Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnline CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 58-15 (2024) 301–306

Identifying the dynamics of interacting objects with applications to scene understanding and video temporal manipulation. [1]

Armand Comas * **Christian Fernandez** * **Sandesh Ghimire** *
Haolin Li * **Octavia Camps** * **Mario Sznaier** *
* *ECE Dept., Northeastern University, Boston, MA 02115 USA*
(e-mail: camps@coe.northeastern.edu, msznaier@coe.northeastern.edu)

Abstract: There is an ongoing effort in the machine learning community to enable machines to understand the world symbolically, facilitating human interaction with learned representations of complex scenes. A pre-requisite to achieving this is the ability to identify the dynamics of interacting objects from time traces of relevant features. In this paper, we introduce GrODID (GRaph-based Object-Centric Dynamic Mode Decomposition), a framework based on graph neural networks that enables Dynamic Mode Decomposition for systems involving interacting objects. The main idea is to model individual, potentially non-linear dynamics using a Koopman operator and identify its corresponding Dynamic Mode Decomposition using deep AutoEncoders, while the interactions amongst systems are captured by a graph, modeled by a Graph Neural Net (GNN). The potential of this approach is illustrated with several applications arising in the context of video analytics: video forward and backwards prediction, video manipulation and achieving temporal super-resolution.

Keywords: Deep Learning, Koopman Operators

1. INTRODUCTION

Understanding complex scenarios often requires unraveling the dynamics of the interactions between several objects. Formally, this problem can be stated as identifying a dynamical graphical model, represented by a graph structure $G = \{V, E\}$, where each vertex is associated with a time series generated by an underlying dynamical system and the edges represent interactions between these systems. In the case of linear dynamics and interactions, the problem can be recast into a block-sparsification form (Wang et al. (2016)) that can be solved using first order methods. However, these results strongly hinge on the linearity assumption. In addition, the resulting models provide the sparsest representation of the underlying graph, but this representation does not necessarily respect physical priors. A second limitation of this approach is that, even if using efficient first order optimization methods to solve the sparsification problem, computational complexity limits the approach to a relatively moderate number of objects and data points. Thus, in general they cannot handle practical problems arising for instance in the context of video analytics and scene understanding.

In this paper, we consider scenarios where both the individual dynamics and the interactions are not necessarily linear and address the problem from a Koopman operator perspective. Our goals are to (**i**) Learn the (possibly non-linear) individual dynamics of interacting systems directly from time series collected from the ensemble of systems. (**ii**) Model systems interactions in a multi-system scenario.

To achieve these goals, we propose a framework, GrODID (GRaph-based Object-Centric Dynamic Mode Decomposition), a framework based on graph neural networks. The main idea is to model individual system dynamics using a Koopman operator and identify its corresponding Dynamic Mode Decomposition using deep AutoEncoders, while the interactions amongst systems are captured by a graph, modeled by a graph neural network (GNN).

The effectiveness of the proposed approach is illustrated with a non-trivial application arising in the context of video-analytics: learning the individual dynamics of individual objects and their interactions directly from pixels in a video. As we will show here, the proposed approach not only learns models with good predictive power for interacting systems, but these models also allow for video manipulation and achieving super-resolution.

The paper is organized as follows: In Section 2 we introduce Koopman operators and in Section 3 we introduce our method in detail. In Section 4 we propose a non-trivial application for our approach. Next, in Section 5, we describe our experiments and results. Finally, we summarize our conclusions in Section 6.

[1] This work was partially supported by NSF grants IIS-1814631 and CNS-2038493, AFOSR grant FA9550-19-1-0005, ONR grant N00014-21-1-2431, and the Sentry DHS Center of Excellence under Award 22STESE00001-03-02.

10.1016/j.ifacol.2024.08.545

2. BACKGROUND ON KOOPMAN OPERATORS

Consider an autonomous system characterized by the nonlinear, time-invariant model:

$$\mathbf{x}_{t+1} = \Phi(\mathbf{x}_t),\ \mathbf{x}_t \in \Re^n \tag{1}$$

The key concept of Koopman operator theory lies in the realization that the dynamics of a finite-dimensional nonlinear system can be represented as an infinite-dimensional linear dynamical system. This is achieved by selecting a suitable Hilbert space of observables, denoted as $\mathbf{g}$, as introduced in Koopman (1931); Mezic (2005). The objective is to find an operator $\mathcal{K} : \Re^m \to \Re^m$ that advances the observables $\mathbf{g}$ one step into the future:

$$\mathbf{g}_t = f_\psi(\mathbf{x}_t), \quad \mathbf{g}_{t+1} = \mathcal{K}\mathbf{g}_t, \tag{2}$$

