BOXLCD: A SIMPLE TESTBED FOR LEARNED SIMULATOR RESEARCH

Anonymous authors
Paper under double-blind review

ABSTRACT

Traditional robotics simulators are inflexible and limited in what they can represent and simulate. Data-driven, or learned simulators, seem a promising way forward. But modeling real world physical systems is challenging and progress in this field remains limited. In this work, we introduce boxLCD, a simplified testbed for developing ideas and algorithms for learning robotics simulators. boxLCD uses the box2D physics engine to simulate simple physical environments, and it renders these environments in low-resolution binarized images (e.g., size 16x16). We provide a set of sample environments and results from training a simple baseline model to predict future video frames. An anonymized GitHub repository with scripts and videos can be found here: bit.ly/3bDMFtc.

1 INTRODUCTION

Simulators are used universally in robotics to develop and debug code which is then run on a physical robot. This enables quicker iteration cycles and helps to mitigate robot and environment damage, among other things. More recently, simulators have also been used as sources of data for training deep learning vision models and policies that then operate in the real world Sadeghi & Levine (2017); Tobin et al. (2017); James et al. (2017); Wilson & Hermans (2019); Chebotar et al. (2019); OpenAI et al. (2020). Sim2real learning provides several advantages over real world training in speed, safety, and environment read + write access. However, the final performance of sim2real relies on both the accuracy of the simulator and the ability to transfer knowledge across whatever reality gap remains. Absent major progress in transfer learning, improving the accuracy of simulators is of crucial importance to the future of sim2real. And perhaps for the future of robotics in general.

Despite the success and promise of simulation, traditional robotics simulators remain inflexible and limited in what they can accurately represent and simulate. Users must arduously specify the many details of the environment they would like to simulate—a haphazard process often requiring substantial effort in manual calibration (Tan et al. (2018); OpenAI et al. (2020)) and still leading to a crude approximation. Furthermore, simulators are often incapable of simulating the relevant physics, no matter how the parameters are set. For example, they don't represent processes like melting and shattering, or strange materials like fire, paint, light switches, and microwaves. And it would require painstaking domain-specific effort to add support for these in the current paradigm.

Data-driven approaches offer some hope of surmounting the issues faced in developing more accurate and usable simulators. Just as humans are unable to write traditional computer programs

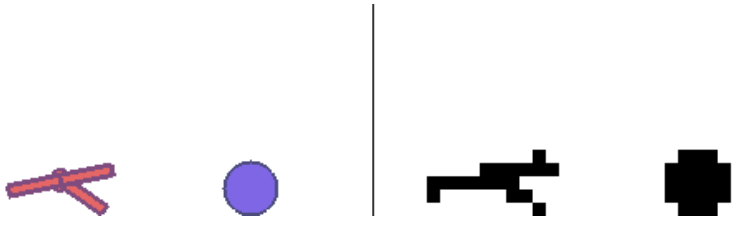

Figure 1: boxLCD sample environment. On the left is a standard RGB image frame, and on the right is a boxLCD low-resolution and binarized image, reminscent of a primitive Liquid Crystal Display.

that outperform neural networks in classifying high-dimensional image data, it seems likely humans will be unable to write computer programs that outperform neural networks in predicting high-dimensional and complicated physical processes. To date, there has been some work on learning parts of simulators from real world data and using them for training. For example, some researchers have learned a model of the physics of a series elastic actuator and used this in the simulator loop Hwangbo et al. (2019); Lee et al. (2020). Others have worked on developing a hybrid learned differentiable simulator Heiden et al. (2020). The scope of these systems, however, remains limited.

Developing a learned robotics simulator to replace traditional ones is a promising endeavor. Simulators learned from real world data are likely to make it easier to solve robotics tasks in the real world. Developing them, however, represents a massive undertaking and research effort. Our philosophy in this work is that it is best to start small, build up strong intuitions, and gradually increase the difficulty of the challenges as progress is made—while always remaining focused on the final goal of systems that work in the real world.

This motivates the idea of boxLCD, which aims to provide a close analogue to the real world problems of learning a robotics simulator, while remaining extremely tractable to work on. In this paper, we describe the design decisions behind boxLCD; we provide a few sample environments; and we provide results from a simple baseline model trained to predict future video frames.

