Learning the 3D Fauna of the Web
– Supplementary Material –

Zizhang Li[1*] Dor Litvak[1,2*] Ruining Li[3] Yunzhi Zhang[1] Tomas Jakab[3] Christian Rupprecht[3]
Shangzhe Wu[1†] Andrea Vedaldi[3†] Jiajun Wu[1†]
[1]Stanford University [2]UT Austin [3]University of Oxford

1. Additional Results

We provide additional visualizations, including shape interpolation and generation, as well as additional comparisons in this supplementary material. Please see https://kyleleey.github.io/3DFauna/ for 3D animations.

1.1. Shape Interpolation between Instances

With the predictions of our model, we can easily interpolate between two reconstructions by interpolating the base embeddings $\tilde{\phi}$, instance deformations and the articulated poses ξ, as illustrated in Fig. 2. Here, we first obtain the predicted base shape embeddings $\tilde{\phi}$ for each of the three input images from the learned Semantic Bank. We then linearly interpolate between these embeddings to produce smooth a transition from one base shape to another, as shown in the last row of Fig. 2. Furthermore, we can also linearly interpolate the predicted articulated the image features ϕ (which is used as a condition to the instance deformation field $f_{\Delta V}$) as well as the predicted articulation parameters ξ, to generate smooth interpolations of between posed shapes, shown in the middle row. These results confirm that our learned shape space is continuous and smooth, and covers a wide range of animal shapes.

1.2. Shape Generation from the Semantic Bank

Moreover, we can also *generate* new animal shapes by sampling from the learned Semantic Bank, as shown in Fig. 3. First, we visualize the base shapes captured by each of the learned value tokens ϕ_k^{val} in the Semantic Bank. In the top two rows of Fig. 3, we show 20 visualizations of these base shapes randomly selected out of the 60 value tokens in total. We can also fuse these base shapes by linearly fusing the value tokens ϕ_k^{val} with a set of random weights (with a sum of 1), and generate the a wide variety of animal shapes, as shown in the bottom two rows.

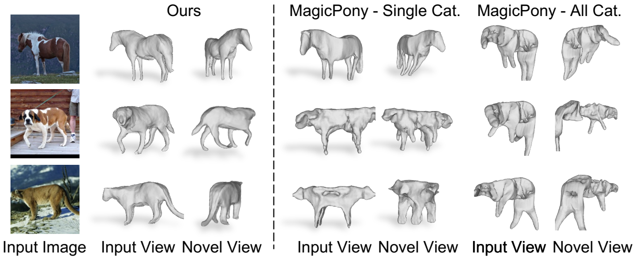

Figure 1. **Qualitative Comparisons** against two variants of MagicPony [11]. In the middle are reconstruction results of the category-specific MagicPony model trained on individual categories. On the right are results of MagicPony trained on all categories jointly, i.e. assuming all quadrupeds belong to one single category.

1.3. Comparisons with Prior Work

Quantitative Results for Each Category. Here, we provide the per-category performance break for the quantitative comparisons in Tab. 1, which correspond to the aggregated results in Tab. 1 of the main paper. On APT36K [13], we evaluate on four categories including horse, giraffe, cow and zebra. On Animal3D [12], we use the available three categories: horse, cow and zebra. Our pan-category model consistently outperforms the MagicPony [11] baseline across all the categories, which highlights the benefits of the joint training of all categories. We also compare to LASSIE [14] and Hi-LASSIE [15] quantitatively by optimizing on three Animal3D categories individually, as each category contains a small size (< 100) of images similar to the default setup proposed in their papers.

MagicPony on All Categories. In Fig. 5 of the main paper, we show that MagicPony [11] fail to produce plausible 3D shapes when trained in a *category-specific* fashion on species with limited (< 100) number of images. Alternatively, we can also train the MagicPony on our entire image dataset of all the animal species, i.e. treating all the images as in one single category. The results are shown in Fig. 1. As MagicPony maintains only one single base shape for all

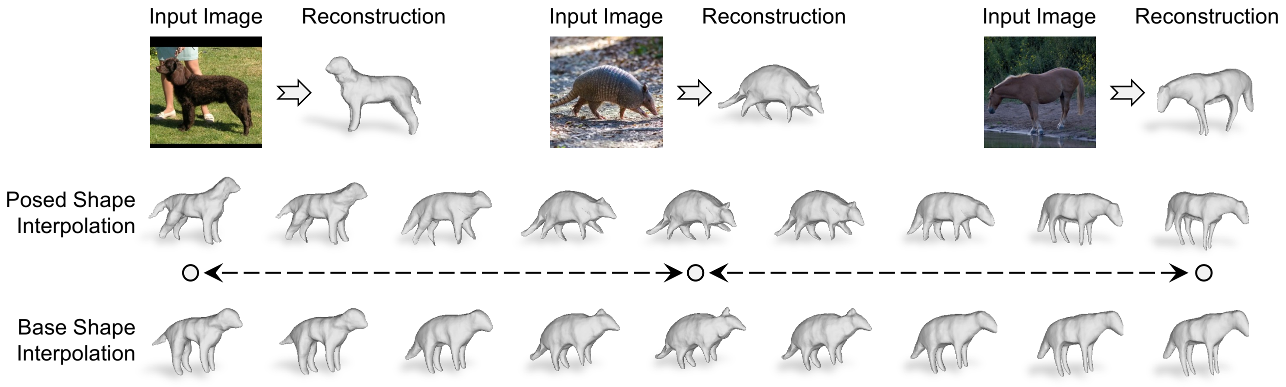

Figure 2. **Shape Interpolation between Instances.** On the top row, we show the 3D reconstructions from three input images. On the second and the third rows, we show the interpolation between the posed shapes and the base shapes.

