Graph-based Task-specific Prediction Models for Interactions between Deformable and Rigid Objects

Zehang Weng*[1], Fabian Paus*[2], Anastasiia Varava[1], Hang Yin[1], Tamim Asfour[2] and Danica Kragic[1]

***Abstract*— We present a simulation environment and a dataset for task-specific manipulation, involving interactions of rigid objects and a deformable bag. The dataset consists of several scenarios with varying object number and size, as well as manipulation actions. We also propose a dynamics prediction approach based on object-centric graph representation and graph neural networks.**

I. Introduction

We address the problem of modelling interaction dynamics between rigid and cloth-like objects. We consider interaction a deformable bag with handles and rigid spheres in several scenarios. We select a sparse set of keypoints on the deformable object's surface and represent the scene state as a fully-connected graph. A video illustrating the scene and graph representation is available at[1]. Building learning-based predictive models for scenes is challenging: *(i)* there is currently no publicly available dataset containing complex interactions with highly deformable objects, and *(ii)* generalization requires an effective model that captures scenes with internal and external relations of a varying number of scene objects.

For predicting action effect on deformable objects, prior work focuses on objects with simple topology or interactions [1], [2], [3], [4], [5], [6], [7], [8], [9]. Considering deformable objects, a new trend is to employ graph neural networks to learn system dynamics with the ability to generalize to scenarios with a different number of objects [4], [10], [11], [12]. We present a publicly available dataset[2] for studying action effect prediction between one deformable object and multiple rigid objects. We also propose a method for predicting the interactions between these.

II. Task Description and Dataset Generation

We generate a novel dataset for task-specific action effect prediction on scenes containing interactions between a deformable bag and a set of rigid objects. The tasks are created by varying action (opening bag, lifting the bag, moving handles in circle, pushing rigid object) and task parameters (bag stiffness, bag content, handle state). The bag can interact with rigid objects and the table. For the deformable bag, we model the mesh in Blender. Compared to the cloth-like objects in previous works, our model has a more complicated hole structure. The whole mesh consists of 1277 particles and 4326 edges.

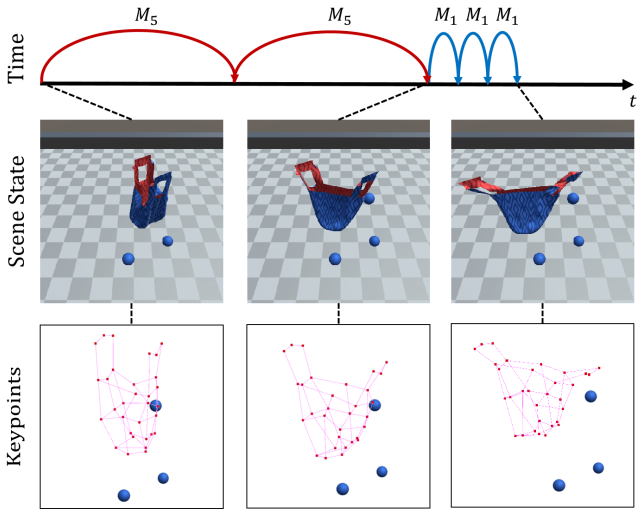

Fig. 1. The mixed-horizon model consists of a short-term prediction model M_1, which can predict the next time step, and a long-term prediction model M_5, which can predict five time steps into the future. This figure shows the scene state at different time steps and our sparse keypoint representation of the scene state at these time steps.

Based on Unity and the Obi Cloth [13] extension, we build our simulation environment and create a dataset grouped into 20 tasks as we want to learn task-specific models. For each task, we simulate 1,000 trajectories, which results in 60,000 recorded time steps. The simulated task data is split into training (80%), validation (10%), and test set (10%).

III. Dynamics Learning and Prediction

Based on the generated dataset, we learn task-specific prediction models for the scene dynamics. Given a scene as a set of rigid and deformable objects O_t, the goal is to learn a dynamics model M to predict the future scene O_{t+1} after performing action a_t at time step t: $O_{t+1} = M(O_t, a_t)$.

The set of rigid objects consists of a variable number of spheres whose state can be represented by their position and radius. The state of the deformable bag consists of the position and connectivity of all vertices. The action a_t is parameterized by the start and end position as well as the radius $(\mathbf{p}_{start}, \mathbf{p}_{end}, r_a)$ of the manipulated target, which can either a rigid object or one of the bag's handles.

We define a graph representation that captures the state of the rigid objects and approximates the state of the deformable

*Authors with equal contribution.

