Neural Implicit Representations for Physical Parameter Inference from a Single Video

Supplementary Material

Florian Hofherr[1] Lukas Koestler[1] Florian Bernard[2] Daniel Cremers[1]
[1]Technical University of Munich [2]University of Bonn

In this supplemental material we give further details on the architecture and the dynamics models (Section 1) as well as on the training procedure (Section 2) to ensure reproducibility of the work. Moreover, we indicate chosen parameter values for all experiments in Section 3, where we also show additional results and figures for all experiments. This section also includes details on the additional loss terms for the spring example (Section 3.1) as well as the analysis on the generalization ability of the Lagrangian variational autoencoder (Section 3.2). Finally, we discuss potential negative societal impact in Section 3.5.

1. Model Details

1.1. Architecture Background and Object Representation

We adopt the architecture used in [3] for both the representation of the background as well as the representations of the objects. See Figure 1 for the basic structure. Since the skip connection did not seem to give a noticeable benefit in our case we did not include it. We follow [4] to obtain the Fourier mapping for $\mathbf{x} \in \mathbf{R}^d$ as

$$\gamma(\mathbf{x}) = [\cos(2\pi \mathbf{B}\mathbf{x}), \sin(2\pi \mathbf{B}\mathbf{x})]^\top, \tag{1}$$

where $\mathbf{B} \in \mathbb{R}^{N_{Fourier} \times d} \sim \mathcal{N}(0, \sigma^2)$ is sampled from a Gaussian distribution and $\sigma \in \mathbf{R}$ is a hyperparameter that is chosen for each scene. We indicate the values chosen for each experiment in the respective sections.

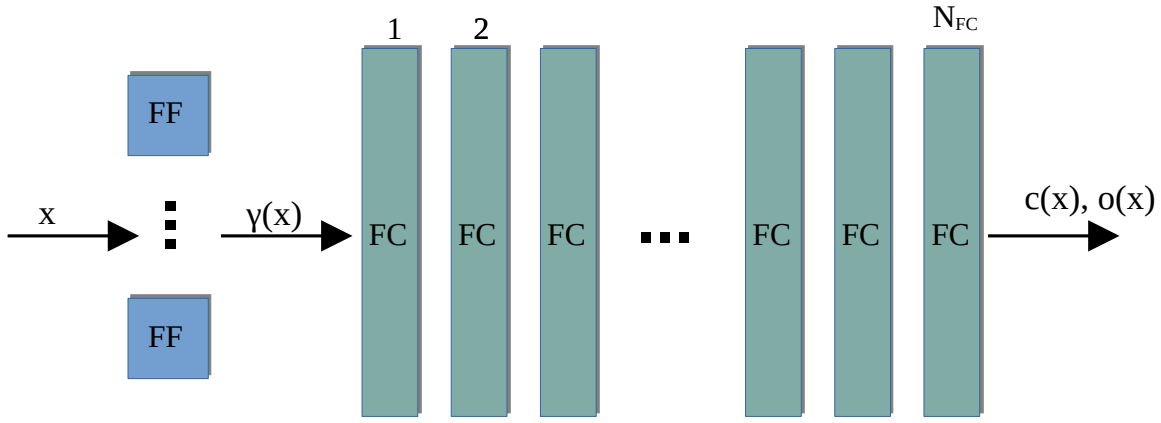

Figure 1: Overview of our architecture for the implicit shape and appearance representations. The input vector $\mathbf{x}$ is passed through a layer of N_{Fourier} Fourier features (FF) to obtain the encoding $\gamma(\mathbf{x})$. The following neural network is constructed from N_{FC} fully connected layers (FC) of width W_{FC} with ReLU activations between the layers. We feed the output of the last layer through a sigmoid function, to achieve values for the color c and the opacity o (only for the local representation) in the range $[0, 1]$. We indicate the values chosen for each experiment in the respective sections.

1.2. Modeling the dynamics

Two Masses Spring system The system is modeled as two-body system where the dynamic of each object is described by Newton's second law of motion, i.e. $F = m\ddot{x}$, where F is the force. Since only the ratio between force and mass can be identified without additional measurement, we fix $m = 1$, analogously to the work of [1]. Using Hooke's law, we write the force applied to object i by object j as

$$F_{i,j} = -k \left((p_i - p_j) - 2l \frac{p_i - p_j}{\|p_i - p_j\|} \right), \tag{2}$$

where k is the spring constant and l is the equilibrium distance. Using the position $p_i(t; k, l)$ of the objects to parametrize the trajectory of two local coordinate systems, we can write the time-dependent 2D spatial transformation to the local coordinate system i as $T_t^{(i)}(x) = x - p_i(t; k, l)$. Besides the initial positions and velocities, l and k are learnable parameters.

Nonlinear damped pendulum A damped pendulum can be modelled as

$$\begin{bmatrix} \dot{\varphi} \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} \omega \\ -\frac{g}{l} \sin(\varphi) - c\omega \end{bmatrix}, \tag{3}$$

where $\varphi \in \mathbb{R}$ is the deflection angle, $\omega \in \mathbb{R}$ is the angular velocity, g is the (known) gravitational acceleration, $l > 0$ is the (physical) length of the pendulum, and $c > 0$ is the damping constant. For the sake of simplicity we assume that the gravitational acceleration g always points downwards in the global image coordinate system. We use the solution curve $\varphi(t; l, c)$ to parameterize the time-dependent 2D spatial transformation as $T_t(x) = R(\varphi(t; l, c))\, x + A$, where $R \in \mathrm{SO}(2)$ is a rotation matrix and $A \in \mathbb{R}^2$ is the pivot point of the pendulum. For the full model A, l, c as well as the initial angle and angular velocity are learnable parameters.

Sliding block We model the sliding block using Newton's second law and gravity that is pointing downward in the global image coordinate system. We model the dynamics as a 1D movement along the inclined plane. Using a friction term with the friction coefficient $\mu > 0$, the ODE for a block on a plane inclined by α reads

$$\begin{bmatrix} \dot{x} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} v \\ g(\sin(\alpha) - \mu \cos(\alpha)) \end{bmatrix}, \tag{4}$$

where $x \in \mathbb{R}$ is the position along the inclined plane, $v \in \mathbb{R}$ is the velocity in this direction and g is again the gravitational acceleration.

