Physics 101: Learning Physical Object Properties from Unlabeled Videos

Jiajun Wu[1]
http://jiajunwu.com
Joseph J. Lim[2]
http://people.csail.mit.edu/lim/
Hongyi Zhang[1]
http://web.mit.edu/~hongyiz/www/
Joshua B. Tenenbaum[1]
http://web.mit.edu/cocosci/josh.html
William T. Freeman[13]
http://billf.mit.edu

[1] CSAIL
Massachusetts Institute of Technology
MA, USA
[2] Department of Computer Science
Stanford University
CA, USA
[3] Google Research
MA, USA

Abstract

We study the problem of learning physical properties of objects from unlabeled videos. Humans can learn basic physical laws when they are very young, which suggests that such tasks may be important goals for computational vision systems. We consider various scenarios: objects sliding down an inclined surface and colliding; objects attached to a spring; objects falling onto various surfaces, *etc*. Many physical properties like mass, density, and coefficient of restitution influence the outcome of these scenarios, and our goal is to recover them automatically. We have collected 17,408 video clips containing 101 objects of various materials and appearances (shapes, colors, and sizes). Together, they form a dataset, named Physics 101, for studying object-centered physical properties. We propose an unsupervised representation learning model, which explicitly encodes basic physical laws into the structure and use them, with automatically discovered observations from videos, as supervision. Experiments demonstrate that our model can learn physical properties of objects from video. We also illustrate how its generative nature enables solving other tasks such as outcome prediction.

1 Introduction

How can a vision system acquire common sense knowledge from visual input? This fundamental question has been of interest to researchers in both the human vision and computer vision communities for decades. One basic component of common sense knowledge is to understand the physics of the world. There is evidence that babies form a visual understanding of basic physical concepts at a very young age; they learn properties of objects from their motions [2]. As young as 2.5 to 5.5 months old, infants learn basic physics even before they acquire advanced high-level knowledge like semantic categories of objects [2, 7]. Both infants and adults also use their physics knowledge to learn and discover latent labels of object properties, as well as predict the physical behavior of objects [3]. These facts suggest the

Pages 39.1-39.12
DOI: https://dx.doi.org/10.5244/C.30.39

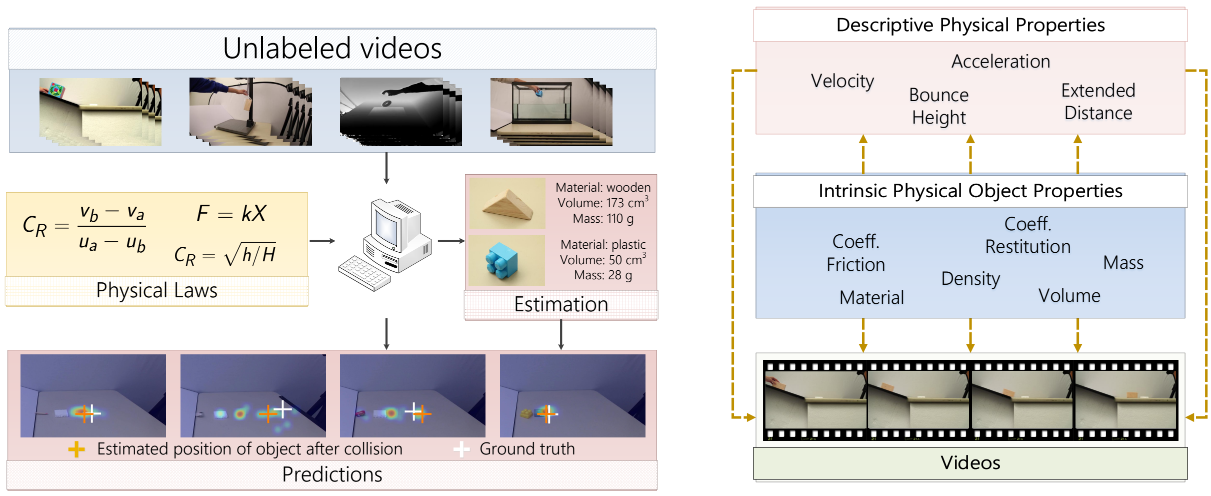


Figure 1: Left: Overview of our model, which learns directly from unlabeled videos, produces estimates of physical properties of objects based on the encoded physical laws, and generalizes to tasks like outcome prediction. Right: An abstraction of the physical world. See discussion in Section 2.

importance for a visual system of understanding physics, and motivate our goal of building a machine with such competency.

