International Journal of Computer Vision (2024) 132:1986–2002
https://doi.org/10.1007/s11263-023-01946-z

SyDog-Video: A Synthetic Dog Video Dataset for Temporal Pose Estimation

Moira Shooter[1] · Charles Malleson[1] · Adrian Hilton[1]

Received: 28 April 2022 / Accepted: 27 October 2023 / Published online: 29 December 2023

Abstract
We aim to estimate the pose of dogs from videos using a temporal deep learning model as this can result in more accurate pose predictions when temporary occlusions or substantial movements occur. Generally, deep learning models require a lot of data to perform well. To our knowledge, public pose datasets containing videos of dogs are non existent. To solve this problem, and avoid manually labelling videos as it can take a lot of time, we generated a synthetic dataset containing 500 videos of dogs performing different actions using Unity3D. Diversity is achieved by randomising parameters such as lighting, backgrounds, camera parameters and the dog's appearance and pose. We evaluate the quality of our synthetic dataset by assessing the model's capacity to generalise to real data. Usually, networks trained on synthetic data perform poorly when evaluated on real data, this is due to the domain gap. As there was still a domain gap after improving the quality of the synthetic dataset and inserting diversity, we bridged the domain gap by applying 2 different methods: fine-tuning and using a mixed dataset to train the network. Additionally, we compare the model pre-trained on synthetic data with models pre-trained on a real-world animal pose datasets. We demonstrate that using the synthetic dataset is beneficial for training models with (small) real-world datasets. Furthermore, we show that pre-training the model with the synthetic dataset is the go to choice rather than pre-training on real-world datasets for solving the pose estimation task from videos of dogs.

Keywords Animal pose estimation · Synthetic data · Domain adaptation · Temporal · Deep learning

1 Introduction

Globally, 33% of households own a dog which makes it man's best friend (from Knowledge, 2016). In general, dog owners want good welfare for their dogs. They want to make sure their dog has a suitable environment, diet, the ability to interact with other animals, the ability to demonstrate normal behaviour patterns and protection from pain, suffering, injury and disease. To address the last two needs, quantifying canine locomotion and gait is essential to the diagnosis of health conditions such as lameness. Traditionally marker based motion capture systems are used to evaluate the gait of canines, however, due to the significant positive results in both human and animal deep learning pose estimation methods over the past years, it is now possible to estimate the pose of animals in a markerless manner. This not only opens applications for veterinary science (Wang et al., 2021) but also ecology (Tuia et al., 2022), robotics (Peng et al., 2020) and entertainment (Luo et al., 2022).

With the exception of Liu et al. (2021) and Russello et al. (2021), many previous animal pose estimation methods (Graving et al., 2019; Nath et al., 2019; Pereira et al., 2018) process video frames individually instead of sequences of frames in an end-to-end manner. These methods ignore valuable temporal context which can lead to inaccurate pose estimations in the event of substantial inter-frame movements and temporary occlusions. Using temporal models could produce more accurate pose estimations from videos of animals in the wild performing actions such as running and interacting with their environment and other animals.

Deep learning methods need a lot of data to perform and generalise well. While the total amount of animal data is

Communicated by Shohei Nobuhara.

✉ Moira Shooter
m.shooter@surrey.ac.uk

Charles Malleson
charles.malleson@surrey.ac.uk

Adrian Hilton
a.hilton@surrey.ac.uk

[1] Centre of Vision, Speech and Signal Processing (CVSSP), University of Surrey, Stag Hill, Guildford GU2 7XH, UK

increasing rapidly from, for example, camera traps, there is still a lack of animal pose (video) datasets and to the best of our knowledge there is no dataset that contains videos of dogs with annotated pose. StanfordExtra (Biggs et al., 2020) is the only large-scale publicly available dataset containing individual images of dogs. Usually, to create a pose dataset, humans are required to manually label a number of anatomical features such as the skeleton joints on many frames of videos. This can be labour-intensive, expensive and prone to errors; particularly when creating datasets containing data of dogs, as within the dog species there is a lot of variation. So, for the model to be able to estimate the pose across different breeds, there is a need for a large dataset with a lot of variation. To tackle the need for large datasets, several previous methods generated synthetic data as training data as it comes with benefits such as producing unlimited diverse data and accurate labels.

In this paper we estimate the pose from videos of dogs in the wild using temporal models. As there is a lack of dog video pose datasets we generate a synthetic dataset by extending on the work of SyDog (Shooter et al., 2021). We produced a synthetic dataset containing 500 videos of different dogs performing different actions labelled with 2D keypoint coordinates, bounding box coordinates and segmentation maps. We evaluate the pose estimation models with a small real-world dataset, which we called *Dogio-11* and which was produced for this work. Deep learning models trained on synthetic data usually perform poorly when evaluated on real data; this is due to the domain gap. To bridge the gap, we initially tried to improve the quality of the synthetic data. However, when evaluating on the real-data we demonstrated that the domain gap still remained, therefore we applied two different transfer-learning methods.

Summary of contribution is: (i) the generation of a large-scale synthetic dataset containing 500 videos of dogs performing different actions with labelled 2D ground truth including bounding box coordinates, keypoint coordinates and segmentation maps. (ii) We demonstrate that it is essential to pre-train models to be able to train on small video datasets (∼1k frames). Additionally, we show that models pre-trained on synthetic data perform better than models pre-trained on large sized real-world datasets. The code and dataset will be made available upon publication.

2 Related Work

Much research has looked into using synthetic data as training data for the following reasons. When creating datasets there is a need to be aware of copyrights and when it comes to humans, privacy. Additionally, manually creating datasets can result in biased datasets and it can be time-consuming, expensive and it can likely have more inaccurate annotations.

2.1 Synthetic Data

The interest in generating and using synthetic data as training data has started to grow since deep learning methods require a large amount of data. Synthetic data has been used for many computer vision tasks such as estimating optical flow (Fischer et al., 2015; Ilg et al., 2016), object detection (Borkman et al., 2021; Kiefer et al., 2021), semantic and instance segmentation (Chen et al., 2018; Gaidon et al., 2016; Park et al., 2021), pose estimation (Chen et al., 2016; Varol et al., 2017) and many more.

Different methods were used to generate computer vision synthetic datasets such as pasting 3D assets onto a real background either in a realistic (Alhaija et al., 2017; Georgakis et al., 2017) or unrealistic manner. Other methods reused 3D environments/assets from games such as GTA V produced by 3D artists (Hu et al., 2019; Hurl et al., 2019; Richter et al., 2016). This inspired other work to use game engines such as Unity 3D (Ebadi et al., 2021; González et al., 2020) and Unreal Engine (Qiu and Yuille, 2016; Tremblay et al., 2018) to create synthetic datasets.

In addition to the application of transfer-learning (Mathis et al., 2018; Sanakoyeu et al., 2020; Yu et al., 2021) and domain adaptation (Cao et al., 2019), synthetic data has been used as training data (Bolaños et al., 2021; Zuffi et al., 2016) to tackle the problem of the lack of animal pose datasets. Mu et al. (2019) created a synthetic dataset including images of different animals (10+). The poses of the animal 3D models were varied using the pre-set animation sequences that came with them. To insert diversity in the dataset, the authors randomized lighting, textures and camera viewpoints. ZooBuilder (Fangbemi et al., 2020) generated a synthetic dataset containing 170k images of cougars by rendering images of one cougar using 12 virtual cameras. They used keyframe animations to obtain various poses and to add more variation to the dataset they added real images into the background. SyDog (Shooter et al., 2021) produced a synthetic dataset containing 32k images of dogs using Unity3D. The dataset was made varied by using different dog models and adding different procedural textures to them. Additionally, different images were sampled for the background and ground geometry and similar to Mu et al. (2019) the lighting conditions and camera viewpoints were randomized. While previous work used a set of pre-set animations to obtain a varied amount of poses, the authors of SyDog were able to control the dog animations using keyboard inputs by leveraging the work from Zhang et al. (2018).

Our work extends on the work of SyDog, but instead of generating synthetic images we generate sequences of frames. Furthermore, we improve the quality of the 3D assets and environment to increase realism and reduce the domain gap.

2.2 Training with Synthetic Data

While generating synthetic data comes with its benefits, using it as training data can come with complexities, especially when it comes to high-level computer vision tasks. Usually when models are trained on synthetic data and evaluated on real data, the models perform poorly and are not able to generalise on the real data. This is called the domain gap which is caused by the synthetic and real data distributions being different. Previous work has attempted to bridge the gap using domain randomization (Tobin et al., 2017) which introduces enough diversity in the training data by randomising parameters in the simulator for the model to consider the out of domain dataset as another variation at evaluation time. Other methods proposed to refine the synthetic data using Generative Adverserial Networks (GANs) (Lee et al., 2019). However, while this might improve the quality of the data, it might not improve the performance of the models. Hence, other methods proposed to apply domain-adaptation to the features of the network, or the network itself. Recently, instead of bridging the gap in later stages of the pipeline, Wood et al. (2021) was able to solve face-related computer vision in the wild tasks by training the network solely on synthetic data. This was done by improving the quality of the synthetic data. We decided to follow the same procedure. Additionally, we have relied on domain randomization to insert diversity into the dataset. In Sect. 5 we demonstrate that there is still a domain gap between the synthetic data and real data, so we carried out different transfer-learning methods such as fine-tuning the networks trained on synthetic data and training with a mixed dataset (synthetic and real data).

3 Data and Methods

In this section we discuss how we acquired the real-world dataset (Sect. 3.1), generated the synthetic dataset (Sect. 3.2), and how the datasets were split depending on what was evaluated.

3.1 Data Acquisition

Our method is evaluated on real-world data, which we sourced from Pexels (Pexels, 2022) and the Youtube-8M dataset (Abu-El-Haija et al., 2016). We acquired 14 videos sampled at 25–30 fps and trimmed to 5-6 s each. The videos contain different types of dog breed, with different backgrounds varying in lighting and camera viewpoint. The videos were annotated with 33 body parts identical to the keypoints labelled in the synthetic dataset. We used coco-annotator (Brooks, 2018) to annotate our data. We tried to label all the 33 keypoints. However, when there was uncertainty we set the keypoint as invisible and did not annotate it.