Thrown ball We model a thrown object using again Newton's law where only gravity is acting on the object. We assume again, that gravity is pointing downwards in the global image coordinate system. The ODE describing the resulting 2D motion reads

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{v}_x \\ \dot{v}_y \end{bmatrix} = \begin{bmatrix} v_x \\ v_y \\ 0 \\ g \end{bmatrix}, \tag{5}$$

where x and y are the positions in the image coordinate system, v_x and v_y are the velocities in the respective directions and g is the gravitational acceleration.

2. Additional training details

2.1. Optimization

We train our model using the Adam optimizer [2] with exponential learning rate decay, which reads

$$r(e) = r_0 \cdot \beta^{e/n_{\text{decay}}} \tag{6}$$

where $r(e)$ is the learning rate depending on the epoch e, r_0 is the initial learning rate, β is the decay rate and n_{decay} is the decay step size.

One important aspect of the training is to use different learning rates for the parameters θ_{bg} and θ_{obj} of the implicit representations on the one hand and the physical parameters $\theta_{\text{ode}}, \mathbf{z}_0$ and θ_+ on the other hand.

Due to the solution of the ODE, our objective function is generally non-convex and non-linear. Therefore, we rely on a good initialization for the ODE parameters and the parameters of the transformation to achieve good convergence in the optimization. In an earlier version of this work, we used object masks in addition to the images of the sequence to supervise the occupancy values by an additional loss term. While we were able to remove the masks for the supervision, we kept the previous approach to estimate initial values for position and velocities based on the masks.

For the initialization of the pendulum we estimate the pivot point A by averaging all masks and use the the pixel with the highest value. Note that this approach will fail when the pivot point is not contained in the image. To obtain an estimate for the initial angle, we perform a principal component analysis (PCA) on the pixel locations covered by the mask and use the angle between the first component and the vertical direction. The angular velocity is estimated as the angular difference between the first principal components of the first and the second frame divided by the time difference. For the synthetic experiments, averaged over all 27 sequences, this leads to an initialization with a relative error of 14% for the pivot point A and of 40% for the initial values (initial angle and angular velocity).

For the remaining systems, we initialize the initial positions at the center of the masks and the initial velocity as positional difference between the first two frames divided by the time difference. We report the initialization of the remaining parameters in the respective subsection of Section 3.

2.2. Loss term

We use a mean squared error photometric loss defined over all the pixel values, which reads

$$\mathcal{L}_{photometric} = \frac{1}{|\mathcal{I}|\,|\mathcal{T}|} \sum_{t \in \mathcal{T}} \sum_{x \in \mathcal{I}} \left\| I\left(\boldsymbol{x}, t\right) - c\left(\boldsymbol{x}, t\right) \right\|^2, \tag{7}$$

where $\mathcal{T}$ is the set of all given time steps, $\mathcal{I}$ is the set of all pixel coordinates and $I\left(\boldsymbol{x}, t\right)$ are the given images. We found, that for some backgrounds with little distinguishable details, a regularization of the mask is helpful. We use the term

$$\mathcal{L}_{maskReg} = \frac{1}{|\mathcal{I}|\,|\mathcal{T}|} \sum_{t \in \mathcal{T}} \sum_{x \in \mathcal{I}} o(\boldsymbol{x})(1 - o(\boldsymbol{x})), \tag{8}$$

that encourages the occupancy o predicted by the local representation to be either close to 1 or close to zero 0. To avoid "burning" artefacts into the masks, we activate this term after N_{reg} epochs. To balance the term with the photometric loss we use a weight of λ_{reg}. This additional loss is only used for the real world examples and the high resolution synthetic data.

For the training, we randomly sample batches of up to $N_{batch} = 2^{16} = 65536$ pixels for each optimization iteration. We found that this large batch size has a stabilizing effect on the optimization.

2.3. Online Training

We adopt the approach from [5] and increase the number of frames used for the loss term during the optimization. Starting from $n_{fr,0}$ we increase the number of frames by 1 every n_{incrT} steps. We found that this strategy improves the convergence behavior of the approach and seems to make it more robust to the initialization of the parameters.

3. Further experimental details and results

In the following we consider specific details for the different experiments.

3.1. Two Masses Spring System

Experimental details As described in Section 1.2, we employ two independent local representations to model the two digits. By using the maximum of both occupancy values, we enable the model to identify the layering of the objects. Since the two local representations are not explicitly assigned to the digits, we found that we need to guide the model with an *additional* loss term. We use a binary cross entropy loss on very coarse object masks in the *first* frame of the sequence, see Figure 2. The loss is initially weighted by a factor of 0.01 compared to the photometric loss. We reduce this loss term every 100 epochs by a factor of 0.2 to enable the model to learn the fine structures.

Figure 2: Coarse occupancy masks used as supervision in the first frame for the spring sequences. We use the masks shown as overlay in the image to supervise the occupancy of the respective local representation using a binary cross entropy loss. This supervision makes the assignment of the two representations to the digits unique. The weighting of the loss is reduced during training to enable the learning of the fine object structures. Note that the rough object masks are only required in the *first* frame of the sequence.

The physical system appears to have a scale freedom in terms of equilibrium length and the points where the spring is attached to the digits.[1] We observe similar effects when overfitting the model of [1] to a single sequence. When training on the full dataset, the effect seems to be averaged out, and is not observed. We add an additional MSE loss to keep the spring attachment close to the origin of the local coordinate system. This loss is weighted by 0.05.

Finally, we use another MSE loss term to keep the opacity value close to zero outside of (but close to) the visible area. We found this to be necessary, since otherwise artefacts might appear in the prediction, when previously unseen parts of the mask appear in the visible area. This term is weighted by a factor of 1.0.

Model parameters For the background we use an MLP with $N_{\text{FC}} = 6$ fully connected layers of width $W_{\text{FC}} = 64$ and a Fourier mapping with $N_{\text{Fourier}} = 64$ Fourier features and variance $\sigma = 5.0$. To represent the local objects we use $N_{\text{FC}} = 6$ fully connected layers of width $W_{\text{FC}} = 64$ and a Fourier mapping with $N_{\text{Fourier}} = 64$ Fourier features and variance $\sigma = 2.2$.