where $f_\psi : \Re^n \to \Re^m$ is the mapping from the state space to the observable space $\mathbf{g}_t \in \Re^m$. Hence:

$$f_\psi \circ \Phi(\mathbf{x}_t) = \mathcal{K} f_\psi(\mathbf{x}_t), \tag{3}$$

where $\circ$ denotes the composition operator. When $\mathcal{K}$ has a countable set of eigenfunctions $\psi_i(.)$ with eigenvalues λ_i, the observables $\mathbf{g}(.)$ can be propagated as follows. Let $\mathbf{a} = [a_1 \ldots]^T$ denote the coordinates of $\mathbf{g}$ in the basis spanned by $\psi(.)$, that is

$$\mathbf{g}(.) = \sum a_i \psi_i(.) \doteq \mathcal{D}(.)\mathbf{a}, \text{ where: } \mathcal{D}(.) = [\psi_1(.) \ldots]$$

Then

$$(\mathcal{K} \circ \mathbf{g})(.) = \sum a_i \mu_i \psi_i(.) = (.)\mathbf{M}\mathbf{a}, \text{where } \mathbf{M} = \text{diag}(\lambda_i)$$

Extended Dynamical Mode Decomposition (EDMD) type approaches seek to identify approximations to Koopman operators over a restricted subspace, defined by the span of the truncated dictionary $\mathbf{D}(.) \doteq [\psi_1(.) \ldots \psi_N(.)]$. In this subspace, the Koopman operator can then be approximated by a matrix $\mathbf{K} \in \mathbb{R}^{N \times N}$ that propagates the coefficients of the expansion, that is, for $\mathbf{g}(.) = \mathbf{D}(.)\mathbf{a}$, then $(\mathcal{K} \circ \mathbf{g})(.) = \mathbf{D}(.)\mathbf{K}\mathbf{a}$. Typically, given experimental data $\mathbf{X} \doteq [\mathbf{x}_1\ \mathbf{x}_2\ \ldots\ \mathbf{x}_T]$, $\mathbf{K}$ is found by minimizing the one-step prediction error over a set of observables. Specifically, this approach considers m observables $\mathbf{g}^{(j)}(.) \doteq \mathbf{D}(.)\mathbf{a}_j$, each defined by a coordinate vector $\mathbf{a}_j$, and solves:

$$\mathbf{K} = \underset{K}{argmin} \sum_{j=1}^{m} \sum_{t=1}^{T-1} \|[\mathbf{D}(\mathbf{x}_{t+1}) - \mathbf{D}(\mathbf{x}_t)\mathbf{K}]\mathbf{a}_j\|_2^2 \tag{4}$$

where $\mathbf{D}(\mathbf{x}_t)$ is the matrix obtained by evaluating the dictionary at the point $\mathbf{x}_t$.

Discovering eigenfunctions can be a challenging task, prompting the development of various algorithms to address this difficulty. Among the most commonly utilized methods are dynamic mode decomposition (DMD) and its extension to nonlinear observables, known as extended DMD (EDMD) (Schmid (2008); Williams et al. (2015)). In earlier studies, researchers employed manually crafted eigenfunctions to model the observable space, selecting them from function families or leveraging prior knowledge of the physics associated with the problem.

Currently, some approaches leverage deep neural networks to represent the observable space (Lusch et al. (2018); Morton et al. (2019); Li et al. (2020); Xiao et al. (2021); Azencot et al. (2020)). Neural networks have the advantage of being universal approximators, proving effective in identifying the Koopman invariant subspace.

3. PROPOSED APPROACH

In this section we introduce the architectures used to learn the individual dynamics and represent the interactions between systems.

3.1 Koopman Embedding

To model the individual dynamics, we propose to learn a suitable set of observables $\mathbf{g}$ and the associated Koopman operator representation $\mathbf{K}$ using a deterministic Auto-Encoder (AE) framework.

An AE is a Neural Network, trained end-to-end through stochastic gradient descent (or similar), that transforms the input data through an Encoder into a latent space, and maps it back to the original domain with a Decoder. The interpretation or utility of the latent space is defined by the task for which the AE is trained. For instance, if the latent space is low-dimensional, the AE will learn a lossy compression of the data. When the goal is prediction by propagating linear dynamics in the latent space, we expect the AE to learn the desired Koopman mapping.

The observables $\mathbf{g}_t^i$ are obtained through the Koopman mapping $f_\mathcal{M} : \Re^n \to \Re^m$ (equivalent to Eq. (2)). $\mathcal{M}$ makes reference to the spaces of observables, respectively. We recover the original state by approximating the inverse function $f_{\mathcal{M}^{-1}} \approx f_\mathcal{M}^{-1}$.

Previous work, has focused on modeling such system in the absence of interactions among particles in a system. A simple multi-layer perception (MLP) has been used to map states to observables in the Koopman space, and vice-versa. This corresponds to a function applied independently for every trajectory as follows:

$$\hat{\mathbf{x}}_{t+1}^i = f_{\mathcal{M}^{-1}} \circ \left(\mathcal{K} \circ f_\mathcal{M}\left(\mathbf{x}_t^i\right)\right) \tag{5}$$

$$= f_{\mathcal{M}^{-1}}\left(\mathcal{K}\mathbf{g}_t^i\right). \tag{6}$$

We define the Koopman operators $\mathcal{K} : \Re^\mathcal{M} \to \Re^\mathcal{M}$ as a parameter matrix. This means that a single Koopman operator must be a valid transition operator for all the data in our dataset.

As interactions occur, correct modeling of the dynamics requires new tools. We propose the use of graph neural networks as described in the next section.

3.2 Koopman Interaction Network

In this paper, we will model interactions between systems using a fully connected graph neural network. The mapping $f_\mathcal{M}$ is parametrized by a network reminiscent of Interaction Networks (Battaglia et al. (2016)). The N vertices of the graph correspond to the individual systems.

We associate the states of the individual systems to the vertices of the graph, together with a binary identifier of the system $\mathbf{v}_t^i = f_\mathtt{V}\left(\left[\mathbf{x}_t^i, \mathtt{ID}^i\right]\right)$, and consider a global state $\mathbf{x}_t$ formed by stacking the individual states $\mathbf{x}_t^i$. The corresponding observables $\mathbf{g}_t$ are partitioned into components $\mathbf{g}_t^i$ conformally to $\mathbf{x}_t^i$. Relations are represented as the identifiers and the difference of the associated vertices $\mathbf{r}_t^{(k,l)} = f_\mathtt{R}\left(\left[\mathtt{ID}^k, \mathtt{ID}^l, \mathbf{v}_t^k - \mathbf{v}_t^l\right]\right), \forall (k,l) \in [1, \ldots, N]$.

The embedding space is the result of a message passing procedure, where we compute an edge effect $\mathbf{e}_t^{(k,l)} = f_{\texttt{EE}}\left(\left[\mathbf{v}_t^l, \mathbf{v}_t^k, \mathbf{r}_t^{(k,l)}\right]\right), \forall (k,l) \in [1,\dots,N]$ and a node effect $\mathbf{g}_t^i = f_{\texttt{NE}}\left(\left[\mathbf{v}_t^i, \sum_{n=1}^N \mathbf{e}_t^{(i,n)}\right]\right)$. All functions f are implemented as MLPs. Similarly to Li et al. (2020), we apply this graph network both as the Koopman mapping $\mathbf{g}_t = f_{\mathcal{M}}(\mathbf{x}_t)$ and its inverse $\mathbf{x}_t = f_{\mathcal{M}^{-1}}(\mathbf{g}_t)$. We do not indicate the nature of interactions supervisedly. Instead, we will assume that interactions between systems will remain the same across training and testing data. Thus, we re-write Eq. (2) as

$$\mathbf{g}_{t+1}^i = \mathcal{K}^{(i)}\mathbf{g}_t \doteq \left(\sum_{n=1}^N \mathcal{K}^{(n,i)}\mathbf{g}_t^n + \sum_{n=1}^N \mathcal{K}^{(n,N\setminus i)}\mathbf{g}_t^n\right) \quad (7)$$

where $\mathcal{K}^{(i)}$, the i^{th} block-row of the global matrix $\mathcal{K}$, has been decomposed into submatrices $\mathcal{K}^{(l,k)}$ associated to the l sender and k receiver systems, respectively. Thus, the interactions are also linear in the Koopman space. The corresponding state at the i^{th} node is given by

$$\begin{aligned}\mathbf{x}_{t+1}^i &= f_{\mathcal{M}^{-1}}\left(\mathbf{g}_{t+1}^i\right) \\ &= f_{\mathcal{M}^{-1}}\left(\sum_{n=1}^N \mathcal{K}^{(n,i)}\mathbf{g}_t^n, \sum_{n=1}^N \mathcal{K}^{(n,N\setminus i)}\mathbf{g}_t^n\right)\end{aligned} \quad (8)$$

