Unsupervised Intuitive Physics from Visual Observations

Sebastien Ehrhardt[1]
hyenal@robots.ox.ac.uk
Aron Monszpart[2]
aron.monszpart.12@cs.ucl.uk
Andrea Vedaldi[1]
vedaldi@robots.ox.ac.uk
Niloy Mitra[2]
n.mitra@cs.ucl.uk

[1] Dept of Engineering Science
University of Oxford
Oxford, UK

[2] Dept of Computer Science
University College London,
London, UK

Abstract

While learning models of *intuitive physics* is an increasingly active area of research, current approaches still fall short of natural intelligences in one important regard: they require external supervision, such as explicit access to physical states, at training and sometimes even at test times. Some authors have relaxed such requirements by supplementing the model with an handcrafted physical simulator. Still, the resulting methods are unable to automatically learn new complex environments and to understand physical interactions within them. In this work, we demonstrated for the first time learning such predictors *directly from raw visual observations and without relying on simulators*. We do so in two steps: first, we learn to track mechanically-salient objects in videos using causality and equivariance, two unsupervised learning principles that do not require auto-encoding. Second, we demonstrate that the extracted positions are sufficient to successfully train visual motion predictors that can take the underlying environment into account. We validate our predictors on synthetic datasets; then, we introduce a new dataset, ROLL4REAL, consisting of real objects rolling on complex terrains (pool table, elliptical bowl, and random height-field). We show that in all such cases it is possible to learn reliable extrapolators of the object trajectories from raw videos alone, without any form of external supervision and with no more prior knowledge than the choice of a convolutional neural network architecture.

1 Introduction

A striking property of natural intelligences is their ability to perform accurate and rapid prediction of physical phenomena using only noisy sensory inputs. Even more remarkable is the fact that such predictors are learned without explicit supervision; rather, natural intelligences induce their internal representation of physics automatically from experiences.

Several authors have recently looked into the problem of learning physical predictors using deep neural networks in order to partially mimic this functionality. Early attempts predicted trajectories in hand-crafted spaces of physical parameters, such as positions and

velocities, assuming that the ground-truth values of such parameters are fully observable during training. Others have considered performing predictions from visual observations, but used full supervision for training. Furthermore, while several papers [3, 5] make use of simulators as a way to generate the required supervisory signals, limited work has been done in transferring such models to real data.

In this paper, we also investigate learning *physical predictors* using deep neural network. However, we do so in a **fully unsupervised manner**, learning from observations of unlabelled video sequences. In contrast to approaches such as the recent de-animation method of [28], we do not require synthetic data, nor do we rely on any handcrafted physical simulator for prediction. Our models are built directly from real data and learn intuitive physics models that empirically outperform more principled, but more brittle, models based on physical parameters [21].

As a working example, we consider video footage of balls rolling on various surfaces, such as pool tables, bowls and random height-fields. Balls interact with the underlying environment and among themselves. For rigorous assessment, we contribute **a new public dataset**, ROLL4REAL, containing a large number of such sequences captured in real-life. We then make two technical contributions. First, inspired by [16], we show that an object **detector** can be learned in an **unsupervised manner** by tuning a convolutional detector to extract tracks that are maximally characteristic of the natural, causal ordering of the frames in a video. Second, we use these trajectories to learn **visual predictors** that automatically learn an internal representation of physics and can extrapolate the trajectory of the balls more reliably than approaches such as Interaction Networks (IN) [3] that use direct measurements of physical parameters.

Empirically, we show that these models more gracefully handle observation noise compared to approaches such as [3, 5] that are learned using physical ground-truth parameters extracted from simulated scenarios. We also show that the Visual Interaction Network (VIN) of [15], which also propose a vision-based physical predictor, fails to account for the interaction of the objects and their environment, whereas the other approaches succeed.

The rest of the paper is organized as follows. We discuss related work in section 2. We then present the technical details of our approach in section 3. Next, we introduce the new ROLL4REAL data in section 4 and use the latter as well as several existing synthetic datasets to evaluate the approach in section 5. We summarise our findings in section 6.