We use an initial learning rate of $r_{\text{MLP, 0}} = 0.001$ for the parameters of the implicit representations and $r_{\text{param, 0}} = 0.005$ for the physical parameters. We set $\beta_{\text{MLP}} = 0.99954, n_{\text{decay,MLP}} = 50$. We do not decay the learning rate for the physical parameters.

For the online training scheme, we start with $n_{fr,0} = 2$ frames and increase the number of frames by one every $n_{incrT} = 30$ steps. We train for 1200 epochs, where one epoch is completed, when all the pixels have been considered.

The initial spring constant is set to $k = 1.5$ and the equilibrium distance is initialized as the distance between the estimates of the initial positions.

Additional results In Figure 3 and Figure 4 we present additional results for sequence 0 and sequence 1 of the test dataset. We see, that for both sequences, overfitting the baseline is not able to produce a reasonable extrapolation of the data and even produces severe artifacts for the reconstruction part of the sequence. One reason for this is that the model is unable to identify the physical parameters correctly as can be seen by the large relative errors. Our model, on the other hand, is able to estimate the parameters with high accuracy that is even slightly better than the baseline trained on the full training dataset, which again shows the strength of our approach, considering, that we use a single video as input.

3.2. Comparison with the Lagrangian Variational Autoencoder

Experimental details Since the data used in this experiment does not include image data, we use a binary cross entropy loss to penalize the discrepancy between the given object masks and our rendered occupancy values. Since the predicted masks are obtained only from the local representation, we do not use an implicit representation for the background in this example.

Model parameters For the local representation we use an MLP with $N_{\text{FC}} = 6$ fully connected layers of width $W_{\text{FC}} = 64$ and a Fourier mapping with $N_{\text{Fourier}} = 64$ Fourier features and variance $\sigma = 0.1$.

We use an initial learning rate of $r_{\text{MLP, 0}} = 0.005$ for the parameters of the implicit representations and $r_{\text{param, 0}} = 0.01$ for the physical parameters. We do not use any learning rate decay in this example.

For the online training scheme, we start with $n_{fr,0} = 5$ and increase the number of frames every $n_{incrT} = 20$ steps. We train for 2000 epochs, where one epoch is completed, when all the pixels have been considered.

We initialize the damping as $c = 0.25$ and the pendulum length as 1.5.

[1] Intuitively, if the motion is only in one direction (linear), we can vary the equilibrium length and adjust the spring attachments without changing the motion. Similar effects are present in particular 2D motions.

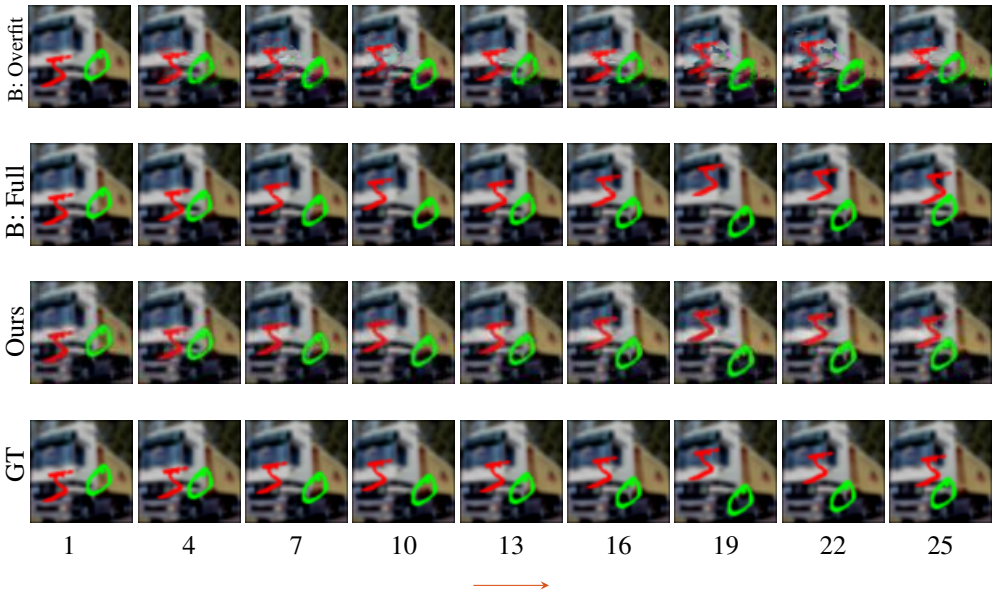

Figure 3: Two masses spring system, where MNIST digits are connected by an (invisible) spring. Reconstruction and prediction for test sequence 0. The arrow indicates where the prediciton starts. For the spring constant and equilibrium distance (k, l) the different methods achieve the following relative errors respectively: $(19.7\%,\ 57.6\%)$ (B: Overfit), $(3.7\%,\ 1.8\%)$ (B: Full), and $(\mathbf{0.3}\%,\ \mathbf{0.7}\%)$ (Ours).

