Learning to Predict Keypoints and Structure of Articulated Objects without Supervision

Titas Anciukevičius
University of Edinburgh
Edinburgh, United Kingdom
Email: titas.anciukevicius@gmail.com

Paul Henderson
University of Glasgow
Glasgow, United Kingdom
Email: paul.henderson@glasgow.ac.uk

Hakan Bilen
University of Edinburgh
Edinburgh, United Kingdom
Email: hbilen@ed.ac.uk

***Abstract*—Reasoning about the structure and motion of novel object classes is a core ability in human cognition, crucial for manipulating objects and predicting their possible motion. We present a method that learns to infer the skeleton structure of a novel articulated object from a single image, in terms of joints and rigid links connecting them. The model learns without supervision from a dataset of objects having diverse structures, in different poses and states of articulation. To achieve this, it is trained to explain the differences between pairs of images in terms of a latent skeleton that defines how to transform one into the other. Experiments on several datasets show that our model predicts joint locations significantly more accurately than prior works on unsupervised keypoint discovery; moreover, unlike existing methods, it can predict varying numbers of joints depending on the observed object. It also successfully predicts the connections between joints, even for structures not seen during training.**

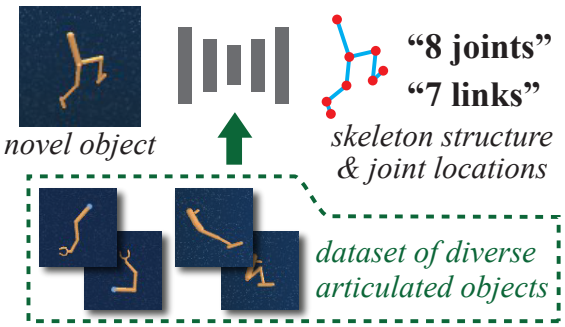

Fig. 1. Our goal is to infer the skeleton structure of an articulated object from a single image—the number of joints, their positions and connectivity. Our method learns from a dataset of objects of diverse structures, using only pairs of images, without manual supervision.

I. Introduction

Learning and predicting the structure of articulated objects from 2D images is a routine task for humans, yet it remains challenging in computer vision. While there has been great progress in predicting *keypoints* for certain object categories such as the human body [1]–[4], successful methods rely heavily on large, labeled datasets of objects. They assume these datasets contain only one object class, meaning the desired set of keypoints (*e.g.* knee, elbow, wrist) is known and fixed *a priori*. For other forms of articulated object such as animals and robots, learning object structure remains an open challenge, due to a lack of annotated datasets, and presence of diverse skeleton structures.

To address the shortage of annotated data, recent works have proposed *unsupervised* object keypoint discovery methods. These either learn to match object parts that are equivariant to geometric transformations [5]–[7], or encode and reconstruct images via structural representations incorporating keypoint locations and appearance [8]–[11]. However, these methods do not learn semantically meaningful keypoints—which for an articulated object typically correspond to *joints* in its skeleton structure. To learn those, existing methods must still rely on either (i) a post-processing step with at least a few human-annotated keypoints, or (ii) a dataset of unpaired poses to which model predictions are aligned [12].

In this work, we develop a model for unsupervised keypoint prediction, that explicitly reasons about skeleton structure, and places keypoints at joint locations (Sec. III). Thus, the keypoints are semantically meaningful. However, this is challenging, as the skeleton structure of the object shown in an image may be unknown *a priori*, and may even vary between instances of the same class (a possibility not considered in earlier works). In addition to joint locations, we therefore predict the *links* in the skeleton structure itself—*i.e.* the connectivity between joints (Fig. 1). This information is also valuable as it allows estimating the kinematics of an unfamiliar object—and thus how to manipulate it physically or reason about occluded parts.

Our model is trained without any manual supervision. Instead, it learns from unannotated pairs of images, where the two images in each pair depict the same object instance in different articulation states and/or poses (such pairs may easily be obtained from videos). The model first extracts a dense set of points belonging to the foreground object in each image, by requiring that the points from one image may be used to reconstruct the object as seen in the other image. From just one image, it then predicts a common skeleton structure whose joints are a subset of those foreground points, and whose links are optimised to best explain the remaining foreground points in both images—ensuring that every foreground point is near some link of the skeleton, and conversely that there are foreground points placed all along each link.

We evaluate the joint locations and skeleton structures predicted by our model on six diverse datasets (Sec. IV). We show qualitatively and quantitatively that, in contrast to prior unsupervised works, the keypoints predicted by our approach

are more physically meaningful and have stable identity across frames, as they correspond to joints in the skeleton structure. Moreover, our model successfully predicts structures for unseen objects, and even for objects having structures that were never seen during training.

To summarize, our contributions are:

- the first unsupervised method that can infer the skeleton structure of a novel articulated object from a single image
- the first method for unsupervised keypoint discovery that explicitly reasons about the skeleton of the object to improve the semantic meaningfulness of keypoints
- the first such method that predicts a *varying* number of keypoints depending on the object shown
- a novel approach to finding a set of foreground points with temporal correspondences, by physically transporting foreground pixels from their location in the source to that in the target frame.

II. Related work

a) Unsupervised keypoint detection: Automatic learning of object structure is an extensively studied problem (*e.g.* [13]–[15]), especially in facial landmark detection [16], [17] and human body pose estimation [18]. Recent unsupervised techniques [19], [20] learn to predict relative transformations between two images of an object. However, these techniques do not learn invariant descriptors and object parts explicitly. [5], [6] learn to explain image parts with descriptors that are invariant to geometric transformations such as thin-plate splines. [7] extends [6] with cross-instance generalisation ability by encouraging transitivity between the embeddings of different object instances. [8] learns to predict keypoint locations from a dataset of image pairs, by requiring that one image in a pair can be reconstructed from the other via a latent representation that factors the appearance and keypoint locations. [21] improves [8] by encouraging the keypoints to lie only on the estimated foreground region. [22] avoids the need for multiple views of each instance, instead learning from many images of a single object class, using a clustering-based approach. [12] extends [8] by estimating a skeleton for each image and associating the discovered keypoints with its joints. Though related to ours, this method requires an empirical skeleton prior (skeleton images obtained from real datasets). In addition, it is limited to learn one object morphology (*e.g.* only faces), while ours can learn from multiple of them. [11] uses a keypoint bottleneck to model appearance changes between two images by replacing features corresponding to the keypoints in a source image with the ones from the target image. Finally [23] and [24] instead discover keypoints from sets of 3D shapes—respectively point-clouds and meshes. These afford richer information, but are much more expensive to obtain than 2D images.

b) Unsupervised part detection: Several other works aim to discover object *parts* without supervision, but without linking those parts into a higher-level structure. [25] learns to predict part masks from pairs of frames showing the same articulated instance, by noting constraints on how the shape and appearance of parts should transform between frames. [26] instead relies on video input, and learns an explicit part-decomposed appearance model; it does not support inference on unseen videos. [27] goes further and estimates a structure matrix describing which parts move together; however, they only predict one matrix for an entire dataset (containing just one object class), instead of predicting structure per-input.

c) Unsupervised structure prediction: Most related to ours, [28] predicts keypoints using [11], and a matrix indicating which of these are 'causally linked' to infer object dynamics and the future state. This model associates pairs of parts that are predictive of each other, hence it does not incorporate any geometric information like ours. Indeed, the trajectories of two physically-connected joints are often *less* correlated than those of two non-connected joints (*e.g.* the motion of the right foot is highly predictive of the left). In addition, it assumes access to ground-truth actions (*e.g.* locations and strengths of applied forces), which convey strong side information about the keypoint locations, and relies on the keypoints of [11] which are typically more error-prone. We compare our method to [28] in Sec. IV.

