Augmenting Physical Simulators with Stochastic Neural Networks: Case Study of Planar Pushing and Bouncing

Authors:

Anurag Ajay, Jiajun Wu, Nima Fazeli, Maria Bauza, Leslie P. Kaelbling, Joshua B. Tenenbaum, Alberto Rodriguez

Authors

- The paper is from MIT, specifically,
- Leslie P Kaelbling:
  - focuses on research about data-driven method for planning.
  - Founded Journal of Machine learning Research
- Joshua B Tenenbaum:
  - focuses on filling in the gap between human cognition and AI

Uncertainties?

- Difference between simulators predictions and real-world observations

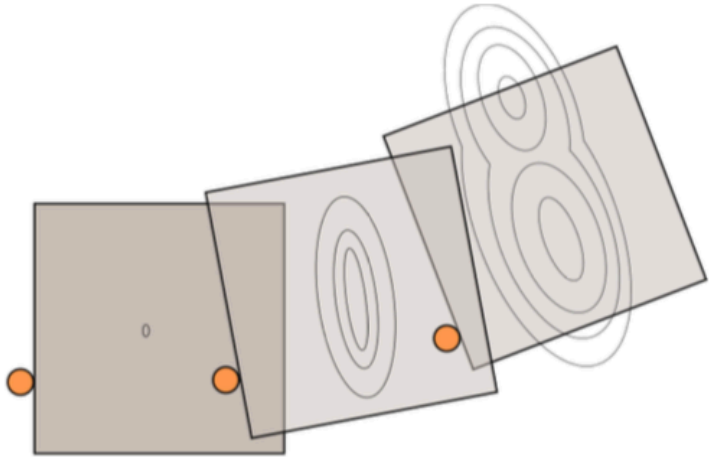


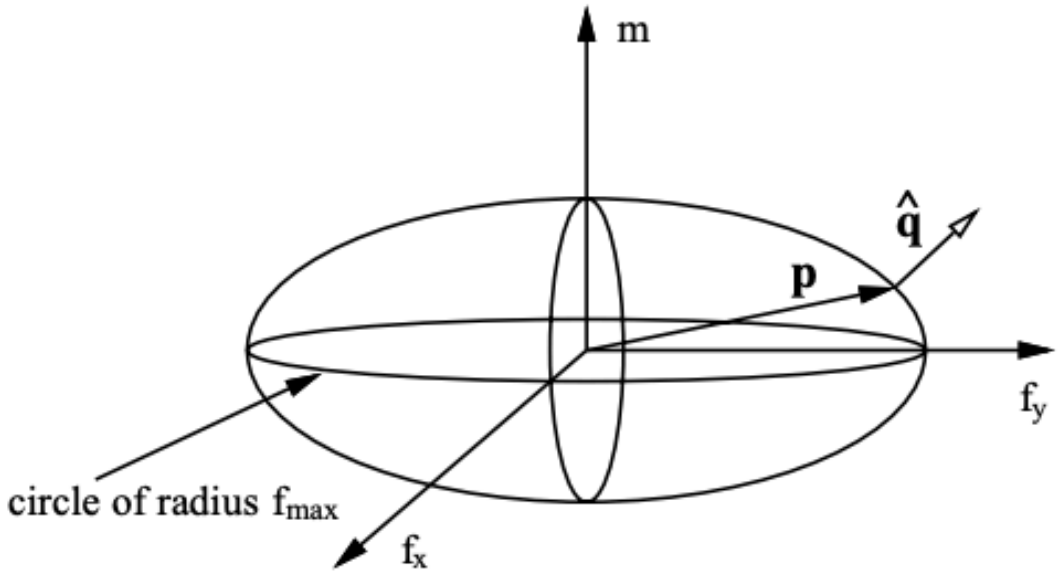
Fig. 1: The motion of an object being pushed appears stochastic and possibly multi-modal due to imperfections in contact surfaces, non-uniform coefficient of friction, stick/slip transitions, and micro surface interactions. In this study, we propose to augment analytical models to more accurately predict such outcomes while reasoning about uncertainty.

Goal

- Fill in this gap between physical simulation and real world observations.

Related Work

- Models for planar pushing:
  - Ellipsoidal Limit Surfaces [Lynch et al.]