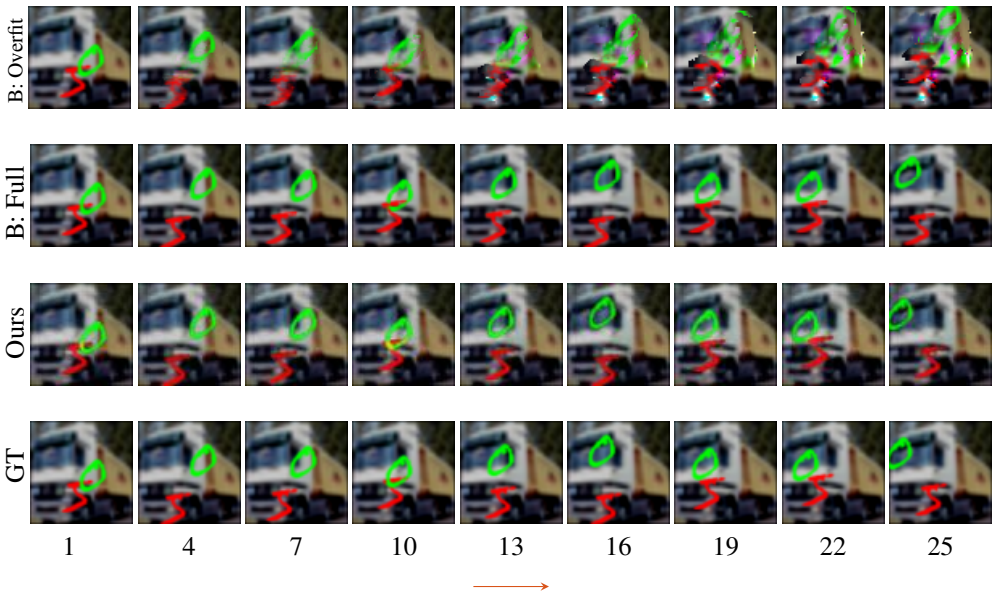

Figure 4: Two masses spring system, where MNIST digits are connected by an (invisible) spring. Reconstruction and prediction for test sequence 1. The arrow indicates where the prediciton starts. For the spring constant and equilibrium distance (k, l) the different methods achieve the following relative errors respectively: $(13.6\%,\ 90.9\%)$ (B: Overfit), $(3.7\%,\ \mathbf{1.8}\%)$ (B: Full), and $(\mathbf{0.1}\%,\ 4.9\%)$ (Ours

Generalization of the Lagrangian Varational Autoencoder One drawback of learning-based approaches for visual estimation of physical models is the poor generalization to data that deviates from the training data distribution. We confirm this

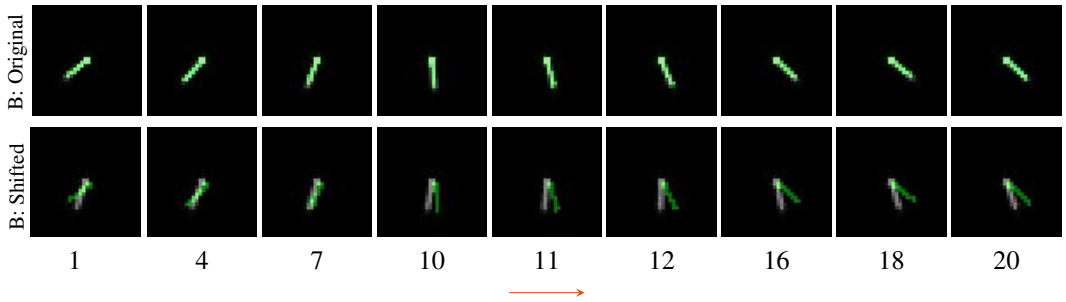

Figure 5: Prediction of the fully trained model of [6] for sequence 3 of the test dataset. While the prediction for the original data is perfect, the prediction for the frames shifted by one pixel in each direction is significantly worse. This shows, that the model does not generalize well to input frames where the pivot point of the pendulum is not in the center of the frame.

for the model of [6] trained on the full dataset. While the IoU averaged over the first 20 sequences of the test set is 0.73, the value drops to 0.21 when we shift the frames of the test dataset by as much as 1 pixel in each direction. This shift corresponds to the case of input videos, where the pivot point of the pendulum is not in the center of the image, which is different from the training data. This effect is visualized in Figure 5, which shows the output of the model for sequence 3 of the test data set with zero control input, both in the original version and in the shifted version. We observe that the small shift of only one pixel in each direction leads to results that are significantly off, and not even the first frame is predicted correctly. While [6] propose to use a coordinate-aware encoder based on spatial transformers, this introduces additional complexity to the model. In contrast, our approach does not suffer from such issues, as we estimate the parameters per scene.

3.3. Synthetic Experiments - High Resolution

Model parameters For the background we use an MLP with $N_{\text{FC}} = 8$ fully connected layers of width $W_{\text{FC}} = 512$ and a Fourier mapping with $N_{\text{Fourier}} = 256$ Fourier features and variance $\sigma = 30.0$. For the representation of the local objects we use $N_{\text{FC}} = 8$ fully connected layers of width $W_{\text{FC}} = 128$ and a Fourier mapping with $N_{\text{Fourier}} = 256$ Fourier features and variance $\sigma = 10.0$.

We use an initial learning rate of $r_{\text{MLP, 0}} = 9e{-}4$ for the parameters of the implicit representations and $r_{\text{param, 0}} = 1e{-}3$ for the physical parameters. We set $\beta_{\text{MLP}} = 0.9, n_{\text{decay,MLP}} = 25$. We do not decay the learning rate for the physical parameters. We activate the mask regularization after $N_{reg} = 400$ epochs and use $\lambda_{reg} = 5e - 4$ to balance the regularization with the photometric loss.

For the online training scheme, we start with $n_{fr,0} = 5$ frames and increase the number of frames by one every $n_{incrT} = 10$ steps. We train for 1200 epochs, where one epoch is completed, when all the pixels have been considered.

We initialize the damping as $c = 0.6$ and the pendulum length as 1.9.

Additional Results Figures 7 and 8 show additional results on the stonewall and woodwall background, also showing the predicted and the groundtruth masks. Figure 6 shows the masks for the sequence considered in the main text, the rendered images are repeated for convenience. The results show, that our method is able to produce excellent reconstruction for unseen time instances, both in terms of visual quality as well as in terms of predicting accurate object masks.

3.4. Real World Examples

Experimental details The pendulum video is recorded at a rate of 30 fps. We extract every third frame into the dataset. For the training we select every second frame of this set and train on 10 frames, covering 1.8 seconds. This leaves frames between the training frames as well as frames to evaluate the extrapolation qualities. We use 31 frames for evaluation, covering 3.9 seconds.

For the sliding block and the ball, the relevant dynamics happen in a significantly shorter amount of time. We record the block at 30 fps and the ball at 120 fps. In both cases the frames cover a time interval of 0.4 seconds. We use again every second frame for training, and use 6 training frames each. This leaves 7 frames for evaluation of the block and 8 for the ball.

Model parameters For the background we use an MLP with $N_{\text{FC}} = 8$ fully connected layers of width $W_{\text{FC}} = 512$ and a Fourier mapping with $N_{\text{Fourier}} = 256$ Fourier features and variance $\sigma = 30.0$ for the ball and the sliding block, and $\sigma = 50.0$ for the pendulum. For the representation of the local objects we use $N_{\text{FC}} = 8$ fully connected layers of width $W_{\text{FC}} = 128$ and a Fourier mapping with $N_{\text{Fourier}} = 128$ Fourier features and variance $\sigma = 5.0$ for the ball and $\sigma = 15.0$ for the sliding block and the pendulum.