2 Related Work

Existing work in learning physics can be organised according to several axis. **Nature of the representation of physics:** Approaches such as [9, 29] use fully-fledged physical simulators to model physics and integrate dynamics. Some focus on representing a small subset of physical parameters such as position and velocities [4, 22, 26, 27, 29]. Others learn an implicit representation of physics, usually as activations in a deep neural network [7, 13, 25]. **Hand-crafted vs learned dynamics:** Some approaches [26], including simulation-based ones [4, 29], extrapolate physics by explicitly integrating parameters such as velocities, while others [4, 29], including methods using an implicit representation of physics, learn recurrent predictors to do so. **Physical vs visual observations:** Many approaches assume direct access to physical quantities such as positions and velocities for prediction, whereas others [4, 9, 11, 13, 14, 22, 25, 29] take as input one or more video frames of a scene. **Qualitative vs quantitative predictions:** While most of the papers discussed above consider

quantitative predictions such as extrapolating trajectory, others have considered *qualitative* predictions such as the stability of stacks of objects [4, 13, 14]. Other papers are in between, and learn *plausible if not accurate physical predictions* [12, 17, 20, 24], often for 3D computer graphics. **Nature of the supervision:** Most approaches are *passive and supervised*, as they are passive observer of physical scenarios and use ground truth information about key physical parameters (positions, velocities, stability) during training. This is contrasted by [2, 6] that attempted to learn intuitive physics of objects through manipulation and hence is *active and unsupervised*. **Scenarios:** Two favorite scenarios in such experiments are bouncing balls, including billard-like environment [10], and block towers [13]. As a variant, [25] consider balls subject to gravitational pulls, ignoring harder-to-model collisions. Most papers make use of simulated data, with limited validation on real data. A different approach [18] is to predict qualitative object forces and trajectories in fully-unconstrained natural images. The approach of [2] considers instead learning from active poking using a real-life robot. In most cases experiments are done on synthetic data. However, approaches such as [14, 27] also used real data; [27] also contributed a dataset of videos of short physical experiments called *Phys-101*.

We relate to such previous work in that we also make physical predictions of the trajectory of ball-like objects. However, we differ in two significant ways. First, our approach, while using only passive observations, is *fully unsupervised*, and yet competitive if not more accurate than supervised counterparts. In particular, while [22, 29] also do not use image labels, they use *a-priori* knowledge of physics for training (a fully-fledged simulator and rendered in the case of [29]). Second, we systematically test on *several real-life scenarios*, both in training and testing, using our new ROLL4REAL. Compared to datasets such as *Phys-101*, ours allows testing long-term ball-rolling prediction in complex scenarios.

3 Method

We first show how to learn to track objects in raw video sequences without any supervision or any prior knowledge of physics (section 3.1). Then, we use the resulting object trajectories to learn visual predictors that can extrapolate the object positions through time, thus implicitly learning physics (section 3.2).

3.1 Unsupervised tracking of a mechanically-salient objects

Single-object detection. Let $\mathbf{x}_t \in \mathbb{R}^{H\times W\times 3}$ be a video frame. We assume we are given video sequences $\mathcal{X} = (\mathbf{x}_1, \dots, \mathbf{x}_N)$, initially containing a single object moving in an environment, such as a single rolling ball. Our goal is to learn a detector function $\Phi(\mathbf{x}_t) = u_t \in \mathbb{R}^2$ that extracts the 2D position u_t of the moving object at any given time (Fig. 1). The challenge is to do so *without* access to any label for supervision.

We start by implementing $\Phi(\mathbf{x}_t)$ as a shallow Convolutional Neural Network (CNN) that extracts a scalar score $f_v \in \mathbb{R}$ for each image pixel $v \in \Omega = \{1, \dots, H\} \times \{1, \dots, W\}$. These are then normalised to a probability distribution using the softmax operator $s_v = e^{f_v} / \sum_{z\in\Omega} e^{f_z}$ and the location u of the object is obtained as the expected value $u = \sum_v v s_v$ [9].

We learn Φ by combining two learning principles. The first one is **causality** and is inspired by [16]. Applied to a video sequence, the detector produces a trajectory $\Phi(\mathcal{X}) = (\Phi(\mathbf{x}_1), \dots, \Phi(\mathbf{x}_N))$. We expect that, when the detector locks properly on the rolling object, the trajectory is physically plausible (e.g., causal/smooth); at the same time, if the frames are shuffled by random permutation π, the resulting trajectory should *not* be plausible anymore.

We incorporate this constraint by learning a discriminator network $D(\Phi(\mathbf{x}_{\pi_1}),\dots,\Phi(\mathbf{x}_{\pi_5}))$ that, for a subsequence, can distinguish between the natural ordering of the frames and a random shuffle (top row of Fig. 1). The permutation π is sampled with 50% probability as a consecutive sequence of 5 frames ($\pi_{i+1}=\pi_i+1$, $i=1,\dots,4$) and with 50% uniformly at random. The discriminator is a 3 layers multi-layer perceptron followed by a sigmoid and the loss

$$\mathcal{L}_{disc}(D,\pi)=\begin{cases}-\log D, & \pi_{i+1}=\pi_i+1,\ i=1,\dots,4,\\ -\log(1-D), & \text{otherwise.}\end{cases}$$

