Nonlinear Dyn (2025) 113:10881–10909
https://doi.org/10.1007/s11071-024-10687-8

RESEARCH

Modeling nonlinear dynamics from videos

Antony Yang · Joar Axås · Fanni Kádár · Gábor Stépán · George Haller

Received: 18 September 2024 / Accepted: 18 November 2024 / Published online: 12 December 2024

Abstract We introduce a method for constructing reduced-order models directly from videos of dynamical systems. The method uses non-intrusive tracking to isolate the motion of a user-selected part in the video of an autonomous dynamical system. In the space of delayed observations of this motion, we reconstruct a low-dimensional attracting spectral submanifold (SSM) whose internal dynamics serves as a mathematically justified reduced-order model for nearby motions of the full system. We obtain this model in a simple polynomial form that allows explicit identification of important physical system parameters, such as natural frequencies, linear and nonlinear damping and nonlinear stiffness. Beyond faithfully reproducing attracting steady states and limit cycles, our SSM-reduced models can also uncover hidden motion not seen in the video, such as unstable fixed points and unstable limit cycles forming basin boundaries. We demonstrate all these features on experimental videos of five physical systems: a double pendulum, an inverted flag in counter-flow, water sloshing in tank, a wing exhibiting aeroelastic flutter and a shimmying wheel.

Keywords Object tracking · Data-driven dynamics · Computer vision · Reduced-order modeling · Spectral submanifolds

A. Yang · J. Axås · G. Haller (✉)
Institute for Mechanical Systems, ETH Zürich, Leonhardstrasse 21, 8092 Zürich, Switzerland
e-mail: georgehaller@ethz.ch

A. Yang
e-mail: thy25@cam.ac.uk

J. Axås
e-mail: jgoeransson@ethz.ch

A. Yang
Department of Applied Mathematics and Theoretical Physics, Centre for Mathematical Sciences, University of Cambridge, Wilberforce Road, Cambridge CB3 0WA, UK

F. Kádár · G. Stépán
Department of Applied Mechanics, Faculty of Mechanical Engineering, Budapest University of Technology and Economics, Műegyetem rkp. 3, Budapest 1111, Hungary
e-mail: fanni.kadar@mm.bme.hu

G. Stépán
e-mail: stepan@mm.bme.hu

MTA-BME Lendület Momentum Global Dynamics Research Group, Budapest University of Technology and Economics, Műegyetem rkp. 3, Budapest 1111, Hungary

1 Introduction

Nonlinear dynamical systems are prevalent in numerous fields of nature and engineering. Examples include sloshing in tank trucks [1], turbulent flows [2], spacecraft motion [3], and vibrations in jointed structures [4]. Full-order modeling of such systems is often challenging due to their high degrees of freedom and uncertain physical parameters. For these reasons, reduced-order modeling of nonlinear mechanical systems has been an increasingly active area of research that promises major benefits in system identification [5], design opti-

mization [6] and model-predictive control [7]. A further recent boost to this effort is the trend is to construct digital twins, i.e., highly accurate, interpretable and predictive models of the current states of physical assets [8].

A common approach to reducing nonlinear dynamical systems to lower-dimensional models is the proper orthogonal decomposition (POD) followed by a Galerkin projection [9,10]. This approach implicitly assumes that model subspaces of the linearized system remain nearly invariant under inclusion of nonlinearities, which is a priori unknown and can often be secured only by selecting an unnecessarily large number of linear modes. Another popular model reduction technique is the Dynamic Mode Decomposition (DMD) and its variants [11,12], which find the best fitting linear autonomous dynamical system for the available observable data. Passing to the dominant modes of this linear system then provides a linear reduced-order model of the full dynamics. While very efficient in capturing linearizable dynamics, DMD methods always return linear systems and hence cannot model intrinsically nonlinear phenomena, such as coexisting isolated attracting fixed points, limit cycles or transitions between such states [13,14].

Machine learning methods based on the training of neural networks [15,16] have also been explored as data-driven model reduction alternatives, but they often lack physical interpretability, are prone to overfitting and require large amounts of data and extensive tuning. Within the category of machine learning, sparse identification of nonlinear dynamics (SINDy) [17] can provide interpretable models but only if an appropriate reduced set of variables is already known. Even in that case, however, the outcome of the process is generally sensitive to the choice of sparsification parameters.

In recent years, reduction to spectral submanifolds (SSMs) has appeared as a new alternative for constructing reduced-order models for nonlinear dynamical systems from data [18–21]. A primary SSM of a dynamical system is the unique smoothest invariant manifold tangent to a nonresonant spectral subspace of the linearized system at a steady state. Such manifolds have long been envisioned and formally approximated as nonlinear normal modes (NNMs) in a series of papers initiated by the seminal work of Shaw and Pierre [22] in the 1990's. The existence, uniqueness and smoothness of such manifolds, however, has only been clarified more recently for various types of steady states [23–26] and general external forcing [27]. Importantly, the internal dynamics of attracting SSMs tangent to a span of slowest eigenmodes provide a mathematically exact reduced-order model with which all nearby trajectories synchronize exponentially fast.

Analytic and data-driven tools, available as open-source MATLAB codes, have been developed for constructing SSM-based reduced-order models. For analytic treatments, the *SSMTool* [28] package computes SSMs directly from governing equations, whereas the data-driven methods *SSMLearn* [19] and *fastSSM* [21] construct models from time-series data. These SSM-reduction algorithms have been successfully applied to several nonlinear systems known from numerical or experimental data. Examples include flow past a cylinder [19], a Brake-Reuss beam [20], water tank sloshing [21], transition to turbulence in pipe flows [29] and finite element models of various structures going beyond a million degrees of freedom [28,30].

While all these studies demonstrate the applicability and robustness of SSM models inferred from data, the scarce availability of experimental measurements in many physical settings motivates the extension of SSM-based model reduction to general video data. Such an extension should ideally work with generic video footage, such as an excerpt from a documentary or instructional video, that was not necessarily generated in a sterile environment for the sole purpose of model reduction. An additional benefit of a purely video-based model reduction would be its nonintrusive nature. This is especially important for slender structures where an attached sensor would alter the behavior of the system. In other cases, video-based system modeling may be the only viable option because the placement of a reliable physical sensor is unrealistic due to extreme temperatures, pressure or humidity.

While video-based analysis has been widely used in various engineering and scientific disciplines, such as robotics [31], medical imaging [32], manufacturing [33], autonomous driving [34], structural health monitoring [35], fluid mechanics (particle image velocimetry) [36], and unsupervised physical scene understanding [37], their application in reduced-order modeling of nonlinear dynamical systems has remained largely unexplored. Indeed, video-based modeling has been mostly tried for identifying parameters in systems whose governing equations are already known (see [38] for a review). The same reference proposes a method to infer nonlinear dynamics by training neural networks

on videos generated by solutions of simple, polynomial ODEs [38]. So far, however, no applications of these methods to videos of real physical experiments have appeared.

A major challenge in video-based reduced-order modeling is the visual tracking of physical systems, which is complicated by changes in object motion, cluttered backgrounds, occlusion, and changes in target appearance [39–41]. Generic tracking, also known as short-term or model-free tracking, involves continuously localizing a target in a video sequence using a single example of its appearance, usually initialized in the first frame. While nearly all existing tracking algorithms focus on object localization, our goal here is to use visual tracking as an experimental sensing method for dynamical systems. In other words, we are interested in the accurate recovery of trajectories rather than locating objects in videos.

The tracking algorithms developed in recent decades, such as correlation-based methods [42–46] and convolutional neural network-based (CNN) methods [47–51], are founded on the assumption that the appearance of the tracked object changes over time, requiring a process that learns such changes. The main objective of these methods is to continuously update the model of the tracked object, learning its evolving shape and size, to ensure its robust and accurate localization. However, due to the changing nature of the object's representation, the tracked features are not constant observables on the phase space, and are therefore typically unsuitable for system identification. In response to the challenge of precise trajectory extraction from videos for dynamical problems, Kara et al. [52] proposed a tracking algorithm based on deep learning, addressing the identity switching problem encountered in particle tracking scenarios such as walking droplets and granular intruders experiments. Their approach, however, relies on a machine learning model that necessitates manual labeling of data for training the tracker.

In this work, we introduce a tracking algorithm that prioritizes trajectory recovery to assemble data for SSM-based model reduction from videos of physical systems. The tracking method is based on the classic template matching technique [53], which achieves marker-free and rotation-aware tracking that requires only a one-time initialization from the user. With the trajectories obtained from our proposed tracker, we train SSM-reduced polynomial models, employing the open-source MATLAB package *SSMLearn* [19]. The main steps of this procedure are summarized in Fig. 1. We illustrate on several examples that the methodology developed here extracts simple, accurate and predictive polynomial ODE models from videos of a diverse set of physical systems. These include a double pendulum, an elastic flag in counter flow, water sloshing in a moving tank, aeroelastic flutter of a tail fin and wheel shimmy.

2 Marker-free tracking with template matching

In this section, we describe our model-free tracking algorithm used for recovering trajectory data from experimental videos, which we have implemented in Python using the open-source OpenCV framework [55]. Figure 2 illustrates the flowchart of this tracking algorithm which we also summarize later in Algorithm 1.

2.1 Template matching and tracking algorithm

Template matching has been actively developed in the computer vision community for the last few decades [56–64] with applications to manufacturing [65,66], medical imaging [67,68], geoinformatics [69], and general object tracking [70–75]. The technique identifies instances of a predefined template image within a larger target image by comparing pixel-by-pixel similarity [53]. This is achieved by computing match scores, typically a cross-correlation or sum of square errors, from the intensity values of the template and those of small patches of the target image with the same pixel dimensions. The optimal match is selected as the location with the best match score. This approach is applicable under our assumption that the appearance and size of the object to be modeled remain constant over time. The matching procedure is summarized in lines 5-14 in Algorithm 1.

More specifically, the template matching process starts by selecting a template, a rectangular sub-image, from a reference video frame $\boldsymbol{I}_{\text{reference}}(x, y) \in \mathbb{R}^c$, where c is the number of channels. For a gray-scale image $c = 1$ and RGB image $c = 3$. We denote the i-th channel of an image with subscript $(\cdot)_i$ (i.e. RGB image $\boldsymbol{I} = (I_1, I_2, I_3)$). The reference video frame is usually the first frame of the video. Let $\boldsymbol{T}(x_t, y_t) \in \mathbb{R}^c$ represent the template image with a separate set of coordinates (x_t, y_t) and $\boldsymbol{I}(x, y) \in \mathbb{R}^c$ represent a target frame

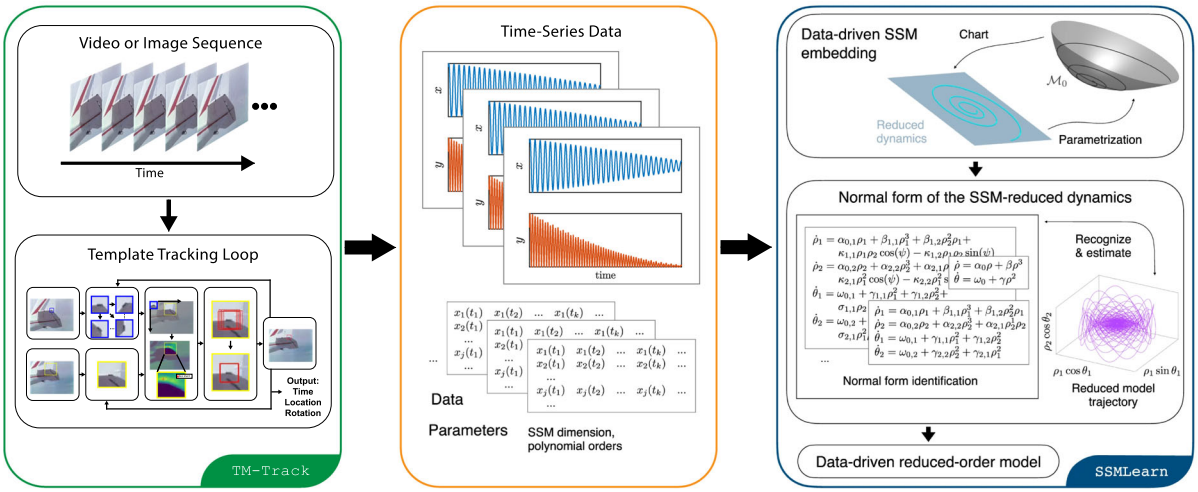

Fig. 1 Schematic of our Tracking and Modeling Approach—(green) First, we prepare input videos or image sequences that capture transient phenomena. We select a template on the video to track. The tracking algorithm outputs the (x, y) pixel and template rotation as time-series data. (orange) Next, the data is preprocessed with possible noise reduction and delay-embedded to produce a suitable observable space. (blue) Finally, we train a reduced-order model with the *SSMLearn* algorithm [19] in the delay-embedded space to make predictions for previously unseen initial conditions or to predict behavior under additional external forcing. (Color figure online)

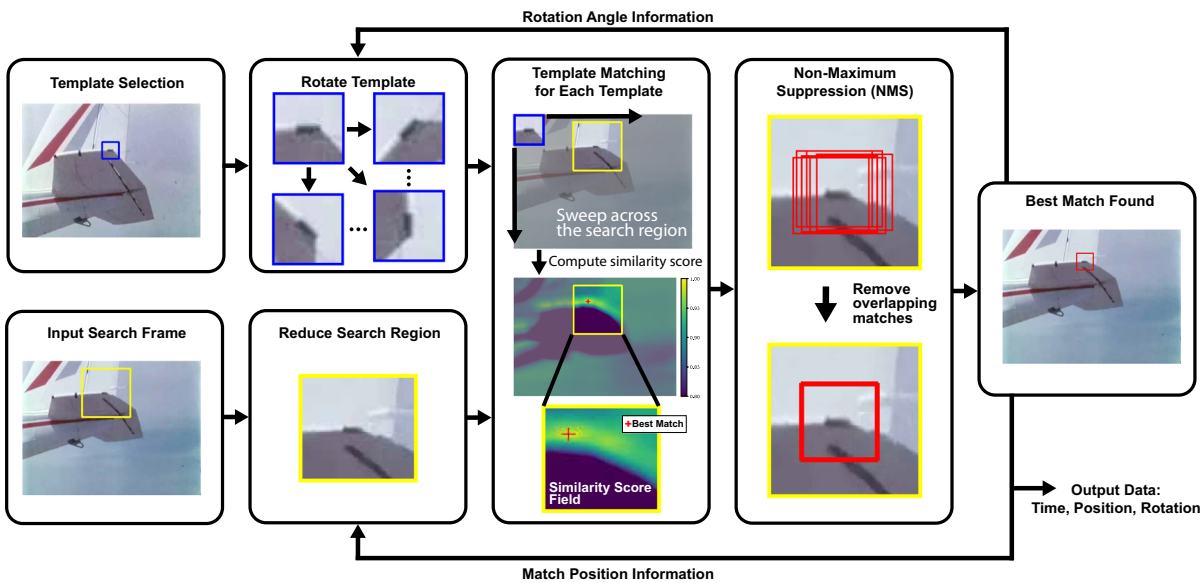

Fig. 2 Schematic of Template-Matching Tracking Algorithm—We initialize one or multiple templates whose centers correspond to the points to track. This initialization is only done in the first frame of the video. On the next frame of the video, we crop out a smaller search region to look for the template we have selected from the initial frame. The template matching algorithm is then applied for each template rotated at fixed intervals, giving optimal match locations for each. Note that the similarity score field in the figure shows $1 - \boldsymbol{R}$ for better visualization. The red cross indicates the location of the optimal match. If multiple good matches overlap each other, we remove the redundant matches with the Non-Maximum Suppression algorithm [54]. With the best match found at the target frame, the match location and rotational angle can be used to update the search region as well as the range of template rotations for the subsequent frames. The output from each iteration is collected as time-series data. (Color figure online)

where we search for the template. The dimensions, in pixel widths and heights, of the template $\boldsymbol{T}$ need to be strictly smaller than the target image $\boldsymbol{I}$. If a background-removed image mask was used, then $T \in \mathbb{B}$ and $I \in \mathbb{B}$, where $\mathbb{B}$ represents the set of Boolean values $\{0, 1\}$.

We use image intensity as the matching quantity and the Normalized Sum of Square Difference (NSSD) as the similarity metric [56]. By computing a similarity score between the template $\boldsymbol{T}$ and all locations on $\boldsymbol{I}$ with the NSSD, we obtain a similarity function $R \in \mathbb{R}$ from

$$R(x, y) = r(\boldsymbol{T}, \boldsymbol{I}) = \frac{\sum_{x_t, y_t, i} \left(T_i(x_t, y_t) - I_i(x_t + x, y_t + y)\right)^2}{\sqrt{\sum_{x_t, y_t, i} T_i(x_t, y_t)^2 \sum_{x_t, y_t, i} I_i(x_t + x, y_t + y)^2}}, \tag{1}$$

where r is an operator that maps a pair of template and target image to a similarity function R. We note that $R(x, y)$ is a scalar function as each summation in (1) is done over all c channels. This process is akin to applying a convolution filter on the target image, where a small template image is swept across the target image and a metric is computed for each location.