We use an initial learning rate of $r_{\text{MLP, 0}} = 9e{-}4$ for the parameters of the implicit representations and $r_{\text{param, 0}} = 1e{-}3$ for the physical parameters. We set $\beta_{\text{MLP}} = 0.9, n_{\text{decay,MLP}} = 25$. We do not decay the learning rate for the physical parameters. We activate the mask regularization after $N_{reg} = 100$ epochs and use $\lambda_{reg} = 1e - 3$ to balance the regularization with the photometric loss.

For the online training scheme, we start with $n_{fr,0} = 5$ frames for the block and the pendulum and $n_{fr,0} = 8$ for the ball. For the ball and the sliding block we increase the number of frames by one every $n_{incrT} = 10$ steps, for the pendulum every $n_{incrT} = 20$ steps. We train for 1200 epochs, where one epoch is completed, when all the pixels have been considered.

We initialize the friction coefficient for the sliding block as $\mu = 0$, the damping for the pendulum as $c = 0.5$ and the pendulum length as 0.4.

Additional Results Figures 9 to 11 and show renderings for the test frames of the real world data, as well as visualizations of the object masks obtained by our method. The results show, that our method is able to achieve highly detailed reconstruction for all 3 real world scenarios. Moreover, We obtain accurate masks for the dynamic objects observed in the scene.

3.5. Potential Negative Societal Impact

This work attempts to learn interpretable physical models from video clips. While the work is mostly fundamental, it enables a user to edit a scene in a physically plausible manner, at least if the dynamics can be modelled explicitly and the camera and the rest of the scene are static. However, for the physical scenarios that we show, we could not think of possible usages of our method, that could be harmful to individuals or groups of people. In our opinion, the potential for harmful missuse of methods operating on videos is given in particular if the model can alter the actions, expressions or in general the behavior of humans in that scene. In the current state, our method is not able to do such things.

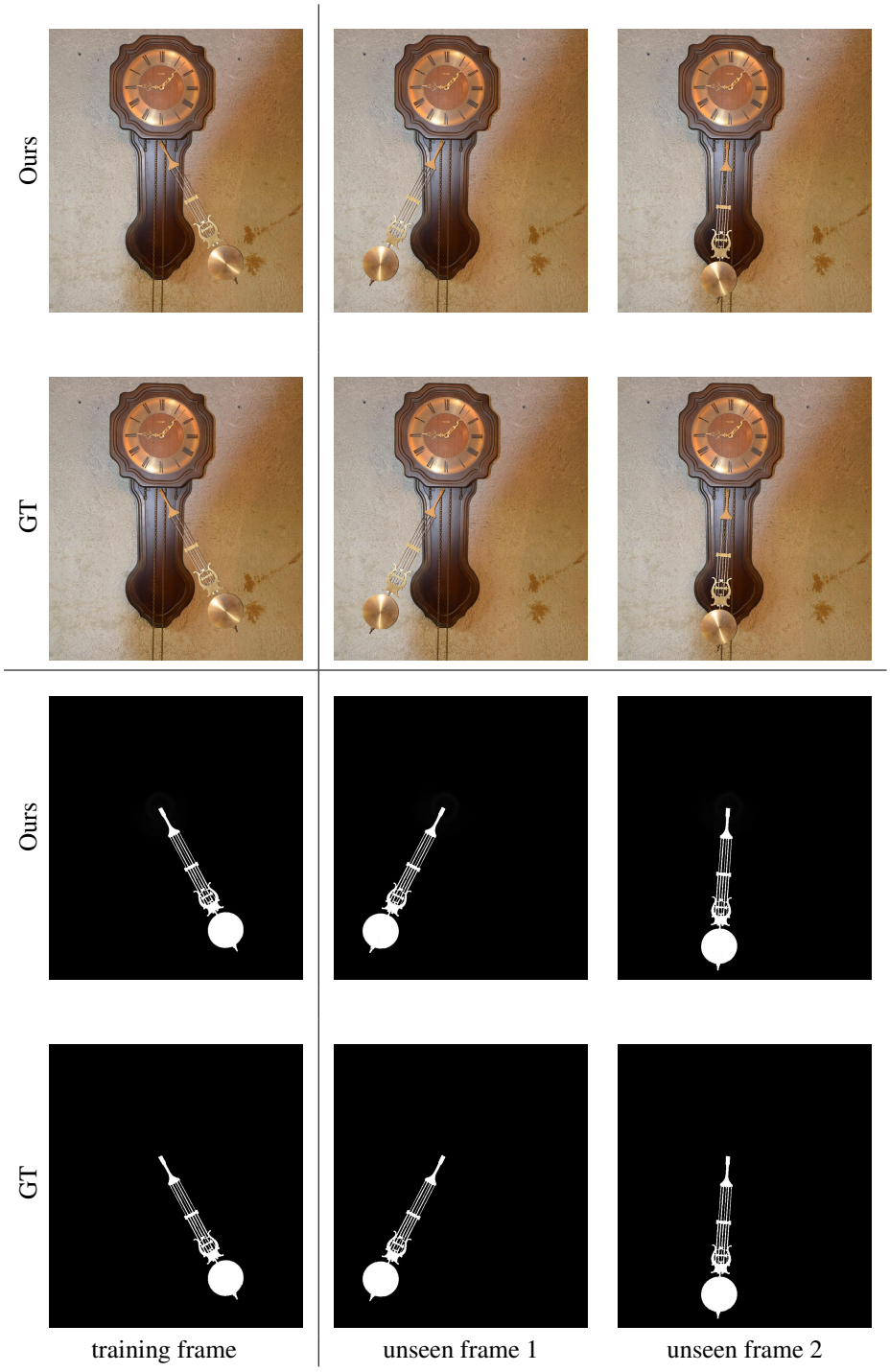

Figure 6: Rendered frames for sequence 1 of the wallclock background (same sequence as in the main text, rendered images are shown again for convenience). The left image is part of the training set, “unseen frame 1” is between two training frames, “unseen frame 2” is a future frame after the interval seen during training. Our method produces photorealistic predictions for the unseen time instances. Also, it predicts accurate segmentation masks for the object.

