SyDog: A Synthetic Dog Dataset for Improved 2D Pose Estimation

Moira Shooter Charles Malleson Adrian Hilton
University of Surrey
Stag Hill, University Campus, Guildford GU2 7XH
{m.shooter,charles.malleson,a.hilton}@surrey.ac.uk

Abstract

Estimating the pose of animals can facilitate the understanding of animal motion which is fundamental in disciplines such as biomechanics, neuroscience, ethology, robotics and the entertainment industry. Human pose estimation models have achieved high performance due to the huge amount of training data available. Achieving the same results for animal pose estimation is challenging due to the lack of animal pose datasets. To address this problem we introduce SyDog: a synthetic dataset of dogs containing ground truth pose and bounding box coordinates which was generated using the game engine, Unity. We demonstrate that pose estimation models trained on SyDog achieve better performance than models trained purely on real data and significantly reduce the need for the labour intensive labelling of images. We release the SyDog dataset as a training and evaluation benchmark for research in animal motion.

1. Introduction

Estimating the pose of animals from video [9, 10, 16, 22] helps to understand the animal motion and this supports many applications and disciplines such as veterinary science where lameness can be diagnosed early and recovery monitored; biomechanical applications where gait is analysed to improve animal performance in sports such as horse racing and dressage; neuroscience where motion is analysed to understand behaviour and/or relate motion to brain activity [18]; robotics where robots learn from animal motion data [21]; and in the entertainment industry to produce more natural and realistic animal animations and to create 3D representations of animals. The traditional and most accurate method to track the motion of subjects of interest is optical motion capture. This involves placing reflective markers on the subject and uses a system with multiple cameras to capture their 3D location. This method has its disadvantages in that it requires expertise and time to set up, it can be stressful to the animal, it can change the animal's behaviour, animals can be uncooperative and in some cases it is impossible to bring the animal into a lab. Another disadvantage is that the lighting conditions need to be fairly controlled, typically restricting such systems to laboratories. Non-contact video-based estimation of animal motion has the potential to overcome these limitations. Deep learning methods are known to perform well with huge amounts of data. The main focus in the literature has been on human pose estimation [6, 8, 11, 19, 25, 26, 27] where large amounts of training data have allowed high accuracy to be obtained. It is challenging to achieve the same quality results for animals as there is less training data available [2, 4, 5, 13]. The standard way to create datasets is to annotate each image manually, but annotating several keypoints in thousands of images is both labour intensive and expensive. However, in recent years the generation of synthetic data has been an accelerator for machine learning [7, 17, 20, 23, 29].

In this work we address the lack of animal datasets by creating a dataset consisting of images of dogs rendered using a real-time game engine. To add variation into the data, the dog's appearance and pose, the environment, the camera viewing points and the lighting conditions were modified. Using this approach, we generated a synthetic dataset containing 32k annotated images.We evaluate the pose estimation models trained with synthetic data on the StanfordExtra dataset [2]. Because networks trained only on synthetic data often fail to generalize to real world examples (the domain gap) [1, 12], two techniques were applied separately: fine-tuning the networks and training the networks with a combination of real and synthetic samples. We demonstrate that models trained on synthetic data increase the models' performances and reduce the need for labour intensive image annotation.

The **main contributions** of this work are: (i) We present a real-time system that generates 2D annotated images containing dogs. (ii) We release SyDog, a large scale annotated dataset of dogs with 2D keypoints and bounding box coordinates. (iii) We show that using the SyDog dataset improves the accuracy of pose estimation models and reduces the need for labour intensive labelling.

