Geometry-Aware Modeling of Rigid Body Physics

Kexin Yi [*1] **Toru Lin** [*2] **Phillip Isola** [2]

Abstract

Rigid body systems are among the most important subjects of study in physics and are widely applied in both simulated and real-world applications. As suggested by theory of classical mechanics, the motion of a rigid body system is strongly governed by its geometric constraints, i.e. how different rigid components are connected and allowed to move with respect to each other. Extracting these information from observations and applying them to the system's dynamics modeling are important steps towards a deeper and more structured understanding of the physical world. In this work, we propose a computational framework that both extracts the geometric constraints and models the forward dynamics of rigid body systems starting from raw pixel observations. Our model first extracts a hierarchical representation, facilitated by keypoints and their groupings, for describing the system's constraints from visual observations in an unsupervised fashion. Then, a dynamics model aware of these constraints is applied to predict the forward dynamics of the state representation. Finally, a reconstruction network recovers the visual frames from the predicted states. Experiment results on classic rigid body control environments show our model is able to accurately infer the constraints, and geometry-aware dynamics modeling leads to more accurate and physically sensible future predictions.

1. Introduction

Building machines that are able to understand the physical world from visual observations has been a long-standing goal in artificial intelligence. Many recent research works focus on studying forward prediction models that operate on the physical states, aiming for making future predictions of the states and perform reasoning based on the outcome (Chen et al., 1990; Battaglia et al., 2013; 2016). These models, while achieving remarkable success on a number of domains in prediction quality, often do not reveal or rely on *structured* domain knowledge of physics. To better understand the physical world, a desired computational model should be able to both *predict* the motion and *abstract* the domain knowledge of the system.

While recent work has studied how dynamical properties are governed by the explicit physical parameters such as gravity and elasticity (Wu et al., 2015; Li et al., 2020), one very important property missing from discussions is the geometric constraints of systems. The geometric constraints define how different parts of a complex system connect and interact with each other, thereby generating a wide range of motion patterns. For example, the rich and complex motions that our arms are able to perform can be associated with two constraints: our hands always remain at a fixed distance to our elbows, and our elbows always remain at a fixed distance to our shoulders. These two simple constraints define the physics of our arms and therefore determine the way we control them.

In this work, we focus on the physics of rigid body systems which not only represent one of the most important subjects in physics for machines to understand, but also widely appear in real world applications such as robotic control. We further identify the following three challenges associated with geometric constraints for physical modeling on these systems. First, how to represent geometric constraints on rigid body? Unlike physical parameters such as gravity, which are often naturally represented as scalars, representing the geometry of a system involves extracting information from all its components and therefore requires a distributed representation. We adopt a hierarchical geometry-aware representation based on keypoints as the physical state representation of the system. As we will see in section 3.2, keypoints provide a natural way to enforce the geometric constraints via grouping. Second, how to ground the state representation on visual observations? We apply a novel self-supervised keypoint extraction network that learns to extract temporal-consistent keypoints from raw pixel observations without ground-truth annotations. The perception module also comes with a visual decoder for reconstructing the frames. Third, how to make use of the geometric constraints in the dynamics modeling of the system?

[*]Equal contribution [1]Harvard University [2]MIT CSAIL. Correspondence to: Kexin Yi <kyi@g.harvard.edu>.

Workshop on Object-Oriented Learning at ICML 2020.

We explore various options for dynamics modeling on the geometry-aware representation. We also introduce a novel *body-centric* model which treats each rigid body as the basic unit for dynamics modeling, enabling constraint-preserving motion updates.

Our framework is able to correctly infer the geometric constraints on several rigid body environments from the DeepMind Control Suite (Tassa et al., 2018). Experimental evaluations also show that dynamics modeling on geometry-aware representations enables accurate and physically sensible long-term future predictions of both the physical states and visual observations.