III. Method

Our goal is to learn a function that predicts the skeleton structure—*i.e.* joint locations and connectivity—of a novel articulated object from a single image $\mathbf{x}$. We split this prediction into two stages. First, a neural network $\pi(\mathbf{x})$ outputs a dense set of points covering the foreground object shown in $\mathbf{x}$. Then, a second neural network $\psi(\mathbf{x}, \pi(\mathbf{x}))$ takes as input the image and foreground points, and outputs the skeleton structure—with the predicted joint locations being a subset of the foreground points. We train this model on a dataset of unannotated image pairs, each showing an articulated object in two different poses. For each pair of *source* and *target* images, we require that the foreground points from the source image can reconstruct the target image using the shape of the target object but the appearance of source object (Sec. III-A). We simultaneously require that the skeleton predicted by ψ from the source image explains the foreground points in both images (Sec. III-B).

A. Predicting foreground points

The point extractor π takes as input a single RGB image $\mathbf{x} \in \Re^{H \times W \times 3}$ of size $H \times W$.[1] It returns the pixel locations of P points covering the foreground object; the output of π should be pose- and viewpoint-invariant, *i.e.* when π is applied to two images of the same object in different poses, the p^{th} point still corresponds to the same physical location on the object (*e.g.* point p is on the nose in both images). However, this is *not* a sparse keypoint representation. We represent the position of each point $\pi(\mathbf{x})|_p \in \{1, \ldots, H\} \times \{1, \ldots, W\}$ as a one-hot categorical variable over all $H \times W$ pixels, indicating the pixel coordinate of the point in the image. Hence, $\pi(\mathbf{x})$ outputs a tensor of size $P \times H \times W$, with each of the P slices corresponding to one foreground point, and interpreted as logits of a single categorical variable over $H \times W$ possible

[1]We implement π with a U-net architecture [29]; architectural details for all networks are in the supplementary

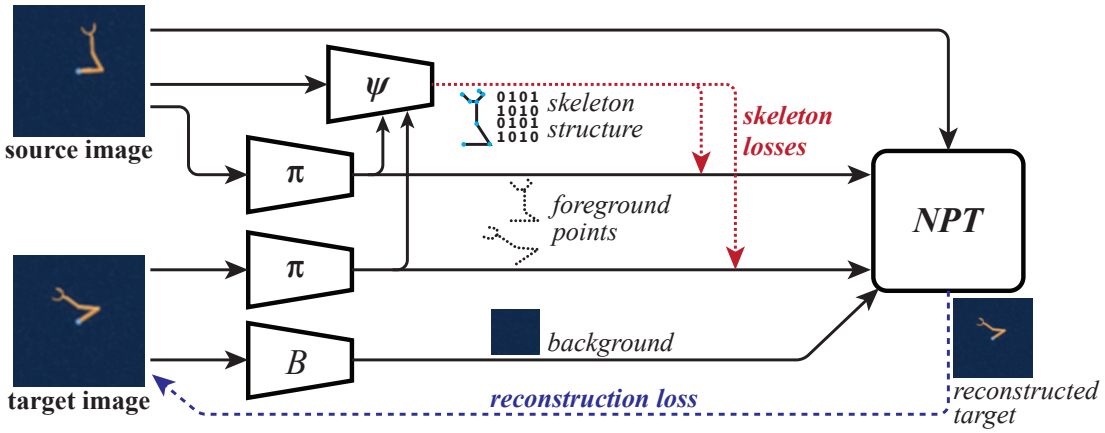

Fig. 2. Our model is trained on pairs of images, *source* and *target*. It has three encoder networks: π predicts a list foreground points from each image; ψ predicts the skeleton structure from the source image, by selecting certain foreground points to use as joints, and specifying which are connected; B predicts the background from the target image. The model is trained for two objectives. A *neural point transporter* (NPT) aims to reconstruct the target image (blue dashed arrow) using the foreground point appearances from the source image, their locations in the target image, and a predicted background. Meanwhile, the skeleton predicted by ψ must be consistent with the foreground points in both images (red dotted arrows).

spatial coordinates. To allow training by gradient descent, we relax the categorical variables to continuous Gumbel-Softmax variables [30], [31]. This representation for point positions is advantageous over using an xy coordinate (*e.g.* given by the expectation of a spatial softmax [32]) used in prior works [8], [9], [11], [12], [33]–[35]. This is because our approach yields informative non-zero gradients through *every* possible location, not just with respect to the mean location; this discourages local optima where points become stuck between two possible locations. In section IV-C, we include an ablation study showing that using [32] reduces performance significantly.

a) Losses: To train π without supervision, we require that the point locations in the target image, combined with the point appearances from the source image, can be used to accurately reconstruct the target image. Specifically, we introduce a novel *neural point transporter* (NPT), which reconstructs $\mathbf{x}_{\text{target}}$ from $\mathbf{x}_{\text{source}}$ by *transporting* the appearance at each foreground point in the source to its corresponding location in the target, and compositing these over a background. We first infer the background b_{target} from the target image itself using an auxiliary network B, *i.e.* $b_{\text{target}} = B(\mathbf{x}_{\text{target}})$. Then, for each point p, we smooth its location map $\pi(\mathbf{x}_{\text{source}})|_p$ by convolving with a 3×3 Gaussian filter to give $\tilde{\pi}_p^{\text{source}}$. Next, we take a weighted average of pixels in the source image according to the smoothed location map, *i.e.* $c_p = \left[\sum_{\omega \in \Omega} \mathbf{x}_{\text{source}}(\omega) \tilde{\pi}_p^{\text{source}}(\omega)\right] / \sum_{\omega \in \Omega} \tilde{\pi}_p^{\text{source}}(\omega)$ where ω ranges over pixels; this has the effect of 'selecting' the colour where each point is likely to be located. Next, a Gaussian splat of colour c_p is placed into a canvas C_p, by convolving the smoothed target location map with a Gaussian kernel $G_{3\times3}$ of colour c_p, *i.e.* $C_p = \pi(\mathbf{x}_{\text{target}})|_p * G_{3\times3} \cdot c_p$. Finally, these canvases are composited together with the background b_{target} using a weighted softmax [36] to give the final reconstructed image, which we denote by $\text{NPT}(\mathbf{x}_{\text{source}}, \pi(\mathbf{x}_{\text{source}}), \pi(\mathbf{x}_{\text{target}}))$. We train the model to minimise the mean squared error between the target image and its reconstruction:

$$\ell_{\text{recon}} = ||\mathbf{x}_{\text{target}} - \text{NPT}(\mathbf{x}_{\text{source}}, \pi(\mathbf{x}_{\text{source}}), \pi(\mathbf{x}_{\text{target}}))||^2. \tag{1}$$

Note that this approach differs from [11], which uses keypoints to select which source and target features are passed to the decoder. Their approach (i) allows permuting predicted keypoints without a change in loss, and (ii) assumes constant background. Hence, it merely segments foreground features that are sufficient for the decoder to reconstruct the target (see Sec. IV-B), but does not localise joints specifically.