There have been several early efforts to build computational vision systems with the physical knowledge of an early child [6], or specific physical properties, such as material, from the perspectives of human and machine perception [1, 17]. Recently, researchers started to tackle concrete scenarios for understanding physics from vision [11, 15, 18, 19, 23, 24, 25, 27], some building on recent advances in deep learning. Different from those approaches, in this work we aim to develop a system that can infer physical properties, *e.g.* mass and density, directly from visual input. Our method is general and easily adaptive to new scenarios, and is more efficient compared to analysis-by-synthesis approaches [23].

Our goal is to discover, learn, and infer physical object properties by combining physics knowledge with machine learning methods. Our system combines a low-level visual recognition system with a physics interpreter and a physical world simulator to estimate physical properties directly from unlabeled videos. We integrate physical laws, which physicists have discovered over many centuries, into representation learning methods, which have shown usefulness in computer vision in recent years. Our incorporation of physical laws into our model distinguish our framework from previous methods introduced in the computer vision and robotics community for predicting physical interactions or properties of objects for various purposes [4, 5, 9, 12, 16, 21, 22, 26].

We present a formulation to learn physical object properties with deep models. Our model is generative and contains components for learning both latent object properties and upper-level physical arguments. Not only can this model discover object properties, but it can also predict the physical behavior of objects. Ultimately, our system can discover physical properties of objects without explicit labels, simply by observing videos with the supervision of a physics interpreter. To train and test our system, we collected a dataset of 101 objects made of different materials and with a variety of masses and volumes. We started by collecting videos of these objects from multiple viewpoints in four various scenarios: objects slide down an inclined surface and possibly collide with another object; objects fall onto surfaces made of different materials; objects splash in water; and objects hang on a spring. These seemingly straightforward setups require understanding multiple physical properties, *e.g.*, material, mass, volume, density, coefficient of friction, and coefficient of restitution, as discussed in Section 3.1. We called this dataset Physics 101, highlighting that we are learning elementary physics, while also indicating the current object count.

Through experiments, we demonstrate that our framework develops some physics competency by observing videos. In Section 5, we also show that our generative model can transfer learned physical knowledge from one scenario to the other, and generalize to other tasks like predicting the outcome of a collision. Some dynamic scenarios we studied, *e.g.* objects sliding down on an inclined surface and possibly hitting other objects, are similar to classic experiments conducted by Galileo Galilei hundreds of years ago [10], and to experimental conditions in modern developmental psychology [2].

Our contributions are three-fold: first, we propose novel scenarios for learning physical properties of objects, which facilitates comparisons with studies of infant vision; second, we build a dataset of videos with objects of various materials and appearances, specifically to study the physical properties of these objects; third, we propose a novel unsupervised deep learning framework for these tasks and conduct experiments to prove its effectiveness and generalization ability.

2 Physical World

There exist highly involved physical processes in daily events in our physical world, even simple scenarios like objects sliding down an inclined surface. As shown in Figure 1, we can divide all involved physical properties into two groups: the first is the intrinsic physical properties of objects like volume, material, and mass, many of which we cannot directly measure from the visual input; the second is the descriptive physical properties which characterize the scenario in the video, including but not limited to velocity of objects, distances that objects traveled, or whether objects float if they are thrown into water. The second group of parameters are observable, and are determined by the first group, while both of them determine the content in videos.

Our goal is to build an architecture that can automatically discover those observable descriptive physical properties from unlabeled videos, and use them as supervision to further learn and infer unobservable latent physical properties. Our generative model can then apply learned knowledge of physical object properties for other tasks like predicting outcomes in the future. To further develop this, in Section 3 we will introduce our novel dataset for learning physical object properties. We then illustrate our model in Section 4.

3 Dataset - Physics 101

The AI and vision community has made much progress through its datasets, and there are datasets of objects, attributes, materials, and scene categories. Here, we introduce a new type of dataset – one that captures physical interactions of objects. The dataset consists of four different scenarios, for each of which plenty of intriguing questions may be asked. For example, in the ramp scenario, will the object on the ramp move, and if so and two objects collide, which of them will move next and how far?

3.1 Scenarios

We seek to learn physical properties of objects by observing videos. To this end, we build a dataset by recording videos of moving objects. We pick an introductory setup with four different scenarios, which are illustrated in Figures 3a and 4. We then introduce each scenario in detail.