The second learning principle is **equivariance**, proposed in [19, 23], which suggests that if a transformation g is applied to a frame $\mathbf{x}_t$ (e.g., a ± 90 degrees rotation), then the output of the detector should change accordingly: $\Phi(g\mathbf{x}_t)=g\Phi(\mathbf{x}_t)$. This is implemented as a Siamese branch (bottom row in Fig. 1) extracting 2D positions $\Phi(g\mathcal{X})=(\Phi(g\mathbf{x}_1),\dots,\Phi(g\mathbf{x}_N))$ from the rotated frames and comparing them to the rotated 2D positions extracted from the orignal frames using a L^2 loss: $\mathcal{L}_{siam}=\frac{1}{N}\sum_t ||g^{-1}\Phi(g\mathbf{x}_t)-\Phi(\mathbf{x}_t)||^2$.

Furthermore, in order to encourage the softmax operator to produce peaky distributions, we minimise the entropy of the resulting distribution $\mathcal{L}_{ent}=-\sum_{v\in\Omega}s_v\log(s_v)$. The final loss is therefore $\mathcal{L}=\lambda_d\mathcal{L}_{disc}+\lambda_e\mathcal{L}_{ent}+\lambda_s\mathcal{L}_{siam}$. In our experiment, $\lambda_d=1$, $\lambda_e=0.01$, and $\lambda_s=0.001$.

Multi-object tracking. We now extend the method from detection of single objects to tracking of multiple objects. In order to do so, the network is fine-tuned to videos containing two or more moving objects of different colors. Due to its structure, the network can only detect a single object, which is also encouraged by the entropy loss. Importantly, however, such detections are encouraged to be *consistent* across frames (as opposed to jumping around) by the causality loss, which would be poor otherwise.

The resulting network therefore consistently tracks a single object selected at random among the visible ones. Once this is done, in the next iteration, a second object is detected by suppressing (setting to zero) a circular region of radius r around the first object location in the activations f_v immediately preceding the softmax operator, and so on. Before the suppression we also add a positive bias to the activations f_v in order to consider the previously detected objects as zero probabilities in the new heatmap.

3.2 Trajectory extrapolation networks

We consider existing network modules for physical prediction. While these modules use external supervision in the original papers, here we apply them to the output of the unsupervised tracker making them fully unsupervised.

We experiment in particular with *PosNet*, *DispNet*, and *ProbNet* from [8], configuring them to take as input the first four frames of a sequence and to produce as output the prediction of future object positions. These models learn an implicit representation of physics, which is extrapolated automatically by a recurrent propagation layer and used to extract estimates of the object positions. The difference between the models is that *PosNet* regresses positions from state, while *DispNet* and *ProbNet* regress displacements from state. Furthermore, *ProbNet* produces a probability estimate over trajectories.

We also consider the *Visual Interaction Network* (VIN) module and its variant *Interaction Network from State* (IFS) [25]. While VIN uses only visual inputs for prediction just like the other networks, IFS works with an explicit state vector of physical parameters, which we set as the stacking of the 2D positions for four past frames which start with positions

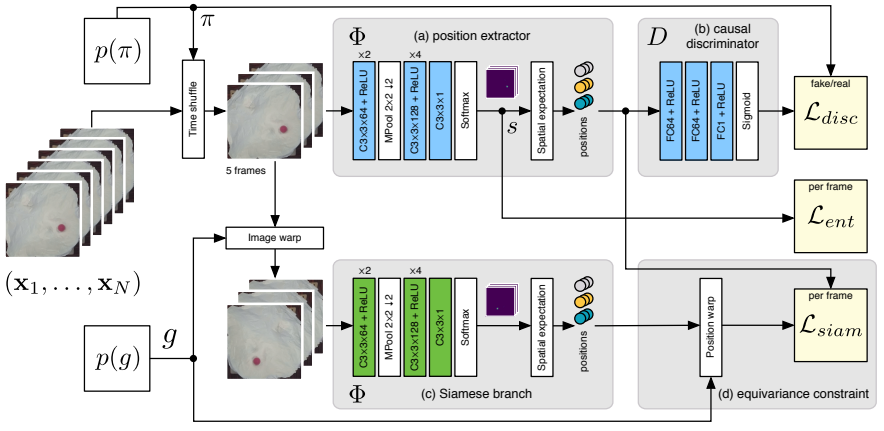

Figure 1: **Overview of our unsupervised object tracker**. Each training point consists of a sequence of five video frames. Top: the sequence is randomly permuted with probability 50%. The position extractor (a) computes a probability map s for the object location, whose entropy is penalised by $\mathcal{L}_{ent}$. The reconstructed trajectory is then fed to a causal/non-causal discriminator network (b) that determines whether the sequence is causal or not, encouraged by $\mathcal{L}_{disc}$. The bottom Siamese branch (c) of the architecture takes a randomly warped version of the video and is expected by $\mathcal{L}_{siam}$ to extract correspondingly-warped positions in (d). Blue and green blocks contain learnable weights and blue blocks are siamese shared ones. At test time only Φ is retained.

