Vision-based system identification and 3D keypoint discovery using dynamics constraints

Anonymous Author(s)
Affiliation
Address
email

Abstract

This paper introduces V-SysId, a novel method that enables simultaneous keypoint discovery, 3D system identification, and extrinsic camera calibration from an unlabeled video taken from a static camera, using only the family of equations of motion of the object of interest as weak supervision. V-SysId takes keypoint trajectory proposals and alternates between maximum likelihood parameter estimation and extrinsic camera calibration, before applying a suitable selection criterion to identify the track of interest. This is then used to train a keypoint tracking model using supervised learning. Results on a range of settings (robotics, physics, physiology) highlight the utility of this approach.

1 Introduction

An understanding of the motion and physics of objects in the real world is a hallmark of the human visual system. Humans have the ability to identify objects and their properties (eg. mass, friction, elasticity) as they move and interact in the world, due to our intuitive understanding of common trajectories, object interactions, and outcomes. This ability is typically studied under the umbrella of *intuitive physics* (5; 52; 20; 4), and often considered a critical component for machines to be able to think more like humans. In the context of machine learning systems, this ability can be distilled to a requirement for *unsupervised* 3D object localization and physical parameter estimation (also known as system identification) from a sensory stream, subject to some inductive bias or intuitive physics prior.

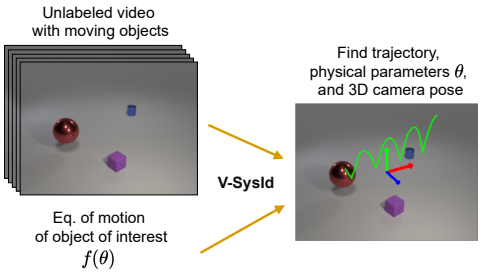


Figure 1: Given an unlabeled video containing moving objects and an equation of motion, our V-SysId identifies the trajectory of the object of interest, along with its physical parameters (e.g. restitution coefficient, initial height), and 3D pose relative to the camera.

Taking inspiration from this view, this paper introduces V-SysId, a novel method that enables simultaneous keypoint discovery, 3D system identification, and extrinsic camera calibration from a single unlabeled video taken from a static camera, using only the family of equations of motion of the object of interest as weak supervision. Crucially, our approach is able to identify the correct object(s) in a scene even in the presence of other moving objects or distractors. This property is key, as it greatly increases applicability to real world scenarios, enabling the system to solve queries like "*find the 3D location of the bouncing ball, and determine its restitution coefficient*".

Submitted to the Workshop on Physical Reasoning and Inductive Biases for the Real World at NeurIPS 2021. Do not distribute.

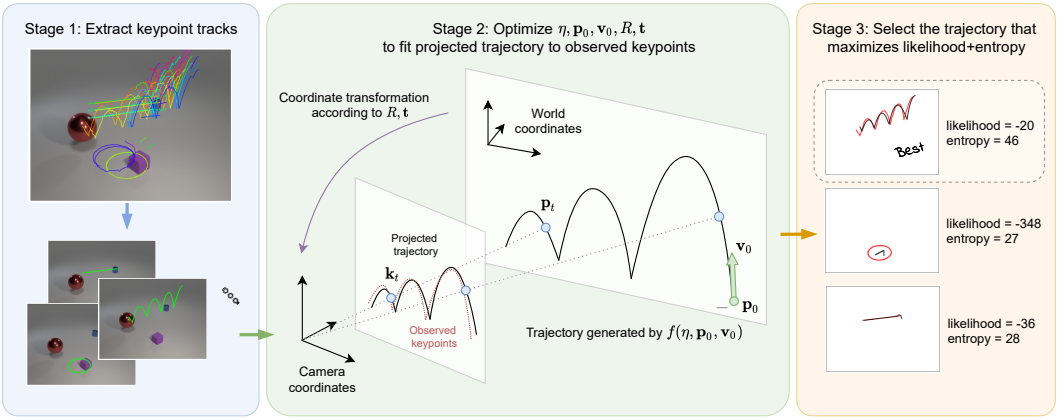


Figure 2: Our V-SysId comprises 3 stages. Stage 1 extracts keypoint tracks from a video using a grid keypoint detector + KLT tracking. Each of these 2D tracks is passed to Stage 2, where the physical parameters $\theta = \{\boldsymbol{\eta}, \mathbf{p}_0, \mathbf{v}_0\}$ of the 3D equation of motion f, and the camera pose parameters $R, \mathbf{t}$ are optimized in order to minimize the difference between the projected 3D trajectory (black, Stage 2) and the 2D keypoint track observed (red, Stage 2). Stage 3 chooses the best trajectory and corresponding parameters as those which maximize the sum of projected likelihood and a trajectory entropy criterion. Here, a bouncing ball scene with 2 moving distractors is shown, where the bouncing ball is correctly discovered as the object that corresponds to the highest entropy motion that fits the equation of motion f.

V-SysId follows a 3-stage process of keypoint track proposal, optimization, and selection, shown in Fig. 2. The optimisation process alternates between maximum likelihood extrinsic camera calibration and maximum likelihood physical parameter estimation for motion tracks detected in video. This joint optimisation can be unstable, which we address through the inclusion of a curriculum-based optimisation strategy, alongside a maximum entropy criterion for keypoint identification. A key benefit of V-SysId is that a neural network is *not* needed for discovery or system identification in our pipeline. This means that V-SysId enables keypoint discovery with high-resolution images; and can also perform system identification in *single* videos, without the need to obtain large datasets, which is particularly useful in robotics applications, where data collection for neural network training can be laborious and time-consuming. The keypoints discovered by V-SysId can be used as pseudo-labels to train a supervised keypoint detector, for downstream tracking or control.

These properties provide significant flexibility to V-SysId, enabling its use in real world environments with important applications for control, physics understanding, and health monitoring. Specifically, we show that the V-SysId can be applied to end-effector localization and extrinsic camera calibration, bouncing ball discovery and physical property estimation, and breathing frequency estimation from chest videos - all unlabeled and without regions of interest provided a priori. This is made possible by the fact that V-SysId identifies keypoints belonging to objects of interest present in scenes, while ignoring any other moving objects or artifacts that do not follow the expected dynamical constraints. This alleviates the need for hand-crafted object segmentation methods or tricks to selectively remove parts of the image that may contain moving distractors; and allows keypoint discovery at a fraction of the computational expense of unsupervised neural methods that learn to identify and model every moving object in an image.

2 Related Work

System identification and physics understanding are key to allow machine learning agents to interact with the real world. System identification is typically performed using proprioceptive trajectory data directly, and there has been extensive research across a range of fields (29; 7; 8; 57; 56; 36) in support of this. Recent contributions include developments in physical parameter estimation (6; 9), simulator learning (42; 47), simulation alignment for robot interaction (2), trajectory generation (27) and compositionality (1; 35).

Unsupervised system identification from vision is a recent area of research that removes the requirements for trajectory data, with approaches including unsupervised physical parameter estimation (24; 31; 40), structured latent space learning (32; 19; 25), and Hamiltonian/Lagrangian learning (18; 51; 58). Unfortunately, these approaches are still relatively limited in the complexity of scene they can model, and typically restricted to toy problems and simulated environments. In this work we aim to improve upon (24; 31; 40)'s limitation to simulated environments by performing physical parameter estimation on real dynamical scenes with distractors.