Ramp We put an object on an inclined surface, and the object may either slide down or keep static, due to gravity and friction. As studied earlier in [23], this seemingly straightforward

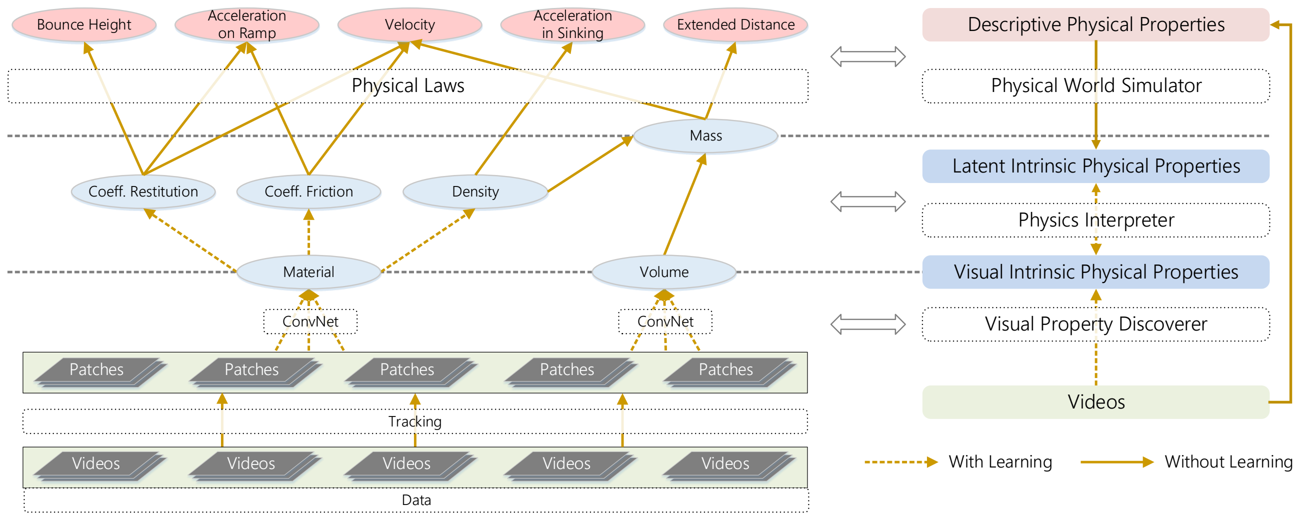
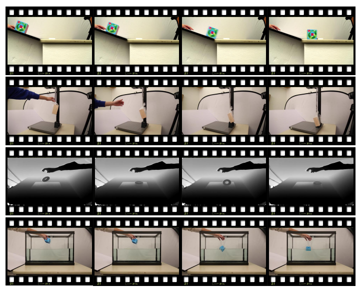
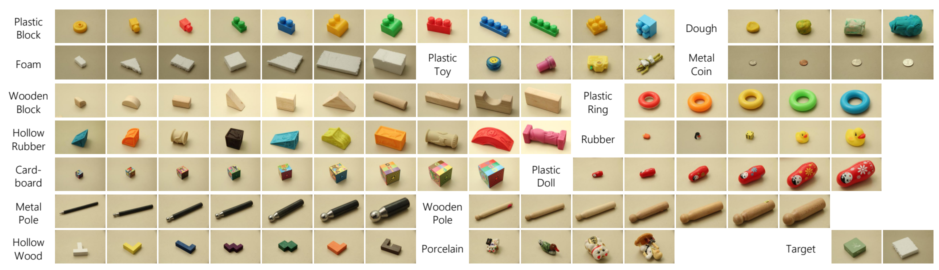


Figure 2: Our model exploits the advancement of deep learning algorithms and discovers various physical properties of objects. We supervise all levels by automatically discovered observations from videos. These observations are provided to our physical world simulator, which transform them to constraints on physical properties. During the training and testing, our model has no labels of physical properties, in contrast to the standard supervised learning approaches.



(a) Our scenario and a snapshot of our dataset, Physics 101, of various objects at different stages. Our data are taken by four sensors (3 RGB and 1 depth).



(b) Physics 101: this is the set of objects we used in our experiments. We vary object material, color, shape, and size, together with external conditions such as the slope of a surface or the stiffness of a string. Videos recording the motions of these objects interacting with target objects will be used to train our algorithm.

Figure 3: The objects and videos in Physics 101.

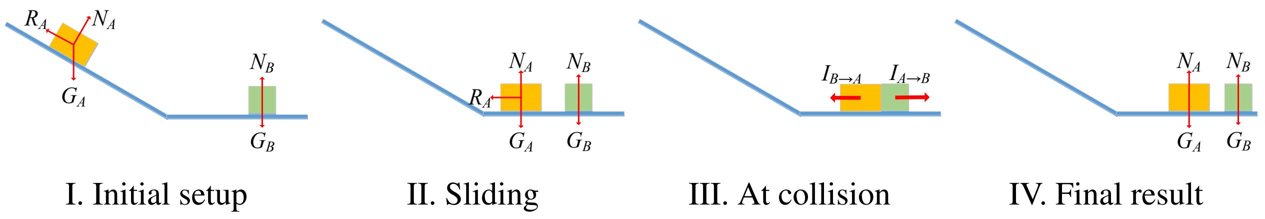
scenario already involves understanding many physical object properties including material, coefficient of friction, mass, and velocity. Figure 4a analyzes the physics behind our setup.