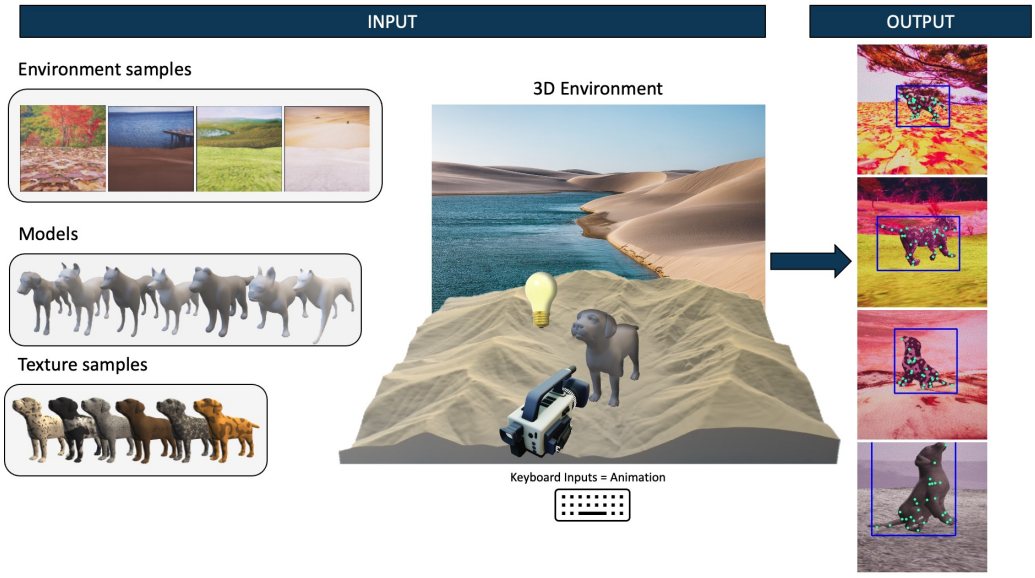

Figure 1: Pipeline showing the process of generating the SyDog dataset. The dog's motion is controlled using keyboard inputs. A virtual camera follows and renders a frame of the dog with different appearance, pose, lighting (post-processing effects), environment, and camera view points. These parameters are randomly sampled to make the data more diverse. RGB images, 2D pose and bounding box coordinate annotation are simultaneously generated.

2. Data Generation

In this section we present how the SyDog dataset was generated. Figure 1 demonstrates the pipeline overview for generating the synthetic data.

We generated **the synthetic dataset** using the game engine Unity3D. We built upon Zhang *et al.*'s project [28] which produces natural animations for quadruped animals from real motion data using a novel neural network architecture which they call Adaptive Neural Networks. By using this system we were able to control the animal's motion using keyboard inputs and make the dataset more varied by transitioning the dog's pose from one state to another. To add more variety into the dataset, the dog's appearance, the environment, the camera viewing points and the lighting conditions (post-processing effects) were randomly modified. We produced 32k images along with annotations of 25 keypoints and bounding box coordinates. We refer the reader to the supplementary material for samples of the SyDog dataset.

Dog models. We used 8 different type of dogs, 1 came with Zhang *et al.*'s project, which we will refer to as the *default* model, 5 were imported from the RGBD-Dog dataset [13], and 2 were a fat and a skinny version of the *default* model. The models represent dogs ranging from big to small sized breeds. The models were manually scaled and rigged based on the *default* model for the models to be correctly imported into the project.

Dog textures. To create different types of textures quickly and without the need of manually UV-unwrapping the models and manually producing textures, the textures were generated procedurally using shaders and mapped onto the surfaces by applying triplanar mapping. Triplanar mapping is a technique which applies textures onto a model from three directions using the world space positions. The initial setup of the shader can take time, but once implemented many textures can be generated by modifying the parameters of the shader such as the colour, size and position of the spots and the main colour of the dog. In total we have generated 12 types of fur texture, which are randomly sampled when rendering the images.

Post-processing effects. The post-processing effects from Unity were used to generate different lighting conditions and add noise to the renders. We added different types of grain which differ in particle size, intensity value, colour, and luminance contribution. We colour graded the image with saturation values which are randomly sampled between [-100, 100]; and with brightness values that range between [-20, 35].

Camera. The camera was set to follow and look at the dog while being randomly positioned around the dog to capture it from different angles. The camera's field of view values were sampled uniformly at random between [50,100] degrees.

Environment. Different environments were created by

modifying the sky and terrain textures. To set the sky texture we randomly sampled 1341 images from the Kaggle Landscape Pictures dataset [24]. Additionally, 10 different terrain textures were collected from the internet consisting of grass, autumn leaves (2x), dry mud (2x), cobble stone, pebbles, sand, snow and tiles.

2D annotation. To save the 2D annotations we located the 3D joint positions in world space and transform them into screen space. When the program runs, the 2D keypoints, the frame number and the bounding box coordinates with the 256x256 RGB image are saved. The bounding box coordinates were computed by adding 10 pixels to the minimum and maximum of the x- and y-coordinates. The average time to produce a synthetic is 33 milliseconds. By contrast, manual annotation of an image with these keypoints typically takes at least a minute. The data was generated on a MacBook Pro 2016 with a 2.9 GHz Quad-Core Intel Core i7 processor and a AMD Radeon Pro 460 4GB.