2. Related Work

Our work is closely related to forward predictive models for physics simulation. Deep neural networks have been widely applied to learning physical dynamics on various systems (Grzeszczuk et al., 1998; Chen et al., 1990). Among those (Battaglia et al., 2016) and (Chang et al., 2016) use graph neural networks to capture the object- and relation-based properties, leading to nice prediction accuracy and generalizability on systems of massive particles. This approach is extended to a wide variety of physical domains, including but not limited to rigid bodies, elastic materials and fluids (Sanchez-Gonzalez et al., 2018; Li et al., 2019; Sanchez-Gonzalez et al., 2020). However, these models assume access to the ground-truth physical states, which are very difficult to access in real world applications.

Other works have studied physical dynamics modeling from raw pixels as well as the applications on video modeling and control (Lerer et al., 2016; Xue et al., 2016; Finn and Levine, 2017; Babaeizadeh et al., 2017; Ha and Schmidhuber, 2018). However, these models relying on latent representations lack the capability of revealing deeper structured knowledge of physics and making long term predictions. (Watters et al., 2017; Wu et al., 2017; Li et al., 2020; Yi et al., 2020) study dynamics modeling on physical states while grounding them on visual observations. Extra supervision is needed for visual grounding on these models.

In search of structured representations able to describe the physical system's constraints, we find keypoint a desirable candidate, for its awareness of geometry and capability of grounding on visual inputs with minimal supervisions (Zhang et al., 2018b; Jakab et al., 2018; Suwajanakorn et al., 2018). Recently, a number of works have studied unsupervised keypoint learning on videos (Minderer et al., 2019; Kulkarni et al., 2019; Jakab et al., 2020). Our work builds on top of these efforts and use the keypoint as foundation for representing structured physical properties and dynamics modeling.

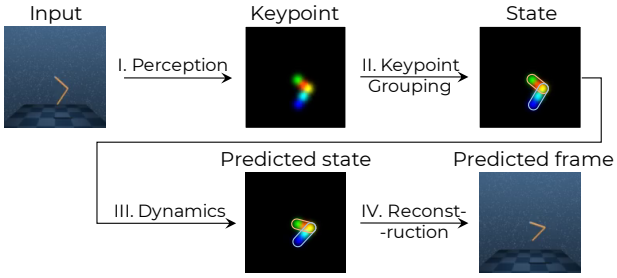

Figure 1. Overview of our model architecture. At each timestep, the perception module extracts keypoints from input visual observation. The keypoint grouping module groups the keypoints by rigid body, and generates a hierarchical state representation. The dynamics module predicts future states from past trajectories. The reconstruction module generates future observations of the system from the predicted states.

3. Method

In this work, we propose a framework to model the physics of dynamic rigid body systems from raw pixel observations. Our model infers a *structured* state representation that incorporates the geometric constraints of the system, and learns a forward dynamics model. Our framework consists of four parts as shown in Figure 1: a self-supervised perception module that extracts keypoints from input visual observations (Fig. 1-I); a keypoint grouping module that computes a hierarchical state representation by grouping the keypoints according to the rigid body they belong to (Fig. 1-II); a dynamics module that predicts forward dynamics of the state representation (Fig. 1-III); and a reconstruction module that generates predicted video frames (Fig. 1-IV). Details of model components and training are presented below.

3.1. Perception and Reconstruction Module

Our perception module learns to extract keypoints from input visual observations through self-supervised training. Given an input frame o_t representing the visual observation at time step t, our perception module maps the frame to a set of keypoints $p_t = (x_i^t, y_i^t)_{i=1}^N$, where x_i^t and y_i^t are the horizontal and vertical coordinates of the i-th keypoint. The total number of keypoints N is a fixed parameter and the keypoints are normalized within $[-1, 1]$ on both directions.