To ensure regularity, we impose an additional loss requiring that for each foreground point, at least one of its neighbours in $\mathbf{x}_{\text{source}}$ should remain so in $\mathbf{x}_{\text{target}}$:

$$\ell_{\text{contiguity}} = -\sum_p \left\{ \max_{q \neq p} N_{pq}^{source} N_{pq}^{target} \right\} \tag{2}$$

loss where p and q index foreground points, and N^{source} and N^{target} are matrices indicating which points are neighbours, *i.e.* lie within distance β of each other. For details of how they are calculated from our categorical representation of point positions, see the supplementary.

B. Predicting joints and their connectivity

The skeleton predictor $\psi(\mathbf{x}, \pi(\mathbf{x}))$ takes as input the image $\mathbf{x}$ concatenated with its foreground points $\pi(\mathbf{x})$, and yields connection indicators $S \in \Re^{P \times P}$ between all pairs of points, *i.e.* a connectivity matrix where $S_{ij} = 1$ indicates the i^{th} and j^{th} points are connected by a link. In practice, we relax S to have values in the range $[0, 1]$. Note that the joint locations are defined implicitly by the foreground points' locations and the connectivity matrix—joints are simply foreground points that are connected to some other. This allows our model to predict a variable number of joints depending on the object.

a) Losses: We require that the skeleton S^{source} predicted from the source image be consistent with both source and target point clouds, $\pi(\mathbf{x}_{\text{source}})$ and $\pi(\mathbf{x}_{\text{target}})$, albeit with different joint angles in the target image. We impose this through three loss terms:

- ℓ_{near} encourages all foreground points p to be near some link in the skeleton. Let $d^f[p||i,j]$ denote the distance from p to the nearest location on the line segment joining points i and j (this line segment will be a link in the skeleton iff $S_{ij}^{\text{source}} = 1$). Then, we minimise:

$$\ell_{\text{near}} = -\sum_{f\in\{\text{source,target}\}} \sum_{p} \max_{ij} \left\{ S_{ij}^{\text{source}} e^{-d^f[p||i,j]} \right\} \tag{3}$$

  To ease optimisation, we replace $\max_{ij}\{x\}$ by the relaxation $\text{argmax}_{ij}\{x\} \cdot \text{softmax}_{ij}\{x\}$.

- ℓ_{sparse} enforces sparsity of the skeleton by minimising the number of links and joints:

$$\ell_{\text{sparse}} = \left(\sum_{ij} \left| S_{ij}^{\text{source}} \right|^p \right)^{1/p} + \left(\sum_{i} \left[\max_{j} S_{ij}^{\text{source}} \right]^p \right)^{1/p} \tag{4}$$

  for some constant $0 < p \leq 1$. Here the first term minimises the number of links, while the second minimises the number of points with at least one link attached—*i.e.* the number of joints.

- ℓ_{uniform} discourages links where not every location along the link has a foreground point nearby:

$$\ell_{\text{uniform}} = \sum_{f\in\{\text{source,target}\}} \left\{ \frac{\sum_{ij} S_{ij}^{\text{source}} [\delta_{ij}^f < \alpha]}{\sum_{ij} [\delta_{ij}^f < \alpha]} \right\} \tag{5}$$

  where δ_{ij}^f is the fraction of pixels along the line between the i^{th} and j^{th} points that are within distance τ of some foreground point, in the source or target image. For details of how these matrices are calculated using our categorical representation of point positions, see the supplementary material.

C. Training

We define an overall loss by summing those given by equations (1)–(5), weighting each according to a hyperparameter (these and other hyperparameters are given in the supplementary material). The neural networks π, ψ, and B are then trained to minimise this total loss, using Adam [37] on minibatches containing 8 pairs of images.

IV. Experiments

We evaluate our method on six datasets, on the tasks of skeleton prediction (Sec. IV-A) and 2D joint detection (Sec. IV-B), and demonstrate that it outperforms recent baselines on both.

a) Evaluation protocol: We emphasise that at test time, our model requires only a single frame as input. However, as in [11], to evaluate our method thoroughly, we run it on each frame of video sequences; this allows us to evaluate consistency of tracking joints over time (Sec. IV-B). In particular, we match predicted joints to ground-truth joints based on their trajectories over the full video. We measure the mean Euclidean distance between predicted and ground-truth locations, and match them with the Hungarian algorithm [38]. Also, for each video, we predict a single connectivity matrix S from one randomly-selected frame, and treat this as representative of the entire video. We binarise S to give S^b by thresholding at 0.5. During training, we randomly sample pairs of frames showing the same object.

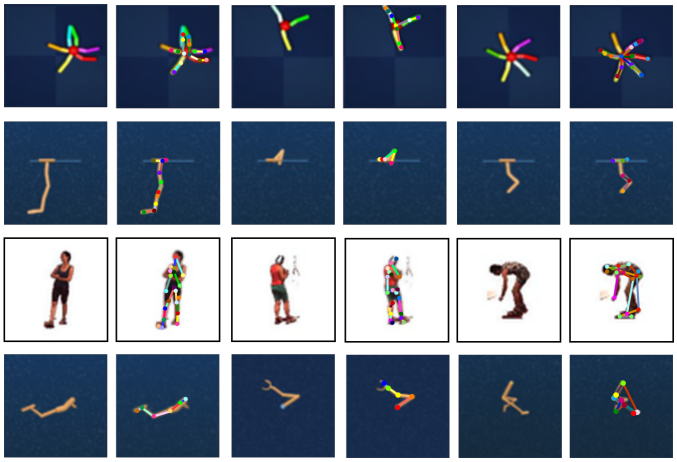

Fig. 3. Examples of keypoints and structure predicted by our model on **Spider** (top row), **Cartpole**, **Human3.6M** and a joint dataset of **Hopper + Walker + Manipulator + Cheetah** (bottom row). For each example we show the input image on the left, then our model's predicted skeleton structure and joint locations.

b) Datasets: To evaluate our method on datasets containing objects with various topologies, we use DeepMind Control Suite [39] to generate 100k episodes of 90 frames each: **1)** ***Spiders*** contains spiders with different number of legs (3 to 8); **2)** ***Cartpoles*** contains cartpoles different number of poles (1 to 5); **3)** ***Swimmers*** contains a snake-like body with 2 to 7 rods; **4)** we also create a joint dataset by combining four other provided datasets (***Walker + Hopper + Cheetah + Manipulator***) containing synthetic animals, robotic arms, and human body. To demonstrate that our method generalises to unseen topologies, we generate **5)** ***Spiders 0-Shot*** where training set contains spiders with 3, 6, 7 or 8 legs but testing set contains spiders with 4 or 5 legs. We ensure that all these datasets exhibit high variability in terms of object pose, size, colour, and movement. To demonstrate that predicted joints are temporally consistent, we include significant pose changes in each sequence. Finally, to show that our method works on real images, we use **6)** ***Human3.6M*** [40] which contains 3.6M frames showing humans performing 17 different activities. For this dataset only, we follow [9], [10] and remove the background using the unsupervised method provided with the dataset. From each dataset, we reserve 60 videos for evaluation. More details on the data generation, splits, etc., are given in the supplementary.

A. Evaluating structure

We measure our model's ability to correctly recover the skeleton structure of an object from a single image by evaluat-

TABLE I
QUANTITATIVE RESULTS ON SKELETON STRUCTURE PREDICTION.