In this scenario, the observable descriptive physical properties are the velocities of the objects, and the distances both objects traveled. The latent properties directly involved are coefficient of friction and mass.

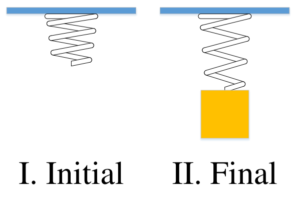
Spring We hang objects on a spring, and gravity on the object will stretch the spring. Here the observable descriptive physical property is length that the spring gets stretched, and the latent properties are the mass of the object and the elasticity of the spring.

Fall We drop objects in the air, and they freely fall onto various surfaces. Here the observable descriptive physical properties are the the bounce heights of the object, and the latent properties are the coefficient of restitution of the object and the surface.

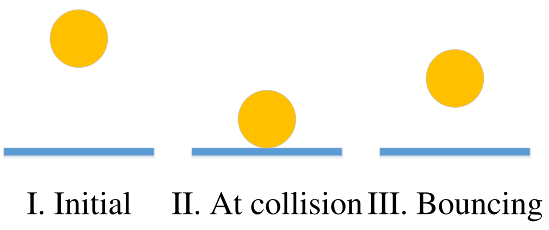
Liquid We drop objects into some liquid, and they may float or sink at various speeds. In this scenario, the observable descriptive physical property is the velocity of the sinking object



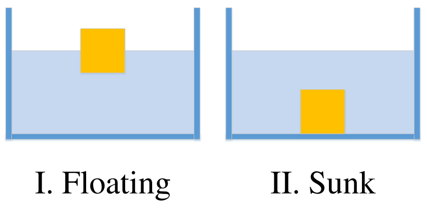
(a) The ramp scenario. Several physical properties will determine if object A will move, if it will reach to object B, and how far each object will move. Here, N, R, and G indicate a normal force, a friction force, and a gravity force, respectively.



(b) The spring scenario



(c) The fall scenario



(d) The liquid scenario

Figure 4: Illustrations of the scenarios in Physics 101

(0 if it floats), and the latent properties are the densities of the object and the liquid.

3.2 Building Physics 101

The outcomes of various physical events depend on multiple factors of objects, such as materials (density and friction coefficient), sizes and shapes (volume), and slopes of ramps (gravity), elasticities of springs, *etc*. We collect our dataset while varying all these conditions. Figure 3b shows the entire collection of our 101 objects, and the following are more details about our variations:

Material Our 101 objects are made of 15 different materials: cardboard, dough, foam, hollow rubber, hollow wood, metal coin, pole, and block, plastic doll, ring, and toy, porcelain, rubber, wooden block, and wooden pole.

Appearance For each material, we have 4 to 12 objects of different sizes and colors.

Slope (ramp) We also vary the angle α between the inclined surface and the ground (to vary the gravity force). We set $\alpha = 10^o$ and 20^o for each object.

Target (ramp) We have two different target objects – a cardboard and a foam box. They are made of different materials, thus having different friction coefficients and densities.

Spring We use two springs with different stiffness.

Surface (fall) We drop objects onto five different surfaces: foam, glass, metal, wooden table, and woolen rug. These materials have different coefficients of restitution.

We also measure the physical properties of these objects. We record the mass and volume of each object, which also determine density. For each setup, we record their actions for $3 \sim 10$ trials. We measure multiple times because some external factors, *e.g.*, orientations of objects and rough planes, may lead to different outcomes. Having more than one trial per condition increases the diversity of our dataset by making it cover more possible outcomes.

Finally, we record each trial from three different viewpoints: side, top-down, and upper-top. For the first two, we take data with DSLR cameras, and for the upper-top view, we use a Kinect V2 to record both RGB and depth maps. We have $4,352$ trials in total. Given we captured videos in three RGB maps and one depth map, there are $17,408$ video clips altogether. These video clips constitute the Physics 101 dataset.