To learn the keypoint extractor in a self-supervised manner, we design the following conditional image generation pretext task. Given a sequence of input frames $O = \{o_t\}_{t=1}^T$, we apply the keypoint extractor to each frame and output a sequence of keypoints $P = \{p_t\}_{t=1}^T$. A feature encoder is also applied to extract the visual feature f_1 from the first frame of the input sequence. Next, the keypoint coordinates P are turned into a sequence of Gaussian heatmaps $H = \{h_t\}_{t=1}^T$ centered at the keypoints, each of which is then concatenated to the visual feature f_1 to form a set of latent representations $\{(f_1, h_t)\}_{t=2}^T$. Finally, a reconstruc-

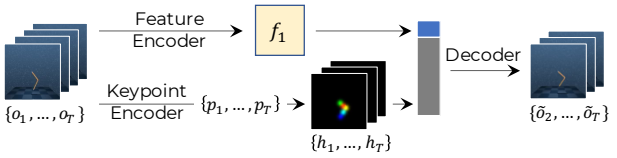

Figure 2. Model architecture of our perception and reconstruction modules. Our model uses a convolutional autoencoder with keypoint bottleneck to extract a structural representation.

tion network is used to generate visual frames $\tilde{O} = \{\tilde{o}_t\}_{t=2}^{T}$ from the latent representation at each time step. The perception module, together with the reconstruction network, is trained to minimize the mean squared error between the predicted and ground truth frames $\mathcal{L} = \text{MSE}(O, \tilde{O})$.

3.2. Keypoint Grouping

Given the keypoints inferred by the perception module, how to obtain a representation that reflects the rigidity of the underlying bodies? Our keypoint grouping module restores the rigidity constraints by computing a grouping of the keypoints according to the rigid body they belong to, based on the assumption that each pair of keypoints on the same rigid body should remain at the same distance throughout the entire motion trajectory.

More formally, the problem of searching for rigid groupings of keypoints can be converted to a search problem on an undirected rigidity graph, where each node represents a keypoint and each edge connects a pair of keypoints that satisfy the rigidity constraint of being at a fixed relative distance. Note that since each pair of keypoints belonging to the same rigid body is connected by an edge, each rigid body in the system should correspond to a *maximum clique*. In our framework, we apply the Bron-Kerbosch algorithm (Bron and Kerbosch, 1973) to compute all maximum cliques of a input graph and generate a grouping of the input keypoints by rigid body. The output keypoint groupings are represented by sets of keypoint indices $(\mathcal{G}_1, \mathcal{G}_2, ...\mathcal{G}_B)$, where B is the number of rigid bodies and $\mathcal{G}_b$ is the set of keypoint indices belonging to the b-th rigid body.

3.3. Dynamics Module

We apply body-centric dynamics modeling on the sequence of keypoints $(p_1, p_2, ...p_T)$ extracted frame-by-frame by the perception module. The model treats each rigid body (i.e. group of keypoints) as the basic unit for dynamics modeling instead of the keypoints themselves. As shown in Figure 3, our dynamics model adopts a sequence-to-sequence architecture consisting of a recurrent encoder and decoder. Details of the architecture is presented below.

Encoder. The encoder inputs a historical window of keypoint trajectories as the initial condition for dynamics prediction. The encoder first generates two 1D thermometer encoding vectors from the keypoint coordinates on both x and y directions. Then these vectors are concatenated and sent to a fully connected embedding layer to generate a flattened embedding. A LSTM (Hochreiter and Schmidhuber, 1997) is used to gather the temporal information from the keypoint embeddings at each input time step and output a hidden state to the decoder.

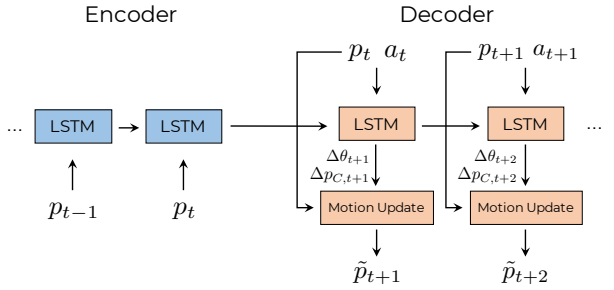

Figure 3. Model architecture of our dynamics module. Our model uses a recurrent network to encode input keypoint trajectories. Then a recurrent decoder predicts the displacement and rotation angle of each rigid body, followed by body-centric motion updates.