The seminal GALILEO model (57) demonstrated physical system identification and simulation alignment using the Physics101 dataset (55). A key shortcoming of Galileo is that it assumes that the camera is parallel to the plane of motion, and relies on manually identified object tracks to lift the visual scenes onto object positions. In contrast V-SysId is able to simultaneously estimate 3D trajectories and camera pose relative to the scene from arbitrary camera angles, greatly increasing its applicability to real world scenes. Furthermore, V-SysId automatically identifies object tracks from keypoint proposals without needing human intervention, allowing us to automatically discover the objects of interest in video that are governed by the relevant equations of motion.

Keypoint discovery Keypoints are a natural representation for object parts, with keypoint detection and tracking one of the earliest and most studied areas of computer vision. Approaches like SIFT (37), FAST (44) and ORB (46) are still widely used to perform SLAM, SFM, VO[1] and other tracking tasks (using, e.g. a KLT tracker (50)). Given keypoint trajectories, the problem of inferring the 3D structure of a 2D trajectory using assumptions about the dynamics has been coined "trajectory triangulation" by (3; 30), who assume that objects follow a straight-line or conic-section trajectory in 3D space, and that physical parameters can be uniquely identified using multiple cameras. In contrast, our method assumes only a single static monocular view. Other approaches to infer moving object structure using motion constraints include (15; 21; 11; 48).

When it comes to 2D keypoint discovery, several recent works have proposed neural network based methods that use a regularized reconstruction objective to discover objects of interest in an image (22; 23; 34; 38; 17; 10), which can be used for downstream control tasks. However, these approaches lack the ability to estimate keypoint depth, limiting their application in realistic control scenarios. Even though these approaches obtain semantically meaningful keypoints (and in some instances are able to ignore scene objects with unpredictable motion (17)), they require visual inspection in order to obtain interpretability. In contrast, V-SysId provides equation-driven keypoint discovery, ensuring a known semantic meaning for learned keypoints. A parallel stream of research tackles this from a geometric perspective, where 3D keypoints are inferred using camera motion cues or geometric constraints (49; 26; 53; 54). Even though this approach has been used in complex real world settings, these keypoints lack semantic meaning, making these unsuitable for semantic discovery queries (eg. *"find the bouncing ball following these dynamics"*).

The use of dynamics as a learning constraint has not been explored in keypoint discovery literature to date. This work proposes a method to integrate dynamical inductive biases into the keypoint discovery process, enabling extrinsic camera calibration and physics-guided discovery of objects of interest alongside the corresponding physical parameter estimation.

3 Method

Our goal is to discover the 3D trajectory of an object of interest in a video with possibly many moving objects, given only its family of motion dynamics, f. To this end, we must estimate: a) 2D keypoint locations $\mathbf{k}_t$ of the object of interest in each frame $\mathbf{I}_t$; b) physical parameters and initial conditions $\boldsymbol{\theta}$, of the equation of motion $f(\boldsymbol{\theta})$; and c) camera rotation and translation relative to the scene $[R, \mathbf{t}]$. Joint estimation of these quantities would be intractable, so we split the objective into tractable components. Our method, V-SysId, has 3 stages (Fig. 2). We first describe the physical parameter+camera pose estimation stage.

3.1 Physical parameter and camera pose estimation

Setup Let us assume we have a set of N 2D keypoint tracks $\mathbf{K} = \{\tilde{\mathbf{k}}^n_{1:T}\}_{n=1}^N$ across the video $\mathbf{I}_{1:T}$, and a family of 3D equations of motion f with unknown physical parameters $\boldsymbol{\eta}$ and initial position

[1] Simultaneous Localisation and Mapping, Structure-from-Motion, Visual Odometry.

and velocity $\mathbf{p}_0$ and $\mathbf{v}_0$, respectively. The equation f can be rolled out over T time steps using a standard integration method in order to obtain a 3D trajectory $\mathbf{p}_{1:T} = f(\boldsymbol{\theta})$, where $\boldsymbol{\theta} = \{\boldsymbol{\eta}, \mathbf{p}_0, \mathbf{v}_0\}$.

Objective Our goal is to maximize the likelihood of the observed keypoint trajectory $\tilde{\mathbf{k}}_{1:T}$ w.r.t. the physical parameters and initial conditions, $\boldsymbol{\theta}$, and the camera rotation and translation, $[R\ \mathbf{t}]$:

$$\boldsymbol{\theta}^*, R^*, \mathbf{t}^* = \underset{\boldsymbol{\theta}, R, \mathbf{t}}{\arg\max}\ p(\tilde{\mathbf{k}}_{1:T} | \boldsymbol{\theta}, R, \mathbf{t}), \tag{1}$$

where we factorize the trajectory likelihood as:

$$p(\tilde{\mathbf{k}}_{1:T} | \boldsymbol{\theta}, R, \mathbf{t}) = \prod_t p(\tilde{\mathbf{k}}_t | \boldsymbol{\theta}, R, \mathbf{t}) = \prod_t \mathcal{N}(\tilde{\mathbf{k}}_t | \mathbf{k}_t(\boldsymbol{\theta}, R, \mathbf{t}), \sigma^2), \tag{2}$$

and $\mathbf{k}_t(\boldsymbol{\theta}, R, \mathbf{t})$ are the 2D projection of the simulated 3D trajectory (given by $f(\theta)$):

$$\mathbf{k}_t(\boldsymbol{\theta}, R, \mathbf{t}) = [\tilde{\mathbf{p}}_{x,t}/\tilde{\mathbf{p}}_{z,t}, \tilde{\mathbf{p}}_{y,t}/\tilde{\mathbf{p}}_{z,t}] \quad ; \quad \tilde{\mathbf{p}}_t = \mathbf{M}\,[R\ \mathbf{t}]\,\mathbf{p}_t \tag{3}$$

with $\mathbf{M}$ being the intrinsic camera matrix. In this work we assume known camera intrinsics.

In order to reduce the space of possible solutions (and therefore local minima) of Step 1 above, we restrict the camera rotation matrix R to have roll $= 0$. This means the camera cannot rotate about its projection axis, which is the case in the vast majority of settings. Using the projection plane in camera coordinates as xy and the projection axis as z, we parametrize R as $R(\alpha, \beta) = \text{EulerRotationMatrix}(\alpha, \beta, 0)$, where α, β and $\gamma = 0$ correspond to the pitch, yaw and roll Euler angles, respectively. We found that this parametrization greatly improves results and optimization stability.

Optimization To maximize (2) we apply an iterative optimization procedure. Given an initial estimate for $\boldsymbol{\theta}$, R and $\mathbf{t}$, we alternate the following steps until convergence:

1. Keeping $\boldsymbol{\theta}$ fixed, maximize (2) w.r.t. R and $\mathbf{t}$ using gradient descent;
2. Keeping R and $\mathbf{t}$ fixed, maximize (2) wrt $\boldsymbol{\theta}$ using gradient descent (with numerical or analytical (6) differentiation) or global optimizer (e.g. CEM (45); BO (39)).