4 Method

In this section, we describe our approach that interprets the basic physical properties and infers the interaction of objects with the world. Rather than training each classifier/regressor with labels, *e.g.* material or volume, with full supervision, we are interested in discovering the properties under a unified system with minimal supervision. Our goals are two-fold. First, we want to be able to discover all involved physical properties like material and volume simply by observing motions of objects in unlabeled videos during training. Second, we also want to infer learned properties and also predict different physical interactions, *e.g.* how far will the object move if it moves at all; or whether the object will float if dropped into water. The setup is shown in Figure 3a.

Our model is shown in Figure 2. Our method is based on a convolutional neural network (CNN) [13], which consists of three components. The bottom component is a *visual property discoverer*, which aims to discover physical properties like material or volume which could at least partially be observed from visual input; the middle component is a *physics interpreter*, which explicitly encodes physical laws into the network structure and models latent physical properties like density and mass; the top component is a *physical world simulator*, which characterizes descriptive physical properties like distances that objects traveled, all of which we may directly observe from videos.

Our network corresponds to our physical world model introduced in Section 2. We would like to emphasize here that our model learns object properties from completely unlabeled data. We do not provide any labels for physical properties like material, velocity, or volume; instead, our model automatically discovers observations from videos, uses them as supervision to the top physical world simulator, which in turn advises what the physics interpreter should discover.

4.1 Visual Property Discoverer

The bottom meta-layer of our architecture in Figure 2 is designed to discover and predict low-level properties of objects including material and volume, which can also be at least partially perceived from the visual input. These properties are the basic parts of predicting any derived physical properties at upper layers, *e.g.* density and mass.

In order to interpret any physical interaction of objects, we need to be able to first locate objects inside videos. We use a KLT point tracker [20] to track moving objects . We also compute a general background model for each scenario to locate foreground objects. Image patches of objects are then supplied to our visual property discoverer.

Material and volume Material and volume are properties that can be estimated directly from image patches. Hence, we have LeNet [14] on top of image patches extracted by the tracker. Once again, rather than directly supervising each LeNet with their labels, we supervise them by automatically discovered observations which are provided to our physical world simulator. To be precise, we do not have any individual loss layer for LeNet components. Note that inferring volumes of objects from static images is an ambiguous problem. However, this problem is alleviated given our multi-view and RGB-D data.

4.2 Physics Interpreter

The second meta-layer of our model is designed to encode the physical laws. For instance, if we assume an object is homogeneous, then its density is determined by its material; the mass of an object should be the multiplication of its density and volume. Based on material and volume, we expand a number of physical properties in this physics interpreter, which will later be used to connect to real world observations. The following shows how we represent each physical property as a layer depicted in Figure 2:

Material A N_m dimensional vector, where N_m is the number of different materials. The value of each dimension represents the confidence that the object belongs to that dimension. This is an output of our visual property discoverer.

Volume A scalar representing the predicted volume of the object. This is an output of our visual property discoverer.

Coefficient of friction and density Each is a scalar representing the predicted physical property based on the output of the material layer. Each output is the inner product of N_m learned parameters and responses from the material layer.

Coefficient of restitution A N_m dimensional vector representing how much of the kinetic energy remains after a collision between the input object with other objects of various materials. The representation is a vector, not a scalar, as the coefficient of restitution is determined by the materials of both objects involved in the collision.

Mass A scalar representing the predicted mass based on the outputs of the density layer and the volume layer. This layer is the product of the density and volume layers.

4.3 Physical World Simulator

Our physical world simulator connects the inferred physical properties to real-world observations. We have different observations for these scenarios. As explained in Section 3, we use velocities of objects and distances objects traveled as observations of the ramp scenario, the length that the string is stretched as an observation of the spring scenario, the bounce distance as an observation of the fall scenario, and the velocity that object sinks as an observation of the liquid scenario. All observations can be derived from the output of our tracker.

To connect those observations to physical properties our model inferred, we employ physical laws. The physical laws we used in our model include

Newton's law $F = mg\sin\theta - \mu mg\cos\theta = ma$, or $(\sin\theta - \mu\cos\theta)g = a$, where θ is angle between the inclined surface and the ground, μ is the friction coefficient, and a is the acceleration (observation). This is used for the ramp case.

Conservation of momentum and energy $C_R = (v_b - v_a)/(u_a - u_b)$, where v_i is the velocity of object i after collision, and u_i is its velocity before collision. All u_i and v_i are observations, and this is also used for the ramp scenario.

Hooke's law $F = kX$, where X is the distance that the string is extended (our observation), k is the stiffness of the string, and $F = G = mg$ is the gravity on the object. This is used for the spring scenario.