[1]The authors are with CAS/RPL, KTH, Royal Institute of Technology, Stocholm, Sweden. {zehang,varava,hyin,dani}@kth.se

[2]The authors are with the Institute for Anthropomatics and Robotics, Karlsruhe Institute of Technology, Karlsruhe, Germany. {paus,asfour}@kit.edu

[1]https://youtu.be/a4ILwCmai9k

[2]https://github.com/wengzehang/deformable_rigid_interaction_prediction/blob/main/docs/dataset.md

bag using a set of sparse keypoints. Using this representation, we formulate a two-stage graph learning problem to facilitate fixed time step predictions. Then, we combine multiple prediction models with different time step horizons to enable predictions of up to 60 time steps into the future.

A. Graph Representation

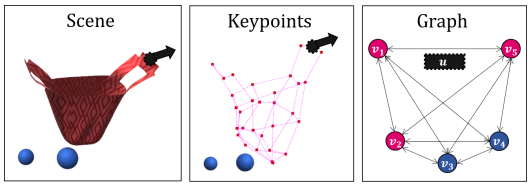

Fig. 2. We transform a scene consisting of deformable and rigid objects into a sparse keypoint representation. Based on the keypoints, we build a fully connected graph, whose vertices represent keypoints and whose edges encode the connectivity between them. The motion of the handle along the black arrow is encoded in the global graph feature $\mathbf{u}$.

We want to represent the state of the scene objects O_t and the action a_t at time step t as a graph $G_t = (V, E, \mathbf{u})$ with vertices V, edges E, and a global feature vector $\mathbf{u}$. We first transform the frame into an action-local coordinate system as in [11] and use V to encode position and state information about the rigid and deformable objects in the scene (see Fig. 2).

We then use edges E to build a fully connected bidirectional graph between the vertices V. E encodes the pairwise position differences of two vertices while the physical connection is indicated by a flag attribute. Global feature vector $\mathbf{u}$ encodes the position change of the manipulated target and the radius of the manipulated object.

B. Two-stage Graph Prediction Model

The goal of the two-stage graph prediction model is, given the current scene state G_t, to predict the scene graph G_{t+h} after h time steps where h is constant. In this work, we focus on single time step predictions ($h = 1$) and longer time steps ($h = 5$). For each prediction horizon h, we learn a dynamics model which consists of two separate modules: Active Prediction Module (APM) and Position Prediction Module (PPM). APM is a binary classifier predicting whether rigid objects or parts of the deformable bag will move in the next time step. The classification is done for every vertex in the scene state G_t. The ground-truth active labels are generated during training based on the position difference between the time steps t and $t + h$. PPM is a regression module that directly predicts the next scene state, i. e. the expected positions of all vertices at time step $t + 1$. Both APM and PPM are implemented as graph neural networks with an encode-process-decode architecture.

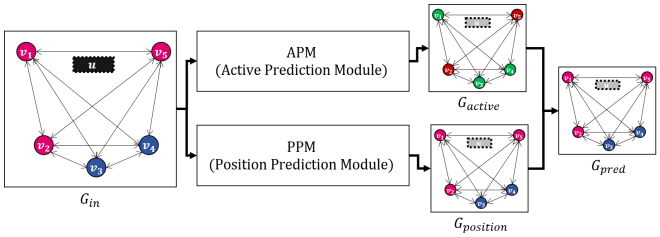

Fig. 3. The two-stage model takes as input the scene state as a graph G_in at a certain time step. This graph is fed into both the APM and PPM. The APM classifies which vertices are active, i. e. will move in the next time step. In the graph G_{active}, the green vertices have been classified as active and the red ones as inactive. The PPM predicts the positions of vertices in the next time step as a graph $G_{position}$. In a final step, only the position updates, whose corresponding vertices have been classified as active in G_{active}, are applied to the prediction result G_{pred}.

APM outputs a binary classification mask through a final softmax activation layer for the vertex features. The classification stage uses cross-entropy loss where N denotes the number of vertices in the scene graph, $y_i^{gt} \in \{0, 1\}$ is the ground-truth active flag and $y_i^{pred} \in [0, 1]$ is the predicted flag. The active flag is set to be 1 when the position difference is above a pre-set threshold.

$$L^{Classification} = \frac{1}{N} \sum_{i=1}^{N} CrossEntropy(y_i^{gt}, y_i^{pred})$$

PPM is a regression model for the scene graph after action execution using a final linear activation layer for the vertex features. The regression stage uses a mean square error loss, where $\mathbf{t}_i^{gt}$ is the ground-truth vertex position and $\mathbf{t}_i^{pred}$ is the predicted position.