Related Work

- Learning Contact Dynamics
  - Neural network for one-step prediction [Kloss et al.]
    - Flaws: only sufficient for the planar pushing case.
    - One-step estimation doesn’t work well for long trajectories
- Uncertainty modeling
  - Model using mixture of Gaussians, giving multi-modality, make it possible to capture it. [Bauza and Rodriguez]

Problems of previous work

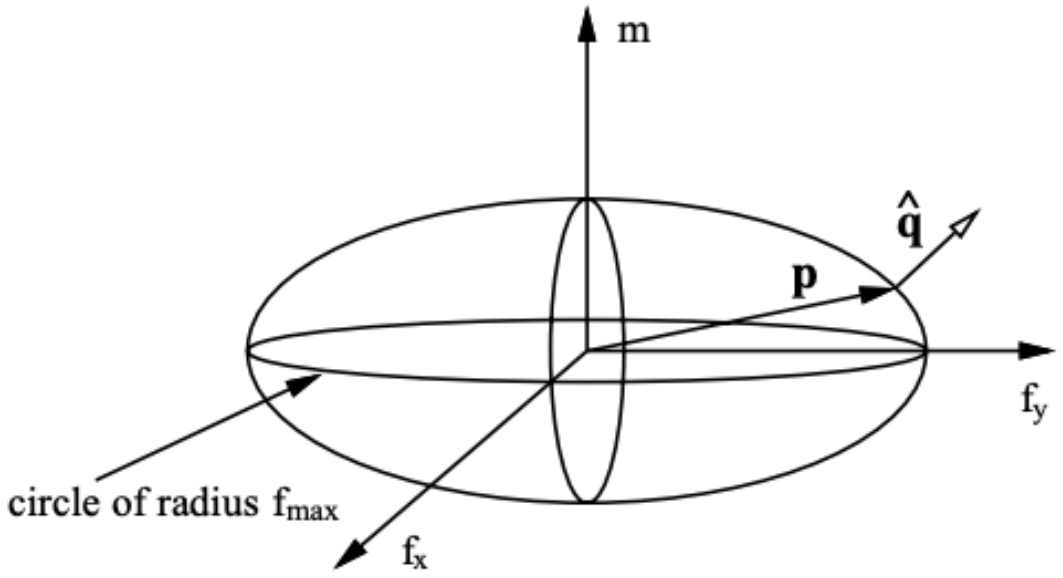
- Physical model:
  - work in nominal cases, yet cannot deal with contact well.
- Existing Data-driven method:
  - not very efficient and very domain-specific

Contribution

- Combine the physical model and data-driven method:
  - Propose Data-Augmented Residual Models
  - use physical simulation and then use a Learns the difference between simulation and real world data

Techniques

- Physics Engine: Ellipsoidal limit surface
  - Compute motion cone
  - Inside: sticking
  - Outside: sliding



Techniques

- Recurrent data-augmented residual model
  - Based on previous model called VRNN.(it cannot condition on additional inputs eg. Pushforces)
  - Make it conditional, namely Conditional VRNNs.

Formulations

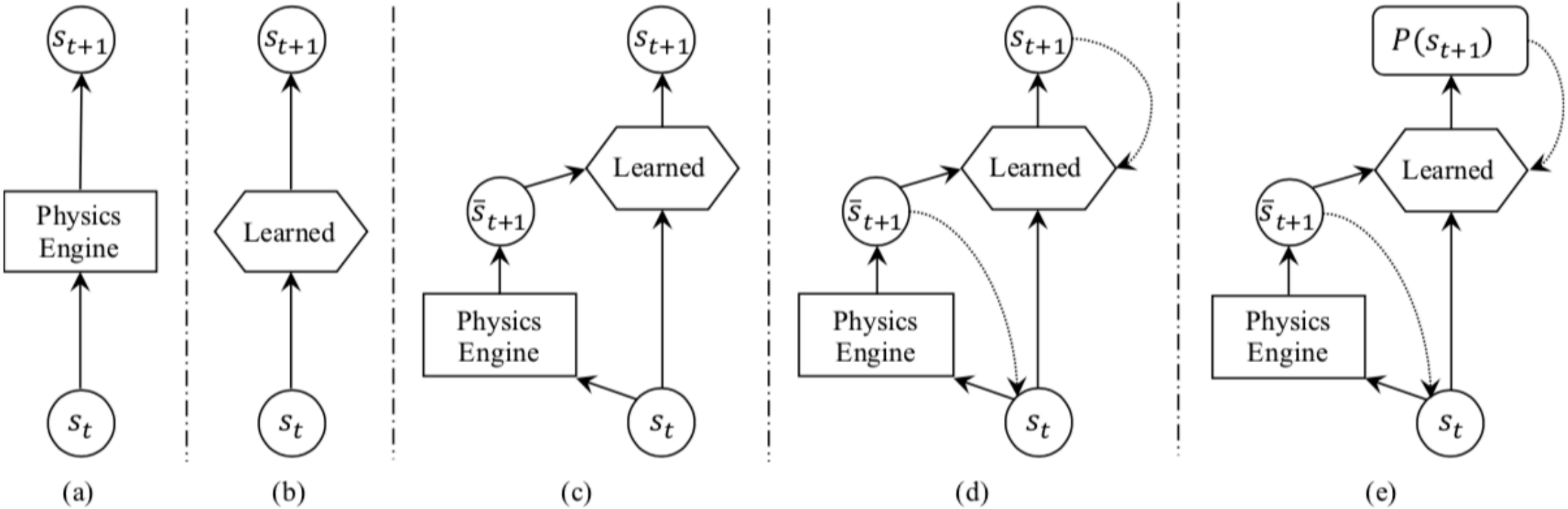
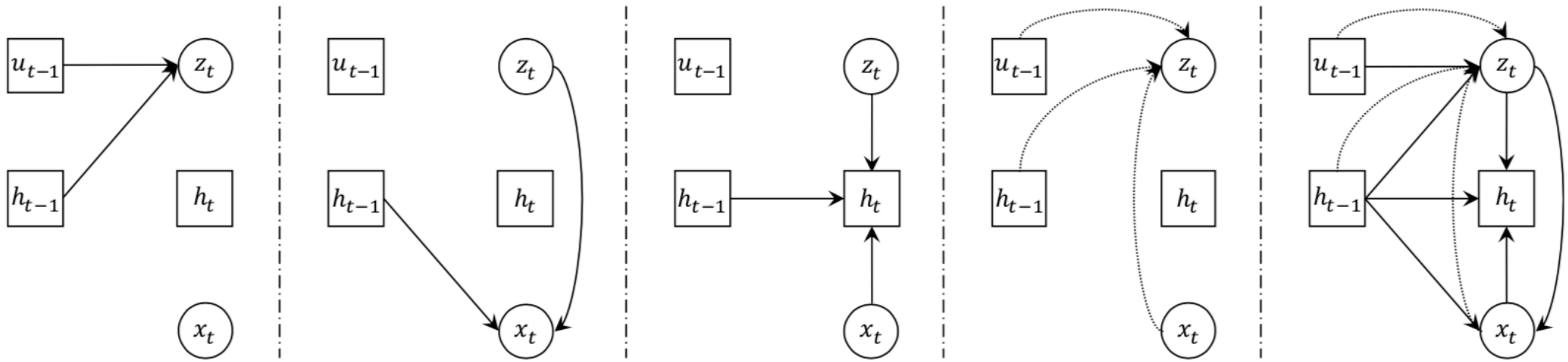