Bounce $C_R = \sqrt{h/H}$, where C_R is the coefficient of restitution, h is the bounce height (observation), and H is the drop height. This can be seen as another representation of energy and momentum conservation, and is used for the fall scenario.

Buoyancy $dVg - d_wVg = ma = dVa$, or $(d - d_w)g = da$, where d is density of the object, d_w is the density of water (constant), and a is the acceleration of the object in water (observation). This is used for the liquid scenario.

5 Experimental Results

In this section, we present experiments in various settings. We start with verifications of our model on learning physical properties. Later, we investigate its generalization ability on tasks like predicting outcomes given partial information, and transferring knowledge across different scenarios.

5.1 Setup

For learning physical properties from Physics 101, we study our algorithm in the following settings:

- Split by frame: for each trial of each object, we use 95% of the patches we get from tracking as training data, while the other 5% of the patches as test data.
- Split by trial: for each trial of each object, we use all patches in 95% of the trials we have as training data, while patches in the other 5% of the trials as test data.
- Split by object: we randomly choose 95% of the objects. We use them as training data and the others as test data.

Among these three settings, split by frame is the easiest as for each patch in test data, the algorithm may find some very similar patch in the training data. Split by object is the most difficult setting as it requires the model to generalize to objects that it has never seen before.

We consider training our model in different ways:

- Oracle training: we train our model with images of objects and their associated ground truth labels. We apply oracle training on those properties we have ground truths labels of (material, mass, density, and volume).
- Standalone training: we train our model on data from one scenario. Automatically extracted observations serve as supervision.
- Joint training: we jointly train the entire network on all training data without any labels of physical properties. Our only supervision is the physical laws encoded in the top physical world simulator. Data from different scenarios supervise different layers in the network.

Oracle training is designed to test the ability of each component and serves as an upper bound of what the model may achieve. Our focus is on standalone and joint training, where our model learns from unlabeled videos directly.

We are also interested in understanding how our model performs at inferring some physical properties purely from depth maps. Therefore, we conduct some experiments where training and test data are depth maps only.

For all experiments, we use Torch [8] for implementation, MSE as our loss function, and SGD for optimization. We use a batch size of 32, and set the learning rate to 0.01.

5.2 Learning Physical Properties

Material perception: We start with the task of material classification. Table 1a shows the accuracy of the oracle models on material classification. We observe that they can achieve nearly perfect results in the easiest case, and is still significantly better than chance on the most difficult split-by-object setting. Both depth maps and RGB maps give good performance on this task with oracle training.

In the standalone and joint training case, given we have no labels on materials, it is not possible for the model to classify materials; instead, we expect it to cluster objects by their

Methods	Frame	Trial	Object
RGB (oracle)	99.9	77.4	52.2
Depth (oracle)	92.6	62.5	35.7
RGB (ramp)	26.9	24.7	19.7
RGB (spring)	29.9	22.4	14.3
RGB (fall)	29.4	25.0	17.0
RGB (liquid)	22.2	15.4	12.6
RGB (joint)	35.5	28.7	25.7
Depth (joint)	38.3	26.9	22.4
Uniform	6.67	6.67	6.67

(a)

Methods	Mass			Density			Volume		
	Frame	Trial	Object	Frame	Trial	Object	Frame	Trial	Object
RGB (oracle)	0.79	0.72	0.67	0.83	0.74	0.65	0.77	0.67	0.61
Depth (oracle)	0.67	0.51	0.33	0.69	0.61	0.49	0.40	0.41	0.33
RGB (spring)	0.40	0.35	0.20	N/A	N/A	N/A	N/A	N/A	N/A
RGB (liquid)	N/A	N/A	N/A	0.33	0.27	0.30	N/A	N/A	N/A
RGB (joint)	0.52	0.32	0.26	0.33	0.34	0.33	0.40	0.37	0.30
Depth (joint)	0.33	0.27	0.25	0.29	0.23	0.17	0.30	0.20	0.22
Uniform	0	0	0	0	0	0	0	0	0

(b)

Table 1: (a) Accuracies (%, for oracle) or clustering purities (%, for joint training) on material estimation. In the joint training case, as there is no supervision on the material layer, it is not necessary for the network to specifically map the responses in that layer to material labels, and we do not expect the numbers to be comparable with the oracle case. Our analysis is just to show even in this case the network implicitly grasps some knowledge of object materials. (b) Correlation coefficients of our estimations and ground truth for mass, density, and volume.

materials. To measure this, we perform K-means on the responses of the material layer of test data, and use *purity*, a common measure for clustering, to measure if our model indeed discovers clusters of materials automatically. As shown in Table 1a, the clustering results indicate that the system learns the material of objects to a certain extent.