$$L^{Regression} = \frac{1}{N} \sum_{i=1}^{N} (\mathbf{t}_i^{gt} - \mathbf{t}_i^{pred})^2$$

We train both models separately on the tasks in the generated dataset. By only applying the regression update to those vertices which have been classified as active, we prevent spurious motion of vertices that are not involved in the interaction between objects in the current time step (see Fig. 3). Under a fixed time step h, we call this combination the *two-stage* model (APM+PPM), whereas the regression stage alone is called *one-stage* model (PPM):

$$\begin{aligned} M_h^{one-stage}(G_i) &= M_h^{PPM}(G_i) \\ M_h^{two-stage}(G_i) &= M_h^{APM}(G_i) \odot M_h^{PPM}(G_i) \end{aligned}$$

Here the operator $\odot$ only applies the position updates from the PPM if the vertices have been classified as active in the APM.

C. Long Horizon Prediction Model

The graph prediction models only predict the scene for a fixed prediction horizon h. The longer horizon model M_5 is trained with a prediction horizon $h = 5$, and the single time step model M_1 is trained with a horizon $h = 1$. By chaining these models recursively together, we can make predictions for any time step t.

If we only use the single time step model M_1, we can predict the scene state G_t after t time steps given the initial

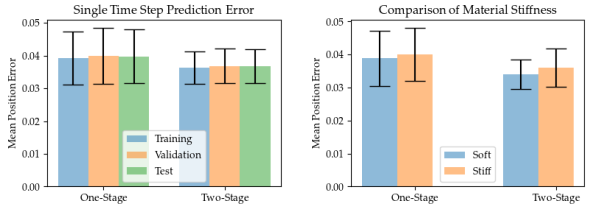

Fig. 4. The left figure shows the single time step prediction errors over all tasks for training, validation, and test set. The right figure shows the single time step prediction errors over all tasks grouped by material stiffness. The mean position error is shown as the bar height and the whiskers show the standard deviation over all tasks.

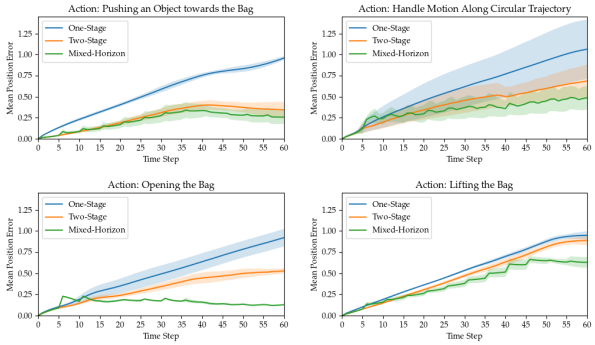

Fig. 5. Long horizon prediction errors per action for the *one-stage*, *two-stage*, and *mixed-horizon* models. The solid lines show the mean position error while the colored area around the line indicate the standard deviation.

scene state G_0:

$$G_t = \underbrace{(M_1 \circ M_1 \circ \cdots \circ M_1)}_{t \text{ times}}(G_0)$$

In this approach, we can either use the *one-stage* or the *two-stage* model. However, this causes the prediction error to accumulate fast. We can alleviate this problem, by also incorporating the longer horizon model M_5. First, we run M_5 recursively for $\lfloor t/5 \rfloor$ steps. Then, M_1 is run for the remaining time steps $t \mod 5$:

$$G_t = \underbrace{(M_1 \circ M_1 \circ \cdots \circ M_1)}_{(t \mod 5) \text{ times}} \circ \underbrace{(M_5 \circ M_5 \circ \cdots \circ M_5)}_{\lfloor t/5 \rfloor \text{ times}}(G_0)$$

We call this combination of a multi-step prediction and a single-step prediction the *mixed-horizon* prediction model (see Fig. 1 for an example).

IV. Evaluation

In the evaluation, we want to investigate the benefits of our proposed method by answering the following questions: *(i)* does the inclusion of the APM (Active Prediction Module) in the *two-stage* model improve the prediction results over the *one-stage* model with the PPM (Position Prediction Module) alone? *(ii)* How does the material stiffness of the deformable bag influence the prediction accuracy? *(iii)* Does the *mixed-horizon* model improve long-term prediction results compared to an iterative application of one-stage or two-stage models?

To answer the first question, we evaluate the single time step prediction performance of the proposed *two-stage* model compared to the *one-stage* model. Fig. 4 shows that the *two-stage* model decreases the mean position errors while also lowering the inter-task variance. APM improves single time step predictions when compared to the PPM alone.