3. Experiments and Results

We trained a 2-stacked hourglass network with 2 blocks (2HG), an 8-stacked hourglass network with 1 block (8HG) and a pre-trained Mask R-CNN model with a ResNet50 as a backbone. We refer the reader to the supplementary material and [11, 19] for further details on the training set up and the networks, respectively.

Six experiments were conducted: Firstly, the networks were trained with solely synthetic data. Secondly, the networks were trained purely on the StanfordExtra dataset. Thirdly, the networks which were trained only on synthetic data were fine-tuned with the StanfordExtra dataset using the same parameters as the first experiment. Then, we repeated the third experiment but with a smaller learning rate. Finally, the networks were trained on a mixed dataset which is a dataset that contains both the StanfordExtra and (either the whole or a fraction of) the SyDog dataset.

3.1. Datasets

The StanfordExtra dataset [2] is based on the Stanford Dogs dataset [14] and contains 12k real images which cover 120 different types of dogs. The 2D joint annotations were modified to reflect our synthetic data labels. Only the common joints were included in the annotations and the joints that differed between the StanfordExtra and the Synthetic Dog datasets, the keypoints' visibility were set to invisible. We used the StandfordExtra training-test split, which are publicly available [3]. To train the networks on synthetic data, **the SyDog dataset** was divided by the different types of dog. 6 dogs were used for training, 1 for validation and 1 for testing. To train the networks with the mixed dataset, either the whole or a fraction of SyDog dataset was made available for training.

Network	PCK (%)	MPJPE (%)
2HG	77.76	6.51
8HG	77.57	6.56
Mask R-CNN	68.98	11.02

Table 1: Average PCK@0.1 and MPJPE from the 2HG, 8HG and the Mask R-CNN on the SyDog test dataset.

3.2. Evaluation metrics

The networks were evaluated using the percentage of correct keypoints (PCK) and the mean per joint position error (MPJPE), which were both normalized with respect to the length of the bounding box diagonal. The PCK measures whether the predicted keypoints are within a threshold from the true keypoints. The threshold was set to 10% of the bounding box diagonal. The MPJPE is the mean of the per joint position error [15]. The evaluation metrics are calculated for visible keypoints only.

3.3. Results for SyDog test dataset

Table 1 shows the pose estimation results for the 2HG, 8H and Mask R-CNN. Some challenges do arise for the Mask R-CNN when it has to predict certain poses such as sitting and when it is presented with certain camera view points such as when the dog is far away.

3.4. Results for StanfordExtra test dataset

The average PCK and MPJPE for all experiments are shown in Table 2.

Isolated training The networks trained solely on synthetic data performed poorly on real data, which was expected due to the domain gap. Another possible reason could be that the SyDog dataset does not cover all breeds in the StanfordExtra dataset. The results from the networks trained only on real data were used as a baseline to evaluate the use of the SyDog dataset.

Fine-tuning There's a significant increase in performance when fine-tuning the stacked hourglass networks with the same learning rate, and there is an even better performance when fine-tuning with a smaller learning rate. When fine-tuning the 2HG and the 8HG with a smaller learning rate the models' PCK performances are increased by 12.51% and 14.15%, respectively. The performance of the Mask R-CNN did not improve when fine-tuning with smaller learning rates, yet it performs better than the Mask R-CNN that was trained solely on real data.

Training with mixed dataset The best performance for the stacked hourglass networks is when the mixed dataset contains the full synthetic dataset, this is different for the Mask R-CNN; the Mask R-CNN performs best when the mixed dataset contains only half of the synthetic dataset, however

Network	Dataset	Learning rate	PCK (%) ↑	MPJPE (%) ↓
2HG	Real	0.001	68.61	15.84
	Synthetic	0.001	16.20	46.26
	FT	0.001 → 0.001	76.57	11.80
	FT	0.001 → 0.000001	**77.19**	**11.32**
	Mixed@0.1	0.001	63.14	19.08
	Mixed@0.5	0.001	68.43	15.50
	Mixed@1.0	0.001	70.46	14.76
8HG	Real	0.001	68.90	15.64
	Synthetic	0.001	17.34	45.08
	FT	0.001 → 0.001	78.31	11.47
	FT	0.001 → 0.00001	**78.65**	**11.19**
	Mixed@0.1	0.001	65.04	17.81
	Mixed@0.5	0.001	71.76	15.19
	Mixed@1.0	0.001	72.09	14.97
Mask R-CNN	Real	0.00001	43.60	21.58
	Synthetic	0.001	13.22	37.49
	FT	0.00001 → 0.00001	**50.77**	**20.03**
	FT	0.00001 → 0.000001	46.58	21.17
	Mixed@0.1	0.001	41.27	22.82
	Mixed@0.5	0.001	47.71	21.64
	Mixed@1.0	0.001	45.77	21.61