2 BOXLCD DESIGN CHOICES

2.1 EMULATING REAL WORLD CHALLENGES

boxLCD aims to serve as a testbed that accurately captures the challenge of building learned robotics simulators. It is thus built to satisfy several requirements:

Physics-based and actuator-based. The real world has consistent physics, with forces like gravity and friction. Also, robots do not move magically; they must coordinate their many actuators to move their bodies in specific ways. Movements like "forward" and "backward" are not primitives; they are built from joint movements. This is reflected in the testbed through the use of the box2D physics engine and how the robots are designed and simulated.

Pixel-based. Humans and robots primarily sense the real world through vision (pixels). boxLCD is fairly focused around pixel-based sensing and predicting physics by predicting future video.

Multi-modal sensing. Robots often have sensors like joint encoders and inertial measurement units (IMUs) that provide complementary information to vision. boxLCD provides support for proprioceptive information and other modalities.

Partially observable. Sensors do not tell the full story of the world. Robots constantly have to make estimates of the underlying state that they are observing evidence of. boxLCD supports this by providing only proprioception and low-res images, not direct observations of the environment.

Interfaceable. A desirable property of future learned simulators is that they be easy to interface with, for example by letting a user specify scenes by providing structured and unstructured information like meshes, natural language, and video frames. The baseline model we train (Section 3.3) demonstrates the ability to take in and complete a video prompt, but it is interesting future work to experiment with other modalities and description formats as well.

2.2 REMAINING TRACTABLE

At the same time, boxLCD aims to remain computationally tractable and easy to work with:

2D physics. box2D deals with things like contacts, friction, gravity, but it is much simpler and lower-dimensional than the real world or even 3D simulators like Mujoco and Bullet.

Low-resolution rendering. boxLCD enables support for both color and high-dimensional image frames, but the sample environments use at most a 16x32 = 544 sized binary images (smaller than MNIST). This greatly reduces the computational cost to process the data, for example eliminating the need to model complex textures and other high-frequency image details.

Procedurally generated and customizable. boxLCD enables users to create custom new scenarios and generate as much data as they need. The programmability of these environments also makes it easy to evaluate RL policies in different settings.

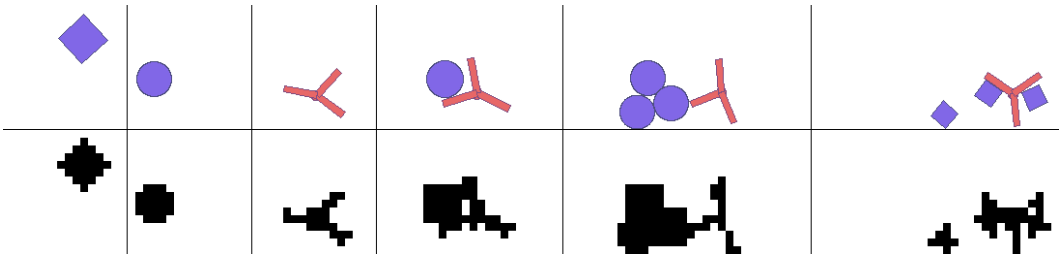

Figure 2: The boxLCD environments we evaluate on in this work. From left to right: Dropbox, Bounce, Urchin, UrchinBall, UrchinBalls, UrchinCubes.

2.3 Related work

While other benchmarks provide some similar features, we are not aware of any work with the same goals as boxLCD, nor ones that simultaneously satisfies the same criteria. For example, PHYRE Bakhtin et al. (2019) and Simulated Billiards Qi et al. (2021) use simple 2D physics systems, but focus on the much simpler case of taking only a single action at the start of the episode. Other environments are either 3D and much more complex and/or do not simulate physics or robots Young & Tian (2019); Xia et al. (2018); Savva et al. (2019); Beattie et al. (2016).

3 Experiments

We present results of training a naive baseline approach on the sample boxLCD tasks showin in Figure 2. The goal is to accurately predict future frames given past frames. For simplicity here, we only deal with the image information, and not multi-modal data like proprioception, making this a straightforward video prediction task. (Code to reproduce these results be found in the example section of the GitHub repository: bit.ly/3bDMFtc).)