The best similarity score (i.e., the smallest value of $R(x, y)$) indicates the optimal match between the template and the search image. We extend this procedure to account for object rotations by computing similarity functions for templates rotated at fixed intervals and finding the minimum NSSD across all the similarity functions with the Non-Maximum Suppression (NMS) algorithm (described in subsection 2.3). By repeating the matching for all subsequent frames in the video, the algorithm achieves accurate motion tracking over time.

To improve computational efficiency, we utilize the previously detected location and angle to narrow the search region and angle sweep of the next frame. Thus, instead of processing an entire frame, we search a smaller sub-image $\boldsymbol{S}(x, y) \in \mathbb{R}^c$ of $\boldsymbol{I}(x, y)$ (see line 4 of Algorithm 1). By assuming the motion in the video data exhibits spatial and temporal coherence, we limit the search area to a window surrounding the previously matched location and rotation range. The size of the search window and the range of rotations must exceed the maximum displacement and rotation observed in the tracked object throughout the video. Therefore, having prior knowledge of the system and video quality (such as frame rate) can inform the selection of these parameters. Based on the examples we presented in this work, a search region of 1 to 2 times the pixel length of the template dimensions and a rotation range within $\pm 15°$ degrees about previously matched template rotation with an interval of 5° is a good starting point.

2.2 Background removal through frame averaging

A helpful preprocessing step for motion extraction involves separating foreground objects from the background. A simple yet effective technique is background subtraction through frame averaging. This preprocessing step is applied to the double pendulum and inverted flag examples presented in Sect. 4. The technique corresponds to lines 1-4 in Algorithm 1.

Frame averaging seeks to extract the foreground objects from a sequence of images or video by subtracting the static background, obtained from time-averaging image intensities from each video frame. This method works under the assumption that the background in a video sequence tends to remain constant or have minimal changes over time, while the foreground objects introduce significant variations in terms of image intensities.

The process of frame averaging for an RGB-colored video $\boldsymbol{I} \in \mathbb{R}^3$ is represented by

$$\boldsymbol{I}_{\text{mean}}(\boldsymbol{x}) = \frac{1}{N} \sum_t \boldsymbol{I}(\boldsymbol{x}, t), \tag{2}$$

where $\boldsymbol{x} = (x, y)$, t is the time corresponding to each frame, and N is the total number of frames in the video. This operation is allowed as the RGB color space is a linear additive space.

The average frame $\boldsymbol{I}_{\text{mean}}$ is subtracted from each individual frame in the video sequence. The result is a difference image $\boldsymbol{D} \in \mathbb{R}^3$,

$$\boldsymbol{D}(\boldsymbol{x}, t) = |\boldsymbol{I}(\boldsymbol{x}, t) - \boldsymbol{I}_{\text{mean}}(\boldsymbol{x})|, \tag{3}$$

that highlights the foreground objects. The larger the magnitude of the difference, the more likely the pixel is in the foreground.

To refine the foreground-background segmentation, we take the maximum value of the c channels of the difference frame, that is

$$D_{\max}(\boldsymbol{x}, t) = \max_{1 \le i \le c} D_i(\boldsymbol{x}, t). \tag{4}$$

By thresholding $D_{\max}$ as

$$D_{\text{mask}}(\boldsymbol{x}, t) = h(D_{\max}, D_{\text{thresh}}) = \begin{cases} 1, & D_{\text{thresh}} \leq D_{\max}(\boldsymbol{x}, t) \\ 0, & \text{otherwise} \end{cases}, \quad (5)$$

we obtain a boolean mask that separates the foreground from the background. The mask contains foreground shape information and we can track features directly in this binary image.

2.3 Non-maximum suppression

To remove redundant bounding boxes or detections, we use the non-maximum suppression (NMS) algorithm [54]. It involves sorting the bounding boxes based on their confidence scores, selecting the box with the highest score, and suppressing other boxes with high overlap. This process ensures that only the most accurate and non-overlapping detections are retained. NMS is widely employed in object detection algorithms to improve accuracy by eliminating duplicate detections. As it is expected that our problems have multiple overlapping close matches from different rotated templates, we use the NMS algorithm to eliminate the ambiguity systematically. In all the examples presented in this work, we keep only the global best match.

The overlap of the bounding boxes is measured with the Jaccard index [76], which is more commonly known as the Intersection over Union (IoU) metric in computer vision, defined as

$$J(A, B) = \left| \frac{A \cap B}{A \cup B} \right| = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}, \quad (6)$$

where $|\cdot|$ indicates the size of set, and $J(A, B)$ has a range of [0, 1] by design.

The NMS algorithm corresponds to lines 15-30 in Algorithm 1 and is summarized in the following steps:

1. Obtain a list of detection boxes $\mathcal{L}$ with a corresponding list of confidence scores $\mathcal{R}$ and rotations $\mathcal{T}$.
2. Find the detection, $\mathcal{L}_m$ with the best similarity score (minimum of $\mathcal{R}$) from set $\mathcal{L}$.
3. Remove the detection $\mathcal{L}_m$ from the set $\mathcal{L}$ and appends it to the set of final output detections $\mathcal{K}$. Repeat this for match scores $\mathcal{R}$ and rotations $\mathcal{T}$.
4. Using the Jaccard index (6) as the overlap metric, remove all detection boxes that have an overlap with $\mathcal{L}_m$ greater than a threshold $\text{IoU}_{\text{thresh}}$ in the set $\mathcal{L}$. Repeat this for match scores $\mathcal{R}$ and rotations $\mathcal{T}$.
5. Repeat step 2-4 until either $\mathcal{L}$ is empty or the size of $\mathcal{K}$ reaches the required number of non-intersecting matches n_{matches}.

As we track a single point in all five video examples, we set $n_{\text{matches}} = 1$. We remark that our proposed tracking algorithm with NMS can potentially track multiple identical objects, however this is not pursued in this paper. Lastly, we find an overlap threshold of $\text{IoU}_{\text{thresh}} = 0.3$ to be effective across all videos.

2.4 Summary

Our template-matching-based tracking algorithm is summarized in Algorithm 1. The tracking algorithm has three main steps: 1. Optional background subtraction pre-processing, 2. Generate candidate templates and apply template matching, 3. Apply non-maximum suppression to obtain optimal estimate of object location. We apply the algorithm to every video frame and collect the output position and rotation of the tracked template as time-series data.

The inputs to the algorithm/tracking routine are:

1. user-selected template in the form of a small patch on the first frame of a video,
2. target frame to search for matches,
3. optional averaged frame to use for background subtraction,
4. threshold value for maximum difference image to refine background removal,
5. a region on the target frame to search for matches which is updated at every frame based on the previous match location,
6. intersection over union threshold to remove the overlapping bounding box from matches,
7. angle sweep bounds and intervals which are updated at every frame based on the rotation previous template match,
8. number of non-overlapping best matches to search for.

In practice, the user only needs to initialize the tracker by selecting a region, to be tracked, in the first frame. The tracking algorithm is evaluated at every frame of the video, outputting the (x, y) locations of

the best matches as well as the corresponding rotation angle of the template. The algorithm then uses the match location information to update the search region and the angle sweep bounds for the next frame to speed up computation. We find updating the search region to a region within 2 times the pixel length of the template centering the previously matched (x, y) location and $\pm 15^\circ$ about the previously matched rotation angle with an interval of 5° work well for the examples presented in this work.

3 Data-driven reduced-order models on spectral submanifolds

Here we summarize available mathematical results on model reduction to SSMs in smooth nonlinear systems and the *SSMLearn* algorithm used in constructing SSM-reduced models from data.

3.1 Problem setup

Consider an n-dimensional dynamical system of the form

$$\begin{aligned}&\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{f}(\boldsymbol{x}),\\ &\boldsymbol{x} \in \mathbb{R}^n,\ \boldsymbol{A} \in \mathbb{R}^{n\times n},\\ &\boldsymbol{f} : \mathbb{R}^n \mapsto \mathbb{R}^n,\ \boldsymbol{f} \sim \mathcal{O}(|\boldsymbol{x}|^2),\ \boldsymbol{f}(\boldsymbol{0}) = \boldsymbol{0},\end{aligned} \tag{7}$$

where $\boldsymbol{f}$ is of differentiability class C^r in $\boldsymbol{x}$ and $\boldsymbol{A}$ is the linear part of the system.

For simplicity of exposition, we assume that $\boldsymbol{A}$ is diagonalizable and $\boldsymbol{x} = \boldsymbol{0}$ is an asymptotically stable fixed point, that is

$$\operatorname{Re}\lambda_k < 0 \quad \forall \lambda_k \in \operatorname{Spect}(\boldsymbol{A}), \tag{8}$$

where $\operatorname{Spect}(\boldsymbol{A}) = \{\lambda_1, \lambda_2, ..., \lambda_n\}$ denotes the eigenvalues of the linear part of the system at the origin. Recent work extended this result from stable to general hyperbolic fixed points with a possible mix of stable and unstable eigenvalues [26].

We select a d-dimensional spectral subspace $E \subset \mathbb{R}^n$, that is, the direct sum of a set of eigenspaces of $\boldsymbol{A}$. We denote the spectrum of $\boldsymbol{A}$ within E by $\operatorname{Spect}(\boldsymbol{A}|_E)$. If the spectral subspace E is non-resonant with the spectrum of $\boldsymbol{A}$ outside of E, i.e.,

$$\begin{aligned}&\lambda_j \neq \sum_{k=1}^{d} m_k\lambda_k, \quad m_k \in \mathbb{N}, \quad \sum_{k=1}^{d} m_k \geq 2,\\ &\lambda_j \in \operatorname{Spect}(\boldsymbol{A})\backslash\operatorname{Spect}(\boldsymbol{A}|_E), \quad \lambda_k \in \operatorname{Spect}(\boldsymbol{A}|_E),\end{aligned} \tag{9}$$

then E generally has infinitely many nonlinear continuations, as invariant manifolds of dimension d emanating from $\boldsymbol{x} = \boldsymbol{0}$ and tangent to E, in the system (7) [25]. Within this family of invariant manifolds, there is a unique smoothest member, which we define as the primary spectral submanifold. This SSM is normally attracting, can be approximated via a Taylor expansion at the fixed point, and is therefore an ideal candidate for nonlinear model reduction. SSMs can be efficiently computed from the equations of motion using the open-source package *SSMTool* [28].

3.2 SSM-based data-driven modeling algorithm

When the equations of motion are unknown, model reduction can also be applied directly to time-dependent observable data describing the evolution of the dynamical system. Recently, SSM theory has been applied to both simulated and experimental data to capture essential nonlinear dynamics and enable accurate predictions for motions not used in the training or for behavior under additional external forcing [18,19,77,78]. Such data-driven models can even outperform analytical models as they are not necessarily bound to the domain of convergence of the Taylor expansions used for the SSMs and their reduced dynamics [21].

To construct SSM-reduced models from data, we use the methodology presented in [19], which is implemented in the open-source MATLAB package *SSMLearn*. The method consists of two main steps: geometry identification and reduced dynamics modeling. As the low-dimensional SSM attracts nearby trajectories, we approximate it as a polynomial using nearby transient training data in the phase space or observable space. We then model the reduced dynamics by identifying a vector field or map from the high-dimensional training trajectories projected onto the SSM local coordinates. Here we provide a summary of the *SSMLearn* algorithm; a complete description can be found in [19].

We start with the geometry identification. To obtain a graph-style model of a d-dimensional SSM from data,

Algorithm 1 Template Matching-Based Tracking Algorithm

Input:
$\boldsymbol{T}(x_t, y_t)$, template image
$\boldsymbol{I}(x, y)$, target frame
$\boldsymbol{I}_{\text{mean}}(x, y)$, background image from frame average
D_{thresh}, difference image threshold value
search_region, region on target frame for matching
$\text{IoU}_{\text{thresh}}$, Intersection over Union (overlap) threshold
$\theta_{\min}, \theta_{\max}, \theta_{\text{interval}}$, angle sweep bounds and interval
n_{match}, number of best matches
Output:
$\mathcal{K}$, list of top left (x, y) coordinates of found matches
$\mathcal{T}$, list of rotation of the found matches

/* preprocess by removing background */
$\boldsymbol{D} \leftarrow |\boldsymbol{I} - \boldsymbol{I}_{\text{mean}}|$
$D_{\max} \leftarrow \max_{1 \le i \le 3} D_i$
$D_{\text{mask}} \leftarrow h(D_{\max}, D_{\text{thresh}})$ ▷ threshold
$\boldsymbol{S} \leftarrow$ crop $\boldsymbol{I}$ or D_{mask} at *search_region*
/* template matching */
$\mathcal{L} \leftarrow \{\}$ ▷ match position array
$\mathcal{R} \leftarrow \{\}$ ▷ match score array
$\mathcal{T} \leftarrow \{\}$ ▷ match angle array
for $\theta \in \{x : \theta_{\min} \le x \le \theta_{\max},\ \frac{x-\theta_{\min}}{\theta_{\text{interval}}} \in \mathbb{Z}\}$ **do**
  rotate $\boldsymbol{T}$ by θ
  $\boldsymbol{R} \leftarrow r(\boldsymbol{T}, \boldsymbol{S})$ ▷ template match
  $\mathcal{L} \leftarrow \mathcal{L} \cup \{(x, y) \mid \boldsymbol{R}(x, y) < 0.5\}$
  $\mathcal{R} \leftarrow \mathcal{R} \cup \{(x, y) \mid \boldsymbol{R}(x, y) < 0.5\}$
  $n \leftarrow |\{(x, y) \mid \boldsymbol{R}(x, y) < 0.5\}|$ ▷ number of potential good matches
  $\mathcal{T} \leftarrow \mathcal{T} \cup \{\theta\}_n$
/* Non Maximum Suppression */
  $\mathcal{K} \leftarrow \{\}$ ▷ match position array for output
  while $\mathcal{L} \neq \emptyset$ and $|\mathcal{K}| < n_{\text{match}}$ **do**
    $m \leftarrow \text{argmin}_j \mathcal{R}_j$
    $\mathcal{K} \leftarrow \mathcal{K} \cup \mathcal{L}_m$
    $\mathcal{L} \leftarrow \mathcal{L} - \mathcal{L}_m$
    $\mathcal{R} \leftarrow \mathcal{R} - \mathcal{R}_m$
    $\mathcal{T} \leftarrow \mathcal{T} - \mathcal{T}_m$
    for $\mathcal{L}_i \in \mathcal{L}$ **do** ▷ remove overlapping matches
      if $J(\mathcal{L}_m, \mathcal{L}_i) \ge \text{IoU}_{\text{thresh}}$ **then**
        $\mathcal{L} \leftarrow \mathcal{L} - \mathcal{L}_i$
        $\mathcal{R} \leftarrow \mathcal{R} - \mathcal{R}_i$
        $\mathcal{T} \leftarrow \mathcal{T} - \mathcal{T}_i$
      end if
    end for
  end while
end for
return $\mathcal{K}, \mathcal{T}$

we construct the SSM as a polynomial parameterized by local coordinates on its tangent space. To this end, we define a matrix $\boldsymbol{V} \in \mathbb{R}^{n \times d}$, whose columns are orthonormal vectors spanning the tangent space of the yet unknown SSM. The reduced coordinates $\boldsymbol{\xi}(t) \in \mathbb{R}^d$ are defined as a projection of the trajectories $\boldsymbol{y}(t) \in \mathbb{R}^n$ onto the tangent space $\boldsymbol{V}$, that is

$$\boldsymbol{\xi} = \boldsymbol{V}^\top \boldsymbol{y}. \tag{10}$$

We seek to approximate and parameterize the manifold with a Taylor expansion in $\boldsymbol{\xi}$ about the fixed point at the origin, $\boldsymbol{y} = \boldsymbol{0}$, in the form

$$\boldsymbol{y} \approx \boldsymbol{M}\boldsymbol{\xi}^{1:m} = \boldsymbol{V}\boldsymbol{\xi} + \boldsymbol{M}_{2:m}\boldsymbol{\xi}^{2:m} = \boldsymbol{v}(\boldsymbol{\xi}). \tag{11}$$

Here $\boldsymbol{M} \in \mathbb{R}^{n \times d_{1:m}}$ is a matrix of manifold parameterization coefficients, with $d_{1:m}$ denoting the number of d-variate monomials from orders 1 up to m. The notation $(\cdot)^{l:m}$ refers to a vector of all monomials at

orders l through m. For example, if $\boldsymbol{x} = [x_1, x_2]^\top$, then $\boldsymbol{x}^{2:3} = [x_1^2, x_1x_2, x_2^2, x_1^3, x_1^2x_2, x_1x_2^2, x_2^3]^\top$. $\boldsymbol{M}_{2:m}$ denotes a submatrix of $\boldsymbol{M}$ that has only columns associated with $(\cdot)^{2:m}$ vector, so $\boldsymbol{M}_{2:m} \in \mathbb{R}^{n\times d_{2:m}}$.