Decoder. The decoder generates predictions of keypoint coordinates in an auto-regressive procedure. It uses a LSTM at its core, same as the one in the encoder, whose hidden state is initialized by the output of the encoder. At each time step, the keypoint from the previous step is embedded (in the same way as the encoder) and then input to the decoder LSTM. If the prediction task involves action, the input action at the current step is also embedded by a fully-connected layer and concatenated to the input. The output from the LSTM is then sent to a body-centric motion update unit to generate the predicted keypoint coordinates at the next time step.

Motion update. The motion update unit is based on the assumption that a rigid body's motion can be decomposed into the translation of its center and rotation with respect to the center. Therefore for *each* rigid body (as indicated by index b) in the system, the decoder LSTM predicts the displacement of the body center Δp_C^b and its rotation angle $\Delta\theta^b$ at the current time step. To obtain the position of the center, we use a set of learnable parameters w associated with each rigid body and compute a weighted average of the keypoints. The relative position of each keypoint with respect to the center is denoted as p', and

$$p_C^b = \sum_{i \in \mathcal{G}_b} w_i^b p_i^b \qquad p_i'^b = p_i^b - p_C^b. \tag{1}$$

Motion update applies to p_C and p' separately. For translation, the predicted displacement is directly added to the center position. For rotation, the predicted angle defines a rotation matrix applied the p's of all keypoints. In summary:

$$\tilde{p}_c^b = p_C^b + \Delta p_C \qquad \tilde{p}_i'^b = \begin{pmatrix} \cos\Delta\theta^b & -\sin\Delta\theta^b \\ \sin\Delta\theta^b & \cos\Delta\theta^b \end{pmatrix} p_i'^b. \tag{2}$$

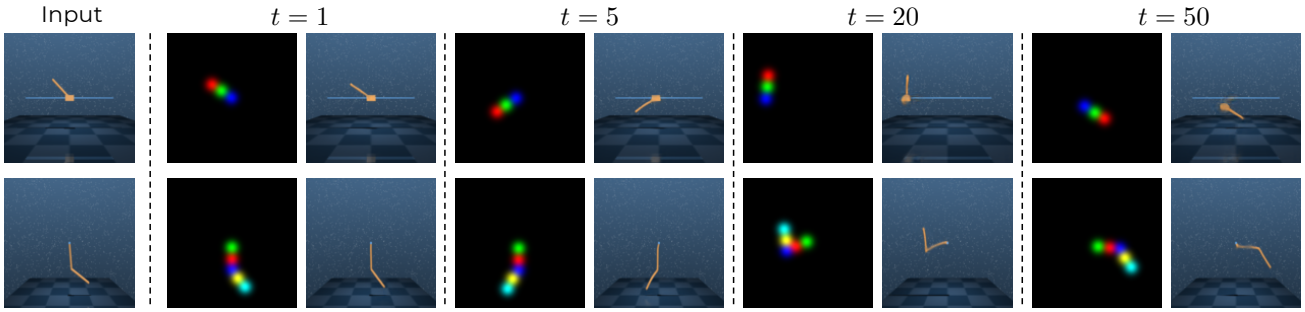

Figure 4. Qualitative results of our model on Cartpole and Acrobot. Our model is able to correctly predict the keypoint motion that satisfies all the geometry constraints of the system over a long period, and generate physically sensible frame reconstructions.

Dataset	Model	$t = 5$	$t = 10$	$t = 15$	$t = 30$	$t = 50$
Cartpole	Ours	0.632	0.741	0.851	1.000	1.069
	IN	0.614	0.870	1.066	1.314	1.375
Acrobot	Ours	0.658	0.751	0.790	0.782	0.837
	IN	0.805	0.907	0.958	1.030	0.981

Table 1. Mean squared error (MSE) on the predicted video frames. The numbers shown are scaled up by 10^3.