3.1 Sample environments

We provide a few sample environments, ranging from fairly trivial to moderately challenging. These environments deal with passive object and environment interactions, robot actuation and action-conditioning, robot-object interactions, and finally robot multi-object interactions. We provide descriptions below, and still frame images in Figure 2.

- **Dropbox (100x16x16):** A large box with random initial XY coordinates and angle.
- **Bounce (200x16x16):** A large a ball with higher restitution (bounciness) than the box, which tends to bounce a few times before coming to rest.
- **Urchin (200x16x16):** A symmetric robot with 3 actuated limbs. This task requires conditioning on actions at every time-step, increasingly the difficulty of the problem.
- **UrchinBall (200x16x24):** Urchin in an environment with a bouncing ball. Induces interactions between a joint-actuated robot and a passive object.
- **UrchinBalls (200x16x32):** Urchin with 3 bouncing balls, introducing complex collisions and multi-object interactions.
- **UrchinCubes (200x16x32):** Urchin with 3 small boxes, like UrchinBalls but more static.

3.2 Baseline approach

Our model is an auto-regressive Transformer (temporally masked), trained to predict the next frame given all past frames. The approach is similar to the GPT models (Radford et al.) but each token is simply the flattened 2D image for that timestep. To train our model, we feed those flat image tokens in; the model produces independent Bernoulli distributions for each pixel in the frame; and we optimize these distributions to match the ground truth (Loss $= -\log p(\mathbf{x})$). To sample the model, we prompt it with the start 10 frames of the episode, and have it predict the rest autoregressively.

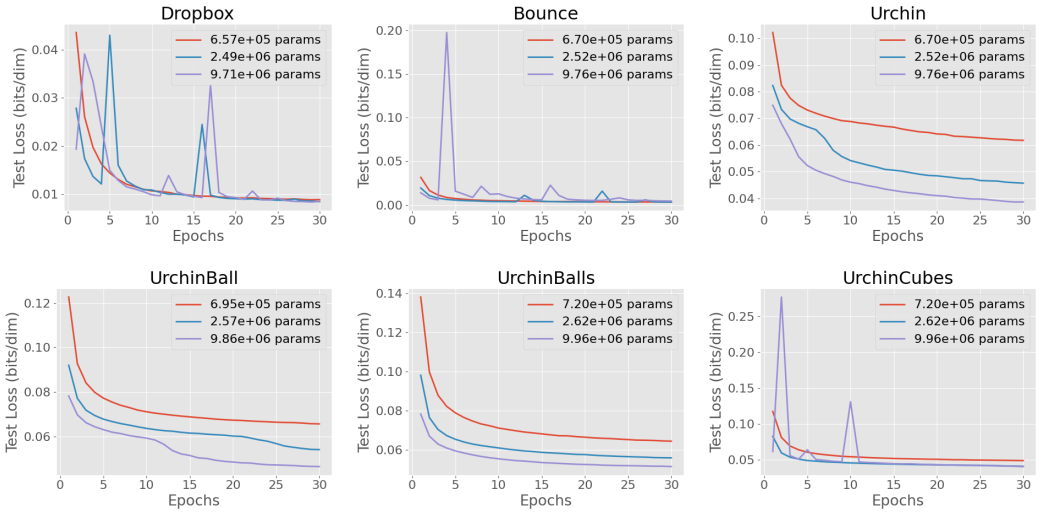

Figure 3: Results for training the baseline approach with different number of parameters.

3.3 Results

We train our baseline model on datasets of 80k rollouts, with a test set of 20k rollouts. Rollout length is determined by the environment, and we report the test loss in bits per dim for various sizes of our baseline model in Figure 3. Video samples can be found here: bit.ly/3cltF1P. (The top frames are ground truths, middles are the predictions, and bottoms are the errors between the two.)

We evaluate on each environment using three sizes of our baseline model, with approximately 500k, 2.5M, and 10M parameters respectively. For Dropbox and Bounce, where the smallest (500k) baseline model achieves nearly pixel perfect predictions, the increase in parameter count does not help and in fact leads to instabilities in training. For the other tasks, though, increasing the parameters improves the loss and the quality of samples, especially for the Urchin task. The small models were trained in as little as 15 minutes, and the larger models were trained for at most 2 hours on a single desktop GPU machine (NVIDIA GTX 1080 Ti).