The matrices $\boldsymbol{V}$ and $\boldsymbol{M}$ are solved for simultaneously by minimizing the cost function

$$(\boldsymbol{V},\ \boldsymbol{M}) = \underset{(\boldsymbol{V}^*,\ \boldsymbol{M}^*)}{\operatorname{argmin}} \sum_{j=1}^{N} \|\boldsymbol{y}_j - \boldsymbol{M}^*(\underbrace{\boldsymbol{V}^{*\top}\boldsymbol{y}_j}_{\boldsymbol{\xi}^*})^{1:m}\|^2, \tag{12}$$

subject to the constraints

$$\boldsymbol{V}^\top\boldsymbol{V} = \boldsymbol{I}, \qquad \boldsymbol{V}^\top\boldsymbol{M}_{2:m} = \boldsymbol{0}. \tag{13}$$

In the second step, after we have found the parameterization of the manifold, we approximate the reduced dynamics up to r-th order on the manifold as

$$\dot{\boldsymbol{\xi}} \approx \boldsymbol{R}\boldsymbol{\xi}^{1:r} = \boldsymbol{r}(\boldsymbol{\xi}), \tag{14}$$

where $\boldsymbol{R} \in \mathbb{R}^{d\times d_{1:r}}$ is a matrix of coefficients of the reduced dynamics. We solve for the matrix $\boldsymbol{R}$ through minimization of the cost function

$$\begin{aligned}\boldsymbol{R} &= \underset{\boldsymbol{R}^*}{\operatorname{argmin}} \sum_{j=1}^{N} \|\dot{\boldsymbol{\xi}}_j - \boldsymbol{R}^*\boldsymbol{\xi}_i^{1:r}\|^2 \\ &= \underset{\boldsymbol{R}^*}{\operatorname{argmin}} \sum_{j=1}^{N} \left\| \frac{d}{dt}(\boldsymbol{V}^\top\boldsymbol{y}_j) - \boldsymbol{R}^*(\boldsymbol{V}^\top\boldsymbol{y}_j)^{1:r} \right\|^2.\end{aligned} \tag{15}$$

We then transform the reduced dynamics to its normal form, which is the simplest complex polynomial form that preserves the local trajectory structure up to a smooth, near-identity deformation with a smooth inverse. Transforming the equations to a normal form therefore provides insights into the qualitative behaviors of the system, such as stability properties, fixed points, limit cycles, and bifurcations [79].

We compute the n-th order normal form of $\boldsymbol{R}$ to obtain a near-identity polynomial transformation of $\boldsymbol{\xi}$ to complex conjugate normal form coordinates $\boldsymbol{z} \in \mathbb{C}^d$. We define the transformations $\boldsymbol{t} : \boldsymbol{z} \mapsto \boldsymbol{\xi}$, its inverse $\boldsymbol{t}^{-1} : \boldsymbol{\xi} \mapsto \boldsymbol{z}$, and the dynamics in normal form $\boldsymbol{n} : \boldsymbol{z} \mapsto \dot{\boldsymbol{z}}$ as

$$\boldsymbol{\xi} = \boldsymbol{t}(\boldsymbol{z}, \boldsymbol{T}) = \boldsymbol{T}\boldsymbol{z}^{1:n} = \boldsymbol{W}\boldsymbol{z} + \boldsymbol{T}_{2:n}\boldsymbol{z}^{2:n}, \tag{16}$$

$$\boldsymbol{z} = \boldsymbol{t}^{-1}(\boldsymbol{\xi}, \boldsymbol{H}) = \boldsymbol{H}\boldsymbol{\xi}^{1:n} = \boldsymbol{W}^{-1}\boldsymbol{\xi} + \boldsymbol{H}_{2:n}(\boldsymbol{W}^{-1}\boldsymbol{\xi})^{2:n}, \tag{17}$$

$$\dot{\boldsymbol{z}} = \boldsymbol{n}(\boldsymbol{z}, \boldsymbol{N}) = \boldsymbol{N}\boldsymbol{z}^{1:n} = \boldsymbol{\Lambda}\boldsymbol{z} + \boldsymbol{N}_{2:n}\boldsymbol{z}^{2:n}, \tag{18}$$

$$\boldsymbol{T} \in \mathbb{C}^{d\times d_{1:n}},\ \boldsymbol{H} \in \mathbb{C}^{d\times d_{1:n}},\ \boldsymbol{N} \in \mathbb{C}^{d\times d_{1:n}},$$

where $\boldsymbol{\Lambda} \in \mathbb{R}^{d\times d}$ is a diagonal matrix of eigenvalues of the linear part $\boldsymbol{R}_{1:1} = \boldsymbol{W}\boldsymbol{\Lambda}\boldsymbol{W}^{-1}$ of $\boldsymbol{R}$, and $\boldsymbol{W} \in \mathbb{R}^{d\times d}$ contains the associated eigenvectors.

The normal form transformation and its dynamics are solved for simultaneously with the minimization of the cost function

$$\begin{aligned}(\boldsymbol{H},\ \boldsymbol{N}) = \underset{(\boldsymbol{H}^*,\ \boldsymbol{N}^*)}{\operatorname{argmin}} \sum_{j=1}^{N} \| &\underbrace{\nabla_{\boldsymbol{\xi}}\boldsymbol{t}^{-1}(\boldsymbol{\xi}_j, \boldsymbol{H}^*)\dot{\boldsymbol{\xi}}_j}_{\dot{z}^*} \\ &- \boldsymbol{n}(\underbrace{\boldsymbol{t}^{-1}(\boldsymbol{\xi}_j, \boldsymbol{H}^*)}_{z^*}, \boldsymbol{N}^*)\|^2.\end{aligned} \tag{19}$$

Finally, with the the matrix $\boldsymbol{H}$ computed, we solve for $\boldsymbol{T}$ with

$$\boldsymbol{T} = \underset{\boldsymbol{T}^*}{\operatorname{argmin}} \sum_{j=1}^{N} \|\boldsymbol{t}(\underbrace{\boldsymbol{t}^{-1}(\boldsymbol{\xi}_j, \boldsymbol{H})}_{z_j}, \boldsymbol{T}^*) - \boldsymbol{\xi}_j\|^2. \tag{20}$$

Our reduced model is thus described fully by (10, 11, 16, 17, 18).

3.3 Delay embedding

In practice, observing or measuring all variables that span the full phase space of a dynamical systems is impractical if not impossible. This prompts us to construct data-driven SSM-reduced models in an appropriate observable space. We achieve this by delay-embedding the SSM in the observable space based on the Takens embedding theorem [80].

Delay embedding involves constructing an embedding space using time-delayed copies of a given observable scalar time series $s(t) = \mu(\boldsymbol{x})$, where μ is a differentiable scalar observable function $\mu : \mathbb{R}^{n_{\text{full}}} \mapsto \mathbb{R}$. The function μ returns a measured feature of the system (7), such as the displacement of a particular material point. A new state vector $\boldsymbol{y} \in \mathbb{R}^p$ is constructed from p delayed measurements of the original scalar time series separated by a timelag Δt, that is

$$\boldsymbol{y}(t) = \begin{bmatrix} s(t) \\ s(t+\Delta t) \\ s(t+2\Delta t) \\ \vdots \\ s(t+(p-1)\Delta t) \end{bmatrix}. \tag{21}$$

Takens's theorem states that if $\boldsymbol{x}(t)$ lies on an d-dimensional invariant manifold in the full phase space, then the embedded state $\boldsymbol{y}(t)$ also lies on a diffeomorphic manifold in the p-dimensional embedding space, if $p \geq 2d+1$ and μ and Δt are generic (as defined in Remark 1 in [81]). This result can also be extended to higher-dimensional observable quantities ($\boldsymbol{\mu} : \mathbb{R}^{n_{\text{full}}} \mapsto \mathbb{R}^{n_{\text{observe}}}$), under appropriate nondegeneracy conditions outlined in [81–83].

Near a fixed point, more can be said about the structure of the embedding. It was recently shown that the orientation of eigenspaces at a fixed point in delay-embedded space is directly determined by the eigenvalues of the system linearized there [81]. Thus local spectral properties of the full phase space have a direct geometrical interpretation in the observable space that can be exploited to improve system identification.

In our context, coordinates extracted from a video represent observables of the underlying physical system. Takens's theorem applies to these observables after we delay-embed them with an appropriate time lag, which we select based on the established method of average mutual information [84] (see Appendix B for more details).

3.4 Backbone curve from the normal form

By expressing $\boldsymbol{z}$ in polar coordinates $(\boldsymbol{\rho}, \boldsymbol{\theta})$, defined as $z_j = \rho_j e^{i\theta_j}$ for $j = 1, ..., m$, we obtain the extended normal form of an oscillatory SSM of dimension $2m$ of the form

$$\dot{\rho}_j = \gamma_j(\boldsymbol{\rho}, \boldsymbol{\theta})\rho_j, \tag{22}$$

$$\dot{\theta}_j = \omega_j(\boldsymbol{\rho}, \boldsymbol{\theta}),$$
$$j = 1, ..., m, \ \boldsymbol{\rho} \in \mathbb{R}^m_+, \ \boldsymbol{\theta} \in \mathbb{T}^m, \tag{23}$$

which enables us to perform analyses such as extracting different modal contributions, applying slow-fast decomposition, and constructing backbone curves.

If there is no resonance in the linearized eigenfrequencies, then γ_j and ω_j from (22) and (23) are functions of the amplitudes $\boldsymbol{\rho}$ only [85]. This allows us to decouple the amplitude dynamics from the phase dynamics.

The zero-amplitude limit of γ_j and ω_j corresponds to the linearized damping and frequency of the jth eigenmode, that is

$$\lim_{||\boldsymbol{\rho}|| \to \boldsymbol{0}} (\gamma_j(\boldsymbol{\rho}, \boldsymbol{\theta}) + i\omega_j(\boldsymbol{\rho}, \boldsymbol{\theta})) = \lambda_j. \tag{24}$$

Thus, γ_j and ω_j represent the nonlinear extension of damping and frequency of the system.

To gain physical insight into the amplitude dependence of instantaneous damping and frequency through backbone curves, the amplitude in the normal coordinates must be transformed back into the observable space. For a general 2D case, the amplitude $\mathcal{A}$ can be determined through

$$\mathcal{A}(\rho) = \max_{\theta \in [0, 2\pi)} |g(\boldsymbol{v}(\boldsymbol{t}(\boldsymbol{z})))|, \ \boldsymbol{z} = \left(\rho e^{i\theta}, \rho e^{-i\theta}\right), \tag{25}$$

where the function $g : \mathbb{R}^n \mapsto \mathbb{R}$ maps from the observable space to the amplitude of a particular observable, such as a degree of freedom.

The backbone and damping curves are expressed as parametric functions

$$\mathcal{B}_{\text{damping}} = \{\gamma(\rho), \mathcal{A}(\rho)\}_{\rho \geq 0}, \tag{26}$$

$$\mathcal{B}_{\text{frequency}} = \{\omega(\rho), \mathcal{A}(\rho)\}_{\rho \geq 0}. \tag{27}$$

4 Applications

We now apply our tracking and *SSMLearn* algorithms to five examples with little to no modification across the examples.

4.1 Double pendulum

As our first example, we consider the compound double pendulum shown in Fig. 3a. The double pendulum in our study consists of a pair of rods; the upper rod weighs 253 g and measures 200 mm, while the lower rod is 180 mm long and weighs 114 g. The rods are connected with a roller bearing and both have a width of 25 mm.

We record videos of the transient decay response of the double pendulum released from different initial

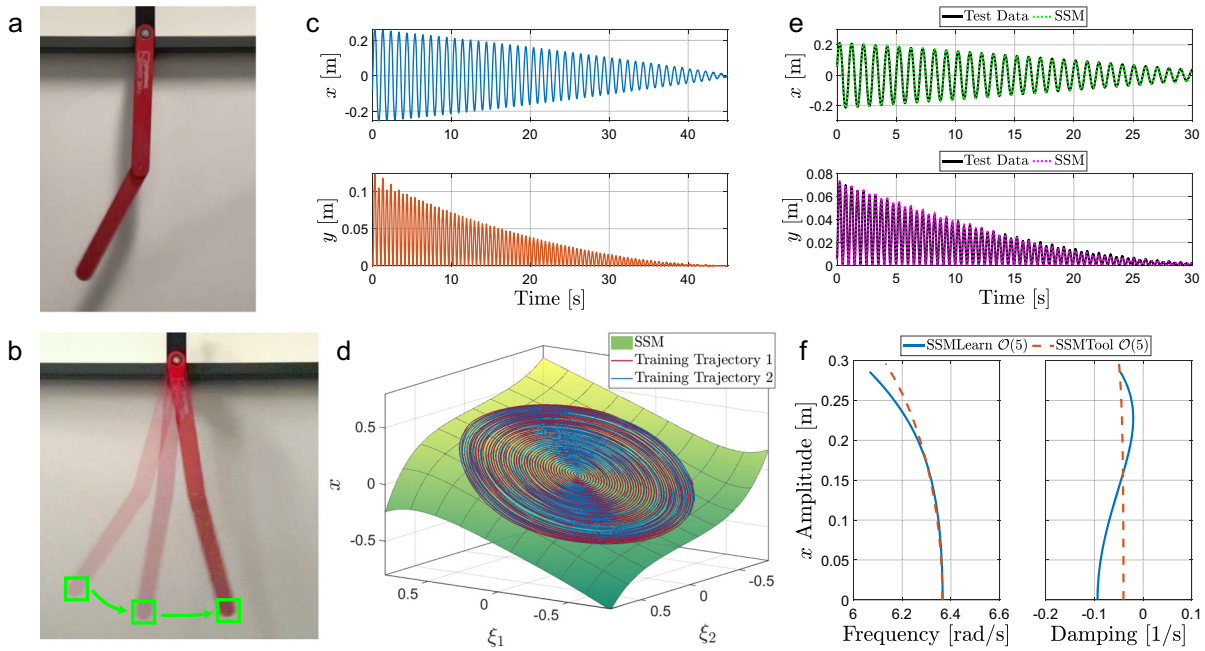

Fig. 3 Data-driven Nonlinear Reduced-order Model on the Slowest SSM of a Double Pendulum—**a** System setup of the double pendulum. **b** Snapshots of processed video frames. **c** Training trajectory in physical length, which was computed by scaling pixels to known physical length from measurements. **d** Training trajectory and fitted manifold plotted in the two reduced coordinates $\xi_{1,2}$ and x. **e** Test trajectory and its reconstruction by the SSM model. **f** Backbone (left) and damping (right) curve output of the *SSMLearn* model, showing the nonlinearities in the system. The analytical result computed from *SSMTool* [28] is plotted together for reference. (Color figure online)

conditions. All videos are recorded with the iPhone 12 Pro, utilizing only the wide camera, at a frame rate of 240 FPS with 1920x1080 resolution. The camera has up to 12 megapixels resolution with f/1.6 aperture.

To extract data from the videos, we initialize the tracker once by selecting a template, in the form of a bounding box, in the first frame of the video. The center of the template corresponds to the end of the lower pendulum rod. The algorithm tracks the template for all subsequent frames in the video and outputs time-series data of the (x, y) pixel locations of the template center. Figure 3b illustrates snapshots of tracked video. One of the two training trajectories is shown in panel (c) of Fig. 3, while the black trajectories in panel (e) are used as test data. Since the physical lengths of the pendulum arms are known, we scale the output pixel data to physical length in meters.

From the video-extracted trajectory data, we aim to learn the reduced dynamics on the slow 2-dimensional SSM emanating from the slow eigenspace of the linear part of the system. The minimal embedding dimension is 5 according to Takens's theorem ($p \geq 2d+1$). Thus we embed the training data (both x and y) five times with a time delay of 0.20s (50 frames of 240 FPS video) to reconstruct the SSM in the observable space.

Figure 3d shows the identified 5th-order,' 2-dimensional SSM in three coordinates: x and the two reduced coordinates, $\xi_{1,2}$. The choice of polynomial order of the SSM parameterization is a compromise between model accuracy, complexity, and the risk of over-fitting. For all the examples in the present paper, we choose a manifold expansion order that results in less than 3% parameterization error (see Appendix A for error quantification).

On the SSM, *SSMLearn* returns the reduced dynamics in a 5th-order normal form

$$\begin{aligned} \dot{\rho}\rho^{-1} &= -0.09352 + 0.8130\rho^2 - 2.256\rho^4, \\ \dot{\theta} &= 6.366 - 0.4733\rho^2 - 1.953\rho^4. \end{aligned} \tag{28}$$

We now use the model (28), trained on a single trajectory, to make predictions of the test trajectory released from a different initial condition. The reduced model takes the initial condition of the test trajectory

as the only input and reconstructs the entire decay response with a reconstruction error of 2.1%.

For comparison, we also derived the equations of motions for this double pendulum system and applied the equation-driven SSM identification package, *SSMTool*, (see Appendix C for more details). In this calculation, the damping from the two joints was modeled as linear using the Rayleigh dissipation function. The geometry of the double pendulum was modeled by two rods with distributed mass. We constrained the motion of the double pendulum to a vertical plane which resulted in a two-degree-of-freedom mechanical model. This four-dimensional dynamical system is fully described by two angles θ_i for $i = 1, 2$ relative to the vertical, and their respective angular velocities $\dot{\theta}_i$.

The frequency and backbone curves obtained from the video data and from the equation of motions are plotted together in Fig. 3f. The frequency learned from data is consistent with the analytical result within 2% error. However, the analytically predicted nonlinear damping curve on the right differs from the actual damping curve identified from the video. This shows that the idealized linear damping used in our mechanical model fails to capture the actual low-amplitude damping characteristic of the system, which is likely dominated by dry friction that is independent of rotational speed. The presence of dry friction is also evident from the more pronounced model mismatch at lower oscillation amplitude. This result illustrates well the use of our video-based SSM-reduction procedure in accurate nonlinear system identification.