Estimation of the physical parameters over the full sequence (possibly hundreds of timesteps) is prone to local minima, as the dependency on the parameters can be highly non-linear[2]. This is further affected by the use of a non-optimized camera pose at the first iteration. In order to address this, we start by performing a step of physical parameter and pose estimation on a small initial trajectory interval, T_0, adding m points to the trajectory at each iteration, as described in Algorithm 1 of Appendix B.

3.2 Trajectory proposal and Selection

Proposal In an unlabeled video, ground-truth 2D keypoints are not available, but keypoint trajectories are required to maximize the likelihood in (2). Joint estimation of physical parameters with a neural network-based keypoint detector would be hard to optimize due to the difficulty of backpropagating through physics rollouts and camera projection into a CNN in a stable manner ((24)). Therefore, we propose a simpler, more robust approach: We extract keypoints from the first frame of the video using a keypoint detector, and track them using an optical-flow-based tracker. This produces a set of 2D keypoint tracks $\tilde{\mathbf{k}}_{1:T}$, and allows physical parameter+pose estimation to be performed for each track independently.

Selection Once the physical parameters and pose are estimated for each keypoint track, the best tracklet can be identified by isolating the highest projection likelihood (2). However, in order to prevent trivial keypoint tracklets from being chosen (since a static keypoint will easily attain maximal likelihood), we add a temporal entropy term to the likelihood, such as the temporal standard deviation of the observed trajectory, resulting in the following selection criterion:

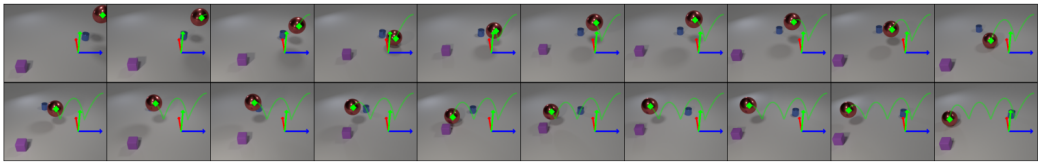
$$n^{\text{best}} = \underset{n \in 1..N}{\arg\max}\ p(\tilde{\mathbf{k}}^n_{1:T} | \boldsymbol{\theta}, R, \mathbf{t}) + \text{Stddev}_t(\tilde{\mathbf{k}}^n_{1:T}) \tag{4}$$

This finds the highest entropy trajectory that satisfies the physical motion constraints.

The full V-SysId procedure is depicted in Fig. 2 and pseudocode is shown in Algorithm 1 in Appendix B.

[2]Global optimizers have a slight advantage in this case, although they require very many iterations to find a good minimum.

Bouncing ball with unknown velocity, initial height, and restitution coefficient.



Archimedes spiral with unknown radius, radius increase rate, and angular velocity.

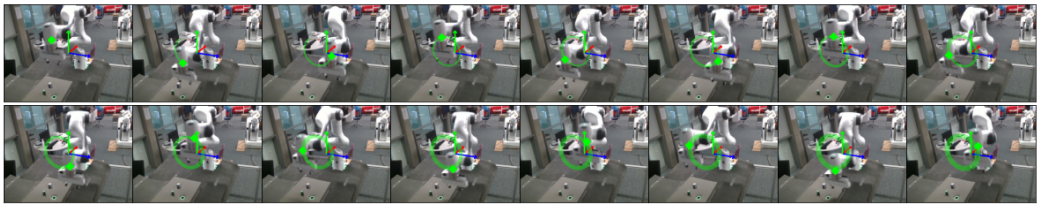


Figure 3: Discovered object and 3D perspective given the only the family of equations above as weak supervision. **Top:** Example bouncing ball scene. More scenes can be found in Fig. 9 in the Appendix. **Bottom:** Spiral robot arm end-effector in a real lab setting.

Inference at run-time Once the V-SysId procedure is complete, keypoints are available for the objects of interest in each frame in the video. These can be treated as pseudo-ground truth keypoints, and used to train a neural network (or another visual object detector) by supervised learning, in order to perform fast keypoint detection at test-time.

4 Experiments

Keypoint detection and tracking: We detect keypoints in the first frame by using taking the locations of a 10x10 grid across the frame, and use the KLT algorithm to track these across the video. We show comparisons between grid, ORB, SuperPoint and LF-Net keypoint detectors in Appendix D.

Track filtering: Since the grid keypoint detector extracts hundreds of keypoints, we remove tracks whose length is less than 60% of the full video, and whose temporal stddev (4) is less than 10 pixel, prior to optimization. This reduces computation, as physical parameter + pose estimation is performed on only the most feasible tracks.

Physical parameter estimation: The gradient-based BFGS (16) is used with numerical derivatives for physical parameter optimization. Although (6) provides an elegant method for analytical differentiation through contacts, we found it much harder to implement, and ultimately slower, than simple BFGS. Since the equations of motion considered here are planar, the z component of $\mathbf{v}_0$ is constrained to 0. The remaining parameters are learnable.

On the first iteration, the initial position $\mathbf{p}_0$ is set to be the reprojection of the first 2D keypoint $\tilde{\mathbf{k}}_0$ onto the $z = 5$ plane in world coordinates. This results in an initial position whose camera projection is the first keypoint. The initial velocity is $\mathbf{v}_0 = [0, 0, 0]$. We found these settings essential to avoid local minima in the incremental optimization.

Camera pose estimation: BFGS is also used with finite differencing for the camera pose optimization step. The parametrization of R on pitch and yaw provides a smooth objective that is easy to optimize, whereas we found the PnP algorithm to result in large and not necessarily optimal jumps between steps. We initialize the camera pose parameters as $\alpha = 0$, $\beta = 0$, and $\mathbf{t} = [0, 0, 0]$.

Curriculum-based optimization: We use 25 input frames to start the optimization, adding 10 frames per iteration until reaching the full length of the sequence.

4.1 Environments

Franka Emika Panda Robot: This sequence consists of a multi-joint robot arm (Franka Emika Panda) in a laboratory setting, where the goal is to find the end-effector's 3D location and the camera pose relative to this. The end-effector was programmed to follow an archimedes spiral in an unknown 2D plane. The spiral is described by: $r = a + b \cdot t \quad ; \quad \theta = \theta_0 + \omega \cdot t$ where r, a, b, θ_0, ω are

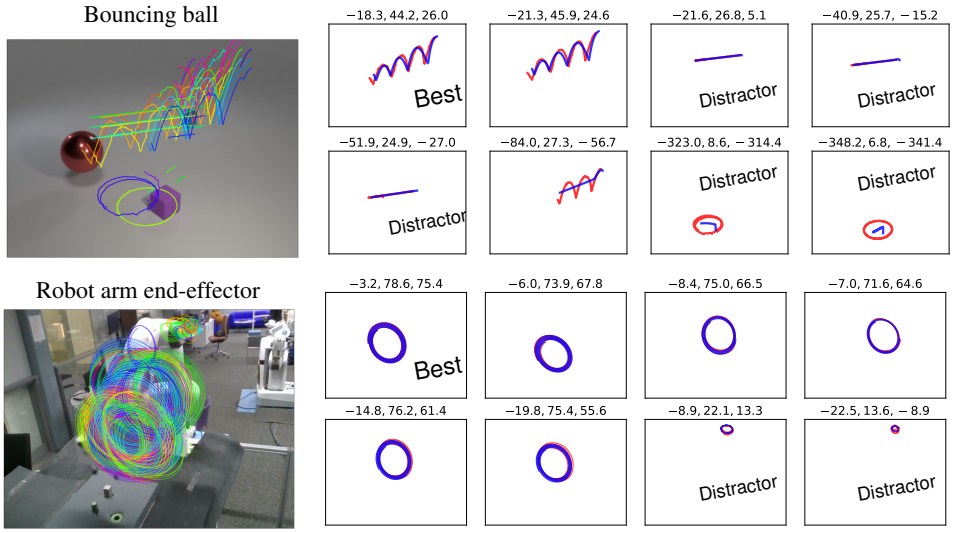


Figure 4: **Left:** Keypoint tracks propsed by a grid keypoint detector + KLT tracker (short or static tracks not shown here for improved visualization). **Right:** Subset of the extracted keypoint tracks (red) and projected fitted trajectories (blue), with the corresponding projection loglikelihood, entropy, and their sum, over each plot.

unknown parameters, to be learned by V-SysId, and t is the time in seconds. A sequence of frames for this environment can be seen on Fig. 3, bottom. The video is 250 frames long, with a resolution of 640×480.

Simulated bouncing ball: This environment consists of a simulated bouncing ball with moving distractor objects. The bouncing ball follows the equation of motion:

$$\begin{cases} a_y = -g \ , \ \text{if } y > \text{floor} \\ v_x = v_{x_0}; \quad v_z = 0; \quad v_y = -\epsilon \, v_y \ , \ \text{if } y = \text{floor} \end{cases} \tag{5}$$

where a is the acceleration, v is the velocity, y is the ball height, $\epsilon \in [0, 1]$ is the restitution coefficient, and $g = 9.8$ is the gravity. The ball moves in the $z = z_0$ plane with constant horizontal velocity, with the pose parameters $R, \mathbf{t}$ being responsible for correctly inferring the location of this plane relative to the camera. Photorealistic scenes are rendered in Blender following the Clevr protocol (28), and trajectories are rolled out using Euler integration.

There are two distractor objects on the floor scene, one moving in a circle, and another in a straight line. This environment is used to obtain thorough quantitative results regarding the physical parameter and camera pose estimation abilities of V-SysId. To this end we generate 108 sequences along the following factors of variation: initial height; initial horizontal velocity; restitution coefficient; camera location; moving/static distractor objects. The physical parameters y_0, v_{y_0}, v_{x_0}, η, and floor height are unknown, and discovered by the optimization process of V-SysId. The sequences are 120 frames long, with a resolution of 320×240.

4.2 Visualizing keypoint proposal and optimization

We start by visually exploring the results obtained by V-SysId on the spiral robot and bouncing ball datasets. Fig. 3 shows the keypoints discovered for two of the scenes. These show that V-SysId correctly identifies objects of interest according to the given equation of motion.

The keypoint proposal and selection process is visualized further in Fig. 4. Fig. 4 (left) shows the proposed keypoint tracks extracted at the proposal stage (Sec 3.2), and Fig. 4 (right) shows the results obtained by the optimization process (Sec 3.1) on a subset of these, ordered by their selection criterion score (the third number above each plot). The trajectory chosen by V-SysId according to the maximum entropy criterion is labeled as "Best". These figures highlight several important points: Firstly, V-SysId is successful despite the large number of distractor keypoints from the various

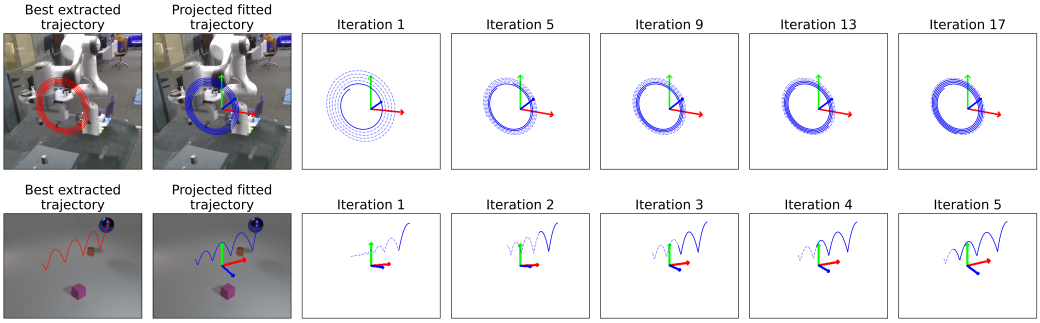


Figure 5: Visualization of the curriculum-based optimization iterations for the spiral robot (top) and bouncing ball (bottom) scenes. The red line corresponds to the extracted keypoint track and the solid blue line corresponds to the trajectory with parameters estimated so far. The dashed blue line corresponds to the predicted trajectory over the full length of the sequence, under the parameters estimated so far. We can see that the curriculum-based optimization progressively improves the physical parameter and pose estimates.

Distractors	Restitution coefficient (%)	Initial height in 3D (%)	Camera angle (°)
With	3.8 ± 1.5	9.7 ± 4.0	8.0 ± 1.8
Without	2.7 ± 0.8	6.7 ± 3.0	9.9 ± 2.6

Table 1: Relative error (percentage) between the ground-truth simulation physical parameters and camera pose, and those estimated by V-SysId, for the bouncing ball scene. Error bounds correspond to a 95% confidence interval.

moving parts of the scene (most notable in the robot arm sequence). Secondly and crucially, the optimization process and the maximum entropy criterion are able to fit and identify the best trajectory, correctly discovering the object corresponding to the motion of interest.

In order to further understand the curriculum-based optimization process, we visualize the optimization iterations of two keypoint tracks selected by V-SysId in Fig. 5. We can see that upon completion (2nd column), the orientation of the trajectory in 3D space is correctly identified by the model, and that each iteration progressively adjusts both the trajectory's shape (parametrized by the physical parameters) and the camera pose. This leads to a stable optimization procedure where both physical parameters and camera pose are identified.

4.3 Evaluating parameter estimation

Even though the scale is generally unidentifiable (this and other limitations are discussed in Sec. A), in the case of a bouncing ball both the initial height and the restitution coefficient are exactly identifiable. This allows us to compare their learned values to the ground truth values used for the simulations. In addition, we can compare the camera angles identified to those used in simulation in order to evaluate the quality of the extrinsic camera calibration.

The percentage error in restitution coefficient, initial height (distance to floor), and camera angle relative to the simulation ground-truth can be seen in Table 1. We can see that all parameters are found with a good degree of accuracy, with physical parameters being slightly more accurate than the camera pose. Notably, the errors are similar with and without moving distractors (within 95% confidence intervals), showing that V-SysId is able to correctly identify the object of interest even in the presence of distractor objects.

In order to highlight the importance of the curriculum-based optimization strategy, we compare the projection likelihood using our incremental alternate optimization with alternate optimization using the full sequence at every step. Averaging over the bouncing ball scenes, we obtain projection RMSE (pixels) of -9.31 and -109.35, respectively. A similar decrease in performance was observed when using CEM and BO optimizers. This shows that gradually increasing sequence length and using a gradient-based optimizer is key to the convergence of V-SysId.

4.4 Tracking by supervised keypoint detection

Once detected, the keypoints discovered by V-SysId can be used as pseudo-ground-truth to train a supervised keypoint detector. For the bouncing ball dataset, the training set consists of 2838 pseudo-labeled frames, and the test set consists of 948 hand-labeled frames from unseen scene configurations. For the robot dataset, the training set consists of 250 pseudo-labeled frames, and the test set consists of 150 hand-labeled frames from unseen end-effector positions. For the supervised keypoint detector, we use a fully convolutional neural network with 6 ReLU layers with 32 channels, with stride 2 on the 3rd layer, and 2 output channels with 2D softmax activation. These maps are converted to $[x, y]$ coordinates by taking the softmax-weighted mean over the output coordinate grid, as per (22). The input images have a downsampling factor of 4 relative to the original frame resolutions, but we report the keypoint error in the original image space. We train the networks for 20 epochs with batch size 16, and Adam (33) (learning rate 3×10^{-4}).

Results are shown in Table 2. The supervised keypoint detector produces highly accurate detections, confirming the quality and usability of the keypoints discovered by V-SysId even on small datasets of high-resolution scenes.

Environment	RMSE (pixel distance)
Simulated bouncing ball (240×320)	8.41 ± 1.50
Spiral robot (480×640)	3.89 ± 0.45

Table 2: Detection error on the held-out test set of the keypoints extracted by the inference neural network, after training using the keypoints discovered by V-SysId as supervision. Bounds correspond to 95% confidence interval.

We recommend reading of the supplementary materials for additional ablations experiments, and an application of V-SysId to breathing rate estimation from video.

4.5 ROI discovery in breathing videos using RANSAC

Setup To further demonstrate the applicability of V-SysId to real world scenarios, we collected 8 videos of people breathing under different pose, lightning, clothing and distractor settings, with the goal of discovering the relevant region region of the image and using it for breathing rate identification. The true breathing rate was obtained by manual annotation. Videos contain between 150 and 300 frames, at 30 fps and 480x640 resolution.

Unlike seminal work in video-based physiology and plethysmography (12), V-SysId does *not* require careful hand selection of the regions of interest and is robust to the existence of distractor motions in the scene. V-SysId simultaneously identifies the region of interest (here, the set of relevant keypoints, rather than a single one) corresponding to sinusoidal motion, and the underlying breathing rate.

Results We have seen how single keypoint discovery can be achieved using V-SysId, but the algorithm can be easily modified to allow discovery of sets of keypoints constituting a region-of-interest. We use the chest video dataset as a prototypical application. The goal is to discover the keypoints in the video corresponding to sinusoidal motion. We start by extracting keypoint tracks as in Stage 1 of V-SysId (filtering out any tracks with a temporal stddev less than 0.7), and transform these 2D tracks into 1D timeseries by taking the projection onto the 1st PCA component of the timeseries (i.e. the 2D direction of highest variance). Each timeseries is standardised, and fit to a sinusoid as per Stage 2 of V-SysId (without the 3D component). In order to identify the best set of tracks, we use a RANSAC inlier count, by measuring the error between a track's fitted sinusoid and all the other extracted tracks, and considering a track an inlier if the MSE is below 0.75. The best track is chosen according to a modified maximum entropy criterion in Stage 3, where the likelihood term is replaced by the inlier count. The ROI is defined as the set of inlier tracks of the best track.

Fig. 6 (top) shows the keypoints discovered for the 8 videos, with Fig. 6 (bottom) showing the timeseries and its sinusoidal fit for one of the keypoints in the ROI. The model correctly identifies keypoints corresponding to the chest area, while ignoring distractor and lower-body keypoints. Comparing the respiratory periods identified with V-SysId with the annotated values results in an MSE of 0.016 (in seconds/breath). In contrast, a baseline that uses the mean of the true rates for all videos obtains an MSE of 0.085. These results demonstrate the accuracy of V-SysId for physical

parameter estimation from an unknown region of interest, using only the knowledge that the motion of interest is sinusoidal as supervision.

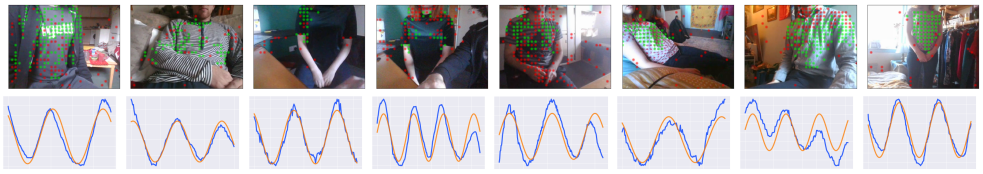


Figure 6: **Top:** Green dots correspond to keypoints identified by V-SysId as relevant for determining the breathing rate. The red dots are discarded keypoints. Note that some the videos contain distractors that move in the scene (rollouts of scenes are shown in Fig. 10 in Appendix). V-SysId with RANSAC is able to automatically discover regions of interest. **Bottom:** Timeseries (blue) and sinusoidal fit (orange) of one keypoint in the ROI for each of the scenes.

5 Conclusion and future work

This paper has introduced V-SysId, a 3-stage method for dynamics-constrained keypoint discovery and system identification, which alternates between maximum likelihood extrinsic camera calibration and maximum likelihood physical parameter estimation for motion tracks detected in video. We enhance the stability of this optimization through the inclusion of a curriculum-based optimisation strategy, alongside a maximum entropy selection criterion for keypoint identification. Future avenues of work include extensions to multiple interacting objects, rigid or fluid body dynamics from video, and incorporation with a neural network for material and volume inference from vision.

References

[1] Ian Abraham, Gerardo De, La Torre, and Todd D Murphey. Model-Based Control Using Koopman Operators. In *RSS*, 2017.

[2] Martin Asenov, Michael Burke, Daniel Angelov, Todor Davchev, Kartic Subr, and Subramanian Ramamoorthy. Vid2Param: Modelling of Dynamics Parameters from Video. In *ICRA*, 2019.

[3] Shai Avidan and Amnon Shashua. Trajectory triangulation: 3D reconstruction of moving points from a monocular image sequence. *PAMI*, 2000.

[4] Chris L. Baker, Julian Jara-Ettinger, Rebecca Saxe, and Joshua B. Tenenbaum. Rational quantitative attribution of beliefs, desires and percepts in human mentalizing. *Nature Human Behaviour*, 2017.

[5] Peter W. Battaglia, Jessica B. Hamrick, and Joshua B. Tenenbaum. Simulation as an engine of physical scene understanding. *PNAS*, 2013.

[6] Filipe De A Belbute-Peres, Kevin A Smith, Kelsey R Allen, Joshua B Tenenbaum, and J Zico Kolter. End-to-End Differentiable Physics for Learning and Control. In *NIPS*, 2018.