extracted from our tracker. Additionally, in the synthetic experiments (first part of Table 1), IFS uses velocity and in BOWLS experiments the ground-truth ellipsoid axes parameters are appended to the state to inform the model of the shape of the ground. IFS and VIN are trained following [8]; in particular, this means that VIN uses the same setting as the original paper (32×32 pixels images).

All such models are trained by showing the network four initial frames of a sequence and the output of the unsupervised tracker up to time $T_{\text{train}} \in \{15, 20\}$ frames. At test time, the networks, which are recurrent, are used to extrapolate the trajectory up to an arbitrary time T, also starting from four video frames. We test in particular $T = T_{\text{train}}$ and $T \gg T_{\text{train}}$ to assess the generalization capabilities of the models learned by the network.

In addition, for some experiments on single object we also consider *linear* and *quadratic* extrapolators as baselines. In both cases we fit a first (respectively second) order polynomial to the 10 first positions given as input (hence with a significant advantage compared to the networks which only observe four frames).

4 ROLL4REAL: A New Benchmark Dataset

In the absence of a suitable real-world dataset to evaluate intuitive physics on objects rolling on complex terrains, we created a new benchmark, ROLL4REAL (R4R).

Dataset content. R4R consists of 1118 short 256×256 videos containing one or two balls rolling on three types of terrains (Fig. 2): a flat pool table (POOLR), a large ellipsoidal 'bowl' (BOWLR), and an irregular height-field (HEIGHTR). More specifically, there are 151 videos (avg. 99 frames/video) for the POOLR dataset with one ball; 216 videos (522 frames/video)

Figure 2: **Physical setup.** In each of the three real-world scenarios (POOLR, BOWLR, HEIGHTR), we show the experimental setup (left) and a sample data frame (right).

for the BOWLR dataset with one ball; 543 videos (356 frames/video) for the HEIGHTR dataset with one ball; and 208 videos (206 frames/video) for the HEIGHTR dataset with two balls. We rolled a total of 7 differently colored balls for the HEIGHTR and BOWLR datasets, varying from 3.5 cm to 7 cm in diameter. The height-field surface fits into a $70 \times 70 \times 28$ cm^3 bounding box, with 76 cm diameter. The bowl was created using a 70 cm diameter ball, and is 60 cm high. Videos were randomly split into *train*, *validation*, and *test* sets. Ground-truth annotations are provided for the test split.

Dataset collection. Both the bowl and height-field terrains were modeled using paper mâché on scaffolds, using a large inflatable ball and a custom-made wire-mesh frame, respectively. For the the POOLR dataset, balls were rolled on the table, while for the other settings, balls were manually dropped from a small height and allowed to roll on. The setup was imaged using a fixed camera (Samsung Galaxy S8) from the top. The POOLR dataset was captured at 30fps (due to low light), while all the others at 240fps in order to reduce motion blur and later downsampled to 80fps. Videos were cropped to only focus on the scenario of interest, i.e., ball(s) and terrain, and trimmed to retain the portion of the video containing motion.

In order to create ground-truth tracks for the ball centers, we used a template-based tracker using zero-normalized cross-correlation in the LAB colorspace, and tracked each frame along with a smoothness term over time. The setup was manually initiated by providing suitable template. The raw results were then manually inspected, corrected, and saved as ground-truth.

5 Results and Discussion

Implementation details. We used Tensorflow [1] on a single NVIDIA Titan X GPU for all the experiments. All the networks were initialised with a learning rate of 10^{-4} that was progressively decreased by a factor of 10 when no improvements were found over K epochs ($K = 100$ for the synthetic datasets). Training was stopped when the loss did not decrease for $2K$ consecutive epochs. Before processing images, we resized all dataset images to 128×128 pixels to fit in the GPU memory.

5.1 Unsupervised tracker

We first evaluate our unsupervised tracker. In addition to POOLR, BOWLR, and HEIGHTR from ROLL4REAL, we also consider two synthetic datasets from [8]: BOWLS for the ellipsoidal bowl with one or two balls and HEIGHTS for the random height-fields. Fig. 3-left reports the mean and 99$^{\text{th}}$ percentile pixel error of the extracted object positions against ground-truth averaged over multiple runs of our experiments.