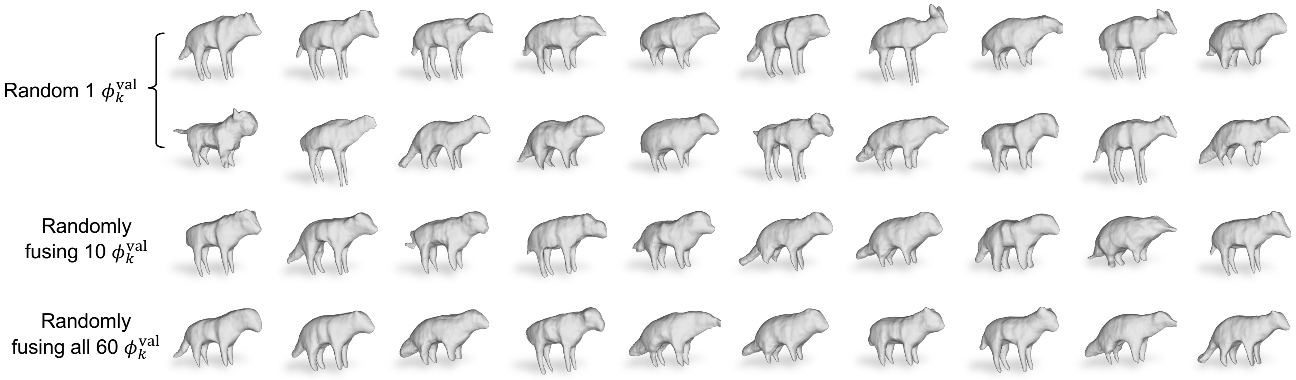
Random 1 ϕ_k^{val}
Randomly fusing 10 ϕ_k^{val}
Randomly fusing all 60 ϕ_k^{val}

Figure 3. **Shape Generation from the Learned Semantic Bank.** On the top two rows, we visualize 20 base shapes generated from the individual value tokens ϕ_k^{val} in the learned Semantic Bank. On the bottom two rows, we show the base shapes obtained by randomly fusing 10 and 60 value tokens ϕ_k^{val}.

	APT-36K			
	Horse	Giraffe	Cow	Zebra
MagicPony [?]	0.775	0.699	0.769	0.778
Ours	**0.853**	**0.796**	**0.876**	**0.840**

	Animal3D		
	Horse	Cow	Zebra
LASSIE [14]	0.850	0.887	0.878
Hi-LASSIE [15]	0.410	0.720	0.704
MagicPony [?]	0.835	0.895	0.919
Ours	**0.884**	**0.903**	**0.942**

Table 1. **Quantitative Comparisons** on APT-36K [13] and Animal3D [12] for each category. Our method consistently performs better than MagicPony [11], LASSIE [14] and Hi-LASSIE [15] on all the categories.

	APT-36K			
	Horse	Giraffe	Cow	Zebra
Final Model	**0.853**	**0.796**	**0.876**	**0.840**
w/o Semantic Bank	0.402	0.398	0.371	0.373
Category-conditioned	0.822	0.776	0.832	0.798
w/o $\mathcal{L}_{\text{adv}}$	0.831	0.782	0.823	0.828

	Animal3D		
	Horse	Cow	Zebra
Final Model	**0.884**	**0.903**	**0.942**
w/o Semantic Bank	0.402	0.701	0.630
Category-conditioned	0.842	0.886	0.910
w/o $\mathcal{L}_{\text{adv}}$	0.813	0.871	0.873

Table 2. **Quantitative Ablation Studies** on APT-36K [13] and Animal3D [12] for each category.

K	2	10	60	100	500
PCK0.1	0.724	0.766	0.782	0.788	0.789

Table 3. **Bank Size Ablation Studies** on PASCAL [1].

animal instances, which is not able to capture the wide variation of shapes of different animal species. On the contrary, our proposed Semantic Base Shape Bank learns various base shapes automatically adapted to different species, based on self-supervised image features.

1.4. Quantitative Ablation Studies

In addition to the qualitative comparisons in Fig. 6 of the main paper, Tab. 2 shows the quantitative ablation studies on APT-36K [13] and Animal3D [12]. As explained in Sec. 5.3 of the paper, we follow CMR [2] and optimize a linear mapping from our predicted vertices to the annotated keypoints in the *input view*. These numerical results are consistent with the visual comparