# A Compositional Object-Based Approach to Learning Physical Dynamics

**Michael Chang**[*], **Tomer Ullman**[**], **Antonio Torralba**[*], and **Joshua B. Tenenbaum**[**]

[*]Department of Electrical Engineering and Computer Science, MIT
[**]Department of Brain and Cognitive Sciences, MIT
{mbchang,tomeru,torralba,jbt}@mit.edu

## Abstract

We present the Neural Physics Engine (NPE), an object-based neural network architecture for learning predictive models of intuitive physics. We propose a factorization of a physical scene into composable object-based representations and also the NPE architecture whose compositional structure factorizes object dynamics into pairwise interactions. Our approach draws on the strengths of both symbolic and neural approaches: like a symbolic physics engine, the NPE is endowed with generic notions of objects and their interactions, but as a neural network it can also be trained via stochastic gradient descent to adapt to specific object properties and dynamics of different worlds. We evaluate the efficacy of our approach on simple rigid body dynamics in two-dimensional worlds. By comparing to less structured architectures, we show that our model's compositional representation of the structure in physical interactions improves its ability to predict movement, generalize to different numbers of objects, and infer latent properties of objects such as mass.

## 1 Introduction

Physical reasoning is a crucial part of learning, perception, planning, inference, and understanding in artificial intelligence, and can be leveraged to accelerate the learning of new tasks (Lake et al., 2016). Accurately modeling a scene involves reasoning about the spatial properties, identities and locations of objects (Eslami et al., 2016; Hinton et al., 2011; Jaderberg et al., 2015; Kulkarni et al., 2015), but also their physical properties, future dynamics, and causal relationships. Such a sense of intuitive physics can be seen as a program (Anderson, 1990; Goodman and Tenenbaum, 2016) that takes input provided by a physical scene and the past states of objects, and outputs the future states and physical properties of relevant objects for a given task. This program should flexibly scale to model worlds with a variable number of objects, object properties, dynamics, interactions, and physical laws. It must be general enough to express various arrangements and behavior in known worlds, and also flexible enough to adapt to unknown worlds with new configurations.

At least two general approaches have emerged in the search for a program that captures common-sense physical reasoning. The top-down approach (Battaglia et al., 2013; Ullman et al., 2014; Wu et al., 2015) formulates the problem as inference over the parameters of a symbolic physics engine, while the bottom-up approach (Agrawal et al., 2016; Fragkiadaki et al., 2015; Lerer et al., 2016; Li et al., 2016; Mottaghi et al., 2015; 2016; Sutskever et al., 2009) learns to directly map physical observations to motion prediction or physical judgments. A program under the top-down approach can generalize across any scenario supported by the entities and operators in its description language. However, it may be brittle under scenarios not supported by its description language, and adapting to these new scenarios requires modifying the code or generating new code for the physics engine itself. In contrast, gradient-based bottom-up approaches can apply the same model architecture and learning algorithm to specific scenarios without requiring the physical dynamics of the scenario to be pre-specified. This often comes at the cost of reduced generality: transferring knowledge to new scenes may require extensive retraining, even in cases that seem trivial to human reasoning.

This paper takes a step toward bridging this gap between expressivity and adaptability, by proposing the Neural Physics Engine (NPE), a model that combines rough symbolic structure with gradient-

based learning for physical inference. This hybrid approach exhibits several strong inductive biases that are explicitly present in symbolic physics engines, such as a notion of objects-specific properties and object interactions. It is also end-to-end differentiable, and thus is able to flexibly tailor itself to the specific object properties and dynamics of a given world through training.

While previous bottom-up approaches (Sec. 4) have coupled learning vision and learning physical dynamics, we take a different approach, and for two reasons. First, we see that disentangling the visual properties of an object from its physical dynamics is a step toward achieving the generality of a physics engine; a program that learns to evolve objects through time can potentially be reused as a subprogram and composed with other programs under a modular framework for applications in areas such as model-based planning and model-based reinforcement learning. Second, we are optimistic that those two components indeed can be decoupled, that a vision model can map visual input to an intermediate state space, and a dynamics model can evolve objects in that state space through time. For example, there is much work in object detection and segmentation for extracting position and velocity, as well as work for extracting latent object properties (Wu et al., 2015). Therefore this paper focuses on learning dynamics in that state space, taking a small step toward emulating a general-purpose physics engine (Lake et al., 2016), with the eventual goal of building a system that exhibits the compositionality, modularity, and generality of physics engine whose internal components can be learned through observation.

This paper's contribution links two levels of factorization and composition in learning physical dynamics. On the level of the physical scene, we factorize the scene into object-based representations (Sec. 2.1), and compose smaller building blocks to form larger objects (Sec. 3.4). This framework of representation adapts to complex scenes and configurations with variable number of objects. On the level of the physics program, the NPE architecture (Sec. 2.2) explicitly reflects a causal structure in object interactions by factorizing object dynamics into pairwise interactions. As a predictive model of physical dynamics, the NPE models the future state of a single object as a function composition of the pairwise interactions between itself and other neighboring objects in the scene. This structure serves to guide learning towards object-based reasoning as (Hinton et al., 2011) does, and, by design, allows physical knowledge to transfer across variable number of objects and for object properties to be explicitly inferred. This approach – starting with a general sketch of a program and filling in the specifics – is similar to ideas presented by Solar-Lezama (2008); Tenenbaum et al. (2011). The NPE's general sketch is its architectural structure, and it extends and enriches this sketch to model the specifics of a particular scene by training on observed trajectories from that scene.

In Sec. 3 we investigate variations on two-dimensional worlds of balls and obstacles from the matter-js physics engine (Brummitt, `http://brm.io/matter-js/`) as a testbed for exploring the NPE's capabilities for modeling simple rigid body dynamics under our state space representation. While these worlds are generated from a simplified physics engine, we believe that learning to model such simple physics under the NPE's framework is a first and necessary step towards emulating the full capacity of a general physics engine, while maintaining a differentiability that can allow it to eventually learn complex real-world physical phenomena (see Sec. 4) that would be challenging to engineer into these physics engines. This paper establishes that important step.

## 2 APPROACH

### 2.1 STATE SPACE

We make two observations (Fig. 1) in our factorization of the scene. First, because physics does not change across inertial frames, it suffices to separately predict the future state of each object conditioned on the past states of itself and the other objects in its neighborhood, similar to Fragkiadaki et al. (2015). Second, because physics is Markovian, this prediction need only be for the immediate next timestep. Therefore, we choose an object-based state representation; a state vector comprises extrinsic properties (position, velocity, orientation, angular velocity), intrinsic properties (mass, object type, object size), and global properties (gravitational, frictional, and pairwise forces) at a given time instance. Sec. 3.4 describes composing objects to represent larger structures.
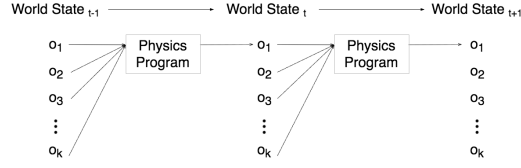
Figure 1: **Physics Programs** We consider the space of physics programs over object-based representations under physical laws that are Markovian and translation-invariant. We consider each object in turn and predict its future state conditioned on the past states of itself and other context objects.

## 2.2 NEURAL PHYSICS ENGINE

**Pairwise Factorization**  Letting a particular object be the *focus* object $f$ and all other objects in the scene be *context* objects $c$, the NPE models the focus object's velocity $v_{t+1}^{(f)}$ as a composition of the pairwise interactions between itself and other neighboring context objects in the scene during time $t-1$ and $t$. This input is represented as pairs of object state vectors $\{(o^{(f)}, o^{(c_1)})_{[t-1,t]}, (o^{(f)}, o^{(c_2)})_{[t-1,t]}, ...\}$. The NPE also predicts angular velocity along with velocity, but for the experiments in this paper we always set angular velocity, as well as gravity, friction, and pairwise forces, to zero. As shown in Fig. 2 (bottom-left), the NPE composes an encoder function and a decoder function. The encoder function $f_{enc}$ summarizes the interaction of a single object pair. The decoder function takes the sum of encodings of all pairs and the focus object's past state as input, and predicts the focus object's velocity $v_{t+1}^{(f)}$. In practice, the NPE predicts the change $\Delta v$ between $t$ and $t+1$ to compute $v_{t+1} = v_t + \Delta v$, and updates position using the velocity as a first-order approximation. We do not include acceleration in the state representation because position and velocity fully parametrize an object's state. Thus acceleration (e.g. collisions) can be learned by observing velocity for two consecutive timesteps, hence our choice for two input timesteps. We explored longer input durations as well and found no additional benefit.

**Neighborhood Mask**  Each $(o^{(f)}, o^{(c)})$ pair is selected to be in the set of neighbors of $f$ by the neighborhood masking function $\mathbb{1}\left[||p^{(c)} - p^{(f)})|| < o_n^{(f)}\right]$, which takes value 1 if the Euclidean distance between the positions $p^{(f)}$ and $p^{(c)}$ of the focus and context object respectively at time $t$ is less the threshold $o_n^{(f)}$. Many physics engines use a collision detection scheme with two phases. *Broad phase* is used for computational efficiency and use a neighborhood threshold to select objects that might, but not necessarily will, collide an object. The actual collision detection is done with *narrow phase* on that smaller subset of objects, which also resolves the collisions for the objects that do collide. Analogously, our neighborhood mask implements broad phase, and the NPE implements narrow phase. In this paper, we found that 3.5 times the focus object's radius worked well as $o_n^{(f)}$. Future work can view $o_n^{(f)}$ as a spatial attention mechanism that can be learned jointly with the other model parameters.

**Function Composition**  Symbolic physics engines evolve objects through time based on dynamics that dictate their independent behavior (e.g. friction, gravity, inertia) and their behavior with other objects (e.g. collision, support, attraction, repulsion). Notably, in a particular object's reference frame, the forces it feels from other objects are additive. The NPE architecture incorporates several inductive biases that reflect this recipe. $f_{enc}$ and $f_{dec}$ induce a causal structure on pairs of objects that can be composed when reasoning about how the object behaves. We provide a loose interpretation of the encoder output $e^{(f,c)}$ as the *effect* of object $c$ on object $f$, and require that these effects are additive as forces are. By design, this allows the NPE to scale naturally to different numbers of neighboring context objects. These inductive biases have the effect of strongly constraining the space of possible programs of predictive models that the NPE can learn, focusing on compositional programs that reflect pairwise causal structure in object interactions.
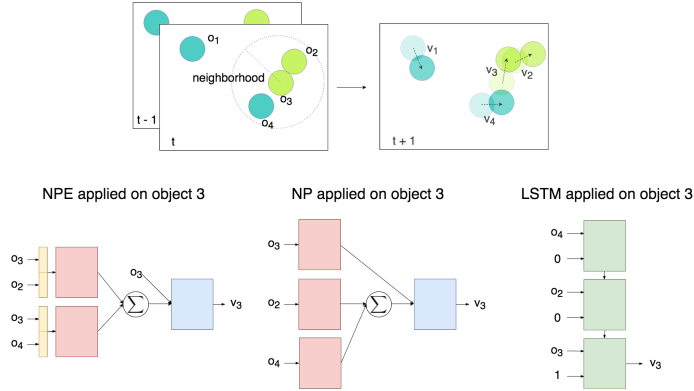
Figure 2: **Scenario and Models** This figure compares the NPE, the NP and the LSTM architectures in predicting the velocity of object 3 for an example scenario [**top**] of two heavy balls (cyan) and two light balls (yellow-green). Objects 2 and 4 are in object 3's neighborhood, so object 1 is ignored. [**bottom-left**]: The NPE encoder comprises a pairwise layer (yellow) and a feedforward network (red) and its decoder (blue) is also a feedforward network. [**bottom-middle**]: The NP encoder is the same as the NPE encoder, but without the pairwise layer. The NP decoder is the same as the NPE decoder. [**bottom-right**]: We shuffle the context objects inputted into the LSTM and use a binary flag to indicate whether an object is a context or focus object.

## 2.3 BASELINES

The purpose of contrasting the NPE with the following two baselines is to illustrate the benefit of pairwise factorization and function composition, which are the key architectural features of the NPE. As the architectures for both baselines have been shown to work well in similar tasks, it is not immediately clear whether the NPE's assumptions are useful or necessary, so these are good baselines to compare with. Viewed in another way, comparing with these baselines is a lesion study on the NPE because each baseline lacks an aspect of the NPE structure.

**No-Pairwise**    The No-Pairwise (NP) baseline is summarized by Fig. 2 (bottom-middle). The NP is very similar to the NPE but has the pairwise layer removed; otherwise the NP's encoder and decoder are the same as the NPE. Therefore the NP most directly highlights the value of the NPE's pairwise factorization. The NP is also a Markovian variant of the Social LSTM (Alahi et al., 2016); it sums the encodings of context objects after encoding each object independently, similar to the Social LSTM's "social pooling." Information for modeling how objects interact would only be present after the encoding step. Thus one interpretation for how such a structure could predict dynamics is if the encoder's object encoding comprises an abstract object representation and a force field created by that object. Therefore the NP decoder could apply the sum of the force fields of all context objects to the abstract object representation the focus object to predict the focus object's velocity. As Alahi et al. has demonstrated the Social LSTM's performance in modeling human trajectories, it is interesting to see how the same architectural assumptions performs for the physics of moving objects.

**LSTM**    Long Short-Term Memory (LSTM) networks (Hochreiter and Schmidhuber, 1997) have been shown to sequentially attend to objects (Eslami et al., 2016), so it is interesting to test whether a LSTM is well-suited for modeling object interactions, when the object states are explicitly given as input. From a cognitive science viewpoint, an LSTM can be interpreted as a serial mechanism in object tracking (Pylyshyn and Annan, 2006). Our LSTM architecture (Fig. 2, bottom-right) accepts the state of each context object until the last step, at which it takes in the focus object's state and predicts its velocity. Because the LSTM moves through the object space sequentially, its lack of factorized compositional structure highlights the value of the NPE's function composition of the independent interactions between an object and its neighbors. Unlike the NPE and NP, the LSTM's structure does not differentiate between focus and context object, so we add a flag to the state representation to indicate to whether an object is a context or focus object. We shuffle the order of the context objects to account for an ordering bias.

## 2.4 IMPLEMENTATION

We trained all models using the rmsprop Tieleman and Hinton (2012) backpropagation algorithm with a Euclidean loss for 1,200,000 iterations with a learning rate of 0.0003 and a learning rate decay of 0.99 every 2,500 training iterations, beginning at iteration 50,000. We used minibatches of size 50 and used a 70-15-15 split for training, validation, and test data.

All models are implemented using the neural network libraries built by Collobert et al.; Léonard et al.. The NPE encoder is consists of a pairwise layer of 25 hidden units and a 5-layer feedforward network of 50 hidden units per layer each with rectified linear activations. The encoding parameters are shared across all object pairs. The decoder is a five-layer network with 50 hidden units per layer and ReLU activations after all but the last layer. The NP encoder architecture is the same as the NPE encoder, but without the pairwise layer. The NP decoder architecture is the same as the NPE decoder. The LSTM has three layers of 100 hidden units and a linear layer after the last layer. It has rectified linear activations after each layer of the LSTM.

We informally explored several hyperparameters, varying the number of layers from 2 to 5, the hidden dimension from 50 to 100, and learning rates in $\{10^{-5}, 3 \times 10^{-5}, 10^{-4}, 3 \times 10^{-4}, 10^{-3}, 3 \times 10^{-3}\}$. Though this is far from an exhaustive search, we found that the above hyperparameter settings work well.

## 3 EXPERIMENTS

Using the matter-js physics engine, we evaluate the NPE on worlds of balls and obstacles. These worlds exhibit self-evident dynamics and support a wide set of scenarios that reflect everyday physics. Bouncing balls have also been of interest in cognitive science to study causality and counterfactual reasoning, as in (Gerstenberg et al., 2012). We trained on 3-timestep windows in trajectories of 60 timesteps. For a world of $k$ objects, we generate 50,000 such trajectories. For experiments where we train on multiple worlds together, we shuffle the examples across all training worlds and train without a curriculum schedule. Plots show results over three independent runs averaged over held-out test data with different random seeds. Randomly selected simulation videos are at `https://drive.google.com/drive/folders/0BxCJLi4FnT_6QW4tcF94dldoLWs?usp=sharing`.

### 3.1 PREDICTION

First we consider simple worlds of four balls of uniform mass (Fig. 3, left column). To measure performance in simulation, we visualize the cosine similarity between the predicted velocity and the ground truth velocity as well as the relative error in magnitude of the predicted velocity and the ground truth velocity, over 50 timesteps of simulation (about 5 seconds). The models take timesteps 1 and 2 as initial input, and then use previous predictions as input to future predictions. To measure progress through training, we also display the Euclidean loss on the velocity.

### 3.2 GENERALIZATION AND KNOWLEDGE TRANSFER

We test whether learned knowledge of these simple physics concepts can be transferred and extrapolated to worlds with a number of objects previously unseen. The unseen worlds (6, 7, 8 balls) in the test data are combinatorially more complex and varied than the observed worlds (3, 4, 5 balls) in the training data. All objects have equal mass. The NPE's predictions are more consistent, whereas the NP and LSTM's prediction begin to diverge wildly towards the end of 50 timesteps of simulation (Fig. 3, middle row). The NPE's performance on this extrapolation task suggests that its architectural inductive biases are useful for generalizing knowledge learned in Markovian domains with causal structure in object interactions. The next paragraph illustrates another aspect of the NPE's strong generalization capability.

### 3.3 MASS INFERENCE

We now show that the NPE can infer latent properties such as mass. This proposal is motivated by the experiments in Battaglia et al. (2013), which uses a probabilistic physics simulator to infer vari-
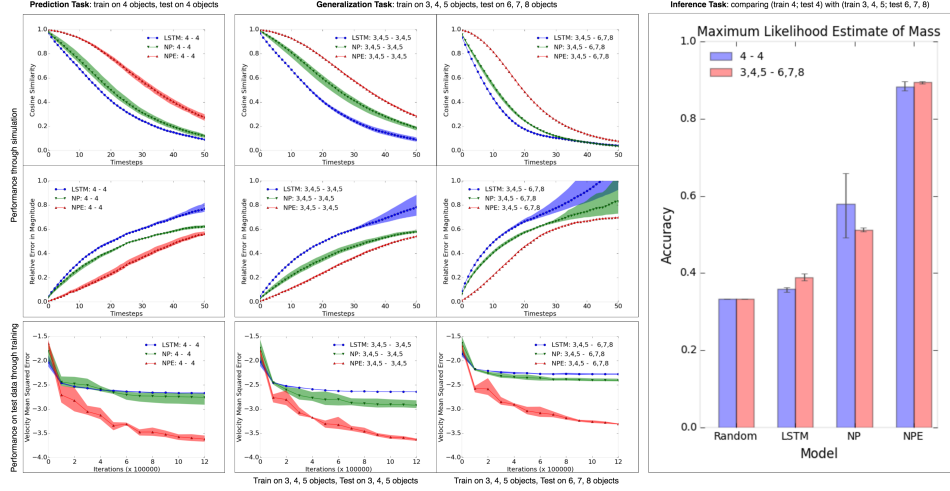
Figure 3: **Quantitative evaluation (balls).** [**Left and center panels**]: Prediction and generalization tasks. *Top two rows*: The cosine similarity and the relative error in magnitude. *Bottom row*: The $\log_{10}$ Euclidean loss of velocity on the test set over the course of training. Because these worlds are chaotic systems, it is not surprising that the models' predictions diverge from the ground truth with time, but NPE consistently outperforms the other two baselines on all fronts, especially when testing on 6, 7, and 8 objects in the generalization task. The NPE's performance continues to improve with training while the NP and the LSTM quickly plateau. We hypothesize that the NPE's structured factorization of the state space guides it from wasting time exploring suboptimal programs. [**Right panel**]: The NPE's accuracy is significantly greater than the baseline models' in mass inference. Notably, the NPE performs similarly well whether in a world it has seen before or in a world with a number of objects it hasn't trained on, further showcasing its strong generalization capabilities. The LSTM performs poorest, reaching just above random guessing (33% accuracy).

ous properties of a scene configuration. Whereas the physical rules of their simulator were manually pre-specified, the NPE learns these rules from observation. We train on the same worlds used in both the prediction and generalization tasks, but we uniformly sampled the mass for each ball from the log-spaced set $\{1, 5, 25\}$. We chose to use discrete-valued masses to simplify our qualitative understanding of the model's capacity to infer. For future work we would like to investigate continuously valued masses and evaluate with binary comparisons (e.g. "Which is heavier?").

As summarized by Fig. 3 (right panel) and Fig. 4 (left panel), we select scenarios exhibiting collisions with the focus object, fix the masses of all other objects, and score the NPE's prediction under all possible mass hypotheses for the focus object. The prediction is scored against the ground-truth under the same Euclidean loss used in training. The hypothesis whose prediction yields the lowest error is the NPE's maximum likelihood estimate of the focus object's mass. The NPE achieves about 90% accuracy, meaning it has 90% probability of inferring the correct mass.

We see that a simulator with a more structured model of the world performs more accurate inferences, and that this simulation capability can be learned from observation. Though we adopted a particular parametrization of an object, the NPE is not limited to the semantic meaning of the elements of its input, so we expect other latent object properties can be inferred this way. Because the NPE is differentiable, we expect that it can also infer object properties by backpropagating prediction error to its a randomly sampled input. This would be especially useful for inferring non-categorical values, such as positions of "invisible" objects, whose effects are felt but whose position is unknown.