	Ours					*Li et al. [28]*				
	Acc ↑	Bcc ↑	F_1 ↑	Prec ↑	Rec ↑	Acc ↑	Bcc ↑	F_1 ↑	Prec ↑	Rec ↑
Spiders	**0.67**	**0.55**	**0.22**	**0.16**	0.38	0.40	0.51	0.21	0.13	**0.66**
Cartpoles	**0.95**	**0.94**	**0.93**	**0.99**	**0.87**	0.56	0.56	0.47	0.41	0.54
M+W+H+C	**0.72**	**0.58**	**0.35**	**0.39**	0.32	0.58	0.50	0.26	0.21	**0.35**
Swimmers	**0.89**	**0.89**	**0.85**	**0.89**	**0.82**	0.55	0.50	0.38	0.42	0.36
Human3.6M	**0.85**	**0.58**	**0.27**	**0.33**	0.23	0.22	0.51	0.21	0.12	**0.89**
Spiders 0-Shot	**0.77**	**0.62**	**0.37**	**0.34**	0.40	0.51	0.55	0.34	0.23	**0.63**

ing whether the predicted connectivity matrix S^b matches the joint connectivities of the ground-truth skeleton, denoted S^{gt}.

a) Metrics: For each predicted joint pair i, j, we check if the predicted connectivity S^b_{ij} is equal to the ground-truth connectivity S^{gt}_{kl}, where k and l are the ground-truth joints matched to i and j. Based on this, we report the **accuracy (Acc)**, **balanced accuracy (Bcc)**, F_1 **score (F_1)**, **precision (Prec)**, and **Recall (Rec)**. For all metrics, higher is better. For all models, we select hyperparameters based on **Bcc**.

b) Baseline: We compare our approach to the recent work of [28], which aims to recover causal dynamic structure without supervision, in terms of a connectivity matrix between keypoints. We provide it with video data, which it requires at both train and test time; by contrast we use a single random frame to test our method. Note that, in addition to videos, this baseline also requires action data, which is not available for our (or most other) datasets; we therefore set these to zero. We tuned their hyperparameters on our data, and trained until convergence or a maximum of 7 days (compared with a maximum of 2 days for our model).

c) Results: The quantitative results in Tab. I show that our method outperforms the baseline on nearly all datasets and metrics. Our model accurately predicts whether links in the skeleton should be present particularly well for Cartpoles, Swimmers, and Human3.6M. We show qualitative results in Fig. 3; we see that the structure of complex objects such as spiders can be accurately predicted—links in the skeleton almost always follow the true legs of the agent. Even for the more challenging Human3.6M data, performance is reasonable (in line with the quantitative results), though there are some spurious joints present. We again emphasise that our method consistently outperforms [28], in spite of ours seeing only a single image at test time, whereas they require full videos. On three datasets, [28] outperforms our method according to recall (but underperforms on other metrics); this is due to it predicting very densely-connected skeletons, whereas ours prefers a more-realistically sparse structure. The most common failure of our model is modelling two parts with one rigid link instead of two (*e.g.* modelling a leg as one long link instead of two smaller plus a knee). This occurs when source and target points are explained equally well by both of these cases (*e.g.* when the leg is straight in both frames). Note, we analyse in the supplementary, the importance of the different losses in our model.

d) Zero-shot generalisation: In contrast to prior work, our method predicts a variable number of joints—and variable connectivity among them—depending on the object observed at test time. We therefore evaluate in an even more challenging setting, where the object at test time has a skeleton structure that was never seen during training. Specifically, we train our model on Spiders with 3, 6, 7 or 8 legs, and test on Spiders with 4 or 5 legs. The results (bottom row of Tab. I) show that our method does indeed still accurately predict object structure, with comparable performance to when all numbers of legs are seen during training.

B. Evaluating joint locations

Our model implicitly outputs joints as endpoints of the links that form the skeleton. In this section, we measure how well the detected joints match the ground-truth, in terms of the number of joints and their locations, and how accurately they are tracked over time.

a) Metrics: To evaluate the accuracy with which joints are detected and localised, we use the metric from [11]. We first match predicted and ground-truth joints as described above. If the distance between matched joints is larger than a threshold ϵ, they are disregarded as a potential match. We then use the matches to report ***precision*** (fraction of predicted joints that match ground truth), ***recall*** (fraction of ground truth joints matched with predicted joints) and $\mathbf{F_1}$, averaged per frame. To more precisely characterise how joints are tracked, we introduce two additional metrics. First, we report the average Euclidean ***distance*** between predicted and matched ground-truth joints, assuming that xy pixel coordinates are normalised to $[-1, 1]$. Secondly, to evaluate whether joints switch which ground-truth joint they are near over the video sequence, we measure tracking ***consistency***. We define this as the fraction of frames where a predicted joint is matched to the same ground-truth when matching is based on distances in one frame, as when it is based on mean distances over the full video. Finally, we report the mean absolute ***difference*** between predicted and ground-truth numbers of joints.

b) Baseline: We compare our method to the unsupervised keypoint discovery method [11], which was shown to outperform other recent works on similar datasets to ours [8], [9]. Note that [11] only predicts a fixed number of keypoints; we

TABLE II
QUANTITATIVE RESULTS ON JOINT DETECTION.

	Ours						*Kulkarni et al. [11]*					
	Dist ↓	Constcy ↑	F_1 ↑	Prec ↑	Rec ↑	Diff ↓	Dist ↓	Constcy ↑	F_1 ↑	Prec ↑	Rec ↑	Diff ↓
Spiders	**0.083**	**0.91**	**0.87**	**0.89**	**0.85**	**4.3**	0.270	0.71	0.33	0.32	0.33	4.4
Cartpoles	**0.086**	**0.92**	**0.84**	**0.86**	0.82	**1.2**	0.126	0.70	0.72	0.63	**0.85**	1.9
M+W+H+C	**0.133**	**1.0**	**0.89**	**0.98**	0.82	**1.5**	0.143	0.71	0.91	0.84	**0.99**	**1.5**
Swimmers	**0.066**	**0.96**	**0.90**	**0.96**	0.85	**0.9**	0.083	0.72	0.87	0.77	**0.98**	1.4
Human3.6M	**0.180**	**1.0**	**0.9**	**0.9**	**0.9**	0.1	0.226	**1.0**	0.86	0.86	0.86	**0.0**
Spiders 0-Shot	**0.090**	**0.98**	**0.70**	**0.91**	**0.57**	5.4	0.240	0.52	0.47	0.44	0.51	**2.5**

set this to the true maximum number of keypoints in each dataset.

c) Results: Our model significantly outperforms the baseline in terms of F_1, distance and consistency on all datasets (Tab. II). We see that joints predicted by our method are much closer to assigned ground-truth joints in terms average per-frame distance than [11]. This is further corroborated by our higher performance on F_1. We attribute the good performance of our model to the joints having a consistent physical interpretation, unlike the arbitrary keypoints of [11] and other earlier works on keypoint discovery. Fig. 4 gives examples of keypoints predicted by [11]; we see these have no physical interpretation and do not track the joints over time (contrast with Fig. 3). Our model also performs performs well according to the consistency metric (Tab. II), showing that predicted joints stay near the same ground-truth joint over time—*i.e.* the skeleton structure is temporally-consistent. Low values for [11] indicate that its keypoints often switch which ground-truth joint they are nearest to over the video sequence. In contrast to prior work, our method predicts a variable number of joints depending on the object; it therefore outperforms the baseline (which had the number fixed to the maximum number in the dataset) according to the ***Difference*** metric, except for Human3.6M which has the same number of keypoints in all images. Finally, we see that our method supports generalisation to skeleton structures with different numbers of joints than any seen during training. Specifically, our model trained on Spiders with 3, 6, 7 or 8 legs still accurately predicts joints on Spiders with 4 or 5 legs (bottom row of Tab. II).