[7] Rune Brincker, Lingmi Zhang, and Palle Andersen. Modal identification of output-only systems using frequency domain decomposition. *Smart materials and structures*, 2001.

[8] Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the national academy of sciences*, 2016.

[9] Miles Cranmer, Alvaro Sanchez-Gonzalez, Peter Battaglia, Rui Xu, Kyle Cranmer, David Spergel, and Shirley Ho. Discovering Symbolic Models from Deep Learning with Inductive Biases. In *NeurIPS*, 2020.

[10] Neha Das, Sarah Bechtle, Todor Davchev, Dinesh Jayaraman, Akshara Rai, and Franziska Meier. Model-Based Inverse Reinforcement Learning from Visual Demonstrations. In *CoRL*, 2020.

[11] Philip David, Daniel Dementhon, Ramani Duraiswami, and Hanan Samet. SoftPOSIT: Simultaneous Pose and Correspondence Determination. *IJCV*, 2004.

[12] Willem de Boer, Joan Lasenby, Jonathan Cameron, Rich Wareham, Shiraz Ahmad, Charlotte Roach, Ward Hills, and Richard Iles. Slp: A zero-contact non-invasive method for pulmonary function testing. In *BMVC*, 2010.

[13] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. SuperPoint: Self-Supervised Interest Point Detection and Description. In *CVPR*, 2018.

[14] Philipp Fischer, Alexey Dosovitskiy, Eddy Ilg, Philip Hausser, Caner Hazırbas, Vladimir Golkov, Patrick Van Der Smagt, Daniel Cremers, and Thomas Brox. FlowNet: Learning Optical Flow with Convolutional Networks. In *ICCV*, 2015.
[15] Andrew W Fitzgibbon and Andrew Zisserman. Multibody Structure and Motion: 3-D Reconstruction of Independently Moving Objects. 2000.
[16] R. Fletcher. *Practical Methods of Optimization*. John Wiley & Sons, Ltd, 2000.
[17] Anand Gopalakrishnan, Sjoerd Van Steenkiste, and J Urgen Schmidhuber. Usupervised Object Keypoint Learning using Local Spatial Predictability. *arxiv.org/abs/2011.12930*, 2020.
[18] Sam Greydanus, Misko Dzamba, and Jason Yosinski. Hamiltonian Neural Networks. In *NeurIPS*, 2019.
[19] Vincent Le Guen and Nicolas Thome. Disentangling Physical Dynamics from Unknown Factors for Unsupervised Video Prediction. In *CVPR*, 2020.
[20] Jessica B Hamrick, Peter W Battaglia, Thomas L Griffiths, and Joshua B Tenenbaum. Inferring mass in complex scenes by mental simulation. *Cognition*, 2016.
[21] Mei Han and Takeo Kanade. Multiple Motion Scene Reconstruction with Uncalibrated Cameras. In *PAMI*, 2003.
[22] Tomas Jakab, Ankush Gupta, Hakan Bilen, and Andrea Vedaldi. Unsupervised Learning of Object Landmarks through Conditional Image Generation. In *NIPS*, 2018.
[23] Tomas Jakab, Ankush Gupta, Hakan Bilen, and Andrea Vedaldi. Learning Landmarks from Unaligned Data using Image Translation. 7 2019.
[24] Miguel Jaques, Michael Burke, and Timothy Hospedales. Physics-as-Inverse-Graphics: Unsupervised Physical Parameter Estimation from Video. In *ICLR*, 2020.
[25] Miguel Jaques, Michael Burke, and Timothy Hospedales. NewtonianVAE: Proportional Control and Goal Identification from Pixels via Physical Latent Spaces. In *CVPR*, 2021.
[26] You-Yi Jau, Rui Zhu, Hao Su, and Manmohan Chandraker. Deep Keypoint-Based Camera Pose Estimation with Geometric Constraints. In *IROS*, 2020.
[27] Marija Jegorova, Joshua Smith, Michael Mistry, and Timothy Hospedales. Adversarial generation of informative trajectories for dynamics system identification. In *IROS*, 2020.
[28] Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *CVPR*, 2017.
[29] Jer-Nan Juang and Richard S Pappa. An eigensystem realization algorithm for modal parameter identification and model reduction. *JGCD*, 1985.
[30] Jeremy Yirmeyahu Kaminski and Mina Teicher. General trajectory triangulation. In *ECCV*. 2002.
[31] Rama Kandukuri, Jan Achterhold, Michael Moeller, and Joerg Stueckler. Learning to Identify Physical Parameters from Video Using Differentiable Physics. In *GCPR*, 2020.
[32] Maximilian Karl, Maximilian Soelch, Justin Bayer, and Patrick Van Der Smagt. Deep Variational Bayes Filters: Unsupervised Learning of State Space Models from Raw Data. In *ICLR*, 2017.
[33] Diederik Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In *ICLR*, 2014.
[34] Tejas Kulkarni, Ankush Gupta, Catalin Ionescu, Sebastian Borgeaud, Malcolm Reynolds, Andrew Zisserman, and Volodymyr Mnih. Unsupervised Learning of Object Keypoints for Perception and Control. In *NIPS*, 2019.
[35] Yunzhu Li, Hao He, Jiajun Wu, Dina Katabi, and Antonio Torralba. Learning compositional koopman operators for model-based control. *ICLR*, 2020.
[36] Yunzhu Li, Toru Lin, Kexin Yi, Daniel M Bear, Daniel L K Yamins, Jiajun Wu, Joshua B Tenenbaum, and Antonio Torralba. Visual Grounding of Learned Physical Models. In *ICML*, 2020.
[37] David G Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *IJCV*, 2004.
[38] Matthias Minderer, Chen Sun, Ruben Villegas, Forrester Cole, Kevin Murphy, and Honglak Lee. Unsupervised Learning of Object Structure and Dynamics from Videos. In *NIPS*, 2019.
[39] Jonas Mockus. *Bayesian Approach to Global Optimization*. Springer, 1989.
[40] J. Krishna Murthy, Miles Macklin, Florian Golemo, Vikram Voleti, Linda Petrini, Martin Weiss, Breandan Considine, Jérôme Parent-Lévesque, Kevin Xie, Kenny Erleben, Liam Paull, Florian Shkurti, Derek Nowrouzezahrai, and Sanja Fidler. gradSim: Differentiable simulation for system identification and visuomotor control, 9 2020.
[41] Yuki Ono, Eduard Trulls, Pascal Fua, and Kwang Moo Yi. LF-Net: Learning Local Features from Images. In *NeurIPS*, 2018.
[42] Yi-Ling Qiao, Junbang Liang, Vladlen Koltun, and Ming C Lin. Scalable Differentiable Physics for Learning and Control. In *ICML*, 2020.
[43] Santhosh K Ramakrishnan, Swarna Kamlam Ravindran, Anurag Mittal, and Iit Madras. CoMaL Tracking: Tracking Points at the Object Boundaries. In *CVPR*, 2016.