4.2 Inverted flag

As our second example, we consider an inverted flag, a flexible elastic sheet with its trailing edge clamped subject to a fluid flow. This is in contrast to a conventional flag where the leading edge is constrained. The experimental configuration, based on [86], is shown in Fig. 4a, where the free end of the flag (in blue) is facing an incoming uniform water flow U. The study of inverted flags has gained much interest in recent years [87–89] for its potential applications in energy harvesting systems.

Across the parameter space of the inverted flag, including flag flexural rigidity and incoming flow properties, the inverted flag motion can display a variety of phenomena, including the undeformed equilibrium state, stable small-deflection state, small-deflection flapping, large-amplitude periodic flapping and chaotic flapping [87]. Here we focus on the large-amplitude periodic flapping regime, wherein the inverted-flag system exhibits greater strain energy than conventional flag flapping [87]. These large bending strains make the inverted-flag system a promising candidate for energy harvesting technologies that convert strain energy to electricity. In the large-amplitude flapping regime, the system has a stable limit cycle and three unstable fixed points, which have been identified numerically as steady-state solutions to the full governing equations [87]. One fixed point is the undeformed equilibrium position, which becomes an unstable saddle point at sufficiently low bending stiffness. The other two symmetric fixed points are located between the saddle point and the limit cycle deflection amplitude.

In the experiments, an inverted flag with a white edge is released from different positions with near-zero velocity and allowed to evolve into a periodic orbit [86]. After preprocessing the videos with background removal, we track the endpoint of the inverted flag with our proposed algorithm. Three frames selected at different times of a training video are shown in Fig. 4b. To construct an SSM model, we train on four trajectories, as shown in Fig. 4c. The pixel coordinates have been normalized against the width of the video.

We delay-embed the x coordinate signal of the flag end point with a time delay of 0.33s (10 frames of 30 FPS video) to create a 5-dimensional observable space, in which *SSMLearn* identifies a 2-dimensional, 3rd-order SSM. Since the origin is a saddle point, *SSMLearn* outputs reduced dynamics corresponding to two real eigenvalues with opposite signs. We identify the reduced dynamics on the SSM up to 9th order, with the full equations presented in Appendix D.

In Fig. 4c, we apply the model to four unseen test trajectories released from different initial positions and plot the reconstructions with the full test dataset. The long term predictions of our model remain bounded with a reconstruction error of up to 6.3%, which we attribute to primarily to phase errors in prediction. Given that a small phase shift leads to high trajectory error, we have further analyzed and quantified phase error growth of our model predictions with windowed-cross-correlation [90] in Appendix E. We find that for all four tests, the phase error remain within 5% of oscillating period.

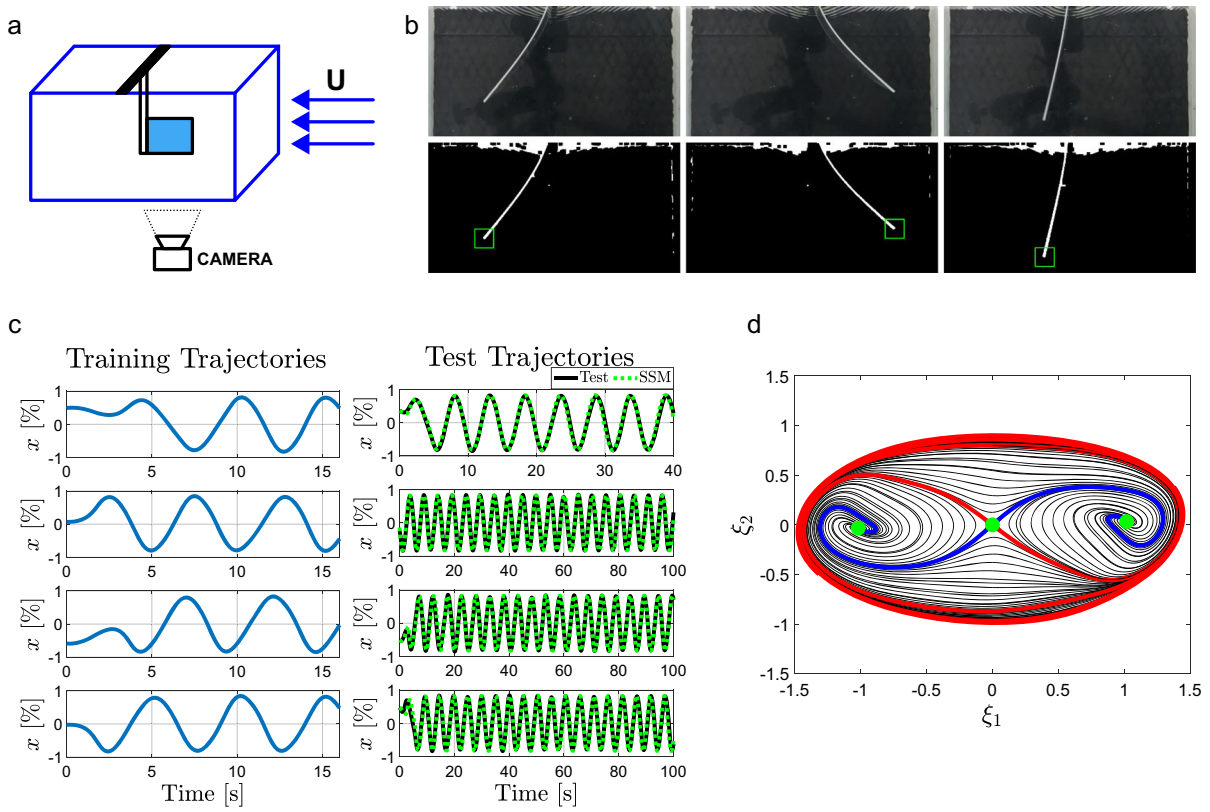

Fig. 4 Data-driven Nonlinear Reduced-order Model on the Mixed-Mode SSM of an Inverted Flag—**a** System setup. **b** Three selected video frames (top row) and the corresponding background-removed and tracked location (bottom row). **c** The training trajectories and test trajectories with model predictions. **d** Phase portrait in reduced coordinates, illustrating the three fixed points in green, stable manifold in blue, and unstable manifold in red. (Color figure online)

In Fig. 4d, we construct a phase portrait by placing various initial conditions on the SSM and advect both forward and backward in time. We find that we recover the stable (blue) and unstable (red) manifold of the saddle point and the three fixed points (green) of the system, along with the stable limit cycle.

4.3 Liquid sloshing

As our third example, we consider fluid oscillation in a partially filled container subject to horizontal external forcing. The sloshing motion of the fluid can be strongly nonlinear [91]. Increased fluid oscillation results in greater shearing at the tank wall, and the damping of the system grows nonlinearly with sloshing amplitude. Further, the study in [19] found a softening response in the system, where the frequency decreases at a higher amplitude. Understanding and modeling the system's nonlinear response is crucial for the design and analysis of structures that involve liquid-containing systems, such as fluid-transporting trucks [1] and spacecraft fuel tanks [92].

Here, we study videos from sloshing experiments performed by [36]. The experiments were performed in a rectangular tank of width 500 mm partially filled with water up to a height of 400 mm, as shown in Fig. 5a. The tank was mounted on a platform excited harmonically by a motor. Videos of the surface profile were recorded with a monochrome camera (1600x1200 pixels at 30 Hz) mounted on a frame moving with the tank. The flow was illuminated by an LED light behind the tank, which creates a distinct shadow of the meniscus. The light source was placed approximately 2 m away from the tank to provide uniform illumination and to avoid heating the fluid.

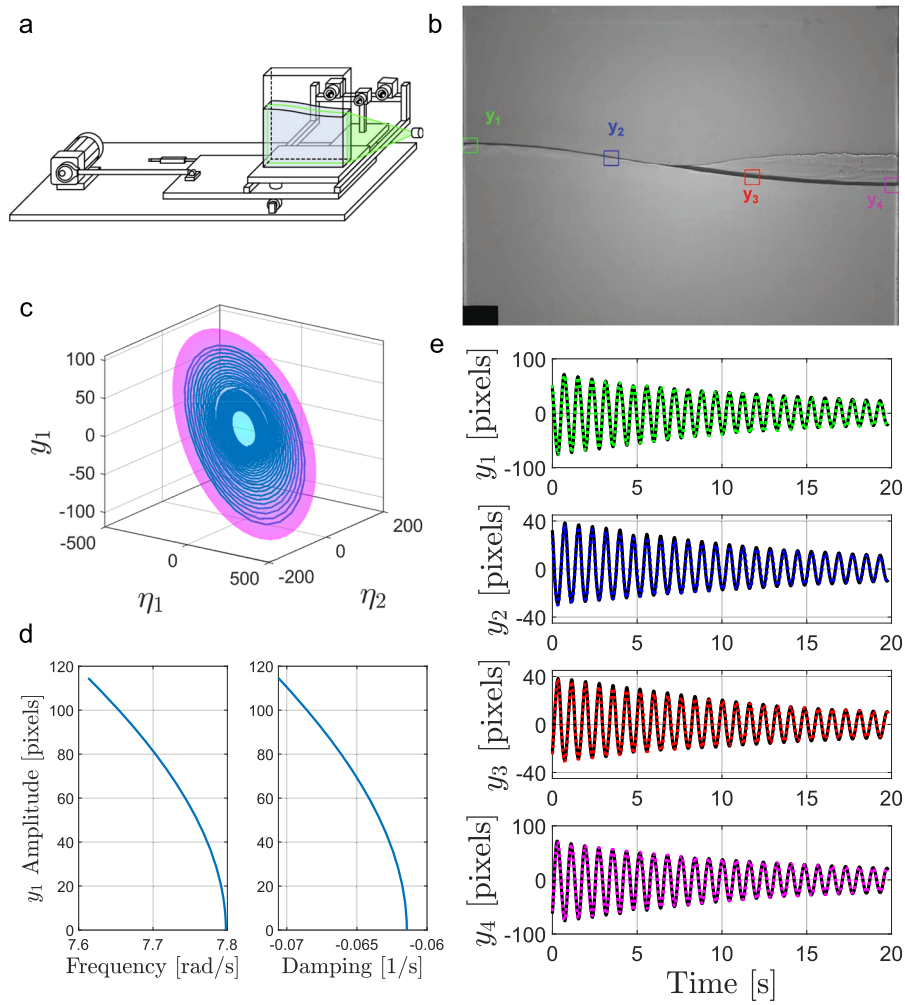

Fig. 5 Data-driven Nonlinear Reduced-order Model on the Slowest SSM of Water Sloshing in a Tank—**a** Sloshing experiment setup with partially filled tank and cameras, mounted on a platform linked to a motor [91]. **b** The four points $y_1, \ldots, y_4$ at the water surface selected for tracking. **c** Training trajectory and identified SSM plotted in reduced coordinates and the leftmost vertical surface elevation y_1 in pixels. **d** Backbone and damping curves produced by the model. **e** Test trajectory for vertical positions of the four material points in black and the model predictions in colors corresponding to (**b**). (Color figure online)

In [36], the surface profile was extracted with a combination of image gradient and thresholding image processing techniques, specialized for the experimental setup and video format. SSM-based models constructed from unforced decaying surface profile data were trained in [19,21]. Here, we will apply our tracking method to extract the liquid surface heights at four, evenly spaced horizontal positions. We initialize four templates, as shown in Fig. 5b with the search region constrained to only the vertical direction. This is the only modification made to our tracking algorithm. The horizontal constraint is required as the shadow of the meniscus is indistinguishable along the surface. We assume the deformation of the surface is locally small, that is, the surface profiles within each template only undergo rigid rotations and vertical translations during sloshing motion.

In Fig. 5e we plot a decaying response signal extracted from experiment videos. We use one such decaying trajectory for training and another one for testing. We delay-embed the four extracted signals $y_{1,2,3,4}$ five times, with $\Delta t = 0.033$s (1 frame of 30 FPS video), to create a 20-dimensional observable space, in which *SSMLearn* identifies a 3rd-order, 2-dimensional SSM illustrated in Fig. 5c.

On the SSM, we identify the reduced dynamics up to 3rd order and compute its 3rd order normal form

$$\begin{aligned} \dot{\rho}\rho^{-1} &= -0.062 - 0.029\rho^2, \\ \dot{\theta} &= 7.80 - 0.60\rho^2, \end{aligned} \tag{29}$$

where the first-order frequency term agrees with an analytical computation of the eigenfrequency [36].

In Fig. 5d, we plot the nonlinear damping and softening response, with respect to the vertical pixels ampli-

tude of the tracked point furthest to the left y_1, captured by our model.

In Fig. 5e we test our model on an unseen trajectory. By inputting only the initial condition and integrating forward in time, our model reconstructs the full test trajectory within 4% error. We find our model to be accurate and correctly capture the essential nonlinearities of the system.

4.4 Aerodynamic flutter

As our fourth example, we consider aeroelastic flutter, a fluid–structure interaction phenomenon characterized by a self-excited structural oscillation wherein energy is drawn from the airstream through the movement of the structure [93]. In a dynamical systems setting, flutter corresponds to a supercritical Hopf bifurcation where the bifurcation parameter is the airspeed.

Modeling aeroelastic flutter is important for the design of flexible aerodynamic structures. In applications where the structural resilience of flexible bodies is crucial, the oscillatory motion plays a significant role in influencing the structure's dynamics and failure. Aside from posing challenges in design scenarios, flutter also emerges as a method for harnessing energy [94]. The simultaneous need to address and enhance flow-induced motion is thus important across diverse engineering fields.

Here, we consider a video of a flutter test from 1966 released by NASA [95]. The test was conducted with a Piper PA-30 Twin Commanche piloted by Fred Haise. We assume the variation in the airspeed is small after the aeroelastic flutter initiation. As the horizontal tailplane in the video moves out of frame, we track the black hook at the bottom of the plane with our tracker and apply inverse translation to the video frame to apply frame stabilization. Figure 6a shows four stabilized frames of the video with three tracked corners of the tailplane in different colors.

We plot the extracted vertical displacements of the three corners in Fig. 6b. There is no reading for the blue and red corners in the last two seconds of the video as the plane moves partially out of the frame. Hence we use only the bottom right corner of the tailplane (green bounding box and signal) to construct our reduced model.

For a two-degree-of-freedom (binary) aeroelastic model at moderate speed, coupling wing bending and torsion, the system has two complex conjugate pairs of eigenvalues corresponding to eigenmodes with positive damping and distinct frequencies [93]. The flutter begins when the damping in one of the modes crosses zero, which occurs when the flow velocity exceeds a critical airspeed and the system undergoes a Hopf bifurcation. The effective dampings in each of the motions must be simultaneously zero and the frequencies of both motions must be identical [96]. Hence, the oscillation will have a single frequency. This argument extends to more than two degrees of freedom.

Further, in Fig. 6c, we plot a spectrogram of the training signal with the power spectral density normalized in each time slice. We find only a single dominant frequency is present in the signals, so we aim to identify a 2D SSM in an appropriate observable space for our model reduction. We delay-embed the signal with $\Delta t = 0.0667$s, to construct an SSM-based model in a 5-dimensional observable space. We identify and parameterize a 3rd-order, 2D SSM, on which *SSMLearn* outputs a 7th-order reduced dynamics and computes a 7th-order normal form

$$\begin{aligned} \dot{\rho}\rho^{-1} &= +0.4844 - 1.679\rho^2 - 8.516\rho^4 + 27.28\rho^6, \\ \dot{\theta} &= 15.90 - 34.64\rho^2 + 377.1\rho^4 - 1213\rho^6. \end{aligned} \tag{30}$$

In Fig. 6d, we integrate in time the initial condition of the original trajectory beyond the original data, we observe the oscillation sustains for all future time and its amplitude remains bounded. The trajectory in the normal space is illustrated in Fig. 6e, where the trajectory grows initially and enters a stable constant amplitude oscillation. We find that the limit cycle is captured by our model and the dynamics predictions remain bounded.

4.5 Wheel shimmy

As our last example, we consider the self-excited vibration of towed wheels, often referred to as wheel shimmy. This phenomenon can occur in a large range of vehicles including motorcycles [97], tractors [98] and aircraft landing gears [99], with significant safety implications.

The rich dynamics of the wheel shimmy originate from the elastic tyres whose ground contact regions are subject to partial sticking and sliding [100]. The

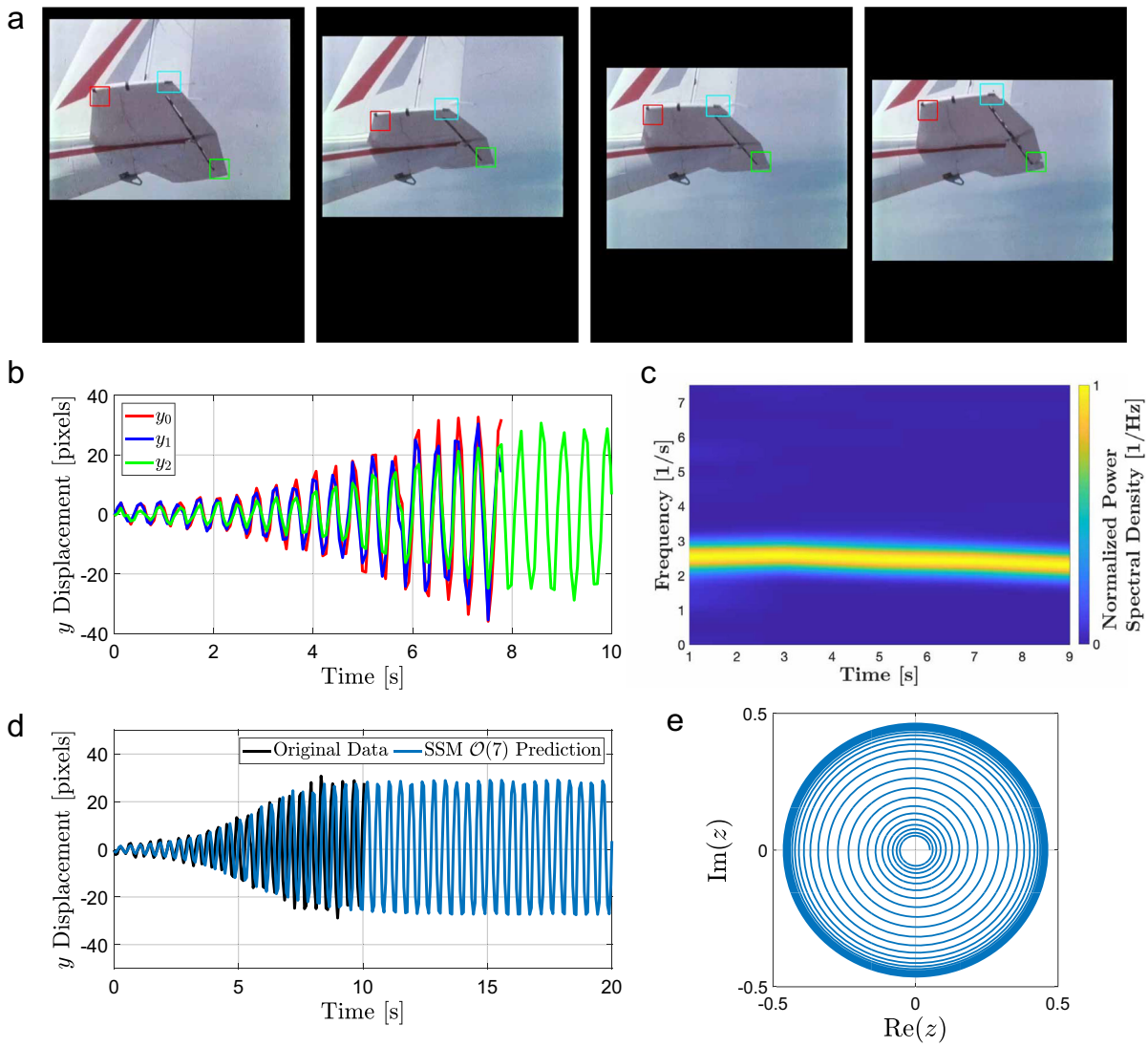

Fig. 6 Data-driven Nonlinear Reduced-order Model on the Unstable Manifold for Tailplane Flutter—**a** Four selected frames of the tailplane flutter video with the three tracked corners in different colors. The black margins of frames are paddings for frame stabilization since the subject is not static in the video. **b** Vertical displacements of the three tracked corners. Since the tailplane moves up and leaves the frame in the last two seconds of the video, there are no valid readings for the blue and red corners for $t > 8$ s. **c** Spectrogram of the displacement signal at the bottom right corner (green) of the horizontal stabilizer. **d** Original trajectory of the vertical displacement of the bottom right corner, y_2, in black and its reconstruction in blue. By advecting the initial condition beyond the time of the original data, we observe a sustained and bounded limit cycle oscillation. **e** Reconstructed trajectory in the normal form coordinates. The model captures the limit cycle behavior. (Color figure online)

time delay in the tyre-ground contact and dry friction result in subcritical Hopf bifurcations in the infinite-dimensional and non-smooth system, giving rise to bistable parameter domains. In the bistable parameter region, the stable rectilinear motion and periodic oscillation coexist with domains of attraction separated by unstable limit cycles. The experimental, analytical, and numerical findings collectively affirm the presence of subcritical bifurcations in the rectilinear motion, wherein bistable parameter domains were observed under fixed caster length and a range of towing speeds [100–102].

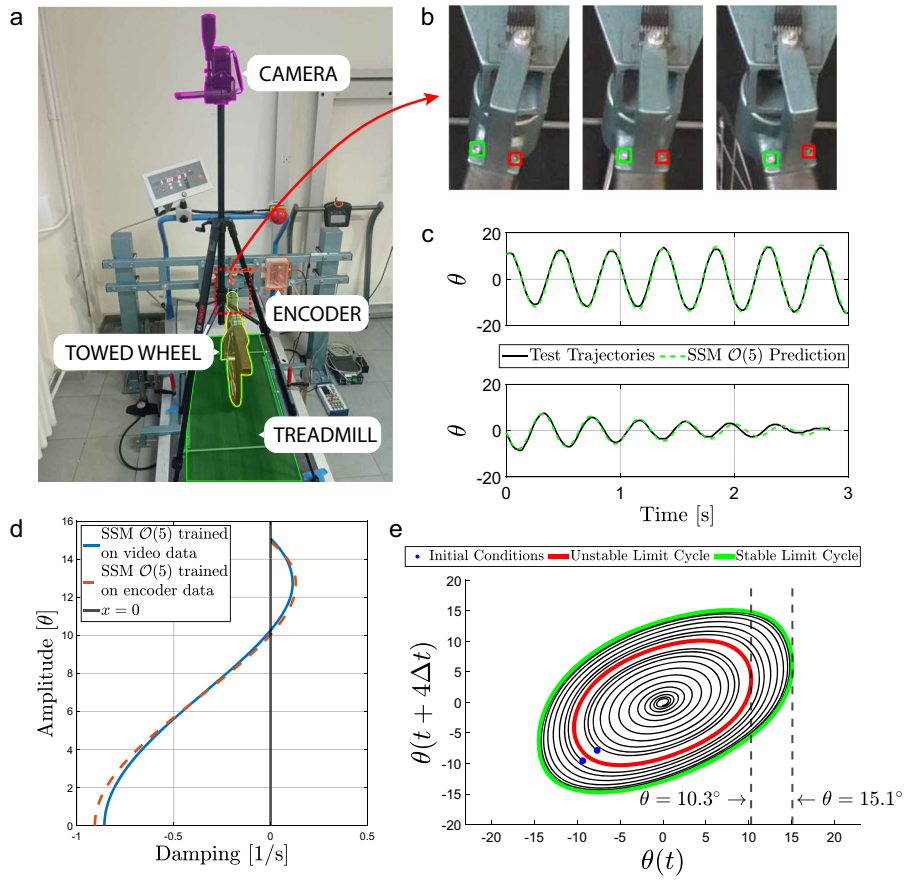

Fig. 7 Data-driven Nonlinear Reduced-order Model on the Slowest SSM of Wheel-Shimmy—**a** The experimental rig with a treadmill in green, towed bicycle wheel in yellow, encoder in orange, and camera in magenta. The red box indicates the region we use for extracting data. **b** Three examples of processed frames, where the tracked bolts are highlighted with green and red bounding boxes. The x and y coordinates of the box centers are used to compute yaw angle θ. **c** Test trajectories for limit cycle and decaying oscillations, and their reconstructions predicted by the trained model. **d** Damping curve output by the SSM-reduced model trained from video data and encoder data. The y-axis zero-crossings correspond to the unstable and stable limit cycle amplitudes. **e** Phase portrait of the system. (Color figure online)

Here, we study a caster-wheel system running on a treadmill with a fixed set of system parameters, caster length and tow speed, in a bistable domain. We record top-down videos (1920x1080 px, 50 Hz) to generate training and test data. The experimental setup is shown in Fig. 7a, which is a modification from the one studied in [100,101]. On top of the treadmill, a bicycle wheel is fixed to a caster, which is mounted to the rack by a rotational joint. A rotational encoder built on the joint provides yaw angle reading for validation.

We track two bolts (see Fig. 7b) on the rotating caster bar to obtain (x, y) pixel positions of the bolt centers, which we use to compute yaw angle readings (see Appendix G for video data validation). Perturbations by hand push were applied to the wheel to generate trajectories that converge to the stable fixed point, $\theta = 0$, and the periodic orbit. Two such trajectories are plotted in Fig. 7c. We use a pair of trajectories, capturing the transients of decaying vibrations and large amplitude oscillations, for training and another pair for testing.

Frequency analysis shows that only a single frequency is present in the signals, so we aim to identify a 2D SSM in an observable space for our model reduction. Based on Takens's embedding theorem, we delay-embed the signal five times with a time delay $\Delta t = 0.02$s (1 frame of 50 FPS video). In the 5-dimensional observable space, *SSMLearn* identifies a 3rd-order, 2-dimensional manifold, on which we identify the reduced dynamics in the 5th-order normal form

$$\begin{aligned} \dot{\rho}\rho^{-1} &= -0.8583 + 12.11\rho^2 - 37.71\rho^4, \\ \dot{\theta} &= 15.17 - 9.155\rho^2 + 7.398\rho^4. \end{aligned} \tag{31}$$

Note that a 5th-order normal form is the minimal order to model the bistable dynamics of the system because we expect two zeros in the damping curve corresponding to the unstable and stable limit cycle amplitudes.

In Fig. 7c, we test our model by advecting two unseen initial conditions and comparing them against the full test trajectories. With our training settings, we

find a reconstruction error of 3.17% for the limit cycle and 5.07% for the decaying oscillation.

As a reference, we train a model using yaw angle readings from the encoder installed on the rotating joint (sampling rate of 200 Hz) with the same trajectories and training setting. We plot the damping curve output by the two models, trained on video data and encoder data, in Fig. 7d. The damping curves from the two models are consistent and predict a similar unstable limit cycle amplitude of 10.2° and the stable limit cycle amplitude of 14.7°, differing by only $< 3.5\%$. The small discrepancy in the damping curves near zero amplitude between the two models is due to a higher signal-to-noise ratio affecting both measurement methods. For the video data, this is caused by the video resolution, while the encoder data is limited by sensor precision. Figure 7e shows the phase portrait constructed by integrating two initial conditions near the unstable limit cycle.

5 Conclusion

We have developed a tracking algorithm based on rotation-aware template matching to extract data from experiment videos, which we have used to construct nonlinear reduced-order models with the open-source MATLAB package *SSMLearn*. We have obtained SSM-based models that accurately captured nonlinear phenomena such as the nonlinear damping of a double pendulum, the softening response of water tank sloshing, multiple coexisting isolated steady states of an inverted flag, stable limit cycle oscillations in the tailplain flutter of an aircfraft and bistable dynamics in wheel shimmy. This broad range of applications demonstrates the versatility of the tracking method and the general applicability of SSM-reduced models constructed from video data.

The main objective of conventional trackers is *object localization*, whereas the purpose of our tracking algorithm is *experimental sensing*. Conventional filter-based trackers are unsuitable for this purpose because the tracked points or features are implicitly defined and updated at each frame, which leads to relatively higher noise, uncertainty in extracted time series data, and drifting issues (see Appendix F for a comparison).

Although the tracking method devised here can extract accurate trajectories for the construction of reduced-order models, a limitation is its computational cost, which is notably higher than state-of-the-art CSRT trackers [46], as well as its limited robustness to environmental changes (see Appendix F for procesing time comparison). Our tracking algorithm takes a brute-force approach to identify optimal matches in target frames, involving the computation of similarity scores between the template and all locations in the search region for multiple potential template rotations. Even though the tracker utilizes information from previous matches to reduce the range of search regions and template rotations, the computational cost remains high.

In future work, we envision reducing the computational time significantly by parallelizing the matching computation without compromising its accuracy. Additionally, one can improve computational efficiency by incorporating motion models to reduce search regions or by calculating matching correlations in the frequency domain similar to the technique used in DCF-based methods [45,46]. Exploring alternative handcrafted features, such as histograms of oriented gradients (HoG) or Gabor filters, could also enhance tracking robustness but at the cost of reduced model interpretability.

Up to a rotation, our template matching tracking method assumes that variations in appearance and lighting are small and a single initialization of the template is therefore sufficient throughout the full video length. It would be possible to extend our template matching to other classes of linear transformations to account for small general rotations and deformations. This, however, would require high-fidelity video and improvements to the computational performance.

These limitations can be addressed in future developments of the algorithm presented here. Our general conclusion is that in situations where the speed of extracting an SSM-reduced model is not critically important, the present form of our algorithm already provides reliable models for nonlinear system identification, trajectory prediction, and localization of hidden features of the dynamics such as unstable limit cycles.

Author contributions A.Y. carried out the analysis and wrote the first draft of the paper; J.A. supervised the research and edited the paper; F.K. carried out the experimental measurements to the wheel shimmy example and edited the paper; G.S. supervised the experimental work and contributed to the analysis; G.H. designed the research, lead the research team and edited the paper.

Funding Open access funding provided by Swiss Federal Institute of Technology Zurich This research was supported by the Swiss National Foundation (SNF) grant no. 200021-214908.

Data availability The videos are available through the link: https://drive.google.com/drive/folders/148Aml_tbgdSoFeBD7w-aSU0X7ekOUYfw?usp=sharing

Code availability The code is available through the link: https://drive.google.com/drive/folders/16IGMR4v87ZZsTmEtgscQVjuxoeOny1Rd?usp=sharing

Declarations

Conflict of interest The authors have no relevant financial or non-financial interests to disclose.

Appendix A Error metric

The root mean square error (RMSE) is a commonly used metric in statistics and machine learning to evaluate the accuracy or performance of a prediction or model. It is defined as

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2},$$

which would not suffice in our context where the observation vector $\boldsymbol{y}(t) \in \mathbb{R}^n$ is not a scalar and for each trajectory, each dimension of the $\boldsymbol{y}(t)$ could take different ranges of values.

In other words, the prediction error for each dimension of the output should be normalized by the corresponding range of the state. For instance, the error of the two phase space variables of a single pendulum, θ and $\dot{\theta}$, should be normalized separately as they could take different ranges of values: $\theta \in [0, 2\pi)$ and $\dot{\theta} \in (-\infty, \infty)$.

To quantify the performance of models that have multi-dimensional output, we use a variation of RMSE extended to vector observations, which will be referred to as the extended root mean square error (ERMSE) defined as

$$\text{ERMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\frac{1}{n}\sum_{j=1}^{n}\left(\frac{\hat{y}_j(t_i) - y_j(t_i)}{\max_i y_j(t_i) - \min_i y_j(t_i)}\right)^2}, \tag{A1}$$

for each $\boldsymbol{y}(t) \in \mathbb{R}^n$ output vector with N samples and the corresponding reconstruction $\hat{\boldsymbol{y}}(t)$ from model.

Alternatively, a modified version of the root mean trajectory error (NMTE) defined in [21] as

$$\text{NMTE} = \frac{1}{N}\frac{1}{||\max_i \boldsymbol{y}(t_i)||}\sum_{i=1}^{N}||\hat{\boldsymbol{y}}(t_i) - \boldsymbol{y}(t_i)||,$$

can also be used. Instead of normalizing the average vectorial error against the maximum norm of $\boldsymbol{y}(t)$, we normalize each dimension/component to its range separately and then compute the average. We will refer to this metric as component-normalized mean trajectory error (CNMTE) defined as

$$\text{CNMTE} = \frac{1}{N}\sum_{i=1}^{N}\sqrt{\sum_{j=1}^{k}\left(\frac{\hat{y}_j(t_i) - y_j(t_i)}{\max_i y_j(t_i) - \min_i y_j(t_i)}\right)^2}. \tag{A2}$$

Even though the expression of (A1) and (A2) are similar, there is a difference in meaning. The ERMSE is an extension of RMSE, which aims to measure the performance of machine-learning models, where models take multiple inputs and produce multiple outputs of different scales. The outputs are not assumed to be related to one another. In contrast, the CNMTE carries a physical meaning as it takes the average of the norm of the normalized vectorial error of the states, which represents trajectory differences.

The choice of which metric to use thus depends on the setting. We will use the ERMSE to quantify the error of manifold parameterization, as this step of the procedure involves fitting a polynomial to a set of scattered data points from all the training trajectories. In contrast, CNMTE will be used to quantify the error of dynamics predictions from the reduced model, a context in which comparing trajectories is important.

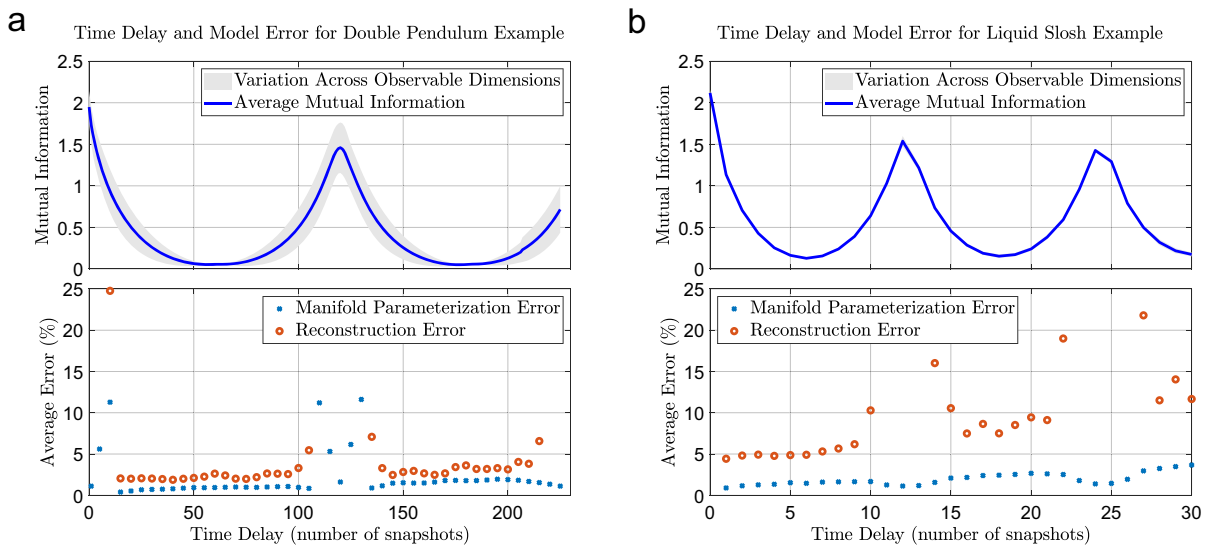

Fig. 8 Mutual Information, Manifold Parameterization Error, and Prediction Error with Varying Time Delay—**a** The averaged mutual information (blue line) with variations across observable dimensions (gray shaded region) are plotted with the manifold parameterization (orange scatter) and trajectory reconstruction (blue scatter) errors of the reduced-model of the double pendulum. **b** Same as (**a**) but for the liquid sloshing reduced-model. The ERMSE error of manifold parameterization is averaged over the training trajectories while the CNMTE error of test trajectory reconstruction is averaged over the test trajectories. Errors larger than 25% are not shown in the plots. (Color figure online)

Appendix B Time delay selection and model robustness

The selection of time delay, in numbers of time snapshots, for each of our models was informed by the method of average mutual information [84,103].

To illustrate the effect of time delay on model performance, we train models for the double pendulum and liquid sloshing systems with varying time delay and compare the behaviour of model errors with average mutual information. The results are plotted in Fig. 8. The small variations in the two errors across a wide range of time delay demonstrate the robustness of our data-driven models, which are constrained only by the mutual information contained in the training data. Thus, we select the time delay close to the first local minima of average mutual information. We note that the behaviour of model errors, with respect to average mutual information, for other systems modelled in this paper shows similar trends.

In both error plots in Fig. 8, we observe a decrease in model prameterization error and a increase in the prediction error near the local maxima of average mutual information. This is a consequence of time delay coinciding with the system periodicty. Delay-embedding the observable vector with these small ranges of time delay collapses the manifold in phase space (hence the small parameterization error), and the essential dynamics of the system cannot be captured in the delay-embedded space. Consequently, this leads to high trajectory prediction error.

Appendix C Equations of motion for the double pendulum

The derivation of the equations of motion begins with the definitions for the coordinates and velocity of the center of masses of the two pendulum rods

$$(x_1, y_1) = \left(\frac{1}{2} l_1 \sin\theta_1,\ -\frac{1}{2} l_1 \cos\theta_1\right),$$

$$(\dot{x}_1, \dot{y}_1) = \left(\frac{1}{2} l_1 \dot{\theta}_1 \cos\theta_1,\ \frac{1}{2} l_1 \dot{\theta}_1 \sin\theta_1\right),$$

$$(x_2, y_2) = \left(l_1 \sin\theta_1 + \frac{1}{2} l_2 \sin\theta_2,\ -l_1 \cos\theta_1 - \frac{1}{2} l_2 \cos\theta_2\right),$$

$$(\dot{x}_2, \dot{y}_2) = \Big(l_1\dot{\theta}_1 \cos\theta_1 + \frac{1}{2} l_2\dot{\theta}_2 \cos\theta_2, \\ l_1\dot{\theta}_1 \sin\theta_1 + \frac{1}{2} l_2\dot{\theta}_2 \sin\theta_2 \Big),$$

where g is the gravitational acceleration constant. These definitions assume the mass centers of the two rods coincide with the halfway point of the physical rod lengths.

The kinetic energy T and potential energy V are

$$\begin{aligned} T &= \frac{1}{2} m_1(\dot{x}_1^2 + \dot{y}_1^2) + \frac{1}{2} m_2(\dot{x}_2^2 + \dot{y}_2^2) \\ &\quad + \frac{1}{2} I_1\dot{\theta}_1^2 + \frac{1}{2} I_2\dot{\theta}_2^2, \\ &= \underbrace{\left(\frac{1}{2} m_1 \left(\frac{1}{2} l_1 \right)^2 + \frac{1}{2} I_1^2 + \frac{1}{2} m_2 l_1^2 \right)}_{A} \dot{\theta}_1^2 \\ &\quad + \underbrace{\left(\frac{1}{2} m_2 \left(\frac{1}{2} l_2 \right)^2 + \frac{1}{2} I_2^2 \right)}_{B} \dot{\theta}_2^2 \\ &\quad + \underbrace{\frac{1}{2} m_2 l_1 l_2}_{C} \dot{\theta}_1\dot{\theta}_2 \cos(\theta_2 - \theta_1), \\ V &= m_1 g y_1 + m_2 g y_2, \\ &= -\underbrace{g l_1 \left(\frac{1}{2} m_1 + m_2 \right)}_{D} \cos\theta_1 - \underbrace{\frac{1}{2} m_2 g l_2}_{E} \cos\theta_2, \end{aligned}$$

where $I_{1,2}$ are the associated moments of inertia, and A, B, C, D, E are all constants consisting of system parameters.

Invoking the parallel axis theorem, we compute the moment of inertia of the pendulum as a summation of contribution from a rectangular prism and two semi-cylinders at both ends.

$$\begin{aligned} I_i &= \frac{1}{12} M_{rod}(l_i^2 + w_i^2) \\ &\quad + 2M_{semicylinder} \left[\left(\frac{1}{16} - \frac{4}{9\pi^2} \right) (w_i)^2 \right. \\ &\quad \left. + \left(\frac{l_i}{2} + \frac{2w_i}{3\pi} \right)^2 \right], \end{aligned}$$

where w_i corresponds to the width of the pendulum arms.

Thus, we can immediately write the Lagrangian of the conservative system as

$$\mathcal{L} = T - V.$$

The equations of motion of the conservative system could be obtained by solving the Euler-Lagrange Equations

$$\frac{\mathrm{d}}{\mathrm{d}t} \left(\frac{\partial \mathcal{L}}{\partial \dot{q}_i} \right) - \frac{\partial \mathcal{L}}{\partial q_i} = 0, \ \text{for } i = 1, 2, \tag{C3}$$

which yield

$$\begin{aligned} \ddot{\theta}_1 &= \frac{1}{K} \Big[C^2 \sin(\theta_1 - \theta_2) \cos(\theta_1 - \theta_2) \dot{\theta}_1^2 \\ &\quad + 2BC \sin(\theta_1 - \theta_2) \dot{\theta}_2^2 \\ &\quad + 2BD \sin\theta_1 - CE \cos(\theta_1 - \theta_2) \sin\theta_2 \Big], \end{aligned} \tag{C4}$$

$$\begin{aligned} \ddot{\theta}_2 &= \frac{1}{K} \Big[-2AC \sin(\theta_1 - \theta_2) \dot{\theta}_1^2 \\ &\quad - C^2 \sin(\theta_1 - \theta_2) \cos(\theta_1 - \theta_2) \dot{\theta}_2^2 \\ &\quad - CD \cos(\theta_1 - \theta_2) \sin\theta_1 + 2AE \sin\theta_2 \Big], \end{aligned} \tag{C5}$$

where
$$\begin{cases} A = \frac{1}{2} m_1 \left(\frac{1}{2} l_1 \right)^2 + \frac{1}{2} I_1^2 + \frac{1}{2} m_2 l_1^2, \\ B = \frac{1}{2} m_2 \left(\frac{1}{2} l_2 \right)^2 + \frac{1}{2} I_2^2, \\ C = \frac{1}{2} m_2 l_1 l_2, \\ D = (\frac{1}{2} m_1 + m_2) g l_1, \\ E = \frac{1}{2} m_2 g l_2, \\ K = C^2 \cos^2(\theta_1 - \theta_2) - 4AB. \end{cases}$$

As the real system is not conservative, we must furthermore include a dissipation in the analytical model.

C.1 Rayleigh dissipation function

In 1881, Lord Rayleigh demonstrated that when a dissipative force $\boldsymbol{F}$ is proportional to velocity, it can be represented by a scalar potential that depends on the generalized velocities $\dot{\boldsymbol{q}}$. This scalar potential is known

as the Rayleigh dissipation function $\mathcal{D}$

$$\mathcal{D}(\dot{\boldsymbol{q}}) \equiv \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\beta_{ij}\dot{q}_i\dot{q}_j, \tag{C6}$$

and it provides an elegant way to incorporate linear velocity-dependent dissipative forces in Lagrangian and Hamiltonian mechanics.

Dissipative drag force in the direction of $\boldsymbol{q}$ is then defined as the negative velocity gradient of the dissipation function (C6), that is

$$\boldsymbol{F} = -\nabla_{\dot{\boldsymbol{q}}}\mathcal{D}(\dot{\boldsymbol{q}}).$$

In the context of a double pendulum, it is reasonable to assume that frictions from the two joints are the main sources of dissipation in the system as opposed to other factors such as air resistance.

If we consider the two joints as free bodies, the dissipation from the first joint only depends on the rate of rotation of the upper arm, whereas the second joint linking the two arms depends on the relative rate of rotation of the two rods. Thus, the dissipation function for the system is of the form

$$\mathcal{D} = \underbrace{\frac{\beta_1}{2}\dot{\theta}_1^2}_{\text{Joint1}} + \underbrace{\frac{\beta_2}{2}(\dot{\theta}_2-\dot{\theta}_1)^2}_{\text{Joint2}}$$
$$\Longrightarrow \begin{cases} F_1 &= -\frac{\partial\mathcal{D}}{\partial\dot{\theta}_1} = -(\beta_1+\beta_2)\dot{\theta}_1 + \beta_2\dot{\theta}_2 \\ F_2 &= -\frac{\partial\mathcal{D}}{\partial\dot{\theta}_2} = \beta_2\dot{\theta}_1 - \beta_2\dot{\theta}_2 \end{cases}.$$

The dissipative generalized forces can then be added to the Euler-Lagrange equation (C3) to obtain

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial\mathcal{L}}{\partial\dot{q}_i}\right) - \frac{\partial\mathcal{L}}{\partial q_i} = F_i, \text{ for } i = 1, 2. \tag{C7}$$

Solving (C7) arrives at a modified version of (C4) and (C5)

$$\begin{aligned}\ddot{\theta}_1 &= \frac{1}{K}\left[\ldots + 2B\left((\beta_1+\beta_2)\dot{\theta}_1 - \beta_2\dot{\theta}_2\right)\right.\\ &\quad \left. + C\cos(\theta_1-\theta_2)\left(\beta_2\dot{\theta}_1 - \beta_2\dot{\theta}_2\right)\right]\\ &= f_3(\theta_1,\theta_2,\dot{\theta}_1,\dot{\theta}_2,\boldsymbol{\mu}),\end{aligned} \tag{C8}$$
$$\begin{aligned}\ddot{\theta}_2 &= \frac{1}{K}\left[\ldots - 2A\left(\beta_2\dot{\theta}_1 - \beta_2\dot{\theta}_2\right)\right.\\ &\quad \left. - C\cos(\theta_1-\theta_2)\left((\beta_1+\beta_2)\dot{\theta}_1 - \beta_2\dot{\theta}_2\right)\right]\\ &= f_4(\theta_1,\theta_2,\dot{\theta}_1,\dot{\theta}_2,\boldsymbol{\mu}),\end{aligned} \tag{C9}$$

where $\boldsymbol{\mu}$ is a vector collecting all system parameters. These are the equations of motion for a double pendulum with modeled joint dissipation. Although the damping model linearly depends on angular velocities, the dissipation is nonlinear due to geometric nonlinearities.

Appendix D SSM-reduced model for the inverted flag

$$\begin{aligned}\dot{\xi}_1 = &-2.325\xi_2 + 0.3512\xi_2^2 + 1.132\xi_2^3\\ &- 1.479\xi_2^4 - 1.806\xi_2^5 + 1.806\xi_2^6\\ &+ 1.742\xi_2^7 - 0.6762\xi_2^8 - 0.5968\xi_2^9\\ &+ 0.09106\xi_1 + 0.128\xi_1\xi_2 - 0.8056\xi_1\xi_2^2\\ &- 0.5981\xi_1\xi_2^3 + 5.407\xi_1\xi_2^4 + 0.3837\xi_1\xi_2^5\\ &- 8.837\xi_1\xi_2^6 + 0.1211\xi_1\xi_2^7 + 4.119\xi_1\xi_2^8\\ &+ 0.2219\xi_1^2 - 0.502\xi_1^2\xi_2 - 1.619\xi_1^2\xi_2^2\\ &+ 0.5419\xi_1^2\xi_2^3 + 3.104\xi_1^2\xi_2^4 + 0.7389\xi_1^2\xi_2^5\\ &- 1.701\xi_1^2\xi_2^6 - 0.608\xi_1^2\xi_2^7 + 0.6853\xi_1^3\\ &+ 0.3954\xi_1^3\xi_2 - 3.645\xi_1^3\xi_2^2 + 0.1459\xi_1^3\xi_2^3\\ &+ 2.215\xi_1^3\xi_2^4 - 0.3787\xi_1^3\xi_2^5 + 0.6783\xi_1^3\xi_2^6\\ &- 0.539\xi_1^4 + 3.49\xi_1^4\xi_2 + 1.664\xi_1^4\xi_2^2\\ &- 5.234\xi_1^4\xi_2^3 - 1.183\xi_1^4\xi_2^4 + 1.699\xi_1^4\xi_2^5\\ &- 1.499\xi_1^5 - 0.8057\xi_1^5\xi_2 + 5.23\xi_1^5\xi_2^2\\ &+ 0.4301\xi_1^5\xi_2^3 - 2.93\xi_1^5\xi_2^4 + 0.4203\xi_1^6\\ &- 3.089\xi_1^6\xi_2 - 0.5586\xi_1^6\xi_2^2 + 2.61\xi_1^6\xi_2^3\\ &+ 0.9821\xi_1^7 + 0.2868\xi_1^7\xi_2 - 1.658\xi_1^7\xi_2^2\\ &- 0.1005\xi_1^8 + 0.7641\xi_1^8\xi_2 - 0.2007\xi_1^9,\end{aligned}$$

$$\begin{aligned}\dot{\xi}_2 = &-0.2462\xi_2 + 0.6229\xi_2^2 + 5.107\xi_2^3\\ &- 1.724\xi_2^4 - 16.8\xi_2^5 + 1.803\xi_2^6\\ &+ 19.39\xi_2^7 - 0.6994\xi_2^8 - 7.426\xi_2^9\\ &- 2.741\xi_1 + 1.084\xi_1\xi_2 + 11.86\xi_1\xi_2^2\\ &- 5.412\xi_1\xi_2^3 - 10.74\xi_1\xi_2^4 + 6.211\xi_1\xi_2^5\\ &- 1.404\xi_1\xi_2^6 - 1.81\xi_1\xi_2^7 + 3.985\xi_1\xi_2^8\\ &+ 0.1505\xi_1^2 + 0.3348\xi_1^2\xi_2 - 4.276\xi_1^2\xi_2^2\\ &- 8.45\xi_1^2\xi_2^3 + 8.499\xi_1^2\xi_2^4 + 16.54\xi_1^2\xi_2^5\end{aligned}$$