We note that our method was able to track the objects with good accuracy in all cases. This is very encouraging especially for challenging real datasets containing less training data and larger variations of the environment (lighting conditions, height-field/ellipsoid vibrations

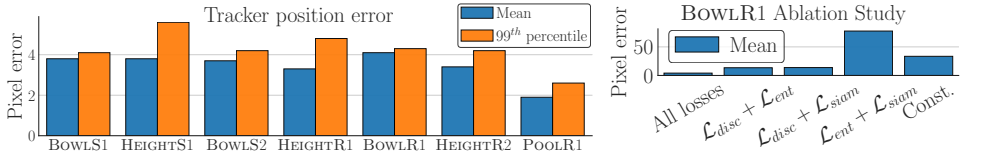

Figure 3: **Tracker errors and Ablation study.** Left: Tracker errors on different dataset. The errors are consistently small across dataset and show that out tracker can perform well on a different range of real situations. Right: Ablation study. We try different combination of tracker losses on the BOWLR dataset. 'Const.' indicates that we are predicting a constant point at the center of the image for reference. For left and right, position errors are reported in pixels. The number of balls in the datasets is appended to the name of the dataset.

and movements). The 99^{th} percentile plots also show that the error is consistent across all experiments. These results show that our method learns to track object robustly in a diverse range of complex scenarios.

We also conducted an ablation study on the BOWLR dataset to measure the impact of each loss term. Fig. 3-right shows that, while each loss contributes to the final results, the best performance is obtained when all the terms are used.

5.2 Extrapolation

Supervised vs unsupervised (single ball synthetic datasets). We now compare training predictors using either ground-truth object positions or the output of the unsupervised tracker. All predictors observe only $T_0 = 4$ frames as input (either positions or video frames) except VIN which uses $T_0 = 6$ and the least squares baselines which use $T_0 = 10$. All the networks are trained to predict T_{train} positions and Table 2 reports the average errors at time T_{train} and $2T_{train}$ to measure the ability of predictors to generalise beyond the training regime.

We see that the *Net* models (*ProbNet*, *DispNet*, *PosNet*) perform well using ground-truth positions or the unsupervised tacker outputs (e.g. *PosNet* error for BOWLS/HEIGHTS is 2.9/6.4 supervised vs 4.9/6.9 unsupervised), whereas IFS does not handle the transition well (3.3/10.4 to 13.3/23.1) and Linear, Quadratic and VIN are not competitive.

The main weakness of the *Net* models is that their performance degrades beyond the training regime at $2T_{train}$, whereas IFS is more resilient. However, in this case, *ProbNet* indicates that the model is uncertain with its estimation.

Synthetic vs real (one ball datasets). On real datasets (Table 2), the *Net* models uniformly outperform others at both T_{train} and $2T_{train}$, with errors comparable to the synthetic case. Note that the real datasets in ROLL4REAL are particularly challenging due to the non-idealities of the surface (e.g. the BOWLR surface is slightly elastic and wobbles as the ball rolls).

One vs multiple balls (real and synthetic datasets). Finally, we move to cases where the balls are interacting with the environment and with each others due to collisions. This is particularly challenging when no ground-truth is used as multiple object tracking is much harder to achieve in an unsupervised setting than tracking a single object.

As shown in Table 3, the *Net* models still perform well. Due to memory limitations, models were trained for a slightly shorter time span T_{train}; since the corresponding predictions are shorter term, their errors are a little lower than before. Overall, the results show that neither perfect ground-truth annotations nor a very large dataset is required to train a reliable