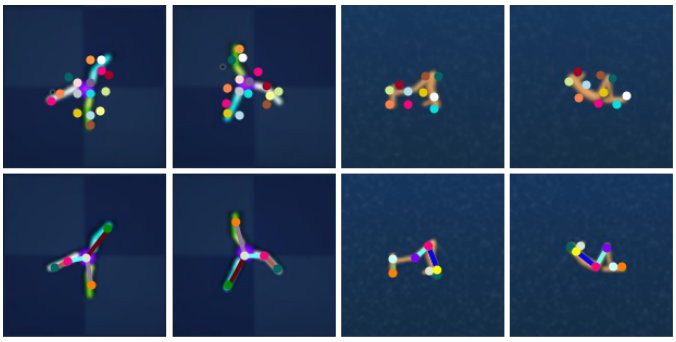

Fig. 4. Qualitative comparison of keypoint tracking over the episode. We sample two random frames from a video sequence from *Spider* (initial two columns) and *Walker+Hopper+Cheetah+Manipulator* (next two) datasets. We show keypoints predicted by the baseline of [11] (top row); and structure and joints predicted by our method (bottom row). Note that in contrast to the baseline, our model preserves joint identity (point colour) throughout two frames and predicts a variable number of keypoints. Moreover, our keypoints reliably lie on the foreground objects.

C. Ablation study – Categorical representation of point location

In contrast to prior work [8], [9], [11], [12], [33]–[35], [41]–[43], we represent the position of each point $\pi(\mathbf{x})|_p$ with a one-hot categorical variable over $H \times W$ pixel locations. This representation is theoretically advantageous over an xy coordinate representation as it receives non-zero gradients through every possible location (Sec. III-A). In Tab. III, we present experimental evidence with an ablation study, where our categorical representation is replaced with an expectation over xy coordinates given by a spatial softmax [32]. Comparing with results from our proposed categorical representation in Tab. I & II, we see that the proposed approach yields significantly higher performance in nearly all metrics (we highlight those which are better than in our main model in Tab. I & II). Without the novel representation, the model is unable to recover object structure, and its predicted keypoints are much further away from ground-truth joints.

TABLE III
ABLATION RESULTS REPLACING OUR NOVEL CATEGORICAL REPRESENTATION OF POINT POSITION WITH A PRIOR APPROACH [32]

	Structure discovery					*Keypoint discovery*					
	Acc	Bcc	F_1	Prec	Rec	Dist	Constcy	F_1	Prec	Rec	Diff
Spiders	**0.76**	0.51	0.16	0.13	0.19	0.410	0.89	0.15	0.16	0.15	4.7
Cartpoles	0.75	0.72	0.72	0.72	0.72	0.120	**1.00**	0.67	0.76	0.60	1.3
M+W+H+C	0.53	0.42	0.20	0.19	0.20	0.145	1.00	0.69	0.98	0.53	3.9
Swimmers	0.32	0.50	0.49	0.32	**1.00**	0.078	**1.00**	0.53	0.96	0.36	3.3
Human3.6M	0.85	0.52	0.12	0.20	0.09	0.305	0.89	0.54	0.45	0.66	8.0

D. Conclusion

We have introduced a model that tackles the novel task of predicting both skeleton structure and joint locations from a single image of an articulated object. We have shown that our model can be trained without manual supervision, from pairs of images in different poses. It learns to accurately predict the number of joints and their connectivity, on a variety of datasets, including for objects having structures not seen during training. Finally, it learns to predict joint locations significantly better than a recent approach to unsupervised keypoint detection, and to track them over time.

References

[1] V. Ferrari, M. Marin-Jimenez, and A. Zisserman, "Progressive search space reduction for human pose estimation," in *2008 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2008, pp. 1–8. 1

[2] A. Toshev and C. Szegedy, "Deeppose: Human pose estimation via deep neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1653–1660. 1

[3] A. Newell, K. Yang, and J. Deng, "Stacked hourglass networks for human pose estimation," in *European Conference on Computer Vision*. Springer, 2016, pp. 483–499. 1

[4] R. A. Güler, N. Neverova, and I. Kokkinos, "Densepose: Dense human pose estimation in the wild," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7297–7306. 1

[5] J. Thewlis, H. Bilen, and A. Vedaldi, "Unsupervised learning of object landmarks by factorized spatial embeddings," in *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*. IEEE Computer Society, 2017, pp. 3229–3238. [Online]. Available: https://doi.org/10.1109/ICCV.2017.348 1, 2

[6] ——, "Unsupervised learning of object frames by dense equivariant image labelling," in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, Eds., 2017, pp. 844–855. [Online]. Available: https://proceedings.neurips.cc/paper/2017/hash/cbcb58ac2e496207586df2854b17995f-Abstract.html 1, 2

[7] J. Thewlis, S. Albanie, H. Bilen, and A. Vedaldi, "Unsupervised learning of landmarks by descriptor vector exchange," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 6361–6371. 1, 2

[8] T. Jakab, A. Gupta, H. Bilen, and A. Vedaldi, "Unsupervised learning of object landmarks through conditional image generation," in *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, S. Bengio, H. M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., 2018, pp. 4020–4031. [Online]. Available: https://proceedings.neurips.cc/paper/2018/hash/1f36c15d6a3d18d52e8d493bc8187cb9-Abstract.html 1, 2, 3, 5, 6

[9] Y. Zhang, Y. Guo, Y. Jin, Y. Luo, Z. He, and H. Lee, "Unsupervised discovery of object landmarks as structural representations," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2694–2703. 1, 3, 4, 5, 6, 11

[10] D. Lorenz, L. Bereska, T. Milbich, and B. Ommer, "Unsupervised part-based disentangling of object shape and appearance," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 10 955–10 964. 1, 4, 11

[11] T. D. Kulkarni, A. Gupta, C. Ionescu, S. Borgeaud, M. Reynolds, A. Zisserman, and V. Mnih, "Unsupervised learning of object keypoints for perception and control," in *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett, Eds., 2019, pp. 10 723–10 733. [Online]. Available: https://proceedings.neurips.cc/paper/2019/hash/dae3312c4c6c7000a37ecfb7b0aeb0e4-Abstract.html 1, 2, 3, 4, 5, 6, 9, 11

[12] T. Jakab, A. Gupta, H. Bilen, and A. Vedaldi, "Self-supervised learning of interpretable keypoints from unlabelled videos," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 8784–8794. 1, 2, 3, 6

[13] T. F. Cootes, C. J. Taylor, D. H. Cooper, and J. Graham, "Active shape models-their training and application," *Computer Vision and Image Understanding*, vol. 61, no. 1, pp. 38–59, 1995. 2

[14] B. Leibe, A. Leonardis, and B. Schiele, "Combined object categorization and segmentation with an implicit shape model," in *Workshop on Statistical Learning in Computer Vision, ECCV*, vol. 2, no. 5, 2004, p. 7. 2

[15] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1627–1645, 2010. 2

[16] O. Wiles, A. S. Koepke, and A. Zisserman, "Self-supervised learning of a facial attribute embedding from video," in *British Machine Vision Conference 2018, BMVC 2018, Newcastle, UK, September 3-6, 2018*. BMVA Press, 2018, p. 302. [Online]. Available: http://bmvc2018.org/contents/papers/0288.pdf 2