These results demonstrate the tractability of boxLCD, but also the challenges it provides. Simple boxLCD tasks like Bounce and Dropbox can be solved with very little compute using a naive approach. And while some progress can be made on the Urchin tasks, the losses remain high and the samples poor quality, showing plenty of room for improvement with more sophisticated methods.

4 Conclusion and Future Work

boxLCD aims to serve as a testbed for learning robotics simulators. Ultimately, the goal is to learn systems in the real world that end up helping us solve real world problems. We believe it is still early days for video prediction, generative modeling, and learned simulation, and that this testbed will help provide greater traction on these problems.

boxLCD is still in active development. The current sample environments are useful for developing learned simulator models. But in the future, we plan to add more complex environments and agents, goal-based tasks, modalities like touch and sound, and other features to emulate what will be necessary in developing real learned simulators.

References

Anton Bakhtin, Laurens van der Maaten, Justin Johnson, Laura Gustafson, and Ross Girshick. Phyre: A new benchmark for physical reasoning. 2019.

Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018.

Charles Beattie, Joel Z Leibo, Denis Teplyashin, Tom Ward, Marcus Wainwright, Heinrich Küttler, Andrew Lefrancq, Simon Green, Víctor Valdés, Amir Sadik, et al. Deepmind lab. *arXiv preprint arXiv:1612.03801*, 2016.

Yevgen Chebotar, Ankur Handa, Viktor Makoviychuk, Miles Macklin, Jan Issac, Nathan Ratliff, and Dieter Fox. Closing the sim-to-real loop: Adapting simulation randomization with real world experience. In *2019 International Conference on Robotics and Automation (ICRA)*, pp. 8973–8979. IEEE, 2019.

Eric Heiden, David Millard, Erwin Coumans, Yizhou Sheng, and Gaurav S Sukhatme. Neuralsim: Augmenting differentiable simulators with neural networks. *arXiv preprint arXiv:2011.04217*, 2020.

Jemin Hwangbo, Joonho Lee, Alexey Dosovitskiy, Dario Bellicoso, Vassilios Tsounis, Vladlen Koltun, and Marco Hutter. Learning agile and dynamic motor skills for legged robots. *Science Robotics*, 4(26), 2019.

Stephen James, Andrew J Davison, and Edward Johns. Transferring end-to-end visuomotor control from simulation to real world for a multi-stage task. In *Conference on Robot Learning*, pp. 334–343. PMLR, 2017.

Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. Learning quadrupedal locomotion over challenging terrain. *Science robotics*, 5(47), 2020.

OpenAI, Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafal Jozefowicz, Bob McGrew, Jakub Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, et al. Learning dexterous in-hand manipulation. *The International Journal of Robotics Research*, 39(1):3–20, 2020.

Haozhi Qi, Xiaolong Wang, Deepak Pathak, Yi Ma, and Jitendra Malik. Learning long-term visual dynamics with region proposal interaction networks. In *ICLR*, 2021.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training.

Fereshteh Sadeghi and Sergey Levine. Cad2rl: Real single-image flight without a single real image. *Robotic: Science and Systems (RSS)*, 2017.

Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, et al. Habitat: A platform for embodied ai research. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9339–9347, 2019.

Jie Tan, Tingnan Zhang, Erwin Coumans, Atil Iscen, Yunfei Bai, Danijar Hafner, Steven Bohez, and Vincent Vanhoucke. Sim-to-real: Learning agile locomotion for quadruped robots. *arXiv preprint arXiv:1804.10332*, 2018.

Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pp. 23–30. IEEE, 2017.

Matthew Wilson and Tucker Hermans. Learning to manipulate object collections using grounded state representations. *Conference on Robot Learning*, 2019.

Fei Xia, Amir R. Zamir, Zhi-Yang He, Alexander Sax, Jitendra Malik, and Silvio Savarese. Gibson Env: real-world perception for embodied agents. In *Computer Vision and Pattern Recognition (CVPR), 2018 IEEE Conference on*. IEEE, 2018.

Kenny Young and Tian Tian. Minatar: An atari-inspired testbed for thorough and reproducible reinforcement learning experiments. *arXiv preprint arXiv:1903.03176*, 2019.