Table 2: Results on the StanfordExtra test dataset. Results are shown from the 2-and 8-stacked hourglass (2HG, 8HG) and the Mask R-CNN trained solely on the StanfordExtra dataset (Real) and solely on the SyDog dataset (Synthetic) together with the fine-tuned (FT) models and the models trained with a mixed dataset (Mixed@fraction). The performance is evaluated using the percentage of correct keypoints (PCK) with a threshold set to 0.1 and the mean per joint per error (MPJPE) which are both w.r.t. the length of the ground truth bounding box diagonal.

using the full synthetic dataset in the mixed dataset produces better results than when the Mask R-CNN is trained solely with real data.

Our results clearly demonstrate the benefits of using the synthetic data generated by our system. We show that using a mixed dataset when training gives a slight boost in the performance and that fine-tuning the networks results in a significant boost in the performance compared to the networks trained only on real data. The synthetic data generated by our system is not very photorealistic. However, it already improves the accuracy of the pose estimation models by 12.51% in the case of the 2-stacked hourglass network. We expect that improvements in photorealism would result in further improvements in pose estimation.

4. Conclusion

To solve the lack of animal datasets, we introduce SyDog a synthetic dataset containing dogs with 2D pose and bounding box annotation which was generated using real-time rendering technology. The dataset was made varied by modifying the dog's appearance, pose, environment, lighting conditions (post-process effects) and camera view points. To evaluate the use of the SyDog dataset we conducted extensive experiments on the real dataset, StanfordExtra. We bridged the domain gap by fine-tuning the networks trained on synthetic data with real data and training the networks with a mixed dataset (synthetic+real). We demonstrated that using the SyDog dataset **increases the performance of pose estimation models** trained solely on real data and significantly **reduces** the need for **labour intensive labelling** which in turn **speeds up the process**. The models trained with a mixed dataset return a slight increase in performance and models that were fine-tuned with real data return a significant increase in performance. The data generated does not look very photorealistic; however we showed that it already notably improves the accuracy of the pose estimation models. Future work would involve improving the photorealism of the data for yet further improvements in pose. In this work we focused on 2D pose estimation and generated data with 2D annotations but with further work, the system could be extended to generate 3D annotations and modified to produce scenes that handle occlusions, multiple dogs and interactions.

References

[1] Igor Barros Barbosa, Marco Cristani, Barbara Caputo, Aleksander Rognhaugen, and Theoharis Theoharis. Looking beyond appearances: Synthetic training data for deep cnns in re-identification. *CoRR*, abs/1701.03153, 2017. 1

[2] Benjamin Biggs, Oliver Boyne, James Charles, Andrew Fitzgibbon, and Roberto Cipolla. Who left the dogs out? 3d animal reconstruction with expectation maximization in the loop, 2020. 1, 3

[3] Benjamin Biggs, Oliver Boyne, James Charles, Andrew Fitzgibbon, and Roberto Cipolla. Who left the dogs out?: 3D animal reconstruction with expectation maximization in the loop. In *ECCV*, 2020. 3

[4] Benjamin Biggs, Thomas Roddick, Andrew Fitzgibbon, and Roberto Cipolla. Creatures great and SMAL: Recovering the shape and motion of animals from video. In *ACCV*, 2018. 1

[5] Jinkun Cao, Hongyang Tang, Haoshu Fang, Xiaoyong Shen, Cewu Lu, and Yu-Wing Tai. Cross-domain adaptation for animal pose estimation. *CoRR*, abs/1908.05806, 2019. 1

[6] Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Openpose: Realtime multi-person 2d pose estimation using part affinity fields. *CoRR*, abs/1812.08008, 2018. 1

[7] Wenzheng Chen, Huan Wang, Yangyan Li, Hao Su, Changhe Tu, Dani Lischinski, Daniel Cohen-Or, and Baoquan Chen. Synthesizing training images for boosting human 3d pose estimation. *CoRR*, abs/1604.02703, 2016. 1