Fig. 2: Model classes: (a) physics-based analytical models; (b) data-driven models; (c) data-augmented residual models; (d) recurrent data-augmented residual models; and (e) stochastic recurrent data-augmented residual models.



(a) Prior (b) Generation (c) Recurrence (d) Inference (e) Overall

Experiments

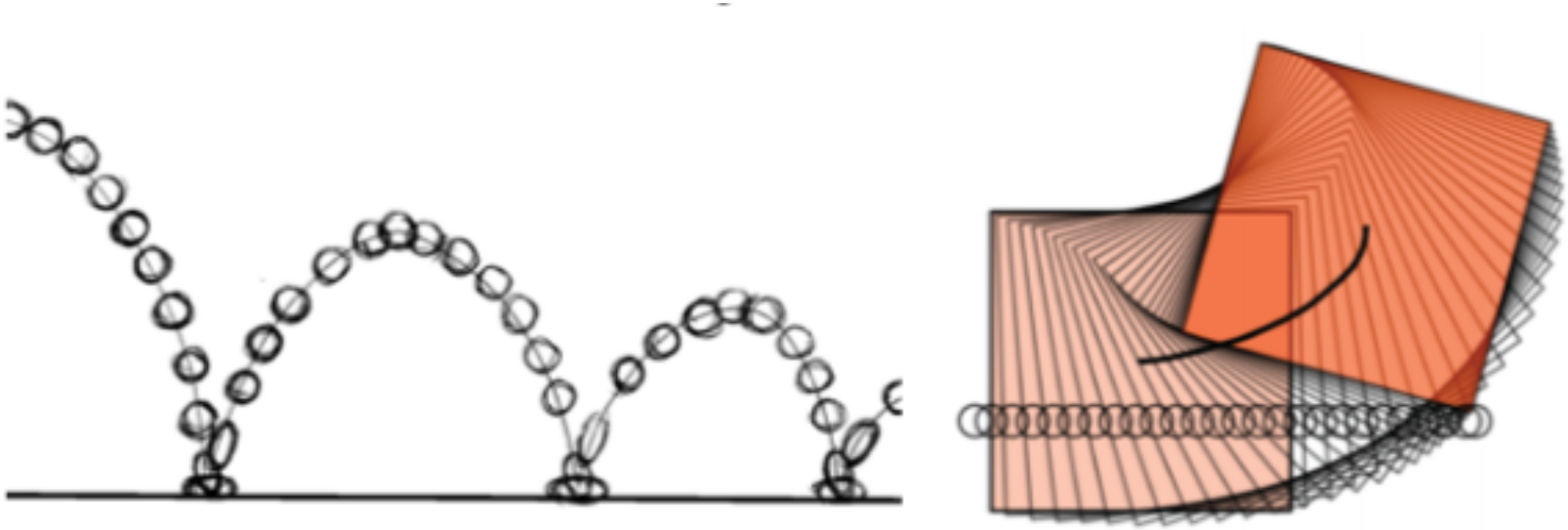


Fig. 4: The two scenarios: ball bouncing and planar pushing.