[44] Edward Rosten and Tom Drummond. Machine learning for high-speed corner detection. In *ECCV*, 2006.
[45] Reuven Y. Rubinstein. Optimization of computer simulation models with rare events. *EJOR*, 1997.
[46] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. ORB: An efficient alternative to SIFT or SURF. In *ICCV*, 2011.
[47] Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter W Battaglia. Learning to Simulate Complex Physics with Graph Networks. In *ICML*, 2020.
[48] Davide Scaramuzza, Friedrich Fraundorfer, Marc Pollefeys, and Roland Siegwart. Absolute scale in structure from motion from a single vehicle mounted camera by exploiting nonholonomic constraints. In *ICCV*, 2009.
[49] Supasorn Suwajanakorn, Noah Snavely, Jonathan Tompson, and Mohammad Norouzi. Discovery of Latent 3D Keypoints via End-to-end Geometric Reasoning. In *NeurIPS*, 2018.
[50] Carlo Tomasi and Takeo Kanade. Detection and Tracking of Point Features. Technical report, 1991.
[51] Peter Toth, Danilo J Rezende, Andrew Jaegle, Sébastien Racanière, Aleksandar Botev, and Irina Higgins. Hamiltonian Generative Networks. In *ICLR*, 2020.
[52] Tomer Ullman, Andreas Stuhlmuller, Noah Goodman, and Joshua Tenenbaum. Learning Physics from Dynamical Scenes. In *CogSci*, 2014.
[53] Sudheendra Vijayanarasimhan, Susanna Ricco, Cordelia Schmid, Rahul Sukthankar, and Katerina Fragkiadaki. SfM-Net: Learning of Structure and Motion from Video. *arXiv preprint arXiv:1704.07804*, 2017.
[54] Xingkui Wei, Yinda Zhang, Zhuwen Li, Yanwei Fu, and Xiangyang Xue. DeepSFM: Structure From Motion Via Deep Bundle Adjustment. In *ECCV*, 2020.
[55] Jiajun Wu, Joseph J Lim, Hongyi Zhang, Joshua B Tenenbaum, and William T Freeman. Physics 101: Learning physical object properties from unlabeled videos. In *BMVC*, 2016.
[56] Jiajun Wu, Erika Lu, Pushmeet Kohli, William T Freeman, and Joshua B Tenenbaum. Learning to See Physics via Visual De-animation. In *NIPS*, 2017.
[57] Jiajun Wu, Ilker Yildirim, J.J. Lim, W.T. Freeman, and J.B. Tenenbaum. Galileo : Perceiving Physical Object Properties by Integrating a Physics Engine with Deep Learning. In *NIPS*, 2015.
[58] Yaofeng Desmond Zhong and Naomi Ehrich Leonard. Unsupervised Learning of Lagrangian Dynamics from Images for Prediction and Control. In *NeurIPS*, 2020.

A Discussion: Challenges and limitations

Scale unidentifiability Due to the projection operation $\mathbf{k}_t(\boldsymbol{\theta}, R, \mathbf{t}) = [\tilde{\mathbf{p}}_{x,t}/\tilde{\mathbf{p}}_{z,t}, \tilde{\mathbf{p}}_{y,t}/\tilde{\mathbf{p}}_{z,t}]$, the 3D trajectory $\mathbf{p}_{1:T}$ can only be determined up to a scale parameter. For this reason, we evaluate the correlation between the true- and learned parameters, not the error. This is also the metric used by GALILEO and Physics101 when doing physical parameter estimation from visual trajectories. Although scale unidentifiability leads to the existence of infinitely many solutions for $\boldsymbol{\theta}$ and $\mathbf{t}$, and therefore instability with joint optimization, our use of alternate optimization steps guarantees that the algorithm converges to a single solution, as $\boldsymbol{\theta}$ and $\mathbf{t}$ are optimized conditioned on one another, not jointly (this is akin to the Expectation-Maximization algorithm, where a marginal distribution is maximized via alternate optimization of conditionals).

Broken trajectories and occlusions: In settings where classical keypoint detectors are unreliable, one can either use state-of-the-art pretrained keypoint networks, like SuperPoint (13) and LF-Net (41), or pretrain an unsupervised keypoint discovery network (22; 38; 34). However, we found show that a simple grid keypoint detector yielded more reliable tracks than using classic (SIFT, ORB, FAST), or modern (SuperPoint, LF-Net) keypoint detectors.

In settings where standard optical flow computation is unreliable, more recent models (eg. FlowNet (14)) could be used to provide flow estimates to the KLT tracker. More recent improvements to the KLT tracker (eg. CoMaL (43)) could also be used. The multi-stage, modular nature of our pipeline allows for the easy replacement of individual components, although we found standard optical flow computation to work very well in practice.

Camera roll set to zero: We found that setting the camera roll angle to zero greatly stabilized the optimization procedure. While this might be perceived as too strong of a constraint on the model, in the vast majority of real settings the camera has zero roll (i.e. it's rare for the camera to rotate around its projection axis). Therefore, imposing this constraint does not reduce the applicability of our method in the vast majority of cases, while providing improved results. Naturally, allowing roll optimization would make the model more general, but this is left as future work.

B V-SysId pseudocode

Algorithm 1 V-SysId

Input: Video V of length T
Input: Equation of motion f of the object of interest
Input: KeypointTrackExtractor # function that outputs a set of keypoint track proposals
Output: Trajectory, physical parameters, and camera pose of the object of interest

Get N keypoint track proposals
Tracks $\leftarrow$ KeypointTrackExtractor(V)

Fit physical parameters and camera pose to trajectory
SelectionCriterion $\leftarrow [\,]$
Params $\leftarrow [\,]$
for $n \in \{1...N\}$ **do**
  $\tilde{\mathbf{k}}_{1:T} \leftarrow$ Tracks$[n]$
  Initialize $\alpha \leftarrow 0$, $\beta \leftarrow 0$, $\mathbf{t} \leftarrow [0, 0, 0]$, $\mathbf{v}_0 \leftarrow [0, 0, 0]$;
  Initialize $\mathbf{p}_0$ as the projection of $\tilde{\mathbf{k}}_0$ onto the $z = 5$ plane in world coordinates;
  Initialize $\boldsymbol{\eta}$ to some sensible initial values (setting dependent);
  for $t \in \{1...T\}$ **do**
    $\boldsymbol{\theta} \leftarrow \arg\max_{\boldsymbol{\theta}}\ p(\tilde{\mathbf{k}}_{1:t}|\boldsymbol{\theta}, R, \mathbf{t})$
    $R, \mathbf{t} \leftarrow \arg\max_{R,\mathbf{t}}\ p(\tilde{\mathbf{k}}_{1:t}|\boldsymbol{\theta}, R, \mathbf{t})$
  end for
  Append $\{\boldsymbol{\theta}, R, \mathbf{t}\}$ to Params
  Append the scalar $p(\tilde{\mathbf{k}}_{1:t}|\boldsymbol{\theta}, R, \mathbf{t}) + H(\tilde{\mathbf{k}}_{1:T})$ to SelectionCriterion
end for

Trajectory selection
$n^* \leftarrow \arg\max$ SelectionCriterion
$\tilde{\mathbf{k}}^* =$ Tracks$[n^*]$
$\boldsymbol{\theta}^*, R^*, \mathbf{t}^* =$ Params$[n^*]$
return $\tilde{\mathbf{k}}^*, \boldsymbol{\theta}^*, R^*, \mathbf{t}^*$

C Evaluating future trajectory prediction

We now evaluate the ability of the optimization process of V-SysId to perform accurate tracking and prediction given a sequence of correct keypoints. At each optimization iteration, we rollout the trajectory under the current parameters (i.e. those learned with the fraction of the sequence up to that iteration), and measure the prediction error relative to the observed keypoint track. This is done in hindsight only for the trajectory chosen by V-SysId, although it could equally be done for a keypoint sequence inferred by a test-time inference neural network. The results for the bouncing ball and spiral robot are shown in Fig. 7. The curves show that the optimization process quickly converges to correct system identification, leading to correct trajectory prediction after only 2 seconds of input.