## 3.4 WALLS AND OBSTACLES

We demonstrate representing large structures as a composition of smaller objects as building blocks in a world of balls and obstacles. These worlds contain 2 balls bouncing around in variations of 4 different wall geometries. "O" and "L" geometries have no internal obstacles and are in the shape
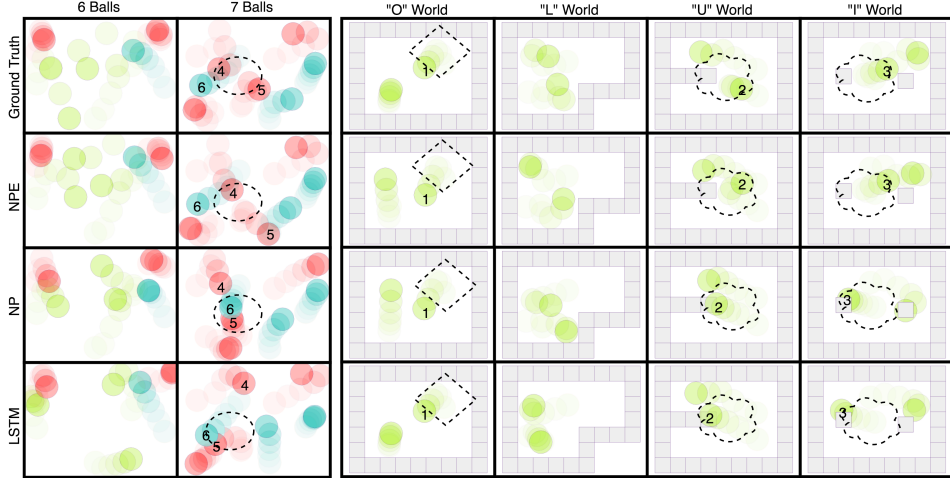
Figure 4: **Visualizations** The NPE scales to complex dynamics and world configurations while the NP and LSTM cannot. The masses are visualized as: cyan = 25, red = 5, yellow-green = 1. [**1,2,3**] At first glance, all models seem to handle collisions well in the "O" world (diamond), but when there are internal obstacles (cloud), only the NPE can successfully resolve collisions. This suggests that the NPE pairwise factorization handles object interactions well, letting it to different world configurations, whereas the NP and LSTM have only memorized the geometry of the "O" world. [**4,5**] Consider the collision in the 7 balls world (circle). In the ground truth, the collision happens between balls 4 and 5, and the NPE correctly predicts this. The NP predicts a slower movement for ball 4, so ball 5 overlaps with ball 6. The LSTM predicts a slower movement and incorrect angle off the world boundary, so ball 5 overlaps with ball 6.
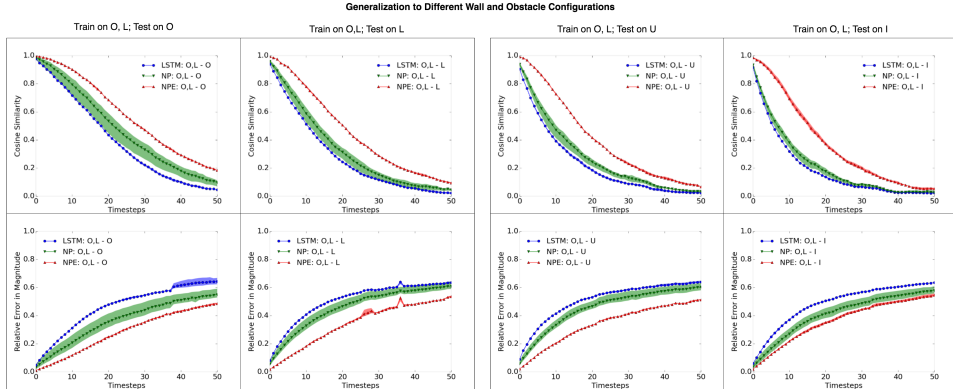


Figure 5: **Quantitative evelution (walls and obstacles)** By design, our compositional state representation simplifies the physical prediction problem to be over a focus object and differently arranged context balls and obstacles, even when the wall geometries are more complex and varied on a macroscopic scale. Therefore, it is not surprising that the models perform consistently across wall geometries. Note that the NPE consistently outperforms the other models, and this gap in performance increases with more varied internal obstacles for the cosine similarity of the velocity angle. This gap is more prominent in "L" and "U" geometries for relative error in magnitude.

of a rectangle and "L" respectively. "U" and "I" have internal obstacles. Obstacles in "U" are linearly attached to the wall like a protrusion, while obstacles in "I" have no constraint in position. We randomly vary the position and orientation of the "L" concavity and the "U" protrusion. We randomly sample the positions of the "I" internal obstacles.

We train on the conceptually simpler "O" and "L" worlds and test on the more complex "U" and "I" worlds. Variations in wall geometries adds to the difficulty of this extrapolation task. However,

our state space representation was designed to be flexible to this variation, by representing walls as composed of uniformly-sized obstacles, similarly to how many real-world objects are composed of smaller components. At most 12 context objects are present in the focus object's neighborhood at a time. The "U" geometries have 33 objects in the scene, the most out of all the wall geometries. As shown in Fig. 4 and 5, using such a compositional representation of the scene allows the NPE to scale to different configurations, which would not be straightforward to do without such a representation.

# 4 RELATION TO PREVIOUS WORK

A recent set of top-down approaches (Battaglia et al., 2013; Bates et al., 2015; Ullman et al., 2014) investigate probabilistic and deterministic (Wu et al., 2015) game physics engines as computational models for physical simulation in humans. However, inverting a physics engine, as the Intuitive Physics Engine (IPE) (Battaglia et al., 2013) does, requires a full specification of the physical laws and object geometries. Inferring how physical laws compose and apply to a given scenario is the IPE's strength, but the nature of these physical laws are pre-defined rather than learned from observation. To manually define, for example, the asymmetrical geometry of a hammer, the fluid motion of a tape dispenser dispensing tape, or the multiple stages of opening a door by its door handle, and in addition the geometry and physical behavior of every other object in a scene, requires extensive hand engineering that is difficult to automate.