[17] W. Li, H. Liao, S. Miao, L. Lu, and J. Luo, "Unsupervised learning of facial landmarks based on inter-intra subject consistencies," *2020 25th International Conference on Pattern Recognition (ICPR)*, pp. 4077–4082, 2021. 2

[18] L. Schmidtke, A. Vlontzos, S. Ellershaw, A. Lukens, T. Arichi, and B. Kainz, "Unsupervised human pose estimation through transforming shape templates," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 2484–2494. 2

[19] A. Kanazawa, D. W. Jacobs, and M. Chandraker, "Warpnet: Weakly supervised matching for single-view reconstruction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 3253–3261. 2

[20] I. Rocco, R. Arandjelovic, and J. Sivic, "Convolutional neural network architecture for geometric matching," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 6148–6157. 2

[21] A. Dundar, K. Shih, A. Garg, R. Pottorff, A. Tao, and B. Catanzaro, "Unsupervised disentanglement of pose, appearance and background from images and videos," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021. 2

[22] D. Mallis, E. Sanchez, M. Bell, and G. Tzimiropoulos, "Unsupervised learning of object landmarks via self-training correspondence," *Advances in Neural Information Processing Systems*, vol. 33, 2020. 2

[23] C. Fernandez-Labrador, A. Chhatkuli, D. P. Paudel, J. J. Guerrero, C. Demonceaux, and L. V. Gool, "Unsupervised learning of category-specific symmetric 3d keypoints from point sets," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXV 16*. Springer, 2020, pp. 546–563. 2

[24] T. Jakab, R. Tucker, A. Makadia, J. Wu, N. Snavely, and A. Kanazawa, "Keypointdeformer: Unsupervised 3d keypoint discovery for shape control," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 12 783–12 792. 2

[25] D. Lorenz, L. Bereska, T. Milbich, and B. Ommer, "Unsupervised part-based disentangling of object shape and appearance," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*. Computer Vision Foundation / IEEE, 2019, pp. 10 955–10 964. [Online]. Available: http://openaccess.thecvf.com/content_CVPR_2019/html/Lorenz_Unsupervised_Part-Based_Disentangling_of_Object_Shape_and_Appearance_CVPR_2019_paper.html 2

[26] L. Del Pero, S. Ricco, R. Sukthankar, and V. Ferrari, "Discovering the physical parts of an articulated object class from multiple videos," in *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*. IEEE Computer Society, 2016, pp. 714–723. [Online]. Available: https://doi.org/10.1109/CVPR.2016.84 2

[27] Z. Xu, Z. Liu, C. Sun, K. Murphy, W. T. Freeman, J. B. Tenenbaum, and J. Wu, "Unsupervised discovery of parts, structure, and dynamics," in *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. [Online]. Available: https://openreview.net/forum?id=rJe10iC5K7 2

[28] Y. Li, A. Torralba, A. Anandkumar, D. Fox, and A. Garg, "Causal discovery in physical systems from videos," in *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., 2020. [Online]. Available: https://proceedings.neurips.cc/paper/2020/hash/6822951732be44edf818dc5a97d32ca6-Abstract.html 2, 5

[29] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241. 2, 10

[30] E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with gumbel-softmax," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. [Online]. Available: https://openreview.net/forum?id=rkE3y85ee 3, 10

[31] C. J. Maddison, A. Mnih, and Y. W. Teh, "The concrete distribution: A continuous relaxation of discrete random variables," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. [Online]. Available: https://openreview.net/forum?id=S1jE5L5gl 3, 10

[32] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *Journal of Machine Learning Research*, vol. 17, no. 39, pp. 1–40, 2016. [Online]. Available: http://jmlr.org/papers/v17/15-522.html 3, 6, 9

[33] R. Boney, A. Ilin, and J. Kannala, "End-to-end learning of keypoint representations for continuous control from images," *arXiv preprint arXiv:2106.07995*, 2021. 3, 6

[34] A. Gopalakrishnan, S. van Steenkiste, and J. Schmidhuber, "Unsupervised object keypoint learning using local spatial predictability," in *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. [Online]. Available: https://openreview.net/forum?id=GJwMHetHc73 3, 6

[35] M. Minderer, C. Sun, R. Villegas, F. Cole, K. P. Murphy, and H. Lee, "Unsupervised learning of object structure and dynamics from videos," in *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett, Eds., 2019, pp. 92–102. [Online]. Available: https://proceedings.neurips.cc/paper/2019/hash/d82c8d1619ad8176d665453cfb2e55f0-Abstract.html 3, 6

[36] S. Liu, T. Li, W. Chen, and H. Li, "Soft rasterizer: A differentiable renderer for image-based 3d reasoning," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 7708–7717. 3

[37] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2015. [Online]. Available: http://arxiv.org/abs/1412.6980 4, 10

[38] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval Research Logistics Quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/nav.3800020109 4

[39] S. Tunyasuvunakool, A. Muldal, Y. Doron, S. Liu, S. Bohez, J. Merel, T. Erez, T. Lillicrap, N. Heess, and Y. Tassa, "dm_control: Software and tasks for continuous control," *Software Impacts*, vol. 6, p. 100022, 2020. 4, 10

[40] C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu, "Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 7, pp. 1325–1339, 2014. 4

[41] T. Karmali, A. Atrishi, S. S. Harsha, S. Agrawal, V. Jampani, and R. V. Babu, "Lead: Self-supervised landmark estimation by aligning distributions of feature similarity," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2022, pp. 623–632. 6

[42] B. Chen, P. Abbeel, and D. Pathak, "Unsupervised learning of visual 3d keypoints for control," in *International Conference on Machine Learning*. PMLR, 2021, pp. 1539–1549. 6

[43] W. Zhao, S. Zhang, Z. Guan, W. Zhao, J. Peng, and J. Fan, "Learning deep network for detecting 3d object keypoints and 6d poses," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 14 134–14 142. 6

[44] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*. PMLR, 2015, pp. 448–456. 10

[45] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel*, J. Fürnkranz and T. Joachims, Eds. Omnipress, 2010, pp. 807–814. [Online]. Available: https://icml.cc/Conferences/2010/papers/432.pdf 10

[46] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf 10

SUPPLEMENTARY MATERIAL

V. NEURAL POINT TRANSPORTER VS. [11]

Our neural point transport operation (Section III-A) is significantly different from Transporter [11], in spite of the similar name. The essential difference is that our approach transports the pixel values c_p under the location of the p^{th} predicted point in the source image to the *corresponding* location of the same point p in the target image, repeating this process for every temporally-consistent point. *This process does not use features or pixel values from the target image, only its point locations.* In contrast, [11] combines (i) *target* image features under the target keypoints and (ii) source image features suppressed at keypoint locations. Such 'transportation' suffers from significant drawbacks:

1) [11] disregards the identity of points—it simply chooses whether to take features from source or target based on whether there is some keypoint at each location. It does not enforce the fact that exactly the appearance 'under' keypoint p in the source image should be moved to the location of keypoint p in the target image. In contrast, our method ensures keypoints track locations of consistent appearance.
2) [11] can learn trivial solutions—*e.g.* consider a case when the number of keypoints is large, then [11] can learn to place 'keypoints' covering the entire image, hence ensuring that only target image features are used in the reconstruction of the target image. Hence, [11] can learn to merely segment foreground features that are sufficient for decoder to reconstruct the target. As we have shown in experiments, keypoints predicted by [11] are often a large distance away from the foreground object. By contrast, our method must learn to explicitly pair target locations with corresponding source locations.
3) [11] assumes the background is identical in the source and target images, whereas our NPT does not (it predicts the completed background from the target image only).