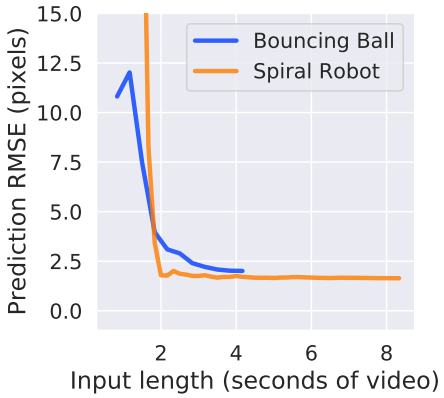


Figure 7: **Left:** Future trajectory prediction error under estimated parameters as a function of input length.

D Comparison of keypoint detectors

Here we provide a visual comparison of the trajectory proposals obtained using grid, ORB, LF-Net and SuperPoint keypoint extractors, in conjunction with a KLT tracker. Fig. 8 shows this comparison for the bouncing ball dataset (after filtering for short and static tracks). It can be seen that despite its simplicity, the grid extractor performs just as well as the more modern keypoint detectors, while running over an order of magnitude faster.

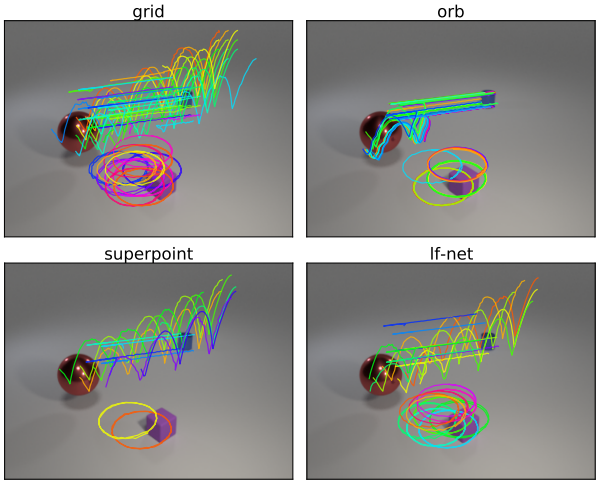


Figure 8: Comparison of various keypoint extraction methods.

E Further visualizations

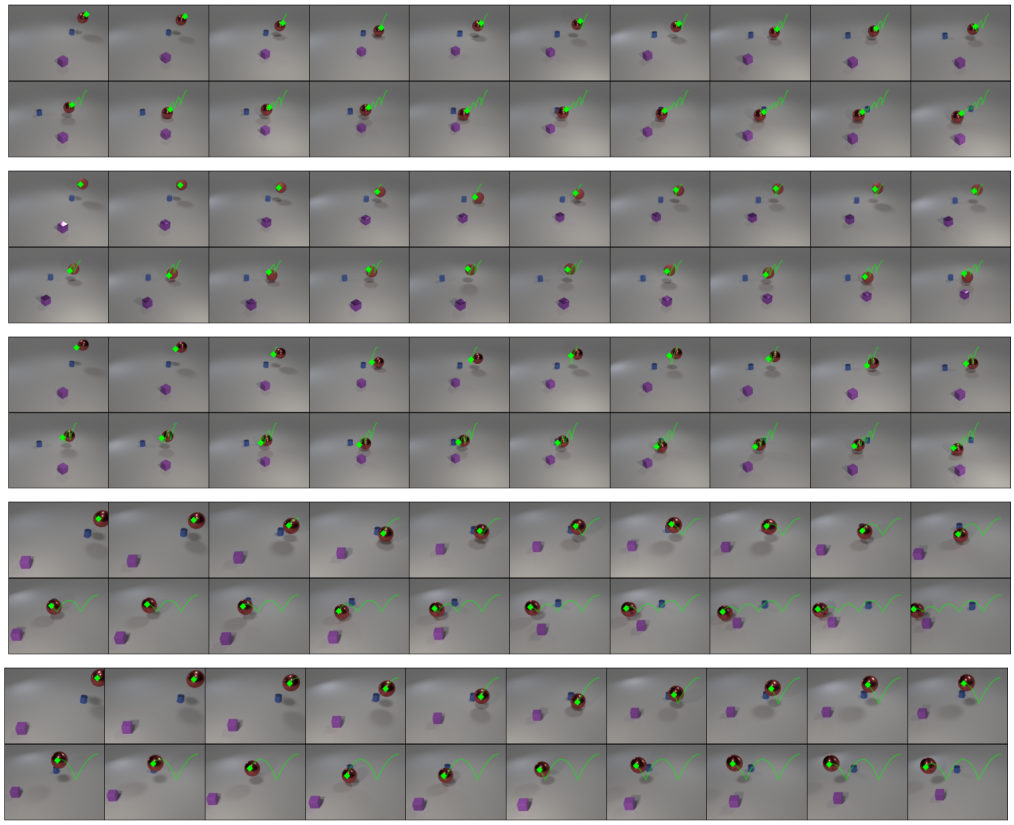


Figure 9: Comparison of various keypoints extractor and trackers on a bouncing ball scene.

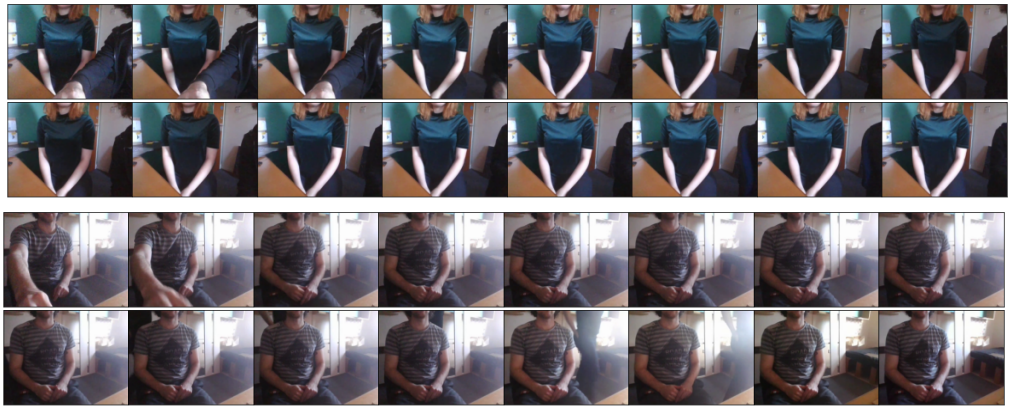


Figure 10: Frames of breathing scenes containing distractors.