3.2 Data Generation

Our work is an extension of SyDog (Shooter et al., 2021), however we modify the generator to synthesise sequences of frames instead of individual frames. Additionally, we improve the quality of the synthetic data by adding fur to the 3D models and integrate the model into the background using high dynamic range images (HDRIs).

3.2.1 Rendering

We generated synthetic videos using the game engine Unity3D (Haas, 2014) and took advantage of the Unity Perception package (Unity Technologies, 2020). The Perception package enables fast and accurate generation of labelled data and enables the effortless application of domain randomization. On a Windows 10 machine with 2.60 GHz 6-Core Intel Core i7, NVIDIA GeForce RTX 2070 with Max-Q Design we generated 17,500 frames labelled with 2D bounding boxes, 33 keypoint labels and segmentation maps in approximately 45 min. This time included the time to write the data to disk. The Perception package enabled us to randomise parameters such as those of the camera. The camera was positioned at various points around the dog, facing the dog's body. The focal length and aperture were varied to simulate various cameras and lenses. Additionally, the yaw of the camera was also randomised.

To light the scene we used one directional light, 2 point lights and as in Wood et al. (2021), we made use of image-based lighting with HDRIs to illuminate the 3D dog model and provide us with a background. We randomised the angle of the directional light by randomising the hour of the day, day of the year and latitude of the light. Further, we randomised the intensity and the temperature of the lights. We recommend the reader to look at Table 11 for more details. For each video we randomly sampled from a collection of 503 HDRIs (Zaal et al., n.d.). We split the HDRIs into a training and test set.

3.2.2 Dog Appearance

We used 5 different 3D dog models which varied in size (Fig. 1). For the models to work properly with the AI4Animation project (Zhang et al., 2018), used in this work, the models needed to be the same scale and shape as the default dog that came with the original project. Due to the dogs not having a similar scale and shape to the default dog, most of the 3D models failed to pose realistically when sitting or lying down, however we chose to keep the models, as

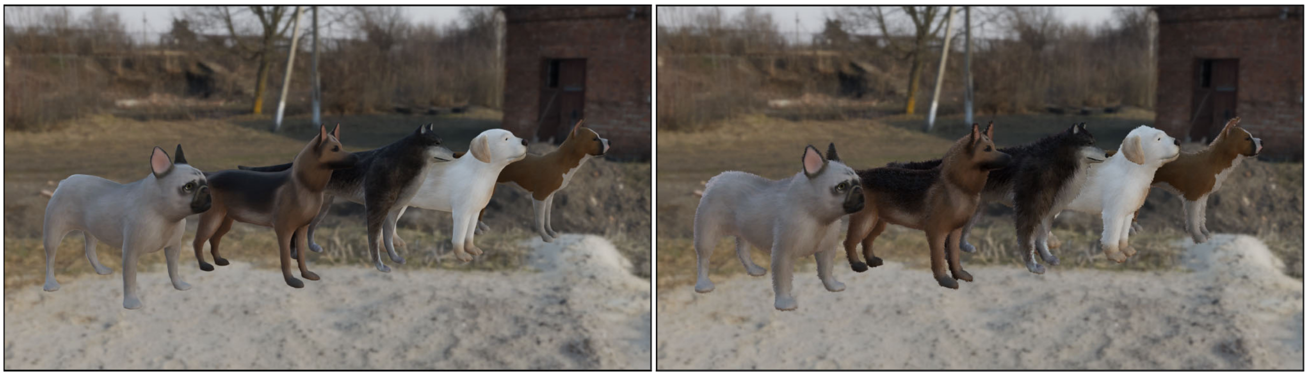
(a) Image of the 5 dog models without fur (b) Image of the 5 dog models with fur

Fig. 1 This figure shows the 5 dog models used in the synthetic generator. Best in colour (Color figure online)

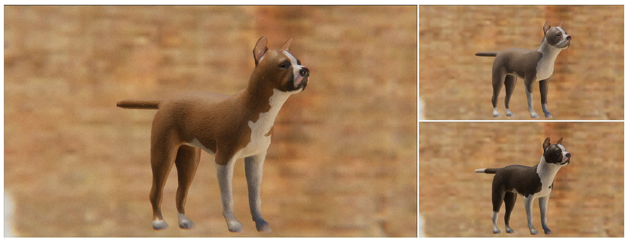
Fig. 2 Examples of hand-painted textures by 3D artist (Color figure online)

the failures were minor, in order to increase the diversity in the dataset.

A 3D artist hand painted most of the textures of the dogs (Fig. 2) in a realistic manner. In the final dataset, each 3D model had 10 different textures to sample from. In addition to the hand-painted textures we added fur to the models by using the Fluffy Grooming Tool (Zeller, 2021). The tool has gravity, wind, physics and colliders built in. However, to make the fur tool work with the Perception package, we had to convert the card-based fur into a 3D mesh. Unfortunately, this meant we could not take advantage of the gravity, wind, physics or colliders.

Initially, the 3D models were animated using the AI4 Animation project and 5 different animations could be executed—walking, running, jumping, sitting and lying down by manually pressing keyboard keys such as the WASD-keys. We implemented a Perception Package *Randomizer* to execute the animations automatically. The animations could be made repeatable by controlling the seed of the randomizer.

3.2.3 Scene Background

As mentioned in Sect. 3.2.1, we used HDRIs as background. Initially, we generated a dataset with a clean background, however we decided to also generate 3 additional synthetic datasets with different distractors/occluders in the background (Sect. 5.6). The distractors we used were 119 3D assets sourced from PolyHaven (Zaal et al., n.d.) which included props, plants and tools. For each video, these 3D assets were randomly positioned and rotated in 3D. Additionally, we sourced 3D human assets, including animations, from Adobe (2022). These human assets were randomly placed at different positions and rotated around the vertical axis on the ground geometry in the scene and were assigned a random animation such as walking, jogging, talking on the phone, breathing, clapping or waving.

3.2.4 Domain Randomization

We depended on domain randomization for the pose estimation models trained on synthetic data to generalise to real data. Parameters such as the type of fur, light conditions and background are randomised to add variety into the synthetic data. All the parameters in the synthetic dataset use a uniform distribution. Usually, for datasets with individual images we would randomise at each frame. But as we are generating video frames, the simulation environment is set to randomise at each iteration (video) instead of at each frame, therefore we implemented the code in the *OnIterationStart()* function instead of the *OnUpdate()* function (Fig. 11).

3.3 Architecture

We used the LSTM Pose Machine (Luo et al., 2017) architecture, originally developed for human pose estimation which is based on the convolutional pose machine network (Cao et al., 2018). The authors converted a multi-stage CNN to a Recurrent Neural Network (RNN). This allowed for placing Long Short-Term memory units between frames, which in turn made the network learn the temporal dependency among video frames and capture the geometric relationships of joints in time.

3.4 Training Procedure

We implemented our approach using PyTorch Lightning (Falcon et al., 2019). We extended the code from Ma (2018). We ran the experiments using a Nvidia GeForce RTX 2080 Ti GPU and tracked the training progress using Tensorboard. For all the experiments we defined the training loss as the Mean Squared Error loss (MSE). We wanted to find the optimal model for when training the models on real data in order to make a fair comparison between the models trained with synthetic data and models trained on real data. We found the optimal model by searching through the hyper-parameters space using the open source hyperparameter optimization framework Optuna (Akiba et al., 2019). When training with the synthetic dataset we set the length of the model to be 5 (i.e. $T = 5$) and set the hyper-parameters as in Luo et al. (2017), except that we set a batch-size to 2 instead of 4.

4 Experiments and Evaluation

4.1 Experiments

We execute different experiments to evaluate the quality of the synthetic data generated:

1. Train the network on real data only.
2. Train the network on synthetic data only.
3. Pre-train the network with synthetic data and then fine-tune it with real data (Fine-tuning).
4. Train the network on synthetic data and real data (Mixed training).

Furthermore, we evaluate if the model trained with synthetic data is able to generalise to real data and to dog breeds not seen by the model. Additionally, we compare the performance of models pre-trained on different types of datasets:

1. Synthetic dataset (*SyDog-Video*)
2. StanfordExtra (Biggs et al., 2020)
3. Animal Pose (Cao et al., 2019)
4. APT-36K (Yang et al., 2022)
5. ImageNet (Deng et al., 2009)

The model trained on ImageNet is modified as such: the feature extractors in the LSTM Pose Machine are replaced with pre-trained ResNets on ImageNet.

Table 1 The 33 keypoints annotated

Index	Synthetic	Real	I_{new}
0	Nose	Top head	2
1	Chin	Chin	1
2	L_eye	Nose	3
3	R_eye	L_eye	4
4	L_ear base	R_eye	5
5	L_ear tip	L_ear base	7
6	R_ear base	R_ear base	6
7	R_ear tip	L_ear tip	8
8	Head	R_ear tip	– 1
9	Top head	Top shoulders	0
10	Top shoulders	Mid spine	9
11	L_shoulder	Top hips	16
12	L_elbow	Start tail	17
13	L_wrist	Mid tail	18
14	LF_paw	End tail	19
15	R_shoulder	L_scapula	21
16	R_elbow	L_shoulder	22
17	R_wrist	L_elbow	23
18	RF_paw	L_wrist	24
19	Mid spine	LF_paw	10
20	Top hips	R_scapula	11
21	Hips	R_shoulder	– 1
22	Start tail	R_elbow	12
23	Mid tail	R_wrist	13
24	End tail	RF_paw	14
25	L_hip	L_hip	25
26	L_stiffle	L_stiffle	26
27	L_ankle	L_hock	27
28	LB_paw	LB_paw	28
29	R_hip	R_hip	29
30	R_stiffle	R_stiffle	30
31	R_ankle	R_hock	31
32	RB_paw	RB_paw	32

Index refers to the keypoints order of the synthetic dataset. I_{new} refers to the new index of the *Dogio-11* dataset. When I_{new} is -1 the keypoint coordinates are set to invisible
L left, *R* right, *LF* left front, *RF* right front, *LB* left back, *RB* right back

4.2 Datasets

4.2.1 Dogio-11

Henceforth we refer to the real dataset as *Dogio-11*. To produce *Dogio-11*, we firstly modified the real dataset keypoints to map to the keypoints of the synthetic dataset (Table 1).