VI. ADDITIONAL DETAILS OF LOSSES

In this section, we give further details of how (i) the contiguity loss $\ell_{\text{contiguity}}$ (Sec. III-A of the main paper) and (ii) the uniformity loss ℓ_{uniform} (Sec. III-B of the main paper) are implemented, using our Categorical (relaxed to Gumbel-Softmax) representation of point locations. Finally, we discuss loss ℓ_{length} that ensures all links in the skeleton have plausible lengths.

a) Contiguity loss $\ell_{\text{contiguity}}$: The contiguity loss defined in Eq. 2 states that in the target image, each point should still be near at least one of its neighbours from the source image. Note that this loss is defined directly on the foreground points, not the skeleton, and discourages areas of the foreground from becoming isolated. $\ell_{\text{contiguity}}$ depends (Eq. 2) on N^{source} and N^{target}, which are square matrices with dimension equal to the number of foreground points. They indicate which pairs of points are within some distance β of each other; however, this is not straightforward to compute due to our spatial Gumbel-Softmax representation of point locations (Sec. III-A), and the requirement for differentiability. Recall $\pi(\mathbf{x}_{\text{source}})|_p$ is the location map for the p^{th} foreground point, *i.e.* a $H \times W$ Gumbel-Softmax sample, indicating at which location in the $H \times W$ image the point is located. We apply max-pooling with a kernel of size β to each $\pi(\mathbf{x}_{\text{source}})|_p$, giving $\hat{\pi}_p^{\text{source}}$. Then, for each pair of foreground points (p, q), we set

$$N_{pq}^{\text{source}} = \max_{\omega \in \Omega} \left\{ \hat{\pi}_p^{\text{source}}(\omega) \cdot \hat{\pi}_q^{\text{source}}(\omega) \right\} \tag{6}$$

where ω ranges over the $H \times W$ pixel locations. N^{target} is defined similarly, but using the locations of points in the target image instead of the source. Thus, two points are regarded as neighbours, if there exists some location in the image where the max-pooled (*i.e.* dilated) location maps for the two points are both large.

b) Uniformity loss ℓ_{uniform}: The uniformity loss defined in Eq. 5 states that every pixel location along every link in the skeleton should have some foreground point nearby. Let L_{ij}^{source} be the line segment joining the i^{th} and j^{th} foreground points in the source image; this may or may not be a link in the skeleton, depending on S_{ij}^{source}. Now, ℓ_{uniform} depends (Eq. 5) on $\delta_{ij}^{\text{source}}$, which conceptually gives the fraction of pixels along L_{ij} that are within distance τ of some foreground point. However, due to our spatial Gumbel-Softmax representation of point locations, both the start- and end-point of L_{ij} are given by distributions over all locations. We avoid converting them to xy coordinates (and thereby introducing local minima in the loss) as follows. Let $\bar{\pi}^{\text{source}} = \max_p \hat{\pi}_p^{\text{source}}$, *i.e.* the maximum over points, of max-pooled location maps; this is an $H \times W$ image indicating whether there is any foreground point near each location. Let $M_{ij}^{\text{source}} = \exp\left(-d^{\text{source}}[p||i,j]\right)$, *c.f.* Eq. 3. Then, $\delta_{ij}^{\text{source}}$ is calculated using the quantities we have just defined, as

$$\delta_{ij}^{\text{source}} = \frac{\sum_{\omega \in \Omega} M_{ij}^{\text{source}}(\omega) \bar{\pi}^{\text{source}}(\omega)}{\sum_{\omega \in \Omega} M_{ij}^{\text{source}}(\omega)}. \tag{7}$$

We define $\delta_{ij}^{\text{target}}$ analogously.

c) Length loss ℓ_{length}: The length loss encourages all links in the skeleton to have plausible lengths in the source and target images. Imposing this explicitly is computationally intractable for our Gumbel-Softmax representation of point locations; hence, for this loss only, we resort to a coordinate representation, similar to [32]. Specifically, we convert $\pi(\mathbf{x}_{\text{source}})|_p$ to an xy coordinate γ_p^{source} according to $\gamma_p^{\text{source}} = \sum_{\omega \in \Omega} \omega \pi(\mathbf{x}_{\text{source}})|_p(\omega)$, and similar for γ_p^{target}. We then have

$$\ell_{\text{length}} = \sum_{f \in \{\text{source,target}\}} \sum_{ij} S_{ij}^{\text{source}} \cdot \left\{ \max(0, l_{\min} - l_{ij}^f) + \max(0, l_{ij}^f - l_{\max}) \right\} \tag{8}$$

where ij index pairs of foreground points, *i.e.* possible links, and $l_{ij}^f = ||\gamma_i^f - \gamma_j^f||$.

VII. ABLATION STUDY – STRUCTURE LOSSES

We require the skeleton predictor $\psi(\mathbf{x}, \pi(\mathbf{x}))$ to explain the foreground object points $\pi(\mathbf{x})$ of both source and target frames

with a sparse skeleton. The sparsity requirement is achieved by minimising the number of links and joints using ℓ_{sparse} (Sec. III-A). In addition, we require that each present link of the skeleton has at least one foreground point near *every* pixel along the link using ℓ_{uniform}. This discourages links that explain point cloud well but have gaps along their length with no foreground points. In this section, we show experimentally that removing either of those losses decreases the performance of our model. In particular, we compare our model with all losses as described in the main text *versus* ablated models with either ℓ_{uniform} or ℓ_{sparse} removed. The results in Table IV show that our full model outperforms the ablated versions on nearly all metrics (marked in bold).

	S.Acc	S.Bcc	S.F_1	S.Prec	S.Rec	J.Dist	J.Constcy	J.F_1	J.Prec	J.Rec	J.Diff
ℓ_{uniform}	**0.09**	**0.064**	**0.066**	**0.132**	-0.026	**-0.0054**	-0.018	**0.104**	**0.01**	**0.158**	**-0.94**
ℓ_{sparse}	**0.13**	**0.112**	**0.166**	**0.06**	**0.206**	**-0.0072**	**0.258**	**0.152**	**0.27**	-0.05	**-6.56**

TABLE IV
QUANTITATIVE COMPARISON BETWEEN OUR MODEL VS. OUR MODEL WITH ABLATED ℓ_{uniform} LOSS (ROW 1) OR ABLATED ℓ_{sparse} LOSS (ROW 2). EACH VALUE GIVES THE DIFFERENCE BETWEEN PERFORMANCE WITH OUR MODEL AND THE ABLATED MODEL, AVERAGED OVER ALL DATASETS.

VIII. IMPLEMENTATION DETAILS

a) Network Architectures: The foreground point predictor $\pi(\mathbf{x})$ takes as input a frame x of size $64 \times 64 \times 3$ and outputs a tensor of size $P \times H \times W$, with each of the P slices interpreted as logits of a single Gumbel-Softmax [30], [31] variable over 64×64 spatial coordinates. It is implemented as a U-net architecture [29]: the first layer is a `Double-Convolution` (two 3×3 `Conv-BatchNorm-ReLU` [44], [45] layers of stride 1 and 1-padding). It is followed by 4 down steps (2×2 max-pooling of stride 2 and `Double-Convolution`) and 4 up steps (upsampling by factor of 2 and passing through `Double-Convolution`). The final layer is a 1×1 convolutional layer.