Results – bouncing ball

	Train				Test			
Models	loss ($\times 10^{-2}$)	trans (%)	pos (mm)	rot (deg)	loss ($\times 10^{-2}$)	trans (%)	pos (mm)	rot (deg)
Zero	N/A	N/A	N/A	N/A	N/A	99.99	359.44	49.46
Physics	N/A	N/A	N/A	N/A	N/A	1.93	6.91	7.71
Neural	0.41	0.72	2.42	1.85	0.68	0.84	2.81	2.48
Hybrid	0.36	0.54	1.86	1.73	0.47	0.60	2.04	2.03

TABLE I: Our hybrid model achieves the best performance in both position and rotation estimation for *rect1*, compared with methods that rely on physics engines or neural nets alone. Here we show results on both training and test sets, as well as the optimization losses. These numbers suggest that our Hybrid model is overfitting to the training set less than the pure Neural model. As we focus on long-term prediction, we include the Zero baseline to show the scale and the challenging nature of the problem.

Results- bouncing ball

prediction, we include the Zero baseline to show the scale and the c

Models	trans (%)	pos (m)	velocity (m/s^2)
Zero	100.00	0.64	1.60
Physics	27.41	0.16	1.06
Neural	9.16	0.058	0.43
Hybrid	2.42	0.016	0.14

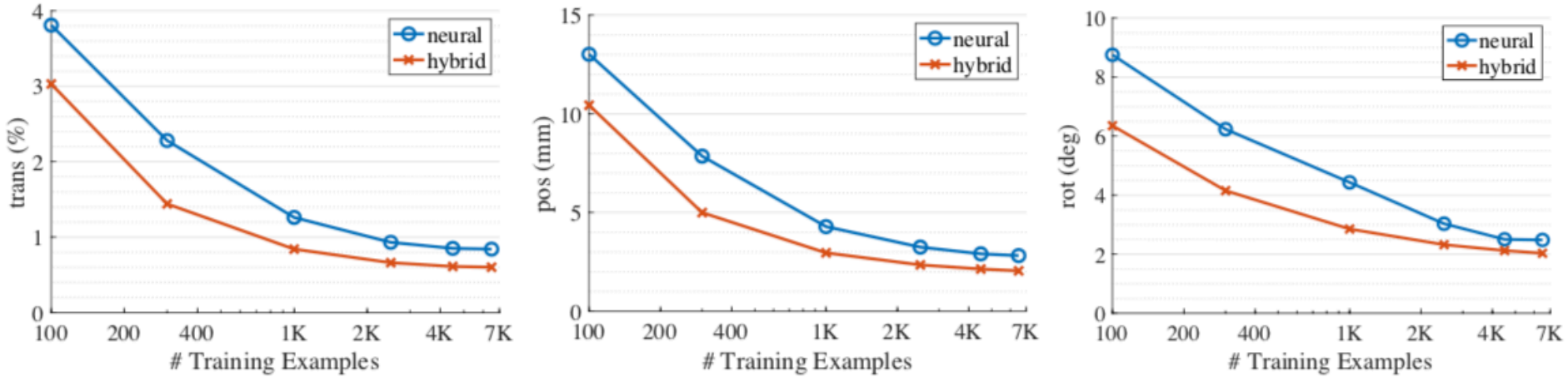
TABLE II: Our hybrid model achieves the best performance in both position and velocity estimation of the ball, compared with methods that rely on physics engines or neural nets alone.

Experiment with different materials

Materials	Models	trans (%)	pos (mm)	rot (deg)
plywood	Zero	99.99	339.12	48.36
	Physics	2.51	5.49	10.38
	Neural	0.92	3.43	2.16
	Hybrid	0.77	2.16	1.65
delrin	Zero	99.99	357.98	52.67
	Physics	1.89	5.78	12.07
	Neural	0.81	2.81	2.50
	Hybrid	0.62	2.09	2.19

TABLE III: Our Hybrid model performs well consistently across object materials. Here for the rectangle made of plywood and delrin, our model again outperforms all other baseline models.

Results – planar pushing



(a) translation (%) (b) position (mm) (c) rotation (deg)

Fig. 5: Prediction errors *vs.* training data size. Our hybrid model not only performs better, but also requires much less data to achieve a given level of performance. In contrast, purely using purely data-driven models requires a larger training set and is not performing as well.

Discussions

- Is the model general enough?
- How can it be adapted to scenarios that is not planar pushing?