The labelled dataset acquired in Sect. 4.2 contains 14 videos of different breeds: Rottweiler (1×), Labrador (1×), Husky (1×), Border Collie (5×), German Shepherd (3×),

Table 2 Number of training and test samples for the *Dogio-11* dataset (T = 5)

Dataset	Train	Test (known)	Test (unknown)	Total
Dogio-11	155	155	96	406

Chihuahua (1×), Miniature Pinscher (1×) and Mountain Cur (1×). Because we evaluate the models with within-domain data and out-of-domain data, we produce a dataset called *Dogio-11* which contains 7 dog breeds (11 videos) for training and 1 dog breed for testing (3 videos). Similarly to Russello et al. (2021), we split the videos into samples of 5 consecutive frames with no overlap. Instead of generating the samples first, we initially split the videos into training and test sets and then split the videos into samples. We refer the reader to Table 2 to have a detailed overview of how we generated and split the *Dogio-11* dataset.

We use *Dogio-11* to evaluate the model's generalisation capacity to unseen frame sequences with known types of dog breeds (known), and to unseen types of dog breeds (unkown). As mentioned earlier we split the dataset into 11 training videos and 3 testing videos. We then sampled the videos into sequences, and took a subset of 50% of random training samples to use for training—the other 50% was used for testing the within-domain robustness. The 3 testing videos, which were used to see whether the models generalised across different type of breeds, were sampled into sequences of 5 frames with no overlap and this produced a total of 96 samples.

4.2.2 SyDog-Video

From now on we refer to the synthetic dataset as *SyDog-Video*. We produce a dataset including 500 synthetic videos of 175 frames (87,500 frames). This dataset included images with an HDRI and a ground geometry (floor/terrain) but did not contain any videos with distractors/occluders such as 3D assets or 3D people. However, in Sect. 5.5 we assess the importance of adding distractors in the background.

To validate the network's performance on the synthetic dataset, we withhold one type of dog breed of the dataset. We split the dataset based on the type of dog breed. We use 4 dogs for training and 1 dog for testing. Additionally, the test dataset contains backgrounds that do not occur in the training dataset. Please refer to Table 3 for an overview of the number of training and test samples of SyDog-Video.

To establish the network's performance is attributed to its acquisition of temporal information rather than the sheer scale of the synthetic video dataset, we conducted an experiment involving pre-training the network with a non-temporal variant of the synthetic video dataset. Consequently, the input to the network consisted of a sequence comprising identical frames.

4.2.3 Animal Pose Datasets

We use the following animal pose datasets including images of animals/dogs to train the network and compare the models with the models (pre-)trained on synthetic data. We follow the same procedure as in Luo et al. (2017) where we build a single image model from the LSTM Pose Machine network. The single image model has the same structure, however, at each stage the input is the same image instead of different frames.

The *StanfordExtra* dataset (Biggs et al., 2020) is a large-scale dataset containing 12k images of 120 different dog breeds based on the Stanford Dogs dataset (Khosla et al., 2011). To train the network we split the data according to the split provided on the StanfordExtra paper (54:32:14). We evaluate the models on the StanfordExtra test dataset for validation. Additionally, the StanfordExtra dataset is used to create a mixed training dataset.

The *Animal Pose* dataset (Cao et al., 2019) is also used for training the networks and to be able to compare these networks with the models trained on synthetic data. The dataset contains more than 4k images of dogs, cats, horses, cows and sheep. Instead of using only the subset of images that contain only a single subject such as in Mathis et al. (2019) from the original dataset, we crop each image based on the bounding box coordinates which result in us having a single animal in the image. We use 80% of the dataset for training and 20% of the dataset for testing.

The *APT-36K* (Yang et al., 2022) is used for training the networks and be able to compare with the network pre-trained on synthetic data. Initially, the dataset comprised 36,000 labeled images featuring a diverse range of animals. Recognizing the value of temporal information, we performed pre-processing on the dataset to generate sequences, allowing us to leverage this temporal aspect. As a result, we obtained a final set of 3,774 sequences. The dataset was divided into training and testing sets, with 80% allocated for training purposes and the remaining 20% utilized for testing.

As mentioned in Sect. 4.1 we evaluate the models trained on the animal datasets with the *Dogio-11* test datasets before and after fine-tuning the models. 155 frames out of 410 frames (37.80%) are categorized as challenging cases due to factors like temporal occlusion or including substantial movements. In Sect. 5.4, we analyze and compare the performance on these challenging cases with that of the easier cases, as well as the overall test set.

4.3 Evaluation Metric

The Percentage of Detected Joints (PDJ) is used to evaluate the pose estimation model. The PDJ metric expresses the percentage of correct keypoints, where a predicted keypoint is considered correct if its distance to the ground-truth keypoint

Table 3 Number of training and test samples for *SyDog-Video* (T=5)

Dataset	Train	Test (known)	Test (unknown)	Total
SyDog-Video	14,000	N/A	3500	17,500

is smaller than a fraction of the bounding box diagonal. For example, PDJ@0.1 is the percentage of the keypoints within the threshold of 10 percent of the bounding box diagonal. In the equation below d_i represents the length of the bounding box diagonal of data/subject i which is calculated from ground-truth. $\mathbf{p}_k$ and $\mathbf{t}_k$ are the predicted and ground-truth location of keypoint k. And finally, Θ represents the proportional threshold.

$$PDJ@\Theta = \frac{1}{N}\sum_{i=1}^{N} \sigma(\|\mathbf{p}_k - \mathbf{t}_k\| - d_i * \Theta) \qquad (1)$$

where $\sigma(x) = 1$ when $x \leq 0$ and $\sigma(x) = 0$ otherwise. We set the threshold to 0.1. Additionally, the Mean per Joint Position Error (MPJPE) metric is also used to evaluate the pose estimation model. It measures the mean of the euclidean distance between the predicted and ground-truth keypoints.

$$MPJPE = \frac{1}{N}\sum_{i=1}^{N} \|\mathbf{p}_k - \mathbf{t}_k\| \qquad (2)$$

The MPJPE metric is normalised with respect to the length of the bounding box diagonal.

5 Results and Discussion

In this work we evaluate the use of the synthetic video dataset we generated called *SyDog-Video*.

5.1 Models Trained Solely on Real Data

To be able to fairly compare the models trained on synthetic data with the models trained on real data, we augmented the appearance (converting to grayscale, sensor noise, brightness and contrast) and geometric properties of the data (rotations, random cropping).

In spite of the fact that we tried to find the optimal model (Sect. 3.4) and augmented the training data, the models were not able to learn. This is most likely due to the small training dataset (115 samples) and inconsistent 2D ground truth.

5.2 Models Trained Solely on Synthetic Data

Table 4 presents the accuracy of the models on various types of *SyDog-Video* test datasets. These datasets consist of sequences featuring unseen backgrounds and dogs to the network. To ensure comprehensive evaluation, we employed a leave-one-out cross validation approach due to the test set containing only one dog breed. This involved training the networks on distinct training datasets, excluding one dog breed each time, and subsequently averaging the obtained results. Our analysis of Table 4 leads us to the conclusion that there is a significant importance associated with maintaining diversity in the shape and size of dogs. The models are evaluated in terms of their ability to generalise to real data (Table 5). The results show that there is still a large domain gap. We address this by fine-tuning the models using the *Dogio-11* training dataset and by training the network with a mixed dataset. Both these methods succeed in bridging the domain gap which we discuss in Sect. 5.3.

Table 4 Results on the different *SyDog-Video* test datasets

Type of test set	PDJ@0.1 ↑	MPJPE ↓
Dog2	85.77	0.11
Lab	75.69	0.15
Pug	75.17	0.17
Pitbull	78.11	0.16
Wolf	72.24	0.17
Average	77.40	0.15

Note that the synthetic test sequences contains backgrounds and a dog type that the model had not seen while training

Table 5 Results on the *Dogio-11* test datasets before fine-tuning

Type of test data	PDJ@0.1 ↑	MPJPE ↓
Dogio-11 (known)	43.07	0.41
Dogio-11 (unknown)	37.36	0.58

It is demonstrated that the model performs poorly when evaluated on real data (domain gap)

Figure 3 illustrates qualitative results on the synthetic test dataset. Note that the test samples contain only unseen dog breeds and unseen background images.

5.3 Transfer-Learning Results

As illustrated in Sect. 5.2 the models trained on synthetic data perform poorly when evaluated on real data. This is due to the domain gap. However, these models do perform better than the models trained solely on real data. We aim to bridge the domain gap by applying transfer-learning methods such as fine-tuning and training the models with a mixed

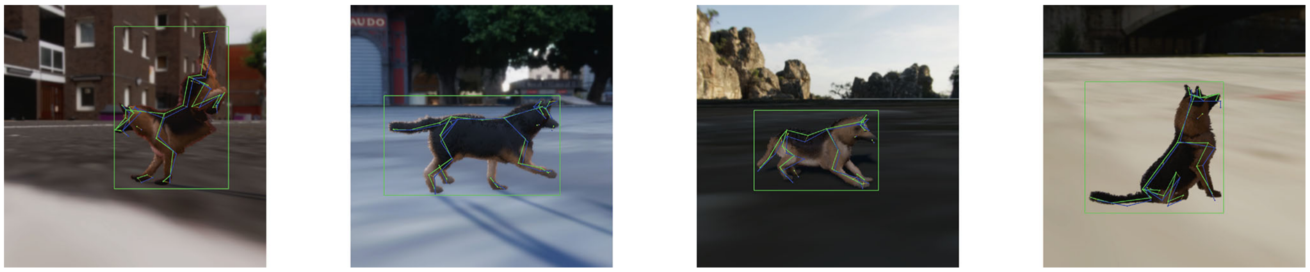

Fig. 3 Samples of the *SyDog-Video* test dataset. The test dataset contains dog breeds that the network had never seen, including backgrounds. The ground truth is coloured in green and the model predictions are coloured in blue (Color figure online)

Table 6 Results on the *Dogio-11* test datasets after the application of transfer-learning methods

Training strategy	PDJ@0.1$_k$ ↑	MPJPE$_k$ ↓	PDJ@0.1$_u$ ↑	MPJPE$_u$ ↓
Before fine-tuning (trained on SyDog-Video)	43.07	0.41	37.36	0.58
Fine-tuning	**77.76**	**0.16**	**41.78**	**0.44**
Mixed dataset	68.82	0.21	37.29	0.46

The bold means the best performance
It is illustrated that pre-training the network results in better performance than using a mixed dataset for training. The subscript k and u indicate to the *known* and *unknown Dogio-11* test datasets. The performance of the model is evaluated using the percentage of detected joints (PDJ) with a threshold set to 10% of length of the bounding box diagonal and the mean per joint per error (MPJPE) which is normalised w.r.t. the length of the ground truth bounding box diagonal

training dataset (synthetic + real samples). As mentioned in Sect. 4.2.3 we used the StanfordExtra dataset to create the mixed training dataset. Table 6 compares the accuracy of the model when fine-tuned to the model that is trained with the mixed dataset. It is illustrated that fine-tuning the network results in better performance than using a mixed dataset for training.

Figure 4 illustrates qualitative results from samples of the *known Dogio-11* test dataset before and after fine-tuning the network. We do not show results from the *unknown* test dataset because the network was not able to generalise well to new dog breeds due to the *Dogio-11* training dataset size being extremely small (115 samples).

As the *Dogio-11* training dataset size is extremely small we also decided to fine-tune the network, pre-trained on the *SyDog-Video* dataset, on larger real-world datasets. Fine-tuning on these datasets and evaluating on the *Dogio-11* test datasets yielded unsatisfactory results due to the different data distributions (Table 8). Consequently, an additional round of fine-tuning was applied using the *Dogio-11* training set. Moving forward, our analysis will focus on contrasting the performance of networks that underwent dual fine-tuning (Table 8) with the networks pre-trained on different datasets (synthetic and real) and fine-tuned only once (Table 7). The performance of the networks fine-tuned twice exhibited improved performance on the *unknown Dogio-11 test set*, which shows that the networks' ability to generalise increased. Conversely, fine-tuning the network on the Stanford Extra dataset and then on the *Dogio-11* training set led to reduced performance on the *known Dogio-11* test set. This is most likely due to the frame-based nature of the Stanford Extra dataset, likely causing the loss of temporal context that was initially beneficial. There was a slight yet marginal increase in the PCK performance of the network that underwent two rounds of fine-tuning, with the first round utilizing the Animal Pose dataset, accompanied by a slight decrease in MPJPE performance on the *known Dogio-11* test set. While the image datasets are larger than the *Dogio-11* training set, this underlines that the network unlearns the temporal context when fine-tuning the network with an image dataset instead of video dataset. Lastly, fine-tuning the network on the APT-35K video dataset followed by fine-tuning on the *Dogio-11* training set resulted in an increase of 11.6 units in PCK on the *known Dogio-11* test set. This highlights the effectiveness of using real-world video datasets for fine-tuning rather than just image-based datasets. While the last variation demonstrates an enhanced performance of the network, it is important to emphasize that the optimal results are achieved by the network pre-trained on the *SyDog-Video* training set and fine-tuned only once on the *Dogio-11* training set.

5.4 Pre-trained on Different Datasets

In this section we discuss the accuracy of the LSTM Pose Machine pre-trained with different types of datasets (synthetic and real) mentioned in Sect. 4.1.

We demonstrate using Table 7 that the model pre-trained with synthetic data is robust on within-domain test data, however, performs poorly when tested on out-of-domain data. In spite of the model not being able to generalise to novel

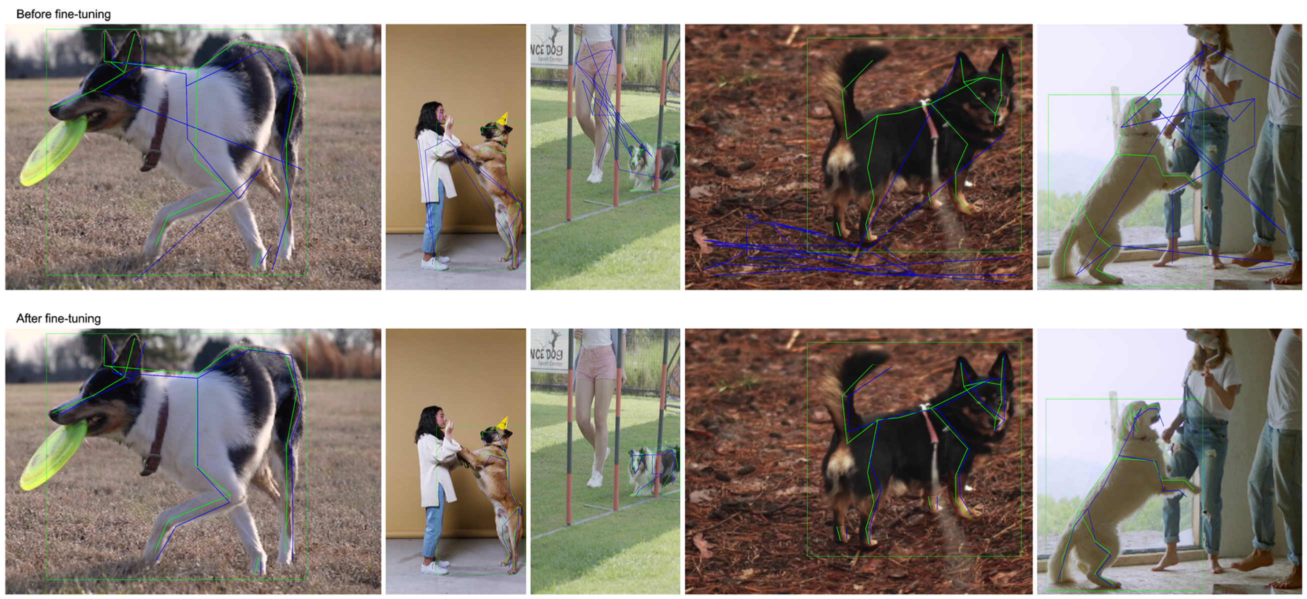

Fig. 4 Samples of known *Dogio-11* test dataset before and after fine-tuning the network. The test dataset contains unseen frames from known dog breeds. The ground truth is coloured in green and the model predictions are coloured in blue (Color figure online)

Table 7 Results on the *Dogio-11* test datasets from models pre-trained on different types of datasets (SyDog-Video (NoTemp), SyDog-Video, StanfordExtra, Animal Pose, APT-36K) and then fine-tuned with the *Dogio-11* training dataset

Dataset	Own test dataset		Dogio-11 test datasets			
	PDJ@0.1	MPJPE	PDJ$_k$@0.1	PDJ$_u$@0.1	MPJPE$_k$	MPJPE$_u$
SyDog-Video(NoTemp)	92.06	0.07	63.35	27.63	0.23	0.51
SyDog-Video	97.44	0.04	**77.76**	**41.78**	0.16	0.44
StanfordExtra	62.34	0.22	73.02	24.34	0.19	0.52
Animal Pose	66.06	0.17	74.03	25.99	**0.15**	0.51
APT-36K	65.47	0.15	60.84	27.04	0.29	**0.30**

The SyDog-Video (NoTemp) refers to the non-temporal version of the SyDog-Video dataset. The values under the *Own test dataset* indicate the accuracy of the model when evaluated on the test dataset of the dataset it was trained on. Bold values show the best accuracy. The subscript *k* and *u* indicate to the *known* and *unknown Dogio-11* test datasets. The performance of the model is evaluated using the percentage of detected joints (PDJ) with a threshold set to 10% of length of the bounding box diagonal and the mean per joint per error (MPJPE) which is normalised w.r.t. the length of the ground truth bounding box diagonal

Table 8 Results on the *Dogio-11* test datasets of the network pre-trained on SyDog-Video dataset followed by fine-tuning on real-world datasets. The subscript *k* and *u* indicate to the *known* and *unknown Dogio-11* test datasets

	PDJ$_k$@0.1	PDJ$_u$@0.1	MPJPE$_k$	MPJPE$_u$
Dataset	Fine-tuned on real-world datasets			
StanfordExtra	35.27	12.85	0.50	0.83
Animal Pose	14.11	4.34	0.86	1.62
APT-36K	47.18	28.33	0.30	0.36
Dataset	Fine-tuned on real-world datasets + Dogio-11			
StanfordExtra	59.75	47.18	0.38	0.38
Animal Pose	74.99	58.72	0.20	0.21
APT-36K	72.44	58.38	0.16	0.18

dog breeds, it performs better than the models pre-trained on the real-world datasets. We expect that adding more diverse videos into the *Dogio-11* training dataset would help the network to generalise to unseen dog breeds. We expect that the reason the model pre-trained on the *SyDog-Video* dataset performs better than the models pre-trained on the real-world animal pose datasets is due to the model learning the temporal context as the *SyDog-Video* dataset contains sequences

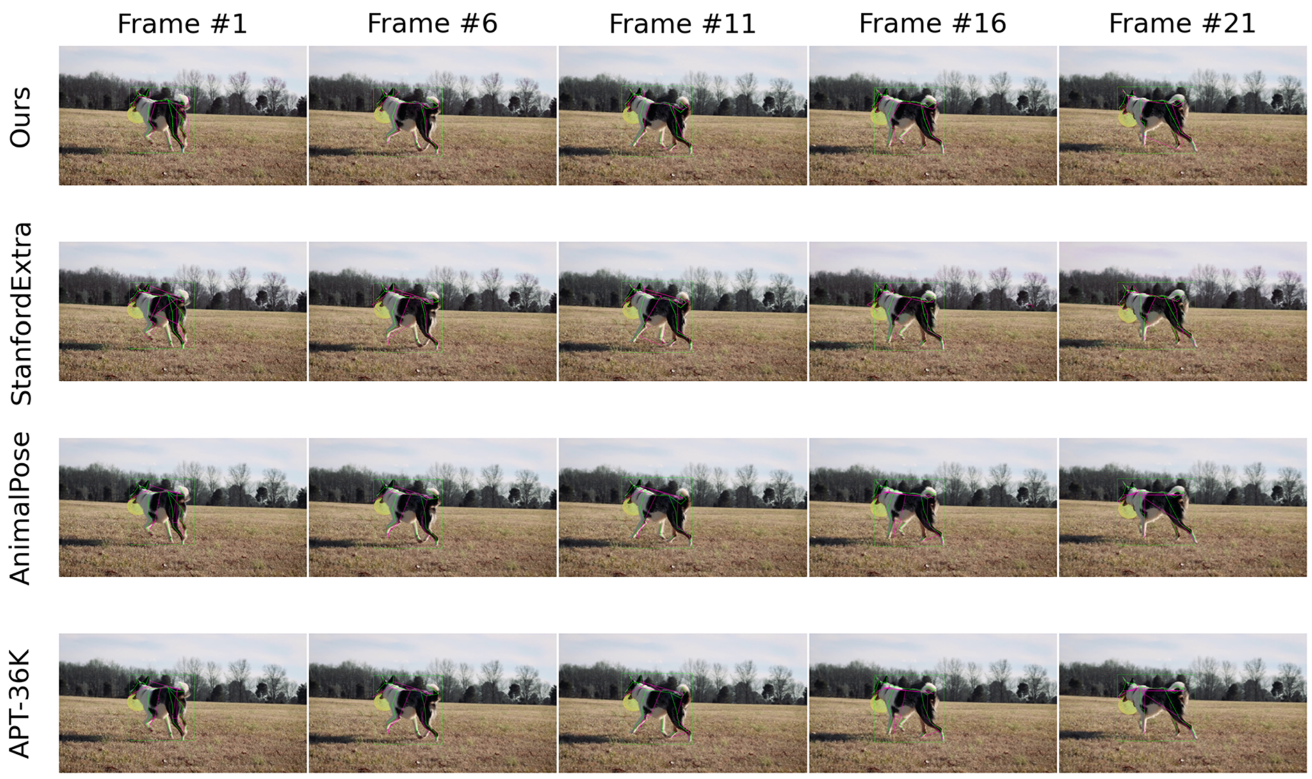

Fig. 5 Image sequences with ground truth and pose predictions of fine-tuned neural network pre-trained on different training datasets. The ground truth is coloured in green and the model predictions are coloured in pink (Color figure online)

of frames and the real-world animal pose datasets do not. It could be argued that increasing the size of the real-world animal pose datasets to the same size of *SyDog-Video* dataset could outperform the model pre-trained on synthetic data. However, labelling real-world images can take a lot of time and it can be expensive. Additionally, it can also be argued that because the model trained on *SyDog-Video* performed better on its own test dataset, it performed better when fine-tuned. The better performance was most probably due to the more accurate 2D ground truth keypoints than the real-world animal pose datasets. *SyDog-Video* contained all 33 keypoint coordinates, even when they were considered invisible, while the real-world animal pose datasets contained only visible keypoint coordinates. It is noticed that the model pre-trained on the Animal Pose dataset achieves a slightly better performance than the model pre-trained with the StanfordExtra dataset. This is most likely due to the Animal Pose dataset having more similar keypoints with the *Dogio-11* dataset than the StanfordExtra dataset with the *Dogio-11* dataset. Another reason could be that the Animal Pose dataset has a different number of animal species, which increases the diversity of the dataset. We also fine-tuned the model pre-trained on ImageNet. However, due to the *Dogio-11* training dataset being so small and the model not being pre-trained on the pose estimation task, the model was not able to learn (Table 8).

Figure 5 shows image sequences with the ground truth and predicted pose predictions of the model pre-trained on different datasets and then fine-tuned with the *Dogio-11* training dataset. We can deduce from the figure that the pose predictions from the model pre-trained on the synthetic dataset is more consistent than the models pre-trained on the real-world datasets. For example, the model pre-trained on the synthetic data predicts the joint coordinates for the right front leg in the first 4 columns in a consistent manner, while the static models pre-trained on the real-world animal pose datasets have more difficulties in doing so. While it is less obvious, the same reasoning counts for the tail. Through the presentation of Table 7, we demonstrate that the superior performance of the model pre-trained on synthetic data can be attributed to its ability to effectively capture and learn the temporal context inherent in the synthetic video training dataset. Figures 6 and 7 demonstrates the performance of the fine-tuned network pre-trained on diverse datasets when faced challenging tasks such as temporal occlusion or when motion blur occurs which indicates to significant movements. These figures reveal that the pose predictions from the network pre-trained with our synthetic dataset exhibit higher consistency in challenging scenes compared to the network pre-trained on real-world datasets. In addition to the qualitative results, we substantiate our findings with quantitative analysis presented in Table 9. This comprehensive comparison assesses the performance

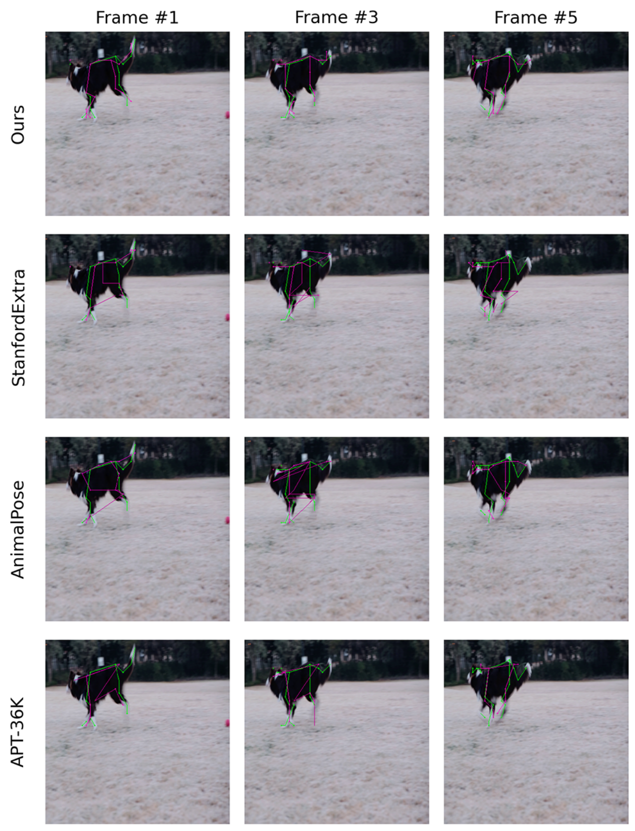

Fig. 6 Samples of frames of dogs executing substantial movements with ground truth (green) and pose predictions (pink) of fine-tuned neural network pre-trained on different training datasets (Color figure online)

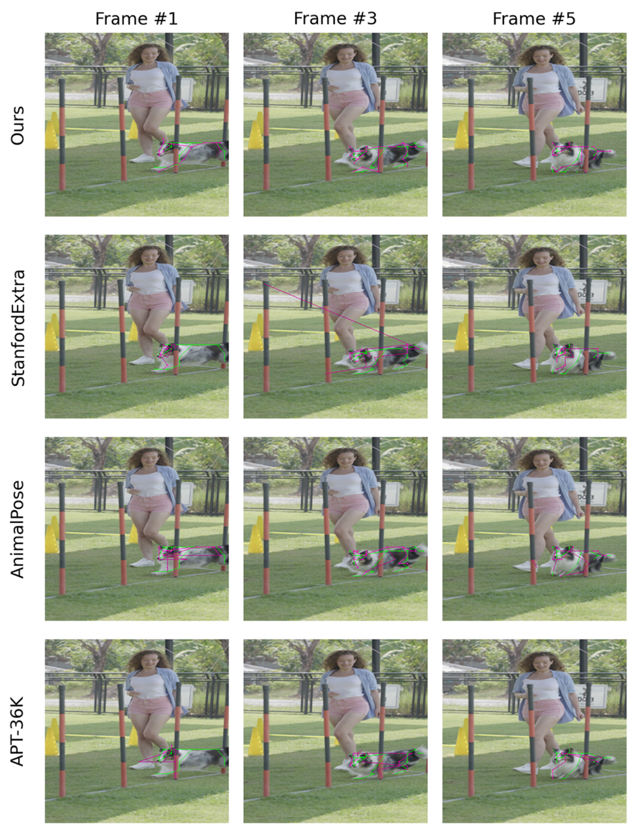

Fig. 7 Samples depicting frames of dogs exhibiting temporal occlusion, showcasing ground truth (green) and pose predictions (pink) generated by a fine-tuned neural network pre-trained on various training datasets (Color figure online)

of networks pre-trained on various datasets across different test distributions, including challenging, easy, and test sets. Notably, our results clearly demonstrate that pre-training the network using our synthetic video dataset consistently outperforms the network pre-trained on real datasets across all test sets. Furthermore, the performance of the network pre-trained on synthetic data indicates versatility in handling different levels of difficulty within the task.

5.5 Are Distractors Important?

As mentioned in Sect. 4.2.2, we assess the importance of distractors in the background of synthetic images (Fig. 8). We define distractors as 3D objects or people which might or might not occlude the dog. We generate 4 different datasets which are similar to the original synthetic dataset, *SyDog-Video*. The datasets differ in backgrounds. To compare the datasets we keep the same seed across datasets, however the seed varies with respect to the type of dog (Table 12). This means that all randomizers are deterministic across datasets, but non-deterministic for each type of dataset. The following datasets are described in more detail:

- *w(ith)_assets*: contains static 3D assets in the background.
- *w(ith)_people*: contains 3D people performing an action such as walking in the background.
- *w(ith)_assetsPlusPeople*: contains static 3D assets and dynamic people in the background.
- *w(ith)o(out)_groundplane*: is identical to the *SyDog-Video* dataset but with no ground geometry.

Figure 9 shows a bar plot of the accuracy of the models pre-trained with different types of synthetic datasets and then fine-tuned on the Dogio-11 training dataset. Additionally, it shows the accuracy of the model when it was trained with a mixed dataset. The model was evaluated on both Dogio-11 test datasets. Recall that one test dataset contains unseen frames of known dogs (known) and the other test dataset contains frames of an unseen dog breed (unkown). For the unknown test dataset. The blue coloured bars show the results on the known test dataset and the green coloured bars show the results on the unknown test dataset. It is demonstrated that the model pre-trained with the dataset without ground plane, outperforms the models trained on the other synthetic datasets. It is also shown that adding 3D people to the syn-

Table 9 Quantitative results of the network pre-trained on different datasets. Comparing the performance of the network on challenging cases (Challenge), easier cases (Easy) and the overall *Dogio-11* known test set (Test)

Dataset	Challenge		Easy		Test	
	PDJ$_k$@0.1	MPJPE$_k$	PDJ$_k$@0.1	MPJPE$_k$	PDJ$_k$@0.1	MPJPE$_k$
SyDog-Video	**70.51**	**0.14**	**81.49**	**0.13**	**77.76**	0.16
StanfordExtra	42.00	0.40	69.75	0.24	73.02	0.19
Animal Pose	47.39	0.31	69.58	0.20	74.03	**0.15**
APT-36K	54.65	0.20	62.20	0.19	60.84	0.29

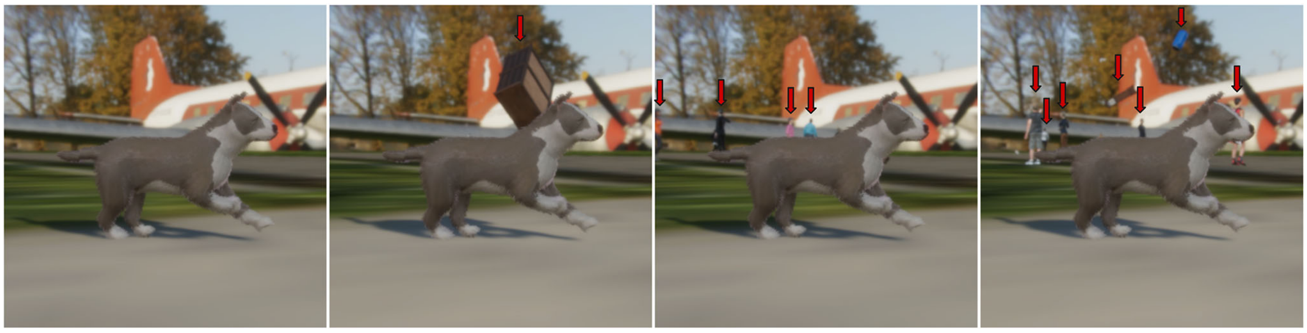

Fig. 8 The same image sample across different datasets: clean_plate (SyDog-Video), w_assets, w_people, w_assetsPlusPeople. The red arrows indicate either 3D assets or 3D people (Color figure online)

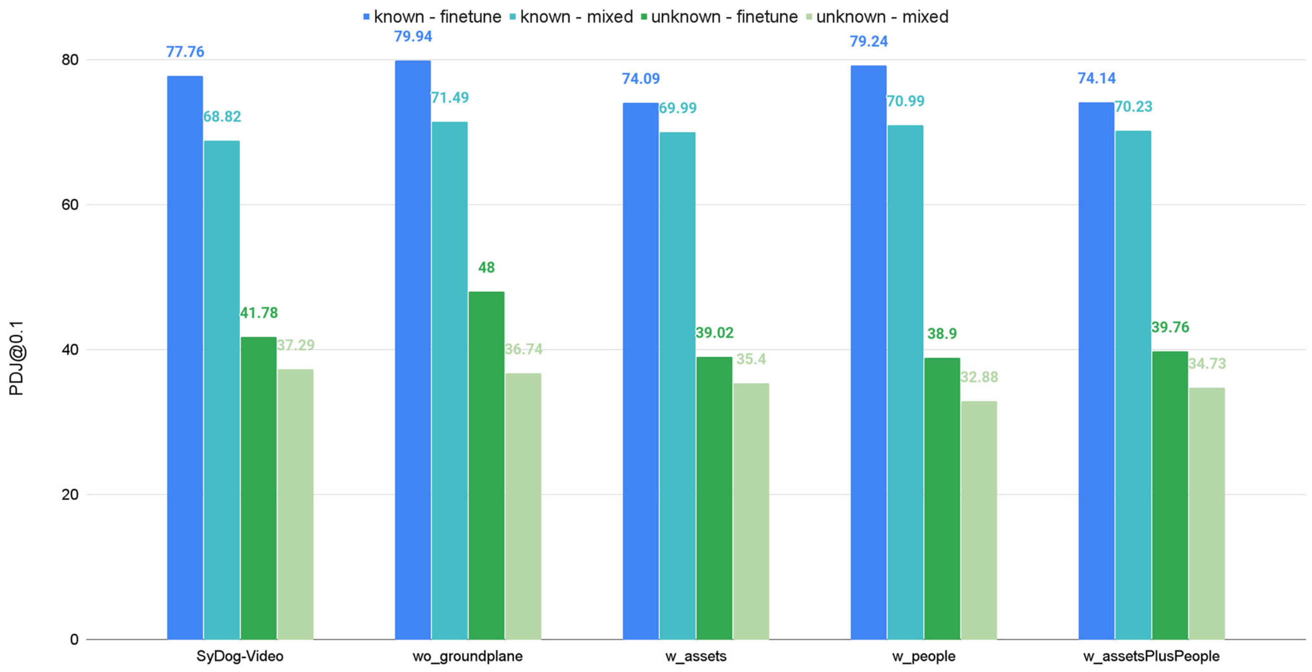

Fig. 9 Bar graph: comparison of the accuracy (PDJ@0.1) of model pre-trained on synthetic data with different backgrounds and fine-tuned with the *Dogio-11* training dataset. Additionally, it also shows the accuracy of the model when trained on a mixed training dataset. The model was evaluated on both *Dogio-11* test datasets. The *unkown* test dataset contains frames of an unseen dog breed and the *known* test dataset contains unseen frames of dogs that the network has already seen (Color figure online)

thetic dataset increases the performance of the model when fine-tuned. It is again deduced that we get a better accuracy of the model when fine-tuning rather than using a mixed dataset. While adding 3D assets or both 3D assets and people does not help the model's performance when training with a mixed dataset.

5.6 Ablation

Firstly, we analyse how the synthetic dataset size influenced the model's performance after fine-tuning. The model, which was pre-trained on the *SyDog-Video* training dataset without ground geometry, is evaluated on both the *Dogio-11* test datasets. Figure 10 demonstrates that the model's perfor-

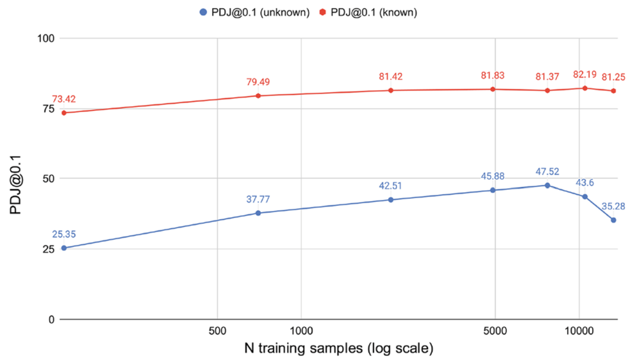

Fig. 10 The model's accuracy (PDJ@0.1) versus the size of the synthetic training data. The model was pre-trained on the *SyDog-Video* dataset but without groundplane. The model's accuracy increases until it decreases or plateaus (Color figure online)

mance increases with the number of the synthetic training samples, before it decreases after 7700 samples. For the *known* test dataset, the model's performance also increases until it plateaus at around 7700 samples. It can be also concluded, that with 140 synthetic training samples the network achieves similar performance levels to the models trained with the real-world animal pose datasets: StanfordExtra (6k training samples) and Animal Pose dataset (3k training samples).

Moreover, we show the importance of data augmentation when training models on synthetic data and follow a similar procedure as in Wood et al. (2021). We train the models with (1) no augmentation, (2) appearance augmentation, and (3) full augmentation which includes appearance and geometric augmentations such as rotations. Table 10 demonstrates that augmenting the dataset increases the model's performance. Applying appearance and geometric augmentations increases the model's performance, however, the model performs best when applying appearance augmentations solely on the synthetic dataset. We also assessed the effect of adding fur to the 3D dog model. Table 10 shows that not adding fur actually increased the model's performance. We believe that the properties of the fur might not have been realistic enough and we think we could improve it by using generative adversarial networks such as in Bolaños et al. (2021).

6 Conclusion

We generated a synthetic dataset, called *SyDog-Video*, containing image sequences of dogs to solve the problem of the lack of pose datasets and to avoid the need to manually label videos as it can be time consuming, costly and be prone to labelling errors. We trained a temporal deep learning model (LSTM Pose Machine) to estimate the pose of dogs from videos as it can result in more accurate pose predictions compared to static deep learning models when temporary occlusions or substantial movements occur. The dataset was made diverse by randomising parameters such as the lighting, backgrounds, camera parameters, and the dog's appearance and pose. We initially aimed to bridge the the domain gap by improving the quality of the synthetic dataset. However, the domain gap still remained and therefore we applied 2 different transfer-learning methods: fine-tuning and using a mixed dataset to train the network.

To the best of our knowledge, there are no publicly available datasets containing videos of dogs with annotated pose data, therefore to evaluate our method we manufactured a real-world pose dataset containing dogs called *Dogio-11*. To label ∼1k frames with 2D bounding box and 33 keypoint coordinates was time consuming and the LSTM Pose Machine network was not able to learn when trained with it due its small training set size (115 samples) and inconsistent labelling. We demonstrate the necessity of pre-training networks in order for the network to effectively learn from a limited training data.

Further, we demonstrate that pre-training the network with the *SyDog-Video* dataset outperforms the models that were trained with real-world animal pose datasets. This is most likely due to the model learning the temporal context of the synthetic videos, as the models trained on the real-world animal pose dataset are single image models instead of temporal models. And because the *SyDog-Video* dataset was automatically and accurately labelled even when some keypoints were considered invisible to the human eyes, while the keypoint coordinates in the real-world animal pose dataset that were considered invisible were not annotated.

Table 10 Results on the *Dogio-11* test datasets after fine-tuning the network that was pre-trained on the synthetic dataset augmented in different ways

Type of augmentation	PDJ@0.1$_k$ ↑	MPJPE$_k$ ↓	PDJ@0.1$_u$ ↑	MPJPE$_u$ ↓
No augmentation	73.63	0.16	32.34	0.63
Appearance augmentation	**78.67**	**0.13**	**39.67**	**0.44**
Full augmentation	77.76	0.16	38.04	0.44
No fur + full augmentation	80.53	38.04	0.12	0.45

Bold values indicate the best accuracy. The subscript k and u indicate to the *known* and *unknown Dogio-11* test datasets. The performance of the model is evaluated using the percentage of detected joints (PDJ) with a threshold set to 10% of length of the bounding box diagonal and the mean per joint per error (MPJPE) which is normalised w.r.t. the length of the ground truth bounding box diagonal

Due to the small scale of the *Dogio-11* training set, the network pre-trained on the synthetic dataset, was fine-tuned on larger real-world datasets either including images or videos and evaluated on the *Dogio-11* test sets. After the initial fine-tuning the networks resulted in poor performance on the *Dogio-11* tests sets due to the different data distributions. To address this, a second round of fine-tuning was conducted using the *Dogio-11* training set. This second round of fine-tuning demonstrated an increase in the networks' ability to generalize. While the image-based datasets are larger than the *Dogio-11* training set, the optimal performance of the networks is achieved through fine-tuning with video datasets, harnessing the inherent temporal information, as opposed to relying on image-based datasets. While fine-tuning the network, initially pre-trained on synthetic data, twice with real-world video datasets yields improvements in pose estimation on the *Dogio-11* test set, the most optimal results are achieved by fine-tuning the network pre-trained on synthetic data with the *Dogio-11* training set only once.

We also show that adding a certain type of distractors to the background of synthetic images helps the performance of the model depending on the transfer-learning method. And illustrate that the size of the synthetic dataset can improve the model's performance up to a certain point, beyond that point the accuracy of the model plateaus or decreases. Finally, we demonstrate that augmenting the synthetic dataset at training time increases the performance of the model.

To conclude, using our synthetic video dataset, *SyDog-Video* as a training set is beneficial for pre-training a temporal model. This temporal model can later be fine-tune with a (small) real-world pose video dataset as it is faster to generate a large-scale synthetic dataset, it is more cost effective and the labels are produced more consistently than labelling real-world videos. Pre-training the model with *SyDog-Video* results in more accurate pose predictions when fine-tuned and evaluated with real-world (small sized) pose datasets.

Limitations the network is not able to generalise to real data before fine-tuning and across new dog breeds before and after fine-tuning. We believe that increasing the diversity in the synthetic dataset by increasing the number of breeds and further improving its photorealism will increase the performance of the model when evaluated on real data and on videos of new dog breeds.

Supplementary information

The article has accompanying supplementary files: 2 videos.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s11263-023-01946-z.

Acknowledgements This research was supported by EPSRC Audio-Visual Media Platform Grant EP/P022529/1 and by the Leverhulme Trust Early Career Fellowship scheme. We thank Michaela Butler for her artistic contributions.

Appendix A Data Generation

Figure 11 shows a diagram visualising the sequence of operations run by typical simulation scenario in the Unity Perception package. Table 11 demonstrates the parameters chosen to randomise in the synthetic data generator made in Unity3D.

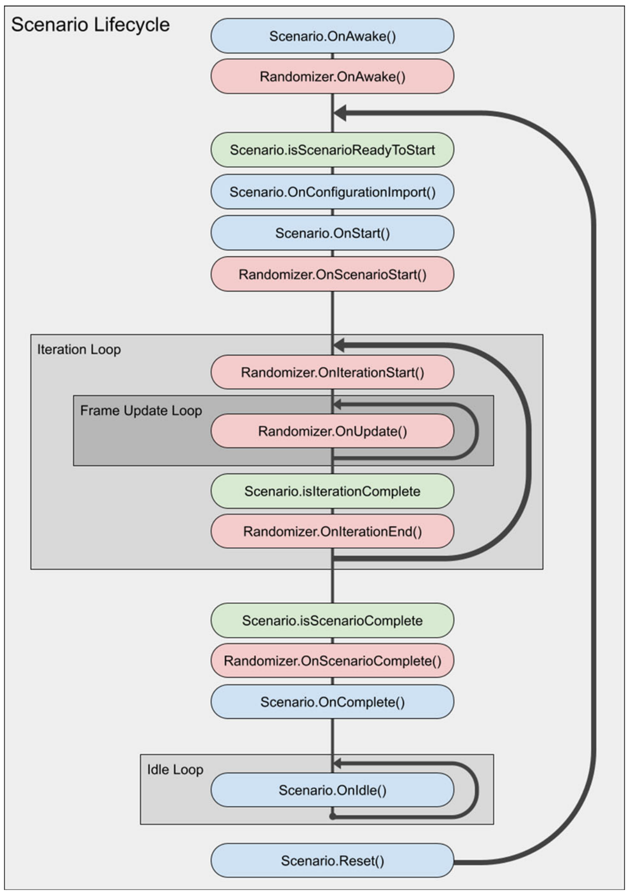
Scenario Lifecycle
Scenario.OnAwake()
Randomizer.OnAwake()
Scenario.isScenarioReadyToStart
Scenario.OnConfigurationImport()
Scenario.OnStart()
Randomizer.OnScenarioStart()
Iteration Loop
Randomizer.OnIterationStart()
Frame Update Loop
Randomizer.OnUpdate()
Scenario.isIterationComplete
Randomizer.OnIterationEnd()
Scenario.isScenarioComplete
Randomizer.OnScenarioComplete()
Scenario.OnComplete()
Idle Loop
Scenario.OnIdle()
Scenario.Reset()

Fig. 11 diagram visualizing the sequence of operations run by a typical simulation scenario. Image source Unity Technologies (2020) (Color figure online)

Table 11 Domain randomization parameters in our data generator

Category	Randomizer	Parameters	Distribution
3D objects	Background object placement	Object placement	Cartesian[Uniform(− 100,100), Uniform(− 2,2), Uniform(− 10,10)]
		Separation distance	Cartesian[Constant(5), Constant(5), Constant(5)]
		Object rotation	Euler[Uniform(0,360), Uniform(0,360), Uniform(0,360)]
		Object scale range	Cartesian[Uniform(1,3), Uniform(1,3), Uniform(1,3)]
	Background people placement	People placement	Cartesian[Uniform(-35,35), Uniform(− 1,1), Uniform(− 5,5)]
		Separation distance	Cartesian[Constant(5), Constant(5), Constant(5)]
		People rotation	Euler[0, Uniform(0, 360), 0]
	People animation	Idle animation	A set of FBX animation clips[a]
		Moving animation	$P(enabled) = 0.5$, $P(disabled) = 0.5$
	Dog animation	Sequence of actions	List of actions shuffled
Textures	Material	material	A set of texture assets[a]
Lights	Sun angle	hour	Uniform(0, 24)
		Day of the year	Uniform(0, 365)
		Latitude	Uniform(− 90, 90)
	Sky	Train hdri's	A set of hdri's (402)[a]
		val hdri's	A set of hdri's (101)[a]
	Light intensity and temperature	Intensity	Uniform(366, 80,000)
		Temperature	Uniform (1500, 20,000)
Camera	Camera	Focal length	Uniform(50, 125)
		Aperature	Uniform(0.7f, 35.0f)
		Yaw	Uniform(− 100, 160)
Post-Processing	Post process volume	Chromatic aberration intensity	Uniform(0.0f, 0.8f)

[a]Each sample of this set is uniformly sampled

Appendix B Ablation

Table 12 shows the seed set in the synthetic data generator for every different type of dog.

Table 12 The seed values change with respect to the type of dog within a dataset but not across the types of datasets

Dog type	Seed
Dog1 (medium)	150,495
Dog2 (medium)	654,321
Dog3 (small)	123,456
Dog4 (medium)	654,123
Dog5 (small)	123,654

References

Abu-El-Haija, S., Kothari, N., Lee, J., Natsev, P., Toderici, G., Varadarajan, B., & Vijayanarasimhan, S. (2016). Youtube-8m: A large-scale video classification benchmark. *CoRR*. Retrieved from arXiv:1609.08675

Adobe. (2022). *Mixamo get animated. Animate 3d characters for games, film, and more*. https://www.mixamo.com/

Akiba, T., Sano, S., Yanase, T., Ohta, T., & Koyama, M. (2019). Optuna: A next-generation hyperparameter optimization framework. *CoRR*. Retrieved from arXiv:1907.10902

Alhaija, H. A., Mustikovela, S. K., Mescheder, L. M., Geiger, A., & Rother, C. (2017). Augmented reality meets computer vision: Efficient data generation for urban driving scenes. *CoRR*. Retrieved from arXiv:1708.01566

Biggs, B., Boyne, O., Charles, J., Fitzgibbon, A., & Cipolla, R. (2020). Who left the dogs out? 3D animal reconstruction with expectation maximization in the loop.

Bolaños, L. A., Xiao, D., Ford, N. L., LeDue, J. M., Gupta, P. K., Doebeli, C., Hu, H., Rhodin, H., & Murphy, T. H. (2021). A three-dimensional virtual mouse generates synthetic training data for behavioral analysis. *Nature Methods, 18*(4), 378–381. https://doi.org/10.1038/s41592-021-01103-9

Borkman, S., Crespi, A., Dhakad, S., Ganguly, S., Hogins, J., Jhang, Y., Kamalzadeh, M., Li, B., Leal, S., Parisi, P., Romero, C., Smith, W., Thaman, A., Warren, S., & Yadav, N. (2021). Unity perception: Generate synthetic data for computer vision. *CoRR*. Retrieved from arXiv:2107.04259

Brooks, J. (2018). *COCO annotator*. https://github.com/jsbroks/coco-annotator/

Cao, J., Tang, H., Fang, H., Shen, X., Lu, C., & Tai, Y. (2019). Cross-domain adaptation for animal pose estimation. *CoRR*. Retrieved from arXiv:1908.05806

Cao, Z., Hidalgo, G., Simon, T., Wei, S., & Sheikh, Y. (2018). Openpose: Realtime multi-person 2d pose estimation using part affinity fields. *CoRR*. Retrieved from arXiv:1812.08008

Chen, W., Wang, H., Li, Y., Su, H., Tu, C., Lischinski, D., Cohen-Or, D., & Chen, B. (2016). Synthesizing training images for boosting human 3D pose estimation. *CoRR*. Retrieved from arXiv:1604.02703

Chen, Y., Li, W., Chen, X., & Gool, L. V. (2018). Learning semantic segmentation from synthetic data: A geometrically guided input–output adaptation approach. *CoRR*. Retrieved from arXiv:1812.05040

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database.

Ebadi, S. E., Jhang, Y., Zook, A., Dhakad, S., Crespi, A., Parisi, P., Borkman, S., Hogins, J., & Ganguly, S. (2021). Peoplesanspeople: A synthetic data generator for human-centric computer vision. *CoRR*. Retrieved from arXiv:2112.09290

Falcon, W. et al. (2019). Pytorch lightning. 3:6. *GitHub. Note* https://github.com/PyTorchLightning/pytorch-lightning

Fangbemi, A. S., Lu, Y. F., Xu, M. Y., Luo, X. W., Rolland, A., & Raissi, C. (2020). Zoobuilder: 2D and 3D pose estimation for quadrupeds using synthetic data. *CoRR*. Retrieved from arXiv:2009.05389

Fischer, P., Dosovitskiy, A., Ilg, E., Häusser, P., Hazirbas, C., Golkov, V., van der Smagt, P., Cremers, D., & Brox, T. (2015). Flownet: Learning optical flow with convolutional networks. *CoRR*. Retrieved from arXiv:1504.06852

From Knowledge, G. (2016). *Man's best friend: Global pet ownership and feeding trends*. https://www.gfk.com/insights/mans-best-friend-global-pet-ownership-and-feeding-trends

Gaidon, A., Wang, Q., Cabon, Y., & Vig, E. (2016). Virtual worlds as proxy for multi-object tracking analysis. *CoRR*. Retrieved from arXiv:1605.06457

Georgakis, G., Mousavian, A., Berg, A. C., & Kosecka, J. (2017). Synthesizing training data for object detection in indoor scenes. *CoRR*. Retrieved from arXiv:1702.07836

González, J. L., Zaccaro, C., Alvarez-Garcia, J., Soria Morillo, L., & Caparrini, F. (2020). Real-time gun detection in CCTV: An open problem. *Neural Networks?: The Official Journal of the International Neural Network Society, 132*, 297–308. https://doi.org/10.1016/j.neunet.2020.09.013

Graving, J. M., Chae, D., Naik, H., Li, L., Koger, B., Costelloe, B. R., & Couzin, I. D. (2019). Deepposekit, a software toolkit for fast and robust animal pose estimation using deep learning. *eLife, 8*, e47994. https://doi.org/10.7554/eLife.47994

Haas, J. K. (2014). A history of the unity game engine.

Hu, Y.-T., Chen, H.-S., Hui, K., Huang, J.-B., & Schwing, A. G. (2019). SAIL-VOS: Semantic amodal instance level video object segmentation—A synthetic dataset and baselines.

Hurl, B., Czarnecki, K., & Waslander, S. L. (2019). Precise synthetic image and lidar (presil) dataset for autonomous vehicle perception. *CoRR*. Retrieved from arXiv:1905.00160

Ilg, E., Mayer, N., Saikia, T., Keuper, M., Dosovitskiy, A., & Brox, T. (2016). Flownet 2.0: Evolution of optical flow estimation with deep networks. *CoRR*. Retrieved from arXiv:1612.01925

Khosla, A., Jayadevaprakash, N., Yao, B., & Fei-Fei, L. (2011). Novel dataset for fine-grained image categorization.

Kiefer, B., Ott, D., & Zell, A. (2021). Leveraging synthetic data in object detection on unmanned aerial vehicles. *CoRR*. Retrieved from arXiv:2112.12252

Lee, C., Batra, T., Baig, M. H., & Ulbricht, D. (2019). Sliced wasserstein discrepancy for unsupervised domain adaptation. *CoRR*. Retrieved from arXiv:1903.04064

Liu, X., Yu, S.-Y., Flierman, N. A., Loyola, S., Kamermans, M., Hoogland, T. M., & De Zeeuw, C. I. (2021). Optiflex: Multi-frame animal pose estimation combining deep learning with optical flow. *Frontiers in Cellular Neuroscience*. https://doi.org/10.3389/fncel.2021.621252

Luo, H., Xu, T., Jiang, Y., Zhou, C., Qiu, Q., Zhang, Y., Yang, W., Xu, L., & Yu, J. (2022). Artemis: Articulated neural pets with appearance and motion synthesis. Retrieved from https://doi.org/10.48550/ARXIV.2202.05628, arXiv:2202.05628

Luo, Y., Ren, J. S. J., Wang, Z., Sun, W., Pan, J., Liu, J., Pang, J., & Lin, L. (2017). LSTM pose machines. *CoRR*. Retrieved from arXiv:1712.06316

Ma, H. (2018). *LSTM pm pytorch*. https://github.com/HowieMa/lstm_pm_pytorch

Mathis, A., Mamidanna, P., Cury, K. M., Abe, T., Murthy, V. N., Mathis, M. W., & Bethge, M. (2018). Deeplabcut: Markerless pose estimation of user-defined body parts with deep learning. *Nature Neuroscience, 21*(9), 1281–1289. https://doi.org/10.1038/s41593-018-0209-y

Mathis, A., Yüksekgönül, M., Rogers, B., Bethge, M., & Mathis, M. W. (2019). Pretraining boosts out-of-domain robustness for pose estimation. *CoRR*. Retrieved from arXiv:1909.11229

Mu, J., Qiu, W., Hager, G. D., & Yuille, A. L. (2019). Learning from synthetic animals. *CoRR*. Retrieved from arXiv:1912.08265

Nath, T., Mathis, A., Chen, A. C., Patel, A., Bethge, M., & Mathis, M. W. (2019). Using deeplabcut for 3D markerless pose estimation across species and behaviors. *Nature Protocols, 14*(7), 2152–2176. https://doi.org/10.1038/s41596-019-0176-0

Park, D., Lee, J., Lee, J., & Lee, K. (2021). Deep learning based food instance segmentation using synthetic data. *CoRR*. Retrieved from arXiv:2107.07191

Peng, X. B., Coumans, E., Zhang, T., Lee, T. E., Tan, J., & Levine, S. (2020). Learning agile robotic locomotion skills by imitating animals. *CoRR*. Retrieved from arXiv:2004.00784

Pereira, T., Aldarondo, D. E., Willmore, L., Kislin, M., Wang, S. S.-H., Murthy, M., & Shaevitz, J. W. (2018). Fast animal pose estimation using deep neural networks. *bioRxiv*. https://doi.org/10.1101/331181. https://www.biorxiv.org/content/early/2018/05/30/331181

Pexels. (2022). *The best free stock photos, royalty free images and videos shared by creators*. Retrieved from https://www.pexels.com

Qiu, W. & Yuille, A. L. (2016). Unrealcv: Connecting computer vision to unreal engine. *CoRR*. Retrieved from arXiv:1609.01326

Richter, S. R., Vineet, V., Roth, S., & Koltun, V. (2016). Playing for data: Ground truth from computer games. *CoRR*. Retrieved from arXiv:1608.02192

Russello, H., van der Tol, R., & Kootstra, G. (2021). T-LEAP: Occlusion-robust pose estimation of walking cows using temporal information. *CoRR*. Retrieved from arXiv:2104.08029

Sanakoyeu, A., Khalidov, V., McCarthy, M. S., Vedaldi, A., & Neverova, N. (2020). Transferring dense pose to proximal animal classes. *CoRR*. Retrieved from arXiv:2003.00080

Shooter, M., Malleson, C., & Hilton, A. (2021). Sydog: A synthetic dog dataset for improved 2d pose estimation. *CoRR*. Retrieved from arXiv:2108.00249

Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W., & Abbeel, P. (2017). Domain randomization for transferring deep neural networks from simulation to the real world. *CoRR*. Retrieved from arXiv:1703.06907

Tremblay, J., To, T., & Birchfield, S. (2018). Falling things: A synthetic dataset for 3d object detection and pose estimation. *CoRR*. Retrieved from arXiv:1804.06534

Tuia, D., Kellenberger, B., Beery, S., Costelloe, B. R., Zuffi, S., Risse, B., Mathis, A., Mathis, M. W., van Langevelde, F., Burghardt, T., Kays, R., Klinck, H., Wikelski, M., Couzin, I. D., van Horn, G., Crofoot, M. C., Stewart, C. V., & Berger-Wolf, T. (2022). Perspectives in machine learning for wildlife conservation. *Nature Communications, 13*(1), 792. https://doi.org/10.1038/s41467-022-27980-y

Unity Technologies. (2020). *Unity perception package*. https://github.com/Unity-Technologies/com.unity.perception

Varol, G., Romero, J., Martin, X., Mahmood, N., Black, M. J., Laptev, I., & Schmid, C. (2017). Learning from synthetic humans. *CoRR*. Retrieved from arXiv:1701.01370

Wang, Y., Li, J., Zhang, Y., & Sinnott, R. O. (2021). Identifying lameness in horses through deep learning. In *Proceedings of the 36th annual ACM symposium on applied computing, SAC '21, New York, NY, USA* (pp. 976–985). Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3412841.3441973

Wood, E., Baltrusaitis, T., Hewitt, C., Dziadzio, S., Johnson, M., Estellers, V., Cashman, T. J., & Shotton, J. (2021). Fake it till you make it: Face analysis in the wild using synthetic data alone. *CoRR*. Retrieved from arXiv:2109.15102

Yang, Y., Yang, J., Xu, Y., Zhang, J., Lan, L., & Tao, D. (2022). Apt-36k: A large-scale benchmark for animal pose estimation and tracking. *Advances in Neural Information Processing Systems*, 35, 17301–17313.

Yu, H., Xu, Y., Zhang, J., Zhao, W., Guan, Z., & Tao, D. (2021). AP-10K: A benchmark for animal pose estimation in the wild. *CoRR*. Retrieved from arXiv:2108.12617

Zaal, G., Tuytel, R., Cilliers, R., Cock, J. R., Mischok, A., Majboroda, S., Savva, D., & Bruger, J. *Poly haven the public 3d asset library*. https://polyhaven.com/

Zeller, D. (2021). *Fluffy grooming tool*. https://assetstore.unity.com/publishers/53628

Zhang, H., Starke, S., Komura, T., & Saito, J. (2018). Mode-adaptive neural networks for quadruped motion control. *ACM Transactions on Graphics, 10*(1145/3197517), 3201366.

Zuffi, S., Kanazawa, A., Jacobs, D. W., & Black, M. J. (2016). 3d menagerie: Modeling the 3d shape and pose of animals. *CoRR*. Retrieved from arXiv:1611.07700