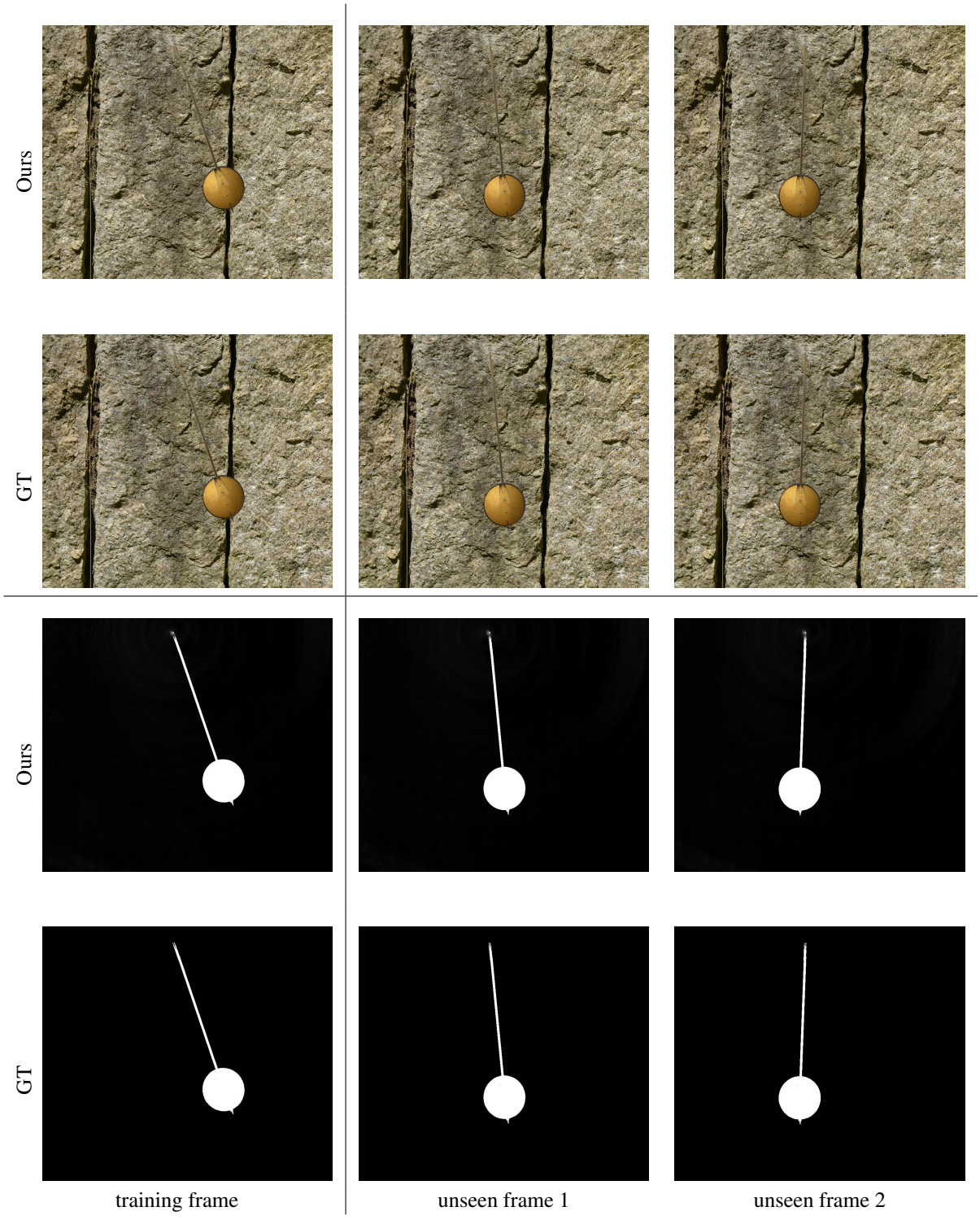

Figure 7: Rendered frames for sequence 5 of the stonewall background. The left image is part of the training set, "unseen frame 1" is between two training frames, "unseen frame 2" is a future frame after the interval seen during training. Our method produces photorealistic predictions for the unseen time instances. Also, it predicts accurate segmentation masks for the object.