Physical parameter estimation: We then test our systems, trained with or without oracles, on the task of physical property estimation. We use Pearson product-moment correlation coefficient as measures. Table 1b shows the results on estimating mass, density, and volume. Notice that here we evaluate the outputs on a log scale to avoid unbalanced emphases on objects with large volumes or masses.

We observe that with oracle our model can learn all physical parameters well. For standalone and joint learning, our model also consistently achieves positive correlation. The uniform estimate is derived by calculating the average of ground truth results, *e.g.*, how far objects indeed travel. As we calculate the correlation between predictions and ground truth, always predicting a constant value (the optimum uniform estimate) gives a correlation of 0.

5.3 Predicting Outcomes

We may apply our model to a variety of outcome prediction tasks for different scenarios. We consider three of them: how far would an object move after being hit by another object; how high an object will bounce after being dropped at a certain height; and whether an object will float in the water. With estimated physical object properties, our model can answer these questions using physical laws.

Transferring Knowledge Across Multiple Scenarios As some physical knowledge is shared across multiple scenarios, it is natural to evaluate how learned knowledge from one scenario may be applied to a novel one. Here we consider the case where the model is trained on all but the fall scenarios. We then apply the model to the fall scenario for predicting how high an object bounces. Our intuition is the learned coefficients of restitution from the ramp scenario can help to predict to some extent.

Results Table 2a shows outcome prediction results. We can see that our method works well, and can also transfer learned knowledge across multiple scenarios. We also compare our method with a CNN (LeNet) trained to directly predict outcomes without physical knowledge.

Tasks	Methods	Frame	Trial	Object
Moving Distance	Ours (joint)	0.65	0.42	0.33
Moving Distance	CNN	0.63	0.39	0.21
Moving Distance	Uniform	0	0	0
Bounce Height	Ours (joint)	0.35	0.31	0.23
Bounce Height	Ours (transfer)	0.32	0.27	0.16
Bounce Height	CNN	0.36	0.25	0.15
Bounce Height	Uniform	0	0	0
Float	Ours (joint)	0.94	0.87	0.84
Float	CNN	0.96	0.86	0.81
Float	Uniform	0.70	0.70	0.70

(a) Correlation coefficients on predicting the moving distance and the bounce height, and accuracies on predicting whether an object floats

Material	Cardboard			Foam		
	H	M	B	H	M	B
cardboard	28.8	40.7	97.0	15.0	77.2	84.0
dough	27.4	25.2	84.4	150.9	105.1	113.4
h_wood	35.7	19.4	108.9	81.0	35.0	21.4
m_coin	13.4	32.2	149.8	31.9	33.3	75.8
m_pole	272.9	257.6	280.0	91.4	188.7	184.0
p_block	29.8	82.1	97.6	46.9	57.2	35.0
p_doll	49.4	23.6	44.0	128.8	41.8	93.9
p_toy	30.1	41.9	121.2	33.3	9.5	70.6
porcelain	138.5	127.0	110.9	196.0	216.6	314.8
w_block	45.9	32.8	36.2	47.3	37.5	14.2
w_pole	78.9	88.0	138.9	58.7	89.8	74.3
Mean	68.2	70.1	115.4	80.1	81.1	98.3

(b) Mean squared errors in pixels of human predictions (H), model outputs (M), or a baseline (B) which always predicts the mean of all distances

Table 2: Results for various outcome prediction tasks

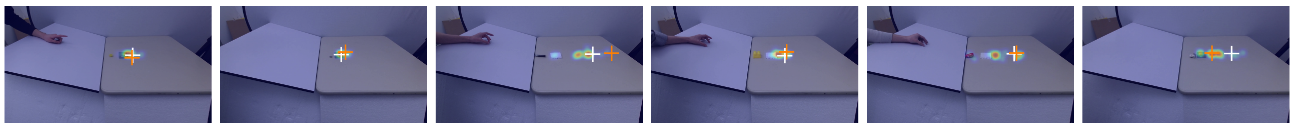


Figure 5: Heat maps of user predictions, model outputs (in orange), and ground truths (in white). Objects from left to right: dough, plastic block, and plastic doll.

Note that even our transfer model is better in challenging cases (split by trial/object). Further, our framework can not only predict descriptive but also intrinsic physical properties like densities, which could be more useful in other domains like robotics.