The skeleton predictor $\psi(\mathbf{x}, \pi(\mathbf{x}))$ takes as input the image $\mathbf{x}$ concatenated with its sampled foreground points (represented as heatmaps, with one channel per point), and yields connection probabilities $S \in \Re^{P \times P}$ between all pairs of points, *i.e.* a connectivity matrix where $S_{ij} = 1$ indicates the i^{th} and j^{th} points are connected by a link. Since the connectivity matrix is symmetric, the network that parametrises $\psi(\mathbf{x}, \pi(\mathbf{x}))$ outputs $\frac{P \times (P-1)}{2}$ parameters. The network is implemented as a convolutional neural network with 8 layers of `Conv-BatchNorm-LeakyReLU`. The kernel size was set to 7 for the first layer, 5 for the second, 1 for the last, and 3 for the remainder. The output of the convolutional network is passed through layer-normalization and a fully connected layer outputing $\frac{P \times (P-1)}{2}$ parameters.

The background network B takes a downsampled frame x of size $16 \times 16 \times 3$ as input and infers the background image of size $64 \times 64 \times 3$. It uses an encoder-decoder architecture, with an encoder that infers a background representation vector and a decoder mapping this to a background image. The encoder is implemented with 3 `Conv-BatchNorm-LeakyReLU` layers followed by a fully-connected network. The generator is implemented with two layers of upsampling and transpose convolutions. The limited capacity discourages B from modeling the foreground object.

b) Hyperparameters: The max-pooling kernel size used in calculating $\ell_{\text{contiguity}}$ (Sec. III-A and Sec. VI) was set to 7. The max-pooling kernel size used in calculating ℓ_{uniform} (Sec. III-B and Sec. VI) was set to 5. δ_{ij}^{f} (the fraction of pixels along the line between the i^{th} and j^{th} points that are near some foreground point) used to calculate ℓ_{uniform} was set to 0.8. The reconstruction loss was calculated over a 5-level scale-space pyramid.

The model is trained with stochastic gradient descent using Adam [37] with a learning rate of 0.005 and $\beta = (0.9, 0.999)$. To increase training stability, gradient clipping is used (both maximum gradient norm and maximum gradient value were set to 1) and weights for ℓ_{uniform} and ℓ_{sparse} were increased linearly over the first 10K iterations. The optimal weight for each loss described in Sec. VI was found using a random search over a grid of parameter values with around 20 to 100 samples per dataset. The selected weights were as follows:

	ℓ_{recon}	$\ell_{\text{contiguity}}$	ℓ_{near}	$\ell_{\text{sparse}}^{joints}$	$\ell_{\text{sparse}}^{links}$	ℓ_{uniform}	ℓ_{length}
Spiders	10	0.000	10	0.005	1	1	1000
Spiders 0-shot	10	0.000	10	0.1	2000	20	100
M+W+H+C	10	0.001	10	0.01	1000	20	1000
Cartpoles	10	0.001	10	0.1	500	10	100
Swimmers	10	0.001	10	0.1	7000	20	1000
Human3.6M	10	0.001	10	0.0002	7000	16	1000

To evaluate keypoint precision and tracking in Section IV-B, the following distance thresholds ϵ were chosen:

Dataset	Spiders	Spiders 0-shot	Human3.6M	Swimmers	M+W+H+C	Cartpoles
ϵ	0.20	0.20	0.30	0.20	0.30	0.15

We implemented our model using PyTorch [46]. Each model instance is trained on a single Nvidia Tesla P100 graphics card.

IX. DATASET GENERATION

To show that our method can generalise to different skeleton structures, we constructed 5 synthetic datasets. Each contains objects of various topologies, with varying skeletons and number of joints. Importantly, our data generation procedure ensures that models cannot 'cheat' by exploiting spatial correlations of joint positions to achieve good results on keypoint detection. For example, when 'foot' is typically at the bottom of the image, such as in ***Human3.6M***, then a model's ability to consistently track joints is poorly tested, as it can simply predict the mean location of the foot over the whole dataset. In contrast, Fig. 5 shows that our data exhibit large enough pose variation that different limbs may appear in the same region of the image in different frames, forcing the model to track them based on the the global structure and appearance.

For each of the synthetic datasets described in the main text, we use several different object models from DeepMind Control Suite [39]. To generate one video, we first sample

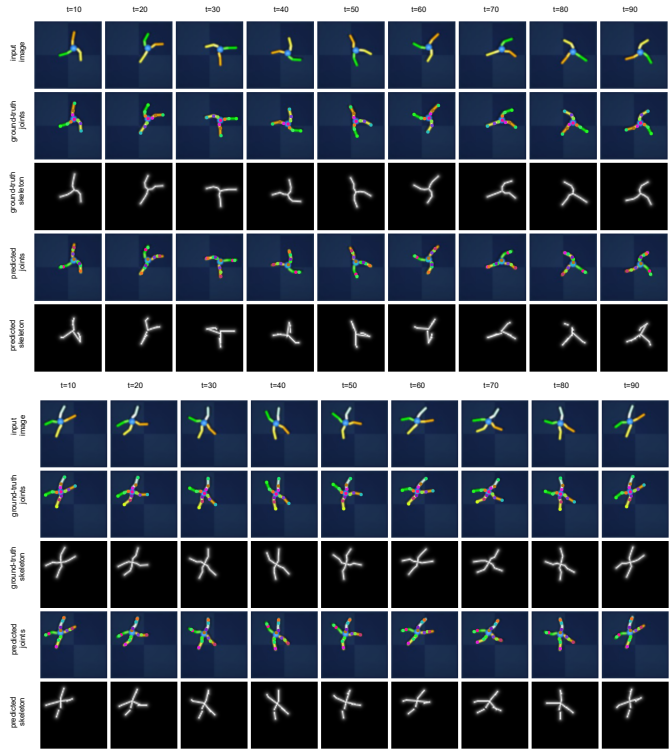

Fig. 5. To demonstrate the temporal consistency of predicted joints, we test our model on a dataset where objects vary significantly in pose over the episode (turning, spinning, falling) and in topology (different number of legs, connections). The figure shows frames from two videos for ***Spiders***, at different time steps (columns). We show input image (1) and its corresponding ground-truth joints (2) and skeleton (3) in the first three rows; as well as predicted joints (4) and predicted skeleton (5) in the last two rows. We use consistent colour of joints for each row—hence, if the colour of joint is preserved in all columns, the joint identity is preserved over time steps.

an object uniformly at random from the set of object models for that dataset. We then randomly sample its pose at the beginning of the episode by drawing a random sample of joint angles. We generate 90 frames using a random policy, taking a random action every 5 steps. For ***Walker + Hopper + Cheetah + Manipulator*** dataset, we ensure that humanoids and animals do not have consistent pose by using zero-gravity conditions—*i.e.* they are as likely to be in upside down or in horizontal poses as being in a vertical standing pose. This contrasts with previously used settings, such as in [11], where datasets contained objects whose joint positions rarely swap over time. For ***Spiders***, we make the data more challenging by randomising the colour of limbs. For all datasets, videos are rendered at 64×64 resolution. During training, we sample a pair by selecting two random frames from the episode. For each dataset, we reserve 60 videos for evaluation.

For ***Human3.6M***, we use the provided foreground masks, similar to [9], [10]; note that these are computed without supervision.