Dataset	Model	$t = 5$	$t = 10$	$t = 15$	$t = 30$	$t = 50$
Cartpole	Ours	0.032	0.041	0.052	0.074	0.089
	IN	0.031	0.049	0.074	0.131	0.163
Acrobot	Ours	0.037	0.050	0.061	0.081	0.104
	IN	0.075	0.114	0.140	0.161	0.148

Table 2. Perceptual loss (LPIPS) on the predicted video frames.

Since rotation transformation preserves the relative distances between the keypoints, the rigidity constraint is implicitly imposed by the above motion update rule.

4. Experiments

In this section, we present experimental evaluations of our framework on two datasets that include a wide range of rigid body motions and different numbers of rigid bodies. Both datasets are generated from the DeepMind Control Suite (Tassa et al., 2018), a set of simulated continuous control environments. Details of the datasets, model and training paradigms are described below.

Datasets We collect data from the "Cartpole" and "Acrobot" environments from DeepMind Control Suite with random discrete actions sampled from $\{-1.0, 0, 1.0\}$. Each action is repeated for 4 times, and we collected one output frame rendered by the environment for every 4 steps of the simulation. On each dataset we generate 5000 sequences for training and 1000 sequences for testing, each containing 200 frames. We use 3 keypoints on Cartpole and 5 keypoints on Acrobot for dynamics modeling.

Perception module. Our perception module applies the same model architecture for its feature encoder, keypoint encoder and decoder as in (Kulkarni et al., 2019) for feature extraction and image reconstruction. The keypoint encoder then uses a 1×1 convolution layer for generating heatmaps of the keypoint coordinates with resolution 64.

Dynamics module. The dynamics module uses a embedding layer of size 128 for input keypoints and an embedding of size 64 for the input actions. The core of both the encoder and decoder is a bi-layer LSTM with hidden size 64. The output from the decoder is passed into a MLP with a single hidden layer of size 64. The dynamics module is trained on keypoints extracted from a trained perception module. The input length is 3 time steps and the decoder is asked to predict the next 97 time steps.

Results. We show quantitative evaluations based on two metrics: pixel mean squared error (MSE) and perceptual loss (Zhang et al., 2018a) on the predicted frames from our model. For comparison, We replace our dynamics module by an interaction network (Battaglia et al., 2016) that treats each keypoint as a node in its underlying graph representation. In other words, the interaction network is not aware of the constraints. As shown in table 1 and 2, our model outperforms the baseline in both metrics, suggesting that structured representation leads to more accurate physics modeling and enables consistent long-term predictions. We also show qualitative results in figure 4, which include predictions of both future keypoints and reconstructed frames. Our model makes predictions that preserve the rigidity constraints on keypoints, and produces high quality reconstructed frames for as many as 50 frames into the future.

5. Conclusion

In this work, we introduce a framework for modeling rigid body physics from visual observations by *disentangling* dynamics modeling from perception. Our model can extract temporally consistent keypoints of rigid body systems, which facilitates a hierarchical state representation aware of the geometrical constraints. This representation enables body-centric dynamics modeling on the rigid bodies, leading to constraint-preserving state predictions over a long time window. Our work takes a small step towards building models with more structured understanding of the physical world, and sheds light on potential applications in video modeling, robotics and model-based control.

References

Mohammad Babaeizadeh, Chelsea Finn, Dumitru Erhan, Roy H Campbell, and Sergey Levine. Stochastic variational video prediction. *arXiv:1710.11252*, 2017.

Peter W Battaglia, Jessica B Hamrick, and Joshua B Tenenbaum. Simulation as an engine of physical scene understanding. *PNAS*, 110(45):18327–18332, 2013.

Peter W. Battaglia, Razvan Pascanu, Matthew Lai, Danilo Rezende, and Koray Kavukcuoglu. Interaction networks for learning about objects, relations and physics. In *NIPS*, 2016.

Coen Bron and Joep Kerbosch. Algorithm 457: finding all cliques of an undirected graph. *Communications of the ACM*, 16(9): 575–577, 1973.

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.

Sheng Chen, SA Billings, and PM Grant. Non-linear system identification using neural networks. *International Journal of Control*, 51(6):1191–1214, 1990.