Behavior Experiments We would like to see how well our model does compared to a human. We thus conducted experiments on predicting the moving distance of an object after collision on Amazon Mechanical Turk. Specifically, among all objects that slide down, we select one object of each material, show AMT workers the videos of the object, but only to the moment of collision. We then ask workers to label where they believe the target object (either cardboard or foam) will be after the collision, *i.e.*, how far the target will move. Before testing, each users are provided four full videos of other objects made of the same material, which contain complete collisions, so that users can simply infer the physical properties associated with the material and the target object in their mind. We tested 30 users per case.

Table 2b shows the mean squared errors in pixels of human predictions (H), model predictions (M), or a baseline which always predicts the mean of all distances (B). We can see that the performance of our model is close to that of human on this task. Figure 5 shows the heat maps of user predictions, model outputs (orange), and ground truths (white).

6 Conclusion

In this paper, we discussed the possibility of learning physical object properties from videos. We proposed a novel dataset, Physics 101, which contains 17,408 videos of 101 objects in four scenarios, with detailed physical properties of these objects. We also proposed a novel model for learning physical properties of objects by encoding physical laws into deep neural nets. Experiments demonstrated that our method works well on Physics 101. We hope our paper could inspire future study on learning physical and other real-world knowledge from visual data.

Acknowledgements This work is in part supported by NSF Robust Intelligence 1212849 Reconstructive Recognition and the Center for Brain, Minds and Machines (NSF STC award CCF-1231216). The authors would like to thank Nvidia for GPU donations.

References

[1] Edward H Adelson. On seeing stuff: the perception of materials by humans and machines. In *SPIE*, 2001.

[2] Renée Baillargeon. Infants' physical world. *Current directions in psychological science*, 13(3):89–94, 2004.

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

[4] Sean Bell, Paul Upchurch, Noah Snavely, and Kavita Bala. Material recognition in the wild with the materials in context database. In *CVPR*, 2015.

[5] Katherine L Bouman, Bei Xiao, Peter Battaglia, and William T Freeman. Estimating the material properties of fabric from video. In *ICCV*, 2013.

[6] Matthew Brand. Physics-based visual understanding. *CVIU*, 65(2):192–205, 1997.

[7] Susan Carey. *The origin of concepts*. Oxford University Press, 2009.

[8] Ronan Collobert, Koray Kavukcuoglu, and Clément Farabet. Torch7: A matlab-like environment for machine learning. In *BigLearn, NIPS Workshop*, number EPFL-CONF-192376, 2011.

[9] Abe Davis, Katherine L Bouman, Justin G Chen, Michael Rubinstein, Frédo Durand, and William T Freeman. Visual vibrometry: Estimating material properties from small motions in video. In *CVPR*, 2015.

[10] Stillman Drake. *Galileo at work: his scientific biography*. Courier Corporation, 2003.

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

[12] Zhaoyin Jia, Andy Gallagher, Ashutosh Saxena, and Tsuhan Chen. 3d reasoning from blocks to stability. *IEEE TPAMI*, 37(5):905–918, 2015.

[13] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

[14] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

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

[16] Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *JMLR*, 17(39):1–40, 2016.

[17] Richard Mann, Allan Jepson, and Jeffrey Mark Siskind. The computational perception of scene dynamics. *CVIU*, 65(2):113–128, 1997.

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

[19] Lerrel Pinto, Dhiraj Gandhi, Yuanfeng Han, Yong-Lae Park, and Abhinav Gupta. The curious robot: Learning visual representations via physical interactions. In *ECCV*, 2016.

[20] Carlo Tomasi and Takeo Kanade. Detection and tracking of point features. *IJCV*, 1991.

[21] Manik Varma and Andrew Zisserman. A statistical approach to material classification using image patch exemplars. *IEEE TPAMI*, 31(11):2032–2047, 2009.

[22] Jacob Walker, Abhinav Gupta, and Martial Hebert. Patch to the future: Unsupervised visual prediction. In *CVPR*, 2014.

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

[24] Lap-Fai Yu, Noah Duncan, and Sai-Kit Yeung. Fill and transfer: A simple physics-based approach for containability reasoning. In *ICCV*, 2015.

[25] Renqiao Zhang, Jiajun Wu, Chengkai Zhang, William T Freeman, and Joshua B Tenenbaum. A comparative evaluation of approximate probabilistic simulation and deep neural networks as accounts of human physical scene understanding. In *CogSci*, 2016.

[26] Bo Zheng, Yibiao Zhao, C Yu Joey, Katsushi Ikeuchi, and Song-Chun Zhu. Detecting potential falling objects by inferring human action and natural disturbance. In *ICRA*, 2014.

[27] Yixin Zhu, Yibiao Zhao, and Song-Chun Zhu. Understanding tools: Task-oriented object modeling, learning and recognition. In *CVPR*, 2015.