To address the second question, we compare the single time step prediction results for soft bag material with results for stiff bag material. Fig. 4 shows the mean position error and the standard deviation for both materials. As can be seen, the tasks with soft bag material have a smaller prediction error. However, the difference is lower than the inter-task variance, indicating that our method is able to handle tasks independent of material stiffness.

For the third question, we compare the long horizon prediction results for the recursive *one-stage*, *two-stage* and *mixed-horizon* models on the test set. We initialize each model with the scene state G_0 at time step $t = 0$ and apply the prediction in an iterative way as described in section III-C. Since long horizon prediction performance varies between actions, Fig. 5 shows the mean position errors and standard deviation for the four actions *Pushing an Object towards the Bag*, *Handle Motion along Circular Trajectory*, *Opening the Bag*, and *Lifting the Bag*. We can see that the *two-stage* model outperforms the *one-stage* model consistently, independent of the action. The difference between the models in the lifting action is quite small, since the almost all parts of the bag move during this action. Therefore, the first movement classification stage is not as helpful as in the other actions. Furthermore, the *mixed-horizon* model outperforms the *two-stage* model for longer term predictions, while sometimes producing worse results for short term predictions. Depending on the action, the *mixed-horizon* model produces much better predictions then the *two-stage* model (e.g. opening the bag), while for others the improvement is marginal (e. g. pushing an object). Overall, the *mixed-horizon* model is better suited for predictions over a longer time periods than the *one-stage* and *two-stage* models.

V. Conclusion

We present a novel dataset for action effect prediction on scenes containing both rigid and cloth-like deformable objects. Our predictive model can generalize to different numbers of vertices in the graph, allowing us to consider different sets of objects. We propose two modules to capture the dynamics based on the graph networks, and implement a mix-horizon model on top of the learned modules to predict the future scene state and evaluate our method on different tasks.

Acknowledgment

The research leading to these results has received funding from the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – Project Number 146371743 – TRR 89 Invasive Computing.

We would also like to show our gratitude to the Swedish Research Council, Knut and Alice Wallenberg Foundation.

REFERENCES

[1] Y. C. Hou, K. S. M. Sahari, and D. N. T. How, "A review on modeling of flexible deformable object for dexterous robotic manipulation," *International Journal of Advanced Robotic Systems*, vol. 16, no. 3, p. 1729881419848894, 2019.

[2] J. Sanchez, J.-A. Corrales, B.-C. Bouzgarrou, and Y. Mezouar, "Robotic manipulation and sensing of deformable objects in domestic and industrial applications: a survey," *The International Journal of Robotics Research*, vol. 37, no. 7, pp. 688–716, 2018.

[3] C. Luible and N. Magnenat-Thalmann, "The simulation of cloth using accurate physical parameters," *CGIM 2008, Insbruck, Austria*, 2008.

[4] P. W. Battaglia, R. Pascanu, M. Lai, D. Rezende, and K. Kavukcuoglu, "Interaction networks for learning about objects, relations and physics," *arXiv preprint arXiv:1612.00222*, 2016.

[5] N. Watters, D. Zoran, T. Weber, P. Battaglia, R. Pascanu, and A. Tacchetti, "Visual interaction networks: Learning a physics simulator from video," in *Advances in neural information processing systems*, 2017, pp. 4539–4547.

[6] M. Yan, Y. Zhu, N. Jin, and J. Bohg, "Self-supervised learning of state estimation for manipulating deformable linear objects," *IEEE robotics and automation letters*, vol. 5, no. 2, pp. 2372–2379, 2020.

[7] Y. J. Oh, T. M. Lee, and I.-K. Lee, "Hierarchical cloth simulation using deep neural networks," in *Proceedings of Computer Graphics International 2018*, 2018, pp. 139–146.

[8] D. Hafner, T. Lillicrap, I. Fischer, R. Villegas, D. Ha, H. Lee, and J. Davidson, "Learning latent dynamics for planning from pixels," in *International Conference on Machine Learning*. PMLR, 2019, pp. 2555–2565.

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

[10] M. Janner, S. Levine, W. T. Freeman, J. B. Tenenbaum, C. Finn, and J. Wu, "Reasoning about physical interactions with object-centric models," in *International Conference on Learning Representations*, 2019, pp. 1–12.

[11] F. Paus, T. Huang, and T. Asfour, "Predicting pushing action effects on spatial object relations by learning internal prediction models," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 10 584–10 590.

[12] A. E. Tekden, A. Erdem, E. Erdem, M. Imre, M. Y. Seker, and E. Ugur, "Belief regulated dual propagation nets for learning action effects on groups of articulated objects," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 10 556–10 562.

[13] *Obi: Unified particle physics for Unity*. [Online]. Available: http://obi.virtualmethodstudio.com