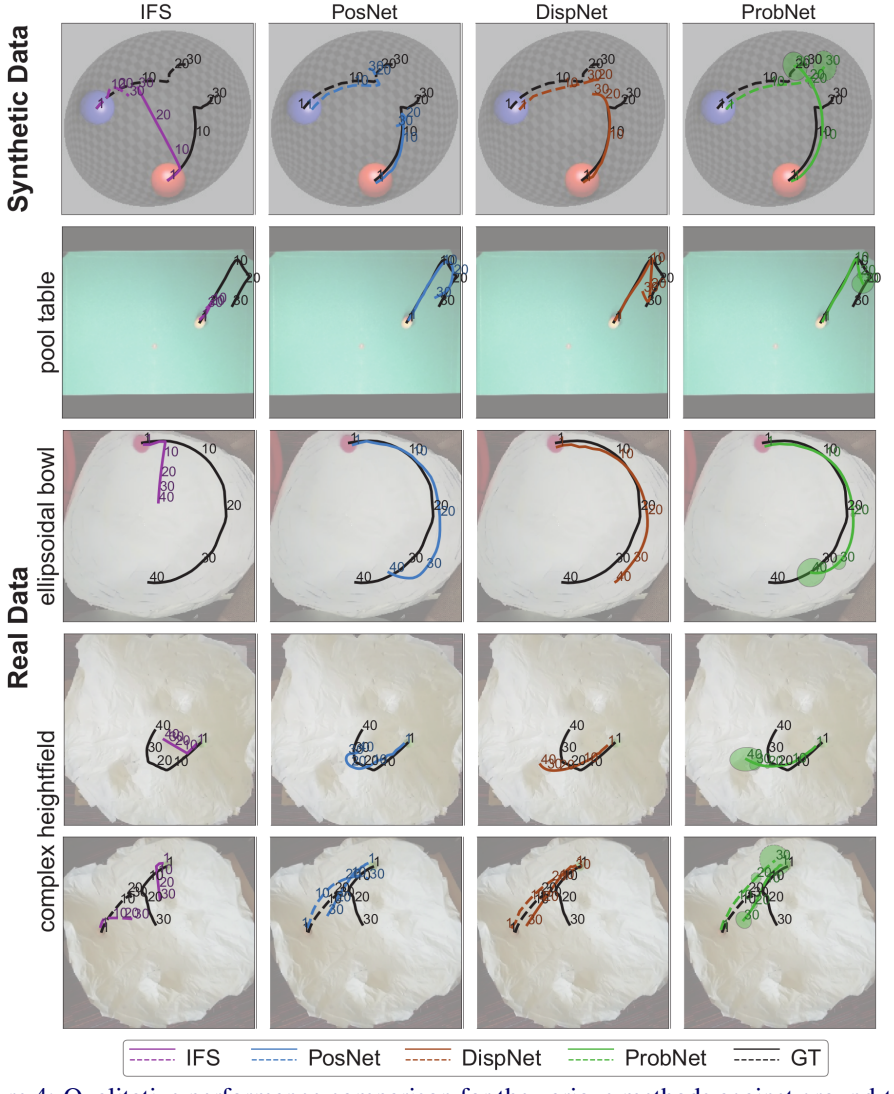

Figure 4: Qualitative performance comparison for the various methods against ground-truth trajectories. Top-to-bottom: two balls colliding on an ellipsoidal bowl; single ball colliding against the walls of a pool table; single ball rolling on an ellipsoidal bow; single ball rolling on complex height-field; and two balls rolling on complex height-field. The top row is on synthetic data, while the other rows are on real-data. The green ellipsoids in the last column show the variance of the predictions estimated by *ProbNet* at selected locations.

physical extrapolator. Still, we noticed that collisions were difficult to predict in the HEIGHTR dataset (see the bottom row of Fig. 4), probably because such events are rare during training.

Table 1: **Long term predictions compared on synthetic datasets with model trained with ground-truth from simulator.** All the models (except VIN, Linear, and Quadratic) are given $T_0 = 4$ frames as input and train to predict first T_{train} positions. We report the average pixel error and perplexity for *PosNet* model at two different times. Perplexity, shown in bracket, is defined as $2^{-\mathbb{E}[\log_2(p(x))]}$ where p is the estimated posterior distribution. *State* shows either the carried forward state is a physical quantity (Exp.), or an implicit vector or tensor (Imp.)

Method	Input	State	BOWLS- T_{train}= 20 Pixel error (Perplexity) at T $T = T_{\text{train}}$	$T = 2\times T_{\text{train}}$	HEIGHTS- T_{train}= 20 Pixel error (Perplexity) at T $T = T_{\text{train}}$	$T = 2\times T_{\text{train}}$
With positions from simulator						
Linear	2D pos.	Exp.	61.9	20.1	21.3	61.9
Quadratic	2D pos.	Exp.	11.7	93.1	26.7	126.0
IFS	2D pos.	Exp.	3.3	**8.9**	10.4	27.6
VIN	Visual	Imp.	24.0	30.2	42.6	42.7
PosNet	Visual	Imp.	**1.6**	24.4	7.2	24.6
DispNet	Visual	Imp.	2.5	20.6	7.7	25.8
ProbNet	Visual	Imp.	2.9 (32.1)	21.8 (54.0)	**6.4** (9.5)	**22.5** (12.7)
With positions from *unsupervised* tracker						
IFS	2D pos.	Exp.	13.3	**23.6**	23.1	38.3
VIN	Visual	Imp.	24.7	30.3	45.8	48.0
PosNet	Visual	Imp.	4.3	29.9	**6.6**	25.6
DispNet	Visual	Imp.	**3.9**	25.6	6.8	**22.7**
ProbNet	Visual	Imp.	4.9 (6.3)	27.0 (20.6)	6.9 (8.3)	23.3 (13.4)

Table 2: **Long term predictions using one ball and real data.** The table has the same format as Table 1. All models are trained using the unsupervised tracker.