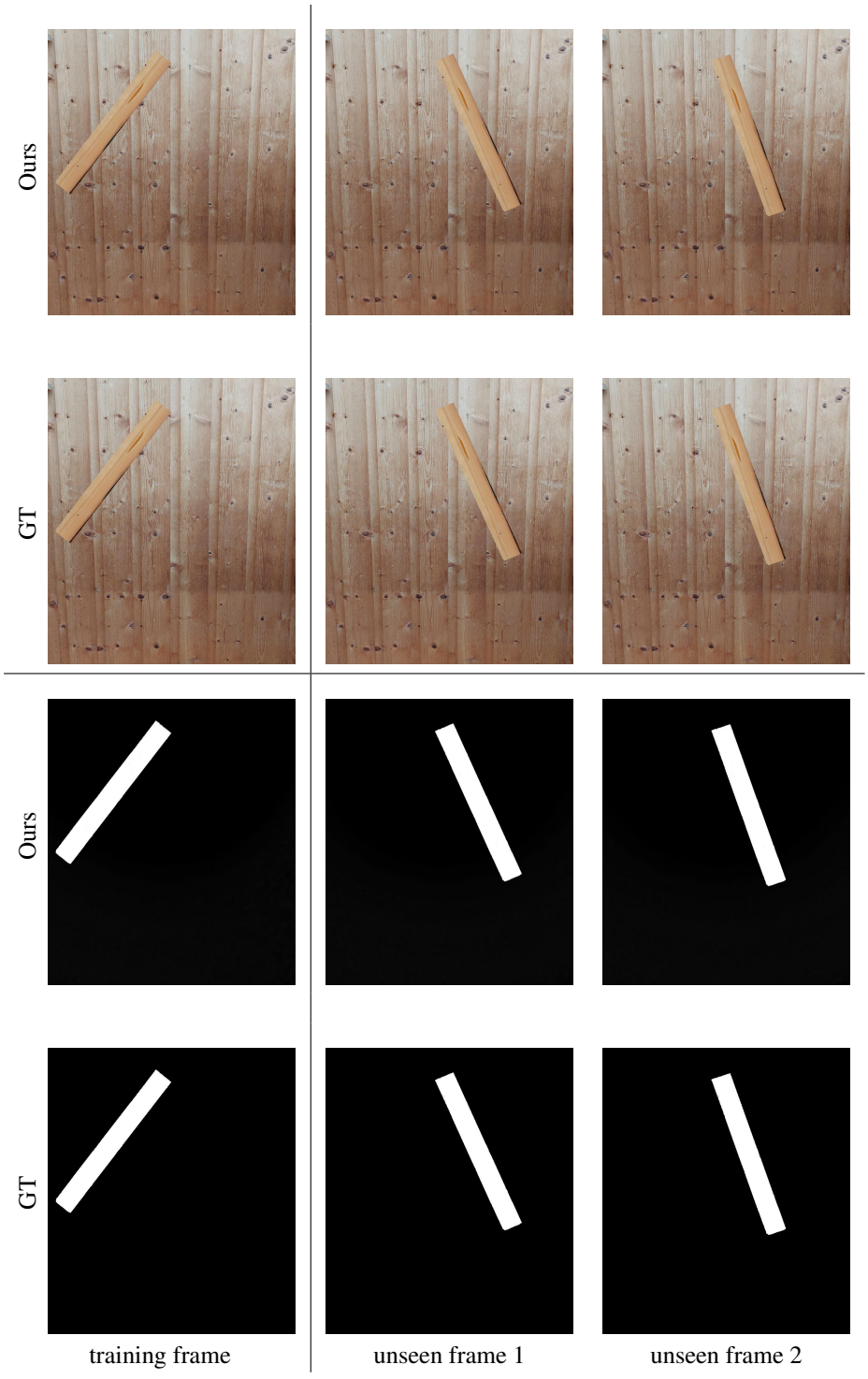

Figure 8: Rendered frames for sequence 7 of the woodwall background. The left image is part of the training set, “unseen frame 1” is between two training frames, “unseen frame 2” is a future frame after the interval seen during training. Our method produces photorealistic predictions for the unseen time instances. Also, it predicts accurate segmentation masks for the object.

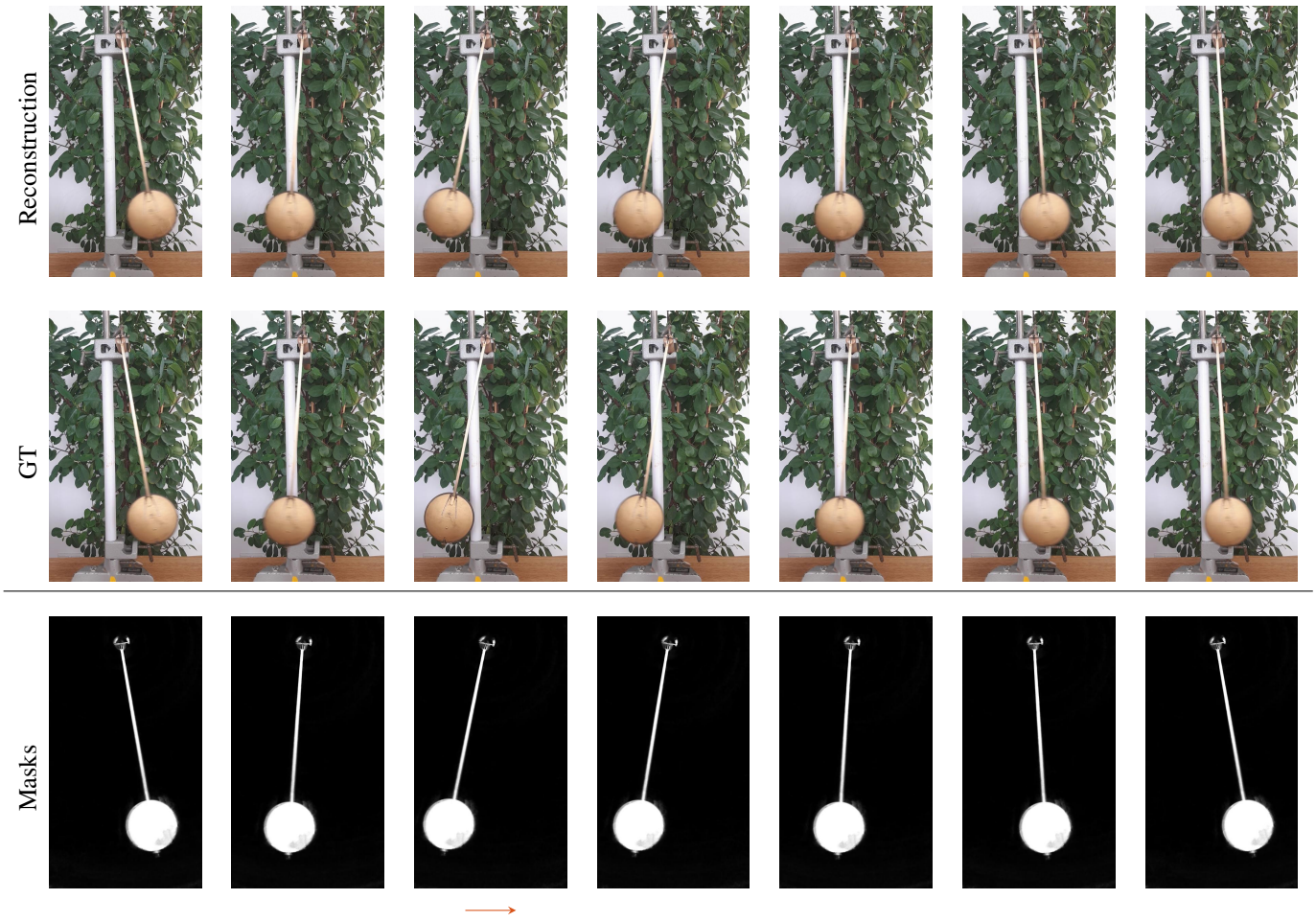

Figure 9: Rendered frames 8-13 of the test set for the real world pendulum sequence. The two left frames are between training frames, the remaining frames are extrapolated (indicated by the red arrow). Our method produces photorealistic predictions for the unseen time instances. Also, it predicts accurate segmentation masks for the object.