To train our model we make use of self-supervision for reconstructing the observed time series $\mathbf{x}_{1:T}$ and predicting that same input from few initial conditions $(1:K)$. We also add regularizers for a better learning of the Koopman embedding, so that the final expression of our objective $\mathcal{J}_\omega$ with respect to the trainable weights ω is:

$$\mathcal{J}_\omega = \min_\omega \left[L_{\texttt{Koop}}\right] = \min_\omega \left[L_{\texttt{fit}} + L_{\texttt{AE}} + L_{\texttt{Rank}\mathcal{K}}\right] \quad (9)$$

$$L_{\texttt{AE}} = \lambda_{\texttt{AE}} \left\|\mathbf{x}_{2:T}^{1:N} - \hat{\mathbf{x}}_{2:T|1:T-1}^{1:N}\right\|_1, \quad L_{\texttt{Rank}\mathcal{K}} = \lambda_{\texttt{r}\mathcal{K}} \left\|\mathcal{K}\right\|_*$$

$$L_{\texttt{fit}} = \lambda_{\texttt{fit}} \left\|\mathbf{g}_{2:T}^{1:N} - \hat{\mathbf{g}}_{2:T|1:T-1}^{1:N}\right\|_2^2, \quad (10)$$

where λs are the weights applied to each one of the loss terms, and $\|\cdot\|_1$, $\|\cdot\|_2^2$ represent the ℓ_1 and ℓ_2 losses respectively and $\|\cdot\|_*$ is the nuclear norm of a matrix, used as a convex surrogate of the rank function.

4. APPLICATION: FINDING OBJECTS AND THEIR INTERACTIONS FROM VIDEO DATA

In this section we illustrate the proposed approach in a non-trivial problem arising in the context of video-analytics: given a video clip containing up to N interacting objects, determine the number of objects, the individual dynamics and their interactions directly from the pixels.

4.1 Object-centric Video Representation

We draw inspiration from previous work on object-centric representation learning to infer dynamics from pixels without explicit supervision. These methods often rely on black-box deep networks to model dynamics. Instead, OKID (Comas et al. (2023)), is a similar methodology that leverages a Koopman operator to model non-linear dynamical systems. This allows for interperability and manipulation of the scene dynamics. OKID, however, is limited to scenes with non-interacting objects. In this work we extend the OKID model to the case of multiple interacting systems employing graph networks and provide new manipulation techniques.

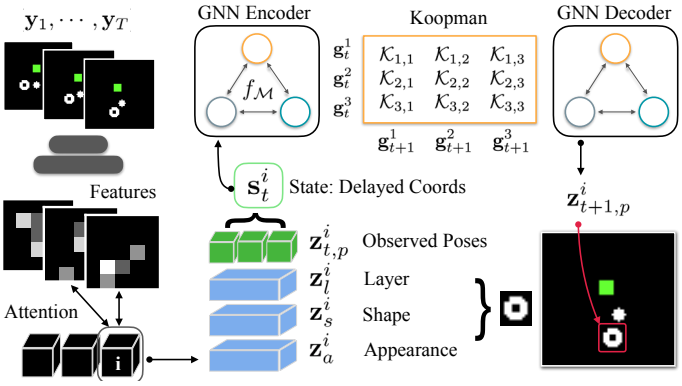

Fig. 1. GrODID architecture. Left: an attention-based recurrent tracker decomposes the scene into its composing objects. Center Down: the object representations are disentangled into Confidence (ommited for clarity), Layer, Shape, Appearance and Pose. Center Top: the dynamic features of Pose are concatenated into a state and forecasted by using a Koopman embedding. The Koopman embedding consists of a Graph-based AutoEncoder with a linear operator in the center (Koopman). Static representations are later used to generate objects and hierarchically compose the scene.

Video latent representations: We leverage the perception pipeline of Comas et al. (2023), where a perceptual module consisting of an array of attention-based trackers, learns to capture object attributes from pixels, without supervision.

The attributes are:

- **Pose** $\mathbf{z}_{t,p} \in (-1,1)^4$: Indicates the x and y coordinates of an object centroid, scale and ratio, scaled to the range $(-1,1)$, thus defining the 2D affine spatial transformation of an object;
- **Appearance** $\mathbf{z}_{t,a} \in \Re^A$: An abstract representation of an object's appearance;
- **Shape** $\mathbf{z}_{t,s} \in (0,1)^S$: Binary object shape mask;
- **Confidence** $z_{t,c} \in (0,1)$: Confidence of our model in the estimated prediction;
- **Layer** $\mathbf{z}_{t,l} \in (0,1)^L$: An indicator of relative depth.

