Curran Associates, Inc., 2020.

[15] Yunzhu Li, Jiajun Wu, Russ Tedrake, Joshua B Tenenbaum, and Antonio Torralba. Learning particle dynamics for manipulating rigid bodies, deformable objects, and fluids. In *ICLR*, 2019.

[16] Yunzhu Li, Jiajun Wu, Jun-Yan Zhu, Joshua B Tenenbaum, Antonio Torralba, and Russ Tedrake. Propagation networks for model-based control under partial observation. In *ICRA*, 2019.

[17] Zhenguo Li, Fengwei Zhou, Fei Chen, and Hang Li. Meta-sgd: Learning to learn quickly for few-shot learning. *arXiv preprint arXiv:1707.09835*, 2017.

[18] 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 S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[19] Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter Battaglia. Learning to simulate complex physics with graph networks. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 8459–8468. PMLR, 13–18 Jul 2020.

[20] Alvaro Sanchez-Gonzalez, Nicolas Heess, Jost Tobias Springenberg, Josh Merel, Martin Riedmiller, Raia Hadsell, and Peter Battaglia. Graph networks as learnable physics engines for inference and control. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4470–4479. PMLR, 10–15 Jul 2018.

[21] Stephen Tian, Suraj Nair, Frederik Ebert, Sudeep Dasari, Benjamin Eysenbach, Chelsea Finn, and Sergey Levine. Model-based visual planning with self-supervised functional distances. In *International Conference on Learning Representations*, 2021.

[22] Benjamin Ummenhofer, Lukas Prantl, Nils Thuerey, and Vladlen Koltun. Lagrangian fluid simulation with continuous convolutions. In *International Conference on Learning Representations*, 2020.

[23] Zhihua Wang, Stefano Rosa, Bo Yang, Sen Wang, Niki Trigoni, and Andrew Markham. 3d-physnet: Learning the intuitive physics of non-rigid object deformations. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 4958–4964. International Joint Conferences on Artificial Intelligence Organization, 7 2018.

[24] Nicholas Watters, Daniel Zoran, Theophane Weber, Peter Battaglia, Razvan Pascanu, and Andrea Tacchetti. Visual interaction networks: Learning a physics simulator from video. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[25] Lin Yen-Chen, Maria Bauza, and Phillip Isola. Experience-embedded visual foresight. In Leslie Pack Kaelbling, Danica Kragic, and Komei Sugiura, editors, *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pages 1015–1024. PMLR, 30 Oct–01 Nov 2020.

# 5 Appendix

## 5.1 DLO@Scale - Additional information

### 5.1.1 Rendering visualizations from the dataset

Our data contains the positions and orientations of all geometries in the simulation. Hence, we can easily generate image observations using any 3D rendering tool. Fig. 5.1.1 presents different color and depth images for the same scene rendered using pyrender [3].

Figure 4: Examples of rendered images created with *pyrender* [3].

### 5.1.2 Rigid object shapes

The geometries of rigid objects are sampled given the set of possible shapes illustrated in Fig. 5.

Figure 5: Types of shapes for rigid objects

## 5.2 Experimental design and details

In the **single-domain-large** experiments, we used feed-forward neural networks consisting of 4 layers with 128 neurons per layer and LeakyReLU activation functions. We predict the future position of all deformable object segments (32x2 dimensional) given its current positions and the poses of all rigid bodies in the scene. The networks are trained using batches of size 64 and a learning rate of 0.0002. We employ a MSE loss on the predicted segment positions. The networks are trained for a maximum number of $10^9$ iterations. To prevent overfitting in the case of small datasets, we stop training if the test error did not decrease after 10 subsequent dataset epochs.

For the **meta-learning** experiments we used a feed-forward neural network consisting of 6 layers of 32 neurons each and LeakyReLU activations. The same architecture is used for both the Multi-Task and the Meta Learning model. We have restricted the prediction to the deformable linear object (n=4) and considered only domains without rigid objects. Both models have been trained on $5 \cdot 10^6$ repeated domains with a learning rate of 0.0001. To evaluate the models performances we used unseen domains corresponding to 25% of the dataset. The used loss is the MSE on the predicted segments positions. For the meta learning model we used learned inner learning rates specific for each layer, like [17]. Finally, we used only one gradient step in the adaptation phase.