Adapting to a scene with a differentiable model such as a neural network offers a possible path to automatically adapt a general architecture to the specific physical properties of a scene without prior human specification. Thus, we view the NPE as a differentiable model that complements and builds on top of the key structural assumptions in these top-down approaches.

Bottom-up approaches have mapped visual observations to physical judgments (Lerer et al., 2016; Li et al., 2016; Mottaghi et al., 2015; 2016) or passive (Lerer et al., 2016; Srivastava et al., 2015; Sutskever et al., 2009) and action-conditioned (Agrawal et al., 2016; Finn et al., 2016; Fragkiadaki et al., 2015) motion prediction. With the exception of works as (Wu et al., 2015), the bottom-up approaches mentioned above do not infer latent properties as our model does. Because the visual and physical are not disentangled, knowledge transfer to conceptually similar worlds, such as those with different numbers of objects, has been a challenge without retraining.

Our work brings a different perspective to recent work that have begun to address this problem. Lerer et al. (2016) used a shared network for both predicting future frames and for making physical judgments. Similarly, our work presents a single model that makes physical judgments beyond frame prediction. However, while their network used a specifically designed branch to predict tower stability, our network is less task-specific and infers mass using the same architecture used for prediction. Translation invariance is an explicit assumption in our model that allows our model to exhibit greater ability to extrapolate to larger numbers of objects.

This assumption is also core to Fragkiadaki et al. (2015), which conditions motion prediction on individual objects rather than the entire scene. However, a key contrast between their and our work is that the NPE's representations operate over a more abstract symbolic state space while theirs operate over visual attention windows. They argued that the combinatorial structure from a variable number of objects with variable properties and nonlinear behavior such as collisions creates a depth in complexity across which it is difficult to define a single state space. We presented a factorization in 2.1 that allows us to define such a state space.

Recently, in work we only learned about after implementing our approach, Battaglia et al. (2016) in parallel and independently developed a similar architecture that they call Interaction Networks for learning to model physical systems. They show how such architectures can apply to several different kind of physical systems, including n-body gravitational interactions and a string falling under gravity. Like their work, our model can simulate over many timesteps very effectively when only trained for next-timestep prediction, and can generalize to different world configurations and different numbers of objects.

Compared to Interaction Networks, a main difference in our architecture is that ours does not take object relations as explicit input, but instead learns the nature of these relations by constraining attention to a neighborhood set of objects. While constraining attention via a neighborhood threshold is more computationally efficient, it cannot capture longer range spatial dependencies such as gravity,

which acts from a remote distance. To address this, we propose for future work to learn an adaptive neighborhood threshold, such that the model would infer which context objects at which distances to pay attention to when computing its prediction. Another difference is that while they train an additional classifier on top of their model to do inference, the same model can be reused for both prediction and inference in our work. Our network produces output given inputs during prediction, but infers inputs given outputs during inference. We believe both their and our work bring valuable perspectives in learning to emulate general-purpose physics engines.

## 5 DISCUSSION

While this paper is not the first to explore predictive models of physics, here we take this opportunity to highlight the value of this paper's contributions. We hope these contributions can seed further research that builds on themes of factorization and composition this paper proposes.

We have presented a factorization of a physical scene into composable object-based representations and also a model architecture whose compositional structure factorizes object dynamics into pair-wise interactions. We applied the NPE to simple two-dimensional worlds ranging in complexity that demonstrate the usefulness of these two levels of factorization and composition. We show that NPE achieves low prediction error, extrapolates learned physical knowledge to previously unseen number of objects and world configurations, and can infer latent properties such as mass.

Further work includes generalizing beyond number of objects to new object types and physical laws such as in world with stacked block towers and liquids. Our results invite questions on how much prior information and structure should and could be given to bottom-up neural networks, and what can be learned without building in such structure. It will be interesting to see whether similar model assumptions work for both objects and agents. This paper works toward emulating a general purpose physics engine under a framework where visual and physical aspects of a scene are disentangled. Future work also includes linking the NPE with perceptual programs that extract properties such as position and mass from visual input. We believe the expressiveness of physics engines and the adaptability of neural networks are ingredients that will become increasingly important for modeling larger and more complex physical systems. The Neural Physics Engine proposes a possible path to link those two ingredients.

## REFERENCES

P. Agrawal, A. Nair, P. Abbeel, J. Malik, and S. Levine. Learning to poke by poking: Experiential learning of intuitive physics. *arXiv preprint arXiv:1606.07419*, 2016.

A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese. Social lstm: Human trajectory prediction in crowded spaces. 2016.

J. R. Anderson. *Cognitive psychology and its implications* . WH Freeman/Times Books/Henry Holt & Co, 1990.

C. J. Bates, I. Yildirim, J. B. Tenenbaum, and P. W. Battaglia. Humans predict liquid dynamics using probabilistic simulation. 2015.