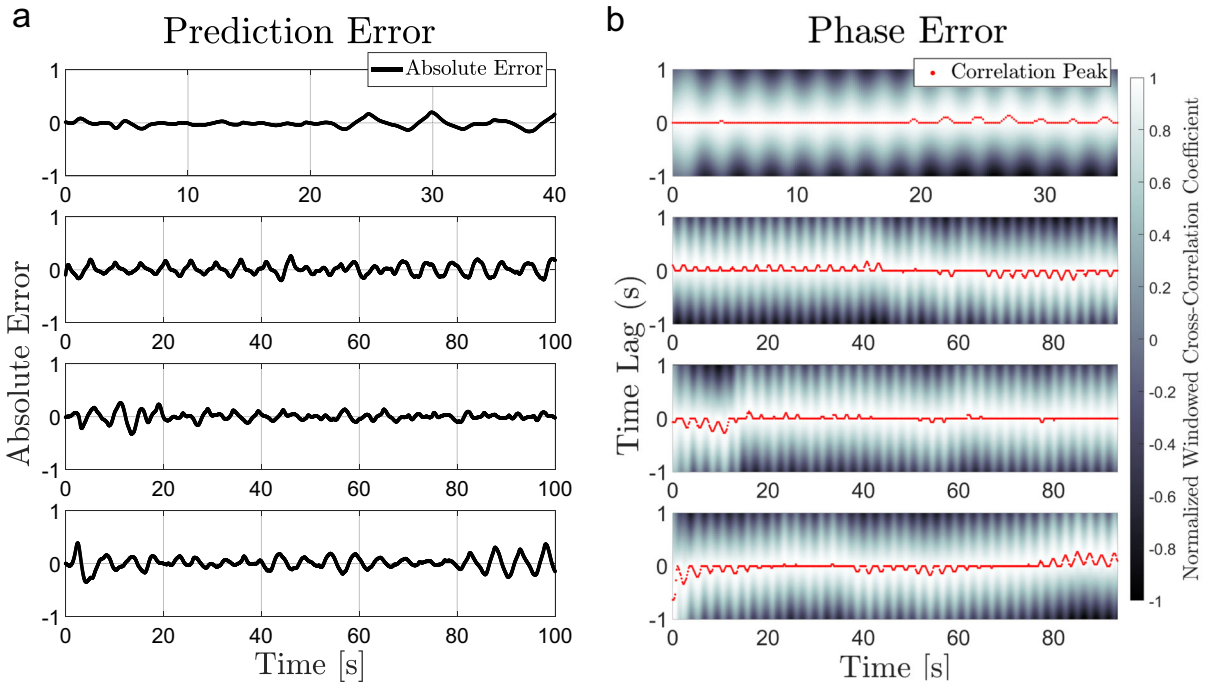

Fig. 9 Error Analysis of Reduced-Model Prediction of Inverted Flag Dynamics—**a** Absolute error of model prediction (c.f. Fig. 4c). **b** Phase error of model prediction measured by windowed cross-correlation, where the coefficient peak across time is plotted in red. The width of sliding window is selected to be one period of oscillation ($\approx$ 5.6 seconds determined from spectrogram similar to the one shown in Fig. 6c). The cross-correlation coefficient is normalized in each time slice. (Color figure online)

$$
\begin{aligned}
&- 4.433\xi_1^2\xi_2^6 - 8.455\xi_1^2\xi_2^7 + 8.555\xi_1^3 \\
&- 0.09146\xi_1^3\xi_2 - 43.53\xi_1^3\xi_2^2 + 4.224\xi_1^3\xi_2^3 \\
&+ 58.71\xi_1^3\xi_2^4 - 3.911\xi_1^3\xi_2^5 - 23.09\xi_1^3\xi_2^6 \\
&- 0.639\xi_1^4 + 7.441\xi_1^4\xi_2 + 4.639\xi_1^4\xi_2^2 \\
&- 10.29\xi_1^4\xi_2^3 - 4.067\xi_1^4\xi_2^4 + 2.415\xi_1^4\xi_2^5 \\
&- 10.49\xi_1^5 - 1.479\xi_1^5\xi_2 + 39.89\xi_1^5\xi_2^2 \\
&+ 0.04915\xi_1^5\xi_2^3 - 28.55\xi_1^5\xi_2^4 + 0.6601\xi_1^6 \\
&- 8.282\xi_1^6\xi_2 - 1.522\xi_1^6\xi_2^2 + 7.384\xi_1^6\xi_2^3 \\
&+ 5.691\xi_1^7 + 0.61\xi_1^7\xi_2 - 10.51\xi_1^7\xi_2^2 \\
&- 0.1828\xi_1^8 + 2.22\xi_1^8\xi_2 - 1.073\xi_1^9.
\end{aligned}
$$

Appendix E Phase error quantification for inverted flag model

We quantify the phase error of our reduced-model predictions for the inverted flag with the windowed cross-correlation [90], which computes cross-correlation of successive segments (or windows) of two signals. A conventional cross-correlation is recovered when the window size is set to the length of the input signals. The windowed cross-correlation captures the temporal variations in the strength and direction of associations between two time series. In an oscillatory system, such as the inverted flag, we could interpret the variations in the correlation between the test trajectories and model predictions as the phase error.

We plot the absolute error of model prediction (from Fig. 4c) and the phase error in Fig. 9. We find the phase error of model predictions across all four test trajectories remain within ± 0.3 s delay, which corresponds to $\approx 5\%$ of oscillating period and less than 1% when averaged across prediction time. We observe that there is a correspondence between localized regions of relatively high absolute prediction error and the phase error, such as at the beginning and end of the bottom test trajectory, which suggests that the main prediction error is associated with phase error rather than amplitude.

Table 1 Summary of Video Details and Settings for Benchmark

Dataset	Resolution	Frames	Angle Sweep[a] [min, max, step] (degrees)	Search Region[b] (multiples of template dimensions)
Double pendulum	1080×680 (grayscale)	12854	−15, 15, 5	1
Inverted flag	680×350 (grayscale)	1570	−15, 15, 5	1
Liquid sloshing	1532×1124 (grayscale)	2373	−15, 15, 5	1
Aerodynamic flutter	697×891 (RGB)	239	−30, 30, 5	2
Wheel shimmy	136×208 (RGB)	16655	0, 0, 0	1

[a] Applicable to the proposed TM tracker only
[b] Applied to all trackers

Table 2 Summary of Process Time of Proposed Template Matching (TM) Tracker with CSR-DCF and MOSSE Tracker

Dataset	Proposed TM tracker process time (seconds)	CSR-DCF tracker process time (seconds)	MOSSE tracker process time (seconds)
Double pendulum	720.8	676.5	476.1
Inverted flag	79.5	74.6	51.6
Liquid sloshing	189.7	154.8	151.1
Aerodynamic flutter	49.1	16.4	4.2
Wheel shimmy	185.2	502.9	310.8

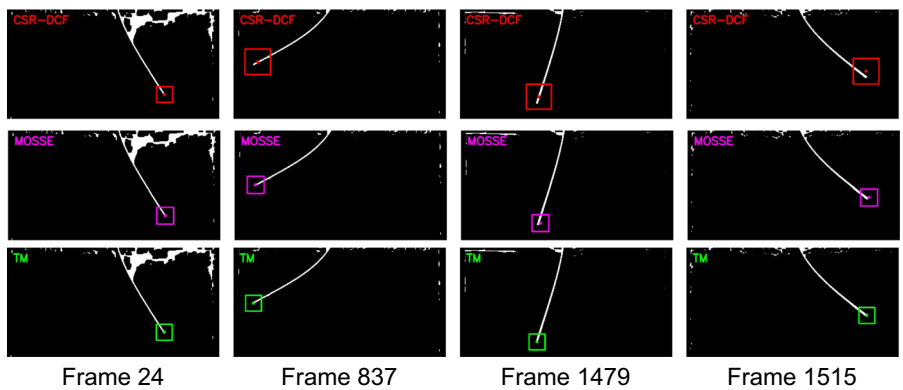

Fig. 10 Qualitative Comparison of Proposed Template Matching (TM) Tracker with CSR-DCF and MOSSE Tracker—This figure illustrates the qualitative performance comparison in an inverted flag experiment video. The proposed TM tracker, along with two advanced correlation filter-based trackers (CSR-DCF [46] and MOSSE [44]), were initialized using an identical template in frame 24, followed by tracking the same video sequence. The center of the bounding box corresponds to the tracked point. Five frames showcase the drift issues common in traditional trackers. The first row (in red) are results from the CSR-DCF tracker. The second row (in magenta) are from the MOSSE tracker. The bottom row (in green) are from our proposed TM tracker. (Color figure online)

Appendix F Comparison with other correlation filter-based trackers

We compare the processing time of the proposed tracking method with two state-of-the-art trackers, CSR-DCF [46] and MOSSE [44], by applying all three tracking algorithms to a sample video from each of the five real-life examples presented in this paper. The hardware used for the benchmark is a 6-core AMD Ryzen™

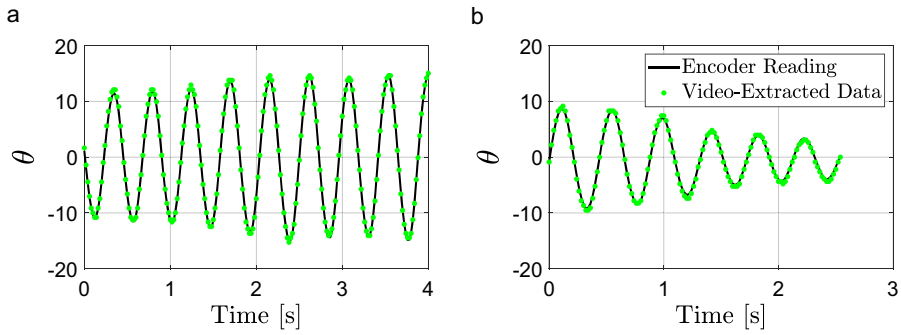

Fig. 11 Comparison of Wheel Shimmy Angle Extracted from Video with Encoder Readings—**a** Limit cycle trajectory used for reduced model training. **b** Decaying trajectory used for reduced model training. The angle extracted from the video (green) in both sub-figures is plotted with the encoder readings (black). (Color figure online)

5 3600 CPU with 3.6GHz base clock speed which Max Boosts (similar to Intel's Turbo Boost) up to 4.2GHz.

The details of the videos and the settings used for the benchmark are summarized in Table 1 and the results are summarized in Table 2. In each video, we initialize all three trackers with the same template with bounding box centers corresponding to the point we want to track, such as the tip of the inverted flag, and allow each tracker to iterate through the entire video sequence. As expected, our proposed TM algorithm is generally slower than the other two for cases where we employ an angle sweep in the template matching process. When angle sweeping is disabled, such as for the wheel shimmy video example, we find our TM tracker to be faster than the CSR-DCF and the MOSSE tracker.

We note that the results presented in Table 2 only illustrate the computational efficiency of the algorithms and not the tracking accuracies, as all videos are of real experiments and ground truth does not exist. Although both the CSR-DCF and MOSSE could track the initialized template through the entire video sequences (except in the liquid sloshing example where both failed), both trackers suffered from drifting issues. We illustrate this problem in Fig. 10 qualitatively, where we provide tracking results of a liquid sloshing video at three selected frames (frame 837, 1479, and 1515) with identical template initialized at frame 24 for all trackers. Using the template center point as the reference point for a qualitative evaluation of tracking accuracy, we find our tracking algorithm to be accurate and not exhibiting any drifting issues as observed by the application of the other two methods.

Appendix G Video tracking error quantification

As a validation of our template matching tracking algorithm, we compare the angle extracted from the wheel shimmy experiment video with the encoder readings (see Fig. 7a for test rig setup). Figure 11a plots the limit cycle and Fig. 11b plots the decaying trajectories used for training the two SSM-based reduced models (one with video data and another with encoder data). Based on the encoder angle readings as the reference, the data extracted with our proposed TM tracker has a trajectory error (measured in CNMTE metric) of 1.1% for the limit cycle trajectory and 1.5% for the decaying trajectory.

References

1. Cheli, F., D'Alessandro, V., Premoli, A., Sabbioni, E.: Simulation of sloshing in tank trucks. Int. J. Heavy Veh. Syst. **20**(1), 1–18 (2013). https://doi.org/10.1504/IJHVS.2013.051099
2. Holmes, P., Lumley, L., Berkooz, G.: Turbulence, coherent structures, dynamical systems and symmetry. Cambridge Monographs on Mechanics, Cambridge (1996)
3. Ogundele, A.D.: Modeling and analysis of nonlinear spacecraft relative motion via harmonic balance and Lyapunov function. Aerosp. Sci. Technol. **99**, 105761 (2020). https://doi.org/10.1016/j.ast.2020.105761
4. Lacayo, R., Pesaresi, L., Groß, J., Fochler, D., Armand, J., Salles, L., Schwingshackl, C., Allen, M., Brake, M.: Nonlinear modeling of structures with bolted joints: a comparison of two approaches based on a time-domain and frequency-domain solver. Mech. Syst. Signal Process. **114**, 413–438 (2019). https://doi.org/10.1016/j.ymssp.2018.05.033
5. Breunung, T., Cilenti, L., You, J.M., Balachandran, B.: Robust identification of nonlinear oscillators from frequency response data. In: Brake, M.R.W., Renson, L., Kuether, R.J., Tiso, P. (eds.) Nonlinear structures & systems, vol. 1, pp. 11–13. Springer, Cham (2024)

6. Detroux, T., Noël, J.-P., Kerschen, G.: Tailoring the resonances of nonlinear mechanical systems. Nonlinear Dyn. **103**(4), 3611–3624 (2021). https://doi.org/10.1007/s11071-020-06002-w
7. Alora, J.I., Cenedese, M., Schmerling, E., Haller, G., Pavone, M.: Practical deployment of spectral submanifold reduction for optimal control of high-dimensional systems. IFAC PapersOnLine **56–2**, 4074–4081 (2023)
8. Perno, M., Hvam, L., Haug, A.: Implementation of digital twins in the process industry: a systematic literature review of enablers and barriers. Comput. Ind. **134**, 103558 (2022). https://doi.org/10.1016/j.compind.2021.103558
9. Awrejcewicz, J., Krys'ko, V.A., Vakakis, A.F.: Order reduction by proper orthogonal decomposition (POD) analysis, pp. 177–238. Springer, Berlin (2004). https://doi.org/10.1007/978-3-662-08992-7_3
10. Lu, K., Jin, Y., Chen, Y., Yang, Y., Hou, L., Zhang, Z., Li, Z., Fu, C.: Review for order reduction based on proper orthogonal decomposition and outlooks of applications in mechanical systems. Mech. Syst. Signal Process. **123**, 264–297 (2019). https://doi.org/10.1016/j.ymssp.2019.01.018
11. Schmid, P.J.: Dynamic mode decomposition of numerical and experimental data. J. Fluid Mech. **656**, 5–28 (2010). https://doi.org/10.1017/S0022112010001217
12. Schmid, P.J.: Dynamic mode decomposition and its variants. Ann. Rev. Fluid Mech. **54**, 225–254 (2022). https://doi.org/10.1146/annurev-fluid-030121-015835
13. Page, J., Kerswell, R.R.: Koopman mode expansions between simple invariant solutions. J. Fluid Mech. **879**, 1–27 (2019). https://doi.org/10.1017/jfm.2019.686
14. Liu, Z., Ozay, N., Sontag, E.D.: Properties of immersions for systems with multiple limit sets with implications to learning Koopman embeddings. (2023). arxiv:2312.17045
15. Simpson, T., Dervilis, N., Chatzi, E.: Machine learning approach to model order reduction of nonlinear systems via autoencoder and LSTM networks. J. Eng. Mech. (2021). https://doi.org/10.1061/(asce)em.1943-7889.0001971
16. Calka, M., Perrier, P., Ohayon, J., Grivot-Boichon, C., Rochette, M., Payan, Y.: Machine-learning based model order reduction of a biomechanical model of the human tongue. Comput. Methods Programs Biomed. **198**, 105786 (2021). https://doi.org/10.1016/j.cmpb.2020.105786
17. Kaheman, K., Kutz, J.N., Brunton, S.L.: SINDy-PI: a robust algorithm for parallel implicit sparse identification of nonlinear dynamics. Proc. R. Soc. A: Math. Phys. Eng. Sci. **476**(2242), 20200279 (2020). https://doi.org/10.1098/rspa.2020.0279
18. Szalai, R., Ehrhardt, D., Haller, G.: Nonlinear model identification and spectral submanifolds for multi-degree-of-freedom mechanical vibrations. Proc. R. Soc. A: Math. Phys. Eng. Sci. **473**(2202), 20160759 (2017). https://doi.org/10.1098/rspa.2016.0759
19. Cenedese, M., Axås, J., Bäuerlein, B., Avila, K., Haller, G.: Data-driven modeling and prediction of non-linearizable dynamics via spectral submanifolds. Nat. Commun. (2022). https://doi.org/10.1038/s41467-022-28518-y
20. Cenedese, M., Axås, J., Yang, H., Eriten, M., Haller, G.: Data-driven nonlinear model reduction to spectral submanifolds in mechanical systems. Philos. Trans. R. Soc. A Math. Phys. Eng. Sci. (2022). https://doi.org/10.1098/rsta.2021.0194
21. Axås, J., Cenedese, M., Haller, G.: Fast data-driven model reduction for nonlinear dynamical systems. Nonlinear Dyn. **111**(9), 7941–7957 (2023). https://doi.org/10.1007/s11071-022-08014-0
22. Shaw, S.W., Pierre, C.: Normal modes for non-linear vibratory systems. J. Sound Vib. **164**(1), 85–124 (1993). https://doi.org/10.1006/jsvi.1993.1198
23. Cabré, X., Fontich, E., de la Llave, R.: The parameterization method for invariant manifolds I: manifolds associated to non-resonant subspaces. Indiana Univ. Math. J. **52**(2), 283–328 (2003)
24. Haro, A., de la Llave, R.: A parameterization method for the computation of invariant tori and their whiskers in quasi-periodic maps: Rigorous results. J. Differ. Equ. **228**(2), 530–579 (2006). https://doi.org/10.1016/j.jde.2005.10.005
25. Haller, G., Ponsioen, S.: Nonlinear normal modes and spectral submanifolds: existence, uniqueness and use in model reduction. Nonlinear Dyn. **86**(3), 1493–1534 (2016). https://doi.org/10.1007/s11071-016-2974-z
26. Haller, G., Kaszás, B., Liu, A., Axås, J.: Nonlinear model reduction to fractional and mixed-mode spectral submanifolds. Chaos Interdiscip. J. Nonlinear Sci. (2023). https://doi.org/10.1063/5.0143936
27. Haller, G., Kaundinya, R.S.: Nonlinear model reduction to temporally aperiodic spectral submanifolds. Chaos (2024). https://doi.org/10.1063/5.0187080
28. Jain, S., Haller, G.: How to compute invariant manifolds and their reduced dynamics in high-dimensional finite element models. Nonlinear Dyn. **107**(2), 1417–1450 (2022). https://doi.org/10.1007/s11071-021-06957-4
29. Kaszás, B., Haller, G.: Capturing the edge of chaos as a spectral submanifold in pipe flows. J. Fluid Mech. **979**, 48 (2024). https://doi.org/10.1017/jfm.2023.956
30. Cenedese, M., Marconi, J., Haller, G., Jain, S.: Data-assisted non-intrusive model reduction for forced nonlinear finite elements models. (2023). arXiv:2311.17865
31. Kragic, D., Vincze, M.: Vision for robotics. Found. Trends® Robot. **1**(1), 1–78 (2009). https://doi.org/10.1561/2300000001
32. Computer Vision in Medical Imaging: World Scientific (2014). https://doi.org/10.1142/8766
33. Zhou, L., Zhang, L., Konz, N.: Computer vision techniques in manufacturing. IEEE Trans. Syst. Man Cybern. Syst. **53**(1), 105–117 (2023). https://doi.org/10.1109/TSMC.2022.3166397
34. Janai, J., Güney, F., Behl, A., Geiger, A.: Computer vision for autonomous vehicles: problems, datasets and state of the art. Found. Trends® Comput. Graph. Vis. **12**(13), 1–308 (2020)
35. Dong, C.Z., Celik, O., Catbas, F.N.: Marker-free monitoring of the grandstand structures and modal identification using computer vision methods. Struct. Health Monit. **18**(5–6), 1491–1509 (2019). https://doi.org/10.1177/1475921718806895
36. Bäuerlein, B., Avila, K.: Phase lag predicts nonlinear response maxima in liquid-sloshing experiments. J. Fluid Mech. **925**, 22 (2021). https://doi.org/10.1017/jfm.2021.576
37. Jaques, M., Burke, M., Hospedales, T.: Physics-as-inverse-graphics: unsupervised physical parameter estimation from

video. In: Proceedings of the international conference on learning representations (ICLR 2020), pp. 1–16 (2020). Eighth International Conference on Learning Representations, ICLR 2020 ; Conference date: 26-04-2020 Through 30-04-2020. https://iclr.cc/Conferences/2020
38. Luan, L., Liu, Y., Sun, H.: Distilling governing laws and source input for dynamical systems from videos. In: De Raedt, L. (ed.) Proceedings of the thirty-first international joint conference on artificial intelligence (IJCAI-22), pp. 3898–3904 (2022). https://doi.org/10.24963/ijcai.2022/541. Main Track
39. Liu, S., Liu, D., Srivastava, G., Połap, D., Woźniak, M.: Overview and methods of correlation filter algorithms in object tracking. Complex Intell. Syst. **7**(4), 1895–1917 (2021). https://doi.org/10.1007/s40747-020-00161-4
40. Dutta, A., Mondal, A., Dey, N., Sen, S., Moraru, L., Hassanien, A.E.: Vision tracking: a survey of the state-of-the-art. SN Comput. Sci. (2020). https://doi.org/10.1007/s42979-019-0059-z
41. Marvasti-Zadeh, S.M., Cheng, L., Ghanei-Yakhdan, H., Kasaei, S.: Deep learning for visual tracking: a comprehensive survey. IEEE Trans. Intell. Transp. Syst. **23**(5), 3943–3968 (2022). https://doi.org/10.1109/tits.2020.3046478
42. Henriques, J.F., Caseiro, R., Martins, P., Batista, J.: High-speed tracking with kernelized correlation filters. IEEE Trans. Pattern Anal. Mach. Intell. **37**(3), 583–596 (2014)
43. Henriques, J.F., Caseiro, R., Martins, P., Batista, J.: Exploiting the circulant structure of tracking-by-detection with kernels. In: Computer Vision–ECCV 2012: 12th European conference on computer vision, Florence, Italy, October 7-13, 2012, Proceedings, Part IV 12, pp. 702–715 (2012). Springer
44. Bolme, D.S., Beveridge, J.R., Draper, B.A., Lui, Y.M.: Visual object tracking using adaptive correlation filters. In: 2010 IEEE computer society conference on computer vision and pattern recognition, pp. 2544–2550 (2010). IEEE
45. Danelljan, M., Häger, G., Khan, F.S., Felsberg, M.: Discriminative scale space tracking. IEEE Trans. Pattern Anal. Mach. Intell. **39**(8), 1561–1575 (2016)
46. Lukežič, A., Vojíř, T., Zajc, L., Matas, J., Kristan, M.: Discriminative correlation filter tracker with channel and spatial reliability. Int. J. Comput. Vision **126**(7), 671–688 (2018). https://doi.org/10.1007/s11263-017-1061-3
47. Lee, D.-H.: CNN-based single object detection and tracking in videos and its application to drone detection. Multimedia Tools Appl. **80**(26), 34237–34248 (2021). https://doi.org/10.1007/s11042-020-09924-0
48. Leal-Taixé, L., Canton-Ferrer, C., Schindler, K.: Learning by tracking: Siamese CNN for robust target association. In: Proceedings of the IEEE conference on computer vision and pattern recognition workshops, pp. 33–40 (2016)
49. Gan, W., Wang, S., Lei, X., Lee, M.-S., Kuo, C.-C.J.: Online CNN-based multiple object tracking with enhanced model updates and identity association. Signal Proc. Image Commun. **66**, 95–102 (2018). https://doi.org/10.1016/j.image.2018.05.008
50. Mahmoudi, N., Ahadi, S.M., Rahmati, M.: Multi-target tracking using CNN-based features: CNNMTT. Multimedia Tools Appl. **78**(6), 7077–7096 (2019). https://doi.org/10.1007/s11042-018-6467-6
51. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 580–587 (2014)
52. Kara, E., Zhang, G., Williams, J.J., Ferrandez-Quinto, G., Rhoden, L.J., Kim, M., Kutz, J.N., Rahman, A.: Deep learning based object tracking in walking droplet and granular intruder experiments. J. Real-Time Image Proc. **20**(5), 86 (2023). https://doi.org/10.1007/s11554-023-01341-4
53. Ballard, D.H.: Computer vision. Prentice-Hall, Englewood Cliffs (1982)
54. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 779–788 (2016)
55. Bradski, G.: The OpenCV library. Dr. Dobb's J. Softw. Tools (2000)
56. Chantara, W., Mun, J.-H., Shin, D.-W., Ho, Y.-S.: Object tracking using adaptive template matching. IEIE Trans. Smart Proc. Comput. **4**, 1–9 (2015). https://doi.org/10.5573/IEIESPC.2015.4.1.001
57. Yan, B., Xiao, L., Zhang, H., Xu, D., Ruan, L., Wang, Z., Zhang, Y.: An adaptive template matching-based single object tracking algorithm with parallel acceleration. J. Vis. Commun. Image Represent. **64**, 102603 (2019). https://doi.org/10.1016/j.jvcir.2019.102603
58. Li, Z., Gao, S., Nai, K.: Robust object tracking based on adaptive templates matching via the fusion of multiple features. J. Vis. Commun. Image Represent. **44**, 1–20 (2017). https://doi.org/10.1016/j.jvcir.2017.01.012
59. Horng, W.-B., Chen, C.-Y.: A real-time driver fatigue detection system based on eye tracking and dynamic template matching. J. Appl. Sci. Eng. **11**, 65–72 (2008). https://doi.org/10.6180/jase.2008.11.1.09
60. Korman, S., Reichman, D., Tsur, G., Avidan, S.: Fast-match: fast affine template matching. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2331–2338 (2013)
61. Prabhakar, N., Vaithiyanathan, V., Sharma, A.P., Singh, A., Singhal, P.: Object tracking using frame differencing and template matching. Res. J. Appl. Sci. Eng. Technol. **4**(24), 5497–5501 (2012)
62. Mao, D., Cao, Y., Xu, J., Li, K.: Object tracking integrating template matching and mean shift algorithm. In: 2011 international conference on multimedia technology, pp. 3583–3586 (2011). IEEE
63. Chantara, W., Ho, Y.-S.: Object detection based on fast template matching through adaptive partition search. In: 2015 12th international joint conference on computer science and software engineering (JCSSE), pp. 1–6 (2015). IEEE
64. Hashemi, N.S., Aghdam, R.B., Ghiasi, A.S.B., Fatemi, P.: Template matching advances and applications in image analysis. (2016). arXiv:1610.07231
65. Guo, X., Liu, X., Gupta, M.K., Hou, S., Królczyk, G., Li, Z.: Machine vision-based intelligent manufacturing using a novel dual-template matching: a case study for lithium battery positioning. Int. J. Adv. Manuf. Technol. **116**(7), 2531–2551 (2021). https://doi.org/10.1007/s00170-021-07649-4

66. Kuo, C.-F.J., Fang, T.-Y., Lee, C.-L., Wu, H.-C.: Automated optical inspection system for surface mount device light emitting diodes. J. Intell. Manuf. **30**(2), 641–655 (2019). https://doi.org/10.1007/s10845-016-1270-6
67. Shi, X., Diwanji, T., Mooney, K.E., Lin, J., Feigenberg, S., D'Souza, W.D., Mistry, N.N.: Evaluation of template matching for tumor motion management with cine-MR images in lung cancer patients. Med. Phys. **41**(5), 052304 (2014). https://doi.org/10.1118/1.4870978
68. Nafisi, V.R., Moradi, M.H., Nasr-Esfahani, M.H.: A template matching algorithm for sperm tracking and classification. Physiol. Meas. **26**(5), 639–651 (2005). https://doi.org/10.1088/0967-3334/26/5/006
69. Kim, T., Park, S.-R., Kim, M.-G., Jeong, S., Kim, K.-O.: Tracking road centerlines from high resolution remote sensing images by least squares correlation matching. Photogramm. Eng. Remote Sens. **70**(12), 1417–1422 (2004). https://doi.org/10.14358/PERS.70.12.1417
70. Jurie, F., Dhome, M.: Real time robust template matching. In: Marshall, D., Rosin, P.L. (eds.) The 13th British machine vision conference (BMVC '02), pp. 123–132. The British Machine Vision Association, Cardiff, United Kingdom (2002). https://inria.hal.science/inria-00548254
71. Zuehlke, D.A., Henderson, T.A., McMullen, S.A.: Machine learning using template matching applied to object tracking in video data. In: Pham, T. (ed.) Artificial intelligence and machine learning for multi-domain operations applications, vol. 11006, p. 110061 (2019). https://doi.org/10.1117/12.2518982 . International Society for Optics and Photonics
72. Wang, L., Tan, T., Hu, W.: Face tracking using motion-guided dynamic template matching. In: Proc. The 5th Asian conference on computer vision, pp. 23–25 (2002)
73. Kim, H.Y., De Araújo, S.A.: Grayscale template-matching invariant to rotation, scale, translation, brightness and contrast. In: Advances in image and video technology: second pacific rim symposium, PSIVT 2007 Santiago, Chile, December 17-19, 2007 Proceedings 2, pp. 100–113 (2007). Springer
74. Li, Y., Liu, J., Tian, J., Xu, H.: A fast rotated template matching based on point feature. In: Zhang, L., Zhang, J., Liao, M. (eds.) MIPPR 2005: SAR and multispectral image processing, vol. 6043, p. 60431 (2005). https://doi.org/10.1117/12.654932 . International Society for Optics and Photonics
75. Kim, D., Lee, D., Myung, H., Choi, H.-T.: Object detection and tracking for autonomous underwater robots using weighted template matching. In: 2012 Oceans-Yeosu, pp. 1–5 (2012). https://doi.org/10.1109/OCEANS-Yeosu.2012.6263501
76. Jaccard, P.: Étude comparative de la distribution florale dans une portion des alpes et des jura. Bull. Soc. Vaudoise Sci. Nat. **37**, 547–579 (1901)
77. Alora, J.I., Cenedese, M., Schmerling, E., Haller, G., Pavone, M.: Data-driven spectral submanifold reduction for nonlinear optimal control of high-dimensional robots. In: 2023 IEEE international conference on robotics and automation (ICRA), pp. 2627–2633 (2023). https://doi.org/10.1109/ICRA48891.2023.10160418
78. Liu, A., Axås, J., Haller, G.: Data-driven modeling and forecasting of chaotic dynamics on inertial manifolds constructed as spectral submanifolds. Chaos (2024). https://doi.org/10.1063/5.0179741
79. Guckenheimer, J., Holmes, P.: Nonlinear oscillations, dynamical systems, and bifurcations of vector fields. Appl. Math. Sci. (1983)
80. Takens, F.: Detecting strange attractors in turbulence. In: Rand, D., Young, L.-S. (eds.) Dynamical systems and turbulence, Warwick 1980, pp. 366–381. Springer, Berlin (1981)
81. Axås, J., Haller, G.: Model reduction for nonlinearizable dynamics via delay-embedded spectral submanifolds. Nonlinear Dyn. **111**(24), 22079–22099 (2023). https://doi.org/10.1007/s11071-023-08705-2
82. Sauer, T., Yorke, J.A., Casdagli, M.: Embedology. J. Stat. Phys. **65**(3), 579–616 (1991). https://doi.org/10.1007/BF01053745
83. Deyle, E.R., Sugihara, G.: Generalized theorems for nonlinear state space reconstruction. PLoS ONE **6**(3), 18295 (2011). https://doi.org/10.1371/journal.pone.0018295
84. Fraser, A.M., Swinney, H.L.: Independent coordinates for strange attractors from mutual information. Phys. Rev. A **33**, 1134–1140 (1986). https://doi.org/10.1103/PhysRevA.33.1134
85. Ponsioen, S., Pedergnana, T., Haller, G.: Automated computation of autonomous spectral submanifolds for nonlinear modal analysis. J. Sound Vib. **420**, 269–295 (2018). https://doi.org/10.1016/j.jsv.2018.01.048
86. Xu, Z., Kaszás, B., Cenedese, M., Berti, G., Coletti, F., Haller, G.: Data-driven modelling of the regular and chaotic dynamics of an inverted flag from experiments. J. Fluid Mech. **987**, 7 (2024). https://doi.org/10.1017/jfm.2024.411
87. Goza, A., Colonius, T., Sader, J.E.: Global modes and nonlinear analysis of inverted-flag flapping. J. Fluid Mech. **857**, 312–344 (2018). https://doi.org/10.1017/jfm.2018.728
88. Kim, D., Cossé, J., Huertas Cerdeira, C., Gharib, M.: Flapping dynamics of an inverted flag. J. Fluid Mech. (2013). https://doi.org/10.1017/jfm.2013.555
89. Ryu, J., Park, S.G., Kim, B., Sung, H.J.: Flapping dynamics of an inverted flag in a uniform flow. J. Fluids Struct. **57**, 159–169 (2015). https://doi.org/10.1016/j.jfluidstructs.2015.06.006
90. Boker, S.M., Rotondo, J.L., Xu, M., King, K.: Windowed cross-correlation and peak picking for the analysis of variability in the association between behavioral time series. Psychol. Methods **7**(3), 338 (2002)
91. Faltinsen, O.M., Timokha, A.N.: Sloshing. Cambridge University Press, Cambridge (2009)
92. Abramson, H.N.: The dynamic behavior of liquids in moving containers, with applications to space vehicle technology. Technical report (1966)
93. Dimitriadis, G.: Introduction to nonlinear aeroelasticity, 1st. edn. (2017)
94. Bryant, M., Garcia, E.: Modeling and testing of a novel aeroelastic flutter energy harvester. J. Vib. Acoust. **133**(1), 011010 (2011). https://doi.org/10.1115/1.4002788
95. Tail Flutter Test by Fred Haise. NASA: https://www.nasa.gov/centers/dryden/history/thisweek/ECN-2845.html
96. Hancock, G., Wright, J., Simpson, A.: On the teaching of the principles of wing flexure-torsion flutter. Aero-

naut. J. **89**(888), 285–305 (1985). https://doi.org/10.1017/S0001924000015050
97. Sharp, R., Limebeer, D.: On steering wobble oscillations of motorcycles. Proc. Inst. Mech. Eng. C J. Mech. Eng. Sci. **218**(12), 1449–1456 (2004). https://doi.org/10.1243/0954406042690434
98. Troger, H., Zeman, K.: A nonlinear analysis of the generic types of loss of stability of the steady state motion of a tractor-semitrailer. Veh. Syst. Dyn. **13**(4), 161–172 (1984). https://doi.org/10.1080/00423118408968773
99. Terkovics, N., Neild, S., Lowenberg, M., Krauskopf, B.: Bifurcation analysis of a coupled nose-landing-gear-fuselage system. J. Aircr. **51**(1), 259–272 (2014). https://doi.org/10.2514/1.C032324
100. Beregi, S., Takács, D., Gyebrószki, G., Stépán, G.: Theoretical and experimental study on the nonlinear dynamics of wheel-shimmy. Nonlinear Dyn. **98**(4), 2581–2593 (2019). https://doi.org/10.1007/s11071-019-05225-w
101. Beregi, S., Takács, D., Stépán, G.: Bifurcation analysis of wheel shimmy with non-smooth effects and time delay in the tyre-ground contact. Nonlinear Dyn. **98**(1), 841–858 (2019). https://doi.org/10.1007/s11071-019-05123-1
102. Beregi, S., Takács, D.: Analysis of the tyre-road interaction with a non-smooth delayed contact model. Multibody Sys.Dyn. **45**(2), 185–201 (2019). https://doi.org/10.1007/s11044-018-09636-2
103. Wallot, S., Mønster, D.: Calculation of average mutual information (AMI) and false-nearest neighbors (FNN) for the estimation of embedding parameters of multidimensional time series in matlab. Front. Psychol. (2018). https://doi.org/10.3389/fpsyg.2018.01679