Scene Rendering and Training. Consider a video sequence $(\mathbf{y}_1, \cdots, \mathbf{y}_T)$ and its inferred attribute representations $\mathbf{z}_t^i = \left[z_{t,c}^i, \mathbf{z}_{t,l}^i, \mathbf{z}_{t,p}^i, \mathbf{z}_{t,s}^i, \mathbf{z}_{t,a}^i\right]$. Given the latent variables, we reconstruct the frame for each object sequentially. We first generate an object by decoding its stochastic appearance $\mathbf{z}_{t,a}^i$. The decoder $f_{\texttt{dec}} : \Re^A \to \Re^R$ is a deconvolutional layer. We then rescale and place the object with a spatial transformer $\mathcal{T}$, according to the predicted poses. Each object is modeled probabilistically by:

$$p(\mathbf{y}_t^i | \mathbf{z}_{t,a}^i) = \mathcal{T}(f_{\texttt{dec}}(\mathbf{z}_{t,a}^i); \mathbf{z}_{t,p}^i) \circ z_{t,c}^i, \quad (11)$$

The final scene is a layer-by-layer composition, as described in He et al. (2019). We compose the decoded object appearances, according to their estimated shape

$\mathbf{z}^i_{t,s}$ and layers $\mathbf{z}^i_{t,l}$, and locate the objects in the predicted locations, with their predicted scale $\mathbf{z}^i_{t+1,p}$.

Similarly to the VAE framework (Kingma and Welling (2013)), we train the model by maximizing the evidence lower bound (ELBO), with self-supervision between the ground-truth and the generated frames. We supervise both the reconstructed frames given initial conditions and the predicted future frames. We add regularizations in our intermediate representations.

Our objective function $\mathcal{J}_\omega$ for the trainable weights ω is:

$$\mathcal{J}_\omega = \min_{\omega} \left[-\text{ELBO} + L_{\texttt{Koop}} \right] \tag{12}$$

Note that this objective contains the Koopman objective described in (9). We add the ELBO loss to account for reconstruction in the pixel space and regularization of the intermediate representations (attributes).

4.2 Identifying the Koopman Operator

In the previous section, we summarized the process to map pixels to object poses $\mathbf{z}^i_{t,p}$, and vice-versa, without supervision. Next, we describe the process to identify the Koopman operator associated with the dynamics of $\mathbf{z}^i_{t,p}$. We define the state by concatenating T_S delayed instances (hereafter called delayed coordinates) of the pose vector $\mathbf{z}^i_p$ for the i_{th} object:

$$\mathbf{s}^i_t = \left[\mathbf{z}^i_{t,p}, \mathbf{z}^i_{t-1,p}, \dots, \mathbf{z}^i_{t-T_S+1,p} \right] \tag{13}$$

Therefore, $\mathbf{s}^i_t \in \mathbb{R}^{4T_s}$. T_s indicates our prior on the maximum number of time-steps needed to model dynamics with an Auto-Regressive (AR) approach. We assume that the effects of the environment on each object remain unchanged through training and testing cases. Thus, these effects are implicitly learned by the model.

4.3 Time reversal and Super-Resolution

A unique feature of the proposed approach is its ability to easily change the discrete time resolution or reverse time. As described in Eqs. (5), (6) and (8) we advance one step in time by using once the Koopman operator $\mathcal{K}$. Thus, `GrODID` is learning to model discrete time at the sampling frequency of the training distribution. In this context, an arbitrarily high temporal resolution can be achieved by manipulating the Koopman operator $\mathcal{K}$. We thus would operate with the eigendecomposition of $\mathcal{K} = Q\Lambda Q^{-1}$, to compute $\mathcal{K}^{1/m} = Q\Lambda^{1/m}Q^{-1}$, raising each complex element of the diagonal to the power $1/m$. to increase resolution of our predictions by a factor of m. This can be generalized to negative values of m. In that case, we would be predicting backwards in time by applying the modified operator. In our experiments, we evaluate the qualitative performance for $m = -1$.

4.4 The Overall System

We will use the architecture shown in Fig. 1 which is similar to the one in Comas et al. (2023), adapted for videos with interacting objects. On the left, we see the perception module, where the input video frames are encoded into features, and a set of trackers attend to objects in the scene through an attention mechanism and output a set of attributes per object (blue for static, green for dynamic). The observed poses $\mathbf{z}^i_{t,p}$ are concatenated into a state $\mathbf{s}^i_t$ as delayed coordinates. We encode all tracked objects at once through a GNN Encoder, propagate the observables $\mathbf{g}^i_t$ T steps in the future, and render an image with the decoded pose and the remaining attributes.