Method	Input	State	POOLR- $T_{train} = 15$ Pix. err. (Perplexity) at T $T = T_{\text{train}}$	$T = 2\times T_{\text{train}}$	HEIGHTR- $T_{train} = 20$ Pix. err. (Perplexity) at T $T = T_{\text{train}}$	$T = 2\times T_{\text{train}}$	BOWLR- $T_{train} = 20$ Pix. err. (Perplexity) at T $T = T_{\text{train}}$	$T = 2\times T_{\text{train}}$
IFS	2D pos.	Exp.	26.0	37.5	48.0	58.1	26.2	39.1
VIN	Visual	Imp.	50.9	40.8	40.2	47.3	33.9	33.0
PosNet	Visual	Imp.	4.6	21.4	**5.6**	29.0	**5.6**	23.0
DispNet	Visual	Imp.	**3.8**	23.6	**5.6**	**28.5**	6.5	**22.6**
ProbNet	Visual	Imp.	4.7(6.3)	**16.3**(11.3)	5.7(5.8)	30.0(22.5)	6.8(6.8)	23.5(13.8)

6 Conclusions

We presented a method that can learn to track physical objects such as balls rolling on complex terrains using only raw video sequences and no supervision. Combined with recent neural networks that can learn an implicit representation of physics, such a system is able to extrapolate object trajectories over time while accounting for object-environment and object-object interactions. To the best of our knowledge, this is the first time that learning long-term physics extrapolation without access to supervision or handcrafted simulators has been demonstrated.

We also contributed a new dataset, ROLL4REAL, of real-life video sequences for complex scenarios such as ball rollings on pool tables, bowls, and height-field, showing that all such methods are applicable to the real world. This data will be made publicly available.

In the future, we plan to train the tracker and the extrapolator end-to-end, further improving tracking of multiple objects. We also aim at improving the generalisation of our predictor beyond the training regime; we believe that the key is to factor knowledge about the environment and the object dynamics to allow the models to remember the first better over longer time spans.

Table 3: **Long term predictions using two balls on real and synthetic data.** Table layout and measures are the same as Table 1. Models are trained with positions from tracker.

Method	Input	State	BOWLS 2b.-$T_{train} = 15$ Pix. err. (Perplexity) at T		HEIGHTR 2b.-$T_{train} = 15$ Pix. err. (Perplexity) at T	
			$T = T_{\text{train}}$	$T = 2 \times T_{\text{train}}$	$T = T_{\text{train}}$	$T = 2 \times T_{\text{train}}$
IFS	2D pos.	Exp.	18.4	30.0	15.6	26.6
VIN	Visual	Imp.	41.3	45.8	45.9	39.8
PosNet	Visual	Imp.	**5.0**	**13.4**	**5.4**	**12.5**
DispNet	Visual	Imp.	5.5	24.7	6.2	15.4
ProbNet	Visual	Imp.	5.6 (7.3)	20.6 (13.7)	6.8 (7.9)	16.9 (12.4)

References

[1] Abadi et al. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] Pulkit Agrawal, Ashvin V Nair, Pieter Abbeel, Jitendra Malik, and Sergey Levine. Learning to Poke by Poking: Experiential Learning of Intuitive Physics. In *Proc. NIPS*, pages 5074–5082, 2016.

[3] Peter Battaglia, Razvan Pascanu, Matthew Lai, Danilo Jimenez Rezende, et al. Interaction networks for learning about objects, relations and physics. In *Proc. NIPS*, pages 4502–4510, 2016.

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

[5] Michael B Chang, Tomer Ullman, Antonio Torralba, and Joshua B Tenenbaum. A compositional object-based approach to learning physical dynamics. In *Proc. ICLR*, 2017.

[6] Misha Denil, Pulkit Agrawal, Tejas D Kulkarni, Tom Erez, Peter Battaglia, and Nando de Freitas. Learning to perform physics experiments via deep reinforcement learning. *Deep Reinforcement Learning Workshop, NIPS*, 2016.

[7] Sébastien Ehrhardt, Aron Monszpart, Niloy J. Mitra, and Andrea Vedaldi. Learning A Physical Long-term Predictor. *arXiv e-prints arXiv:1703.00247*, March 2017.