P. Battaglia, R. Pascanu, M. Lai, D. Jimenez Rezende, and K. Koray. Interaction networks for learning about objects, relations and physics. In *Advances in Neural Information Processing Systems*, 2016.

P. W. Battaglia, J. B. Hamrick, and J. B. Tenenbaum. Simulation as an engine of physical scene understanding. *Proceedings of the National Academy of Sciences*, 110(45):18327–18332, 2013.

L. Brummitt. http://brm.io/matter-js. URL `http://brm.io/matter-js`.

R. Collobert, K. Kavukcuoglu, and C. Farabet. Torch7: A matlab-like environment for machine learning. In *BigLearn, NIPS Workshop*, number EPFL-CONF-192376, 2011.

S. Eslami, N. Heess, T. Weber, Y. Tassa, K. Kavukcuoglu, and G. E. Hinton. Attend, infer, repeat: Fast scene understanding with generative models. *arXiv preprint arXiv:1603.08575*, 2016.

C. Finn, I. Goodfellow, and S. Levine. Unsupervised learning for physical interaction through video prediction. *arXiv preprint arXiv:1605.07157*, 2016.

K. Fragkiadaki, P. Agrawal, S. Levine, and J. Malik. Learning visual predictive models of physics for playing billiards. *arXiv preprint arXiv:1511.07404*, 2015.

T. Gerstenberg, N. Goodman, D. A. Lagnado, and J. B. Tenenbaum. Noisy newtons: Unifying process and dependency accounts of causal attribution. In *In proceedings of the 34th*. Citeseer, 2012.

N. D. Goodman and J. B. Tenenbaum. Probabilistic models of cognition, 2016. URL `http://probmods.org`.

G. E. Hinton, A. Krizhevsky, and S. D. Wang. Transforming auto-encoders. In *Artificial Neural Networks and Machine Learning–ICANN 2011*, pages 44–51. Springer, 2011.

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

M. Jaderberg, K. Simonyan, A. Zisserman, et al. Spatial transformer networks. In *Advances in Neural Information Processing Systems*, pages 2008–2016, 2015.

T. D. Kulkarni, P. Kohli, J. B. Tenenbaum, and V. Mansinghka. Picture: A probabilistic programming language for scene perception. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4390–4399, 2015.

B. M. Lake, T. D. Ullman, J. B. Tenenbaum, and S. J. Gershman. Building machines that learn and think like people. *arXiv preprint arXiv:1604.00289*, 2016.

N. Léonard, S. Waghmare, and Y. Wang. rnn: Recurrent library for torch. *arXiv preprint arXiv:1511.07889*, 2015.

A. Lerer, S. Gross, R. Fergus, and J. Malik. Learning physical intuition of block towers by example. *arXiv preprint arXiv:1603.01312*, 2016.

W. Li, S. Azimi, A. Leonardis, and M. Fritz. To fall or not to fall: A visual approach to physical stability prediction. *arXiv preprint arXiv:1604.00066*, 2016.

R. Mottaghi, H. Bagherinezhad, M. Rastegari, and A. Farhadi. Newtonian image understanding: Unfolding the dynamics of objects in static images. *arXiv preprint arXiv:1511.04048*, 2015.

R. Mottaghi, M. Rastegari, A. Gupta, and A. Farhadi. " what happens if..." learning to predict the effect of forces in images. *arXiv preprint arXiv:1603.05600*, 2016.

Z. W. Pylyshyn and V. Annan. Dynamics of target selection in multiple object tracking (mot). *Spatial vision*, 19(6):485–504, 2006.

A. Solar-Lezama. *Program synthesis by sketching*. ProQuest, 2008.

N. Srivastava, E. Mansimov, and R. Salakhutdinov. Unsupervised learning of video representations using lstms. 2015.

I. Sutskever, G. E. Hinton, and G. W. Taylor. The recurrent temporal restricted boltzmann machine. In *Advances in Neural Information Processing Systems*, pages 1601–1608, 2009.

J. B. Tenenbaum, C. Kemp, T. L. Griffiths, and N. D. Goodman. How to grow a mind: Statistics, structure, and abstraction. *science*, 331(6022):1279–1285, 2011.

T. Tieleman and G. Hinton. Lecture 6.5—RmsProp: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning, 2012.

T. Ullman, A. Stuhlmüller, and N. Goodman. Learning physics from dynamical scenes. 2014.

J. Wu, I. Yildirim, J. J. Lim, B. Freeman, and J. Tenenbaum. Galileo: Perceiving physical object properties by integrating a physics engine with deep learning. In *Advances in Neural Information Processing Systems*, pages 127–135, 2015.