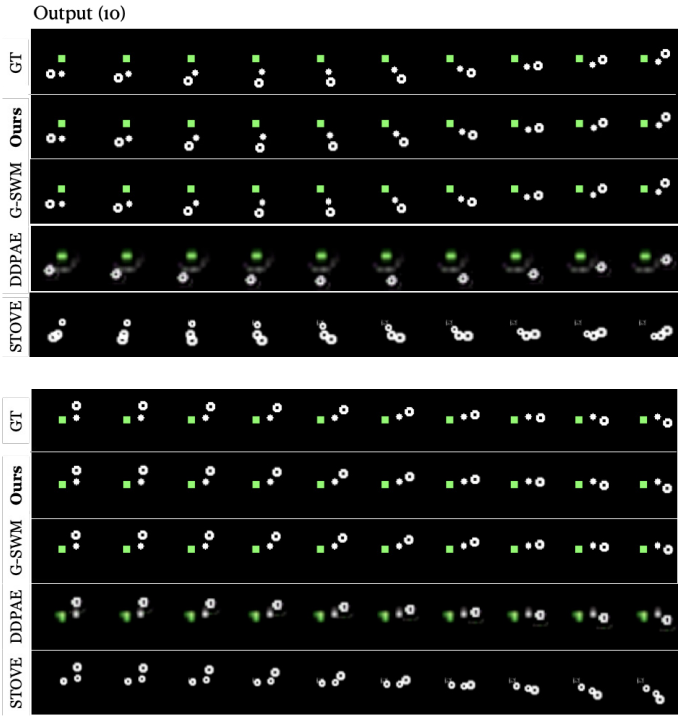

Fig. 2. Qualitative comparison of `GrODID` against the baselines for the Double Pendulum experiments. `GrODID` and G-SWM generate very accurate frames. DDPAE and STOVE fail both to generate sharp and accurate objects, and to correctly predict their trajectory.

5. EXPERIMENTS

With our experiments, we attempt to demonstrate the ability of our model to **(i)** Forecast videos with interacting objects and **(ii)** Manipulate dynamics in videos. We evaluated `GrODID`'s performance in the presence of interactions, with realistic dynamics, by performing experiments with data from a real Double Pendulum. Also, we show quantitatively and qualitatively `GrODID`'s ability to perform temporal super-resolution and backwards sequence forecasting. For this we use the Moving MNIST dataset.

The dimension of the observable space $\mathbf{g}^i_t : \Re^{\mathcal{M}}$, ranges from 15 to 30 per object. We use an appearance vector of size 46 for Moving MNIST and 26 for the Double Pendulum. The number of delayed coordinates per state $\mathbf{s}^i_t$, T_s, corresponds to the number of input frames.

Baselines. We choose our baselines to be established state-of-the-art methods for decomposed self-supervised video generation: DDPAE, G-SWM Lin et al. (2020), and STOVE Kossen et al. (2020). All baselines model dynamics with black-box neural modules.

Evaluation Metrics. Our quantitative results report per-frame metrics, including Mean Square Error (MSE), Mean Absolute Error (MAE), Structural Similarity (SSIM), and the Perceptual Similarity Metric (LPIPS) Zhang et al. (2018). The quantitative comparisons are meant to show that our performance is competitive while providing the benefits of being able to decompose, interpret, and manipulate the learned dynamics.

5.1 Double Pendulum Experiments

We propose to learn object-centric dynamic modes from video of a double pendulum in motion. A double pendulum exhibits a rich dynamic behavior with a strong sensitivity to initial conditions and noises in the environment, despite being a simple physical system. We employ the Double Pendulum Chaotic (DPC) dataset (Asseman et al. (2018)) to generate video samples.

In this dataset, 21 individual runs of a real double pendulum were recorded. Each of the recorded sequences lasts around 40s and consists of around 17,500 frames. Three pairs of marker positions (x, y) were extracted, representing the coordinates of the anchor, and the extremes of the arms. We constructed a dataset by locating three objects with different appearances at the coordinate locations provided by the DPC dataset (see Fig. 2).

For this experiment, we predict 10 frames from an input of 5, at a sample rate 5× lower than the original. Our setup encompasses approximately 280k frames for training, 70k for validation and 9k for testing. We train `GrODID` for 400 epochs.

Results. The correct modeling of the three moving objects dynamics is not possible by treating them as independent entities. Hence, `GrODID` must face challenges specific to this dataset. **i.** Assigning each scene object to a particular tracker. Not only across time frames, but across video instances. And **ii.** Learning the correct Koopman mappings so that both individual dynamics and interactions can be modeled linearly by $\mathcal{K}$.