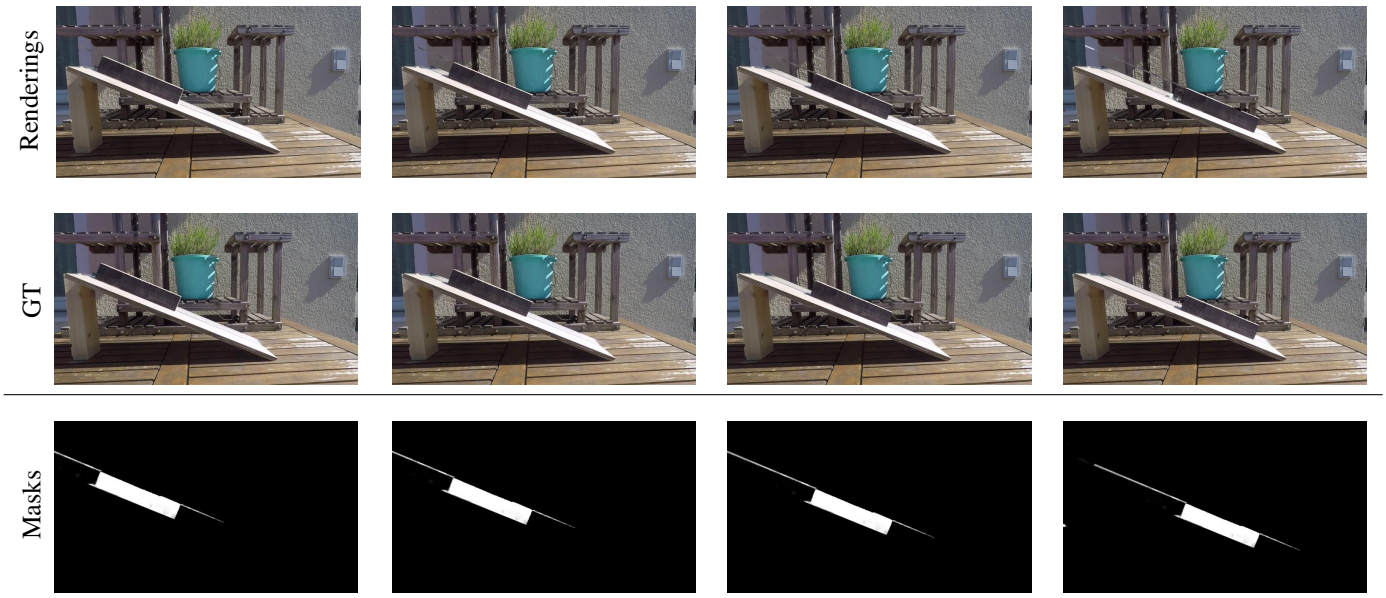

Figure 10: Rendered frames 1, 3, 5, and 7 of the test set for the real world sliding block sequence. The frames are between training frames. Our method produces photorealistic predictions for the unseen time instances. Also, it predicts accurate segmentation masks for the object.

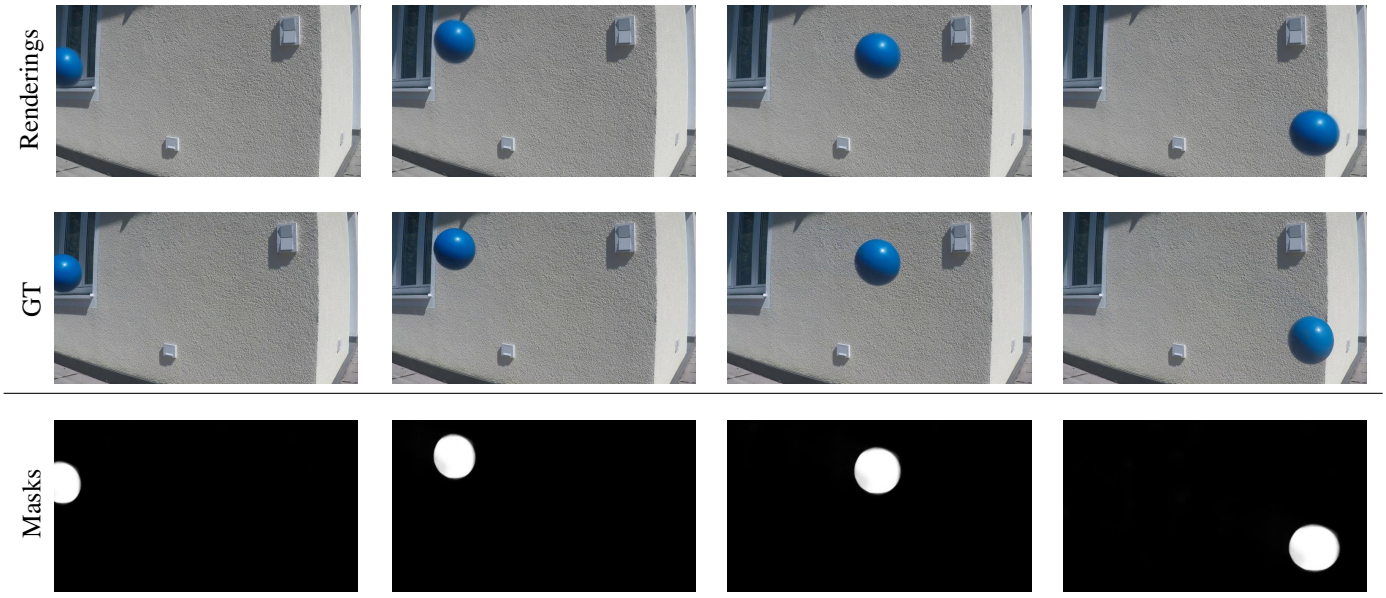

Figure 11: Rendered frames 1, 3, 5, and 7 of the test set for the real world ball sequence. The frames are between training frames. Our method produces photorealistic predictions for the unseen time instances. Also, it predicts accurate segmentation masks for the object.

References

[1] Miguel Jaques, Michael Burke, and Timothy M. Hospedales. Physics-as-inverse-graphics: Unsupervised physical parameter estimation from video. In *International Conference on Learning Representations (ICLR)*, 2020.

[2] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

[3] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *European Conference on Computer Vision (ECCV)*, 2020.

[4] Matthew Tancik, Pratul P. Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T. Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[5] Wentao Yuan, Zhaoyang Lv, Tanner Schmidt, and Steven Lovegrove. Star: Self-supervised tracking and reconstruction of rigid objects in motion with neural rendering. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[6] Yaofeng Desmond Zhong and Naomi Ehrich Leonard. Unsupervised learning of lagrangian dynamics from images for prediction and control. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2020.