[8] Haoshu Fang, Shuqin Xie, and Cewu Lu. RMPE: regional multi-person pose estimation. *CoRR*, abs/1612.00137, 2016. 1

[9] Adam Gosztolai, Semih Günel, Marco Pietro Abrate, Daniel Morales, Victor Lobato Ríos, Helge Rhodin, Pascal Fua, and Pavan Ramdya. Liftpose3d, a deep learning-based approach for transforming 2d to 3d pose in laboratory animals. *bioRxiv*, 2020. 1

[10] Jacob M Graving, Daniel Chae, Hemal Naik, Liang Li, Benjamin Koger, Blair R Costelloe, and Iain D Couzin. Deepposekit, a software toolkit for fast and robust animal pose estimation using deep learning. *eLife*, 8:e47994, 2019. 1

[11] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask R-CNN. *CoRR*, abs/1703.06870, 2017. 1, 3

[12] Tadanobu Inoue, Subhajit Chaudhury, Giovanni De Magistris, and Sakyasingha Dasgupta. Transfer learning from synthetic to real images using variational autoencoders for precise position detection. *CoRR*, abs/1807.01990, 2018. 1

[13] Sinead Kearney, Wenbin Li, Martin Parsons, Kwang In Kim, and Darren Cosker. Rgbd-dog: Predicting canine pose from rgbd sensors. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 1, 2

[14] Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, and Li Fei-Fei. Novel dataset for fine-grained image categorization. In *First Workshop on Fine-Grained Visual Categorization, IEEE Conference on Computer Vision and Pattern Recognition*, Colorado Springs, CO, June 2011. 3

[15] Sijin Li and Antoni B. Chan. 3d human pose estimation from monocular images with deep convolutional neural network. In *ACCV*, 2014. 3

[16] Alexander Mathis, Pranav Mamidanna, Kevin M. Cury, Taiga Abe, Venkatesh N. Murthy, Mackenzie Weygandt Mathis, and Matthias Bethge. Deeplabcut: markerless pose estimation of user-defined body parts with deep learning. *Nature Neuroscience*, 21(9):1281–1289, Sep 2018. 1

[17] Jiteng Mu, Weichao Qiu, Gregory D. Hager, and Alan L. Yuille. Learning from synthetic animals. *CoRR*, abs/1912.08265, 2019. 1

[18] Elon Musk. An integrated brain-machine interface platform with thousands of channels. *J Med Internet Res*, 21(10):e16194, Oct 2019. 1

[19] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. *CoRR*, abs/1603.06937, 2016. 1, 3

[20] OpenAI, Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, Jonas Schneider, Nikolas Tezak, Jerry Tworek, Peter Welinder, Lilian Weng, Qiming Yuan, Wojciech Zaremba, and Lei Zhang. Solving rubik's cube with a robot hand. *CoRR*, abs/1910.07113, 2019. 1

[21] Xue Bin Peng, Erwin Coumans, Tingnan Zhang, Tsang-Wei Edward Lee, Jie Tan, and Sergey Levine. Learning agile robotic locomotion skills by imitating animals. In *Robotics: Science and Systems*, 07 2020. 1

[22] T.D. Pereira, D. E. Aldarondo, L. Willmore, M. Kislin, S. S.-H. Wang, M. Murthy, and J. W. Shaevitz. Fast animal pose estimation using deep neural networks. *bioRxiv*, 2018. 1

[23] Stephan R. Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games. *CoRR*, abs/1608.02192, 2016. 1

[24] Arnaud Rougetet. Landscape pictures, 2019. 3

[25] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. *CoRR*, abs/1902.09212, 2019. 1

[26] Jonathan Tompson, Ross Goroshin, Arjun Jain, Yann LeCun, and Christoph Bregler. Efficient object localization using convolutional networks. *CoRR*, abs/1411.4280, 2014. 1

[27] Alexander Toshev and Christian Szegedy. Deeppose: Human pose estimation via deep neural networks. *CoRR*, abs/1312.4659, 2013. 1

[28] He Zhang, Sebastian Starke, Taku Komura, and Jun Saito. Mode-adaptive neural networks for quadruped motion control. *ACM Trans. Graph.*, 37(4), July 2018. 2

[29] Silvia Zuffi, Angjoo Kanazawa, Tanya Y. Berger-Wolf, and Michael J. Black. Three-d safari: Learning to estimate zebra pose, shape, and texture from images "in the wild". *CoRR*, abs/1908.07201, 2019. 1