Chelsea Finn and Sergey Levine. Deep visual foresight for planning robot motion. In *ICRA*, pages 2786–2793. IEEE, 2017.

Radek Grzeszczuk, Demetri Terzopoulos, and Geoffrey Hinton. Neuroanimator: Fast neural network emulation and control of physics-based models. In *SIGGRAPH*, 1998.

David Ha and Jürgen Schmidhuber. Recurrent world models facilitate policy evolution. In *Advances in Neural Information Processing Systems*, pages 2450–2462, 2018.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, 1997.

Tomas Jakab, Ankush Gupta, Hakan Bilen, and Andrea Vedaldi. Unsupervised learning of object landmarks through conditional image generation. In *Advances in Neural Information Processing Systems*, pages 4016–4027, 2018.

Tomas Jakab, Ankush Gupta, Hakan Bilen, and Andrea Vedaldi. Self-supervised learning of interpretable keypoints from unlabelled videos. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

Tejas D Kulkarni, Ankush Gupta, Catalin Ionescu, Sebastian Borgeaud, Malcolm Reynolds, Andrew Zisserman, and Volodymyr Mnih. Unsupervised learning of object keypoints for perception and control. In *Advances in Neural Information Processing Systems*, pages 10723–10733, 2019.

Adam Lerer, Sam Gross, and Rob Fergus. Learning physical intuition of block towers by example. In *ICML*, 2016.

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.

Yunzhu Li, Toru Lin, Kexin Yi, Daniel Bear, Daniel LK Yamins, Jiajun Wu, Joshua B Tenenbaum, and Antonio Torralba. Visual grounding of learned physical models. *arXiv preprint arXiv:2004.13664*, 2020.

Matthias Minderer, Chen Sun, Ruben Villegas, Forrester Cole, Kevin P Murphy, and Honglak Lee. Unsupervised learning of object structure and dynamics from videos. In *NIPS*, 2019.

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 *ICML*, 2018.

Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter W Battaglia. Learning to simulate complex physics with graph networks. *arXiv preprint arXiv:2002.09405*, 2020.

Supasorn Suwajanakorn, Noah Snavely, Jonathan J Tompson, and Mohammad Norouzi. Discovery of latent 3d keypoints via end-to-end geometric reasoning. In *Advances in Neural Information Processing Systems*, pages 2059–2070, 2018.

Y. Tassa, Y. Doron, A. Muldal, T. Erez, Y. Li, D. de Las Casas, D. Budden, A. Abdolmaleki, J. Merel, A. Lefrancq, T. Lillicrap, and M. Riedmiller. DeepMind Control Suite. *arXiv:1801.00690*, 2018.

Nicholas Watters, Andrea Tacchetti, Theophane Weber, Razvan Pascanu, Peter Battaglia, and Daniel Zoran. Visual interaction networks. In *NIPS*, 2017.

Jiajun Wu, Ilker Yildirim, Joseph J Lim, William T Freeman, and Joshua B Tenenbaum. Galileo: Perceiving physical object properties by integrating a physics engine with deep learning. In *NIPS*, 2015.

Jiajun Wu, Erika Lu, Pushmeet Kohli, Bill Freeman, and Josh Tenenbaum. Learning to see physics via visual de-animation. In *NIPS*, 2017.

Tianfan Xue, Jiajun Wu, Katherine Bouman, and Bill Freeman. Visual dynamics: Probabilistic future frame synthesis via cross convolutional networks. In *NIPS*, 2016.

Kexin Yi, Chuang Gan, Yunzhu Li, Pushmeet Kohli, Jiajun Wu, Antonio Torralba, and Joshua B. Tenenbaum. Clevrer: Collision events for video representation and reasoning. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=HkxYzANYDB.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018a.

Yuting Zhang, Yijie Guo, Yixin Jin, Yijun Luo, Zhiyuan He, and Honglak Lee. Unsupervised discovery of object landmarks as structural representations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2694–2703, 2018b.