Table 1 shows the quantitative evaluation of our method. `GrODID` outperforms the baselines except for G-SWM. While, in that case accuracy is worse but fairly similar [2], our approach has the benefit of interpretability bringing along the capability of increasing resolution.

Figure 2 illustrates our model's qualitative performance. We show 10 predictions. Objects look sharp and dynamics are correctly modeled. We can see how `GrODID` correctly identifies the objects, and generates them with high accuracy. `GrODID` and G-SWM predict correctly the trajectories and generate accurate frames, however, G-SWM models dynamics with Recurrent Neural Networks.

5.2 Time reversal and Super-Resolution Experiments

Similarly to Comas et al. (2023), in our experiments, we found that small eigenvalues have almost no effect on the dynamics. To handle unnecessary amplification during time reversal, we suppress the effect of inverting eigenvalues close to zero. Concretely, we avoid the effect of small eigenvalues by clipping them to a maximum module of 1.5 after applying the negative power to $\mathcal{K}$.

[2] Following indications from their paper, STOVE pre-processes and evaluates data in grayscale.

Table 1. Quantitative Double Pendulum results. We evaluate 10 frame prediction. **SR** for Super-Resolution. We evaluate `GrODID` predicting the sequence at the original sampling rate. Instead of 10, it predicts 54 frames.

Pendulum - Prediction	MSE ↓	MAE ↓	SSIM ↑	LPIPS ↓
DDPAE	171.72	339.02	0.85	0.119
STOVE	438.98	532.78	0.83	0.15
G-SWM	112.45	137.74	0.95	0.019
`GrODID`	120.82	162.44	0.95	0.026
`GrODID` **(SR)**	133.06	174.69	0.94	0.029

Figure 3 shows an example of our results trained on Moving MNIST dataset. It displays how we can increase the temporal resolution by a factor of 3 or 15 or reverse time prediction by manipulating $\mathcal{K}$. Note that the model has only been trained to predict in the forward direction at a specific time resolution. For the Double Pendulum experiments, we quantitatively evaluate how adding super-resolution hurts performance. As seen in Table 1, with `GrODID` (SR) we predict 54 frames at the original sampling rate. Here we interpolate the missing steps. Accuracy drops, but remains similar.

5.3 Conditioning the Koopman Operator

Some engineering decisions must be made to properly condition matrix $\mathcal{K}$. The operator is initialized to very small values, although we make sure that no eigenvalue is 0 so that the matrix can be safely inverted. We expect it to learn the needed eigenvalues (or poles) by optimization during training. Therefore, at the beginning of training, $\mathcal{K}$ will map any value of $\mathbf{g}$ to a vector of 0 (or very small values). Some of those eigenvalues will remain practically static during training, as they might not be needed to model the observed object dynamics.

Finally, having a negative real eigenvalue can also originate problems when applying an arbitrary power to $\mathcal{K}^{(1/m)}$. However, in principle only frequency components at the Nyquist sampling rate in the training data could lead to having an eigenvalue in such location. In our experiments we observe that our sampling rates are much higher than the maximum frequency in the dynamics. Hence, we assume that a negative real eigenvalue will be an artifact of training with no observable effect on the object dynamics, therefore we suppress its effect.

6. CONCLUSION

Many problems of practical interest require identifying the dynamics of interacting objects from time traces of relevant features. In this paper, we address this problem using a combination of Koopman operator, learned using an auto-encoder, and graph neural net ideas. The Koopman operator models individual, potentially non-linear dynamics, while the Graph Neural Net (GNN) captures the interactions between systems and the effects of the environment. The potential of this approach was illustrated with a non-trivial problem: given a video clip containing up to N interacting objects, determine the number of objects, the