[8] Sébastien Ehrhardt, Aron Monszpart, Andrea Vedaldi, and Niloy J. Mitra. Learning to Represent Mechanics via Long-term Extrapolation and Interpolation. *arXiv preprint arXiv:1706.02179*, June 2017.

[9] Chelsea Finn, Xin Yu Tan, Yan Duan, Trevor Darrell, Sergey Levine, and Pieter Abbeel. Deep spatial autoencoders for visuomotor learning. In *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, pages 512–519. IEEE, 2016.

[10] Katerina Fragkiadaki, Pulkit Agrawal, Sergey Levine, and Jitendra Malik. Learning visual predictive models of physics for playing billiards. In *Proc. NIPS*, 2016.

[11] Ken Kansky, Tom Silver, David A Mély, Mohamed Eldawy, Miguel Lázaro-Gredilla, Xinghua Lou, Nimrod Dorfman, Szymon Sidor, Scott Phoenix, and Dileep George. Schema networks: Zero-shot transfer with a generative causal model of intuitive physics. In *International Conference on Machine Learning*, pages 1809–1818, 2017.

[12] Ladický, Jeong, SoHyeon, Barbara Solenthaler, Marc Pollefeys, Markus Gross, et al. Data-driven fluid simulations using regression forests. *ACM Trans. on Graphics (TOG)*, 34(6):199, 2015.

[13] Adam Lerer, Sam Gross, and Rob Fergus. Learning physical intuition of block towers by example. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, pages 430–438, 2016.

[14] Wenbin Li, Aleš Leonardis, and Mario Fritz. Visual stability prediction and its application to manipulation. *AAAI*, 2017.

[15] Pauline Luc, Natalia Neverova, Camille Couprie, Jacob Verbeek, and Yann LeCun. Predicting deeper into the future of semantic segmentation. *ICCV*, 2017.

[16] Ishan Misra, C Lawrence Zitnick, and Martial Hebert. Shuffle and learn: unsupervised learning using temporal order verification. In *European Conference on Computer Vision*, pages 527–544. Springer, 2016.

[17] Aron Monszpart, Nils Thuerey, and Niloy Mitra. SMASH: Physics-guided Reconstruction of Collisions from Videos. *ACM Trans. on Graphics (TOG)*, 2016.

[18] Roozbeh Mottaghi, Hessam Bagherinezhad, Mohammad Rastegari, and Ali Farhadi. Newtonian scene understanding: Unfolding the dynamics of objects in static images. In *IEEE CVPR*, 2016.

[19] David Novotny, Samuel Albanie, Diane Larlus, and Andrea Vedaldi. Self-supervised learning of geometrically stable features through probabilistic introspection. 2018.

[20] R. Riochet, M. Ynocente Castro, M. Bernard, A. Lerer, R. Fergus, V. Izard, and E. Dupoux. IntPhys: A Framework and Benchmark for Visual Intuitive Physics Reasoning. *ArXiv e-prints*.

[21] Adam N Sanborn, Vikash K Mansinghka, and Thomas L Griffiths. Reconciling intuitive physics and newtonian mechanics for colliding objects. *Psychological review*, 120(2):411, 2013.

[22] Russell Stewart and Stefano Ermon. Label-free supervision of neural networks with physics and domain knowledge. In *AAAI*, pages 2576–2582, 2017.

[23] James Thewlis, Hakan Bilen, and Andrea Vedaldi. Unsupervised learning of object frames by dense equivariant image labelling. In *Advances in Neural Information Processing Systems (NIPS)*, pages 844–855, 2017.

[24] J. Tompson, K. Schlachter, P. Sprechmann, and K. Perlin. Accelerating Eulerian Fluid Simulation With Convolutional Networks. *ArXiv e-print arXiv:1607.03597*, 2016.

[25] N Watters, D Zoran, T Weber, P Battaglia, R Pascanu, and A 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 30*, pages 4542–4550. Curran Associates, Inc., 2017.

[26] Jiajun Wu, Ilker Yildirim, Joseph J Lim, Bill Freeman, and Josh Tenenbaum. Galileo: Perceiving physical object properties by integrating a physics engine with deep learning. In *Proc. NIPS*, pages 127–135, 2015.

[27] Jiajun Wu, Joseph J Lim, Hongyi Zhang, Joshua B Tenenbaum, and William T Freeman. Physics 101: Learning physical object properties from unlabeled videos. In *Proc. BMVC*, 2016.

[28] Jiajun Wu, Erika Lu, Pushmeet Kohli, Bill Freeman, and Josh Tenenbaum. Learning to see physics via visual de-animation. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems (NIPS) 30*, pages 153–164. Curran Associates, Inc., 2017.

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