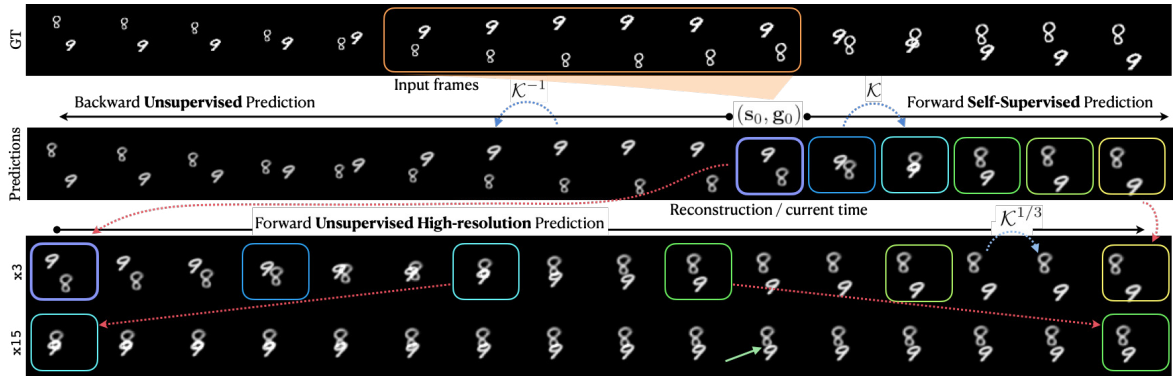

Fig. 3. Visualization of `GrODID`'s unsupervised temporal super-resolution and backward predictions. Top row: Ground Truth sequence, in orange we highlight the frames from which we estimate the current dynamic state $\mathbf{s}_0$. Middle row: Forward prediction (right) at the input sample rate; Backward unsupervised prediction (left) by applying the inverse Koopman operator $\mathcal{K}^{-1}$. Note that although the input frames are encoded into $\mathbf{s}_0$, $\mathbf{s}_{[-1:-5]}$ are also predictions from $\mathbf{s}_0$. Bottom rows: Unsupervised interpolation of the future frames. The green arrow indicates the exact instant where the objects in the predicted scene do not superpose anymore. Only with high temporal resolution not present in the input data we could find that instant.

individual dynamics and their interactions directly from the pixels. As shown in the paper, the proposed Koopman based architecture successfully accomplishes these tasks, allowing for forward and backwards prediction. In addition, by manipulating the eigenvalues of the operator, the proposed approach can also achieve super-resolution, assuming that the sequence is sampled fast enough.

REFERENCES

Asseman, A., Kornuta, T., and Ozcan, A. (2018). Learning beyond simulated physics. In *Modeling and Decision-making in the Spatiotemporal Domain Workshop*.

Azencot, O., Erichson, N.B., Lin, V., and Mahoney, M. (2020). Forecasting sequential data using consistent koopman autoencoders. In *ICML*.

Battaglia, P., Pascanu, R., Lai, M., Jimenez Rezende, D., and kavukcuoglu, k. (2016). Interaction networks for learning about objects, relations and physics. In *NeurIPS*.

Comas, A., Lopez, C.F., Ghimire, S., Li, H., Sznaier, M., and Camps, O. (2023). Learning object-centric dynamic modes from video and emerging properties. In *L4DC*, 745–769.

He, Z., Li, J., Liu, D., He, H., and Barber, D. (2019). Tracking by animation: Unsupervised learning of multi-object attentive trackers. *CVPR*, 1318–1327.

Kingma, D.P. and Welling, M. (2013). Auto-encoding variational Bayes. *CoRR*, abs/1312.6114.

Koopman, B.O. (1931). Hamiltonian systems and transformation in hilbert space. *Proceedings of the National Academy of Sciences*, 17 5, 315–8.

Kossen, J., Stelzner, K., Hussing, M., Voelcker, C., and Kersting, K. (2020). Structured object-aware physics prediction for video modeling and planning. In *ICLR*.

Li, Y., He, H., Wu, J., Katabi, D., and Torralba, A. (2020). Learning compositional Koopman operators for model-based control. In *ICLR*.

Lin, Z., Wu, Y.F., Peri, S.V., Fu, B., Jiang, J., and Ahn, S. (2020). Improving generative imagination in object-centric world models. In *ICML*.

Lusch, B., Kutz, J.N., and Brunton, S. (2018). Deep learning for universal linear embeddings of nonlinear dynamics. *Nature Communications*, 9.

Mezic, I. (2005). Spectral properties of dynamical systems, model reduction and decompositions. *Nonlinear Dynamics*, 41, 309–325.

Morton, J., Witherden, F., and Kochenderfer, M.J. (2019). Deep variational koopman models: Inferring koopman observations for uncertainty-aware dynamics modeling and control. In *IJCAI*.

Schmid, P. (2008). Dynamic mode decomposition of numerical and experimental data. *J. of Fluid Mechanics*, 656, 5–28.

Wang, Y., Sznaier, M., and Camps, O. (2016). A super-atomic norm minimization approach to identifying sparse dynamical graphical models. In *ACC*, 1962–1967.

Williams, M., Kevrekidis, I., and Rowley, C. (2015). A data–driven approximation of the koopman operator: Extending dynamic mode decomposition. *J. of Nonlinear Science*, 25, 1307–1346.

Xiao, Y., Xu, X., and Lin, Q. (2021). Cknet: A convolutional neural network based on koopman operator for modeling latent dynamics from pixels. *ArXiv*.

Zhang, R., Isola, P., Efros, A.A., Shechtman, E., and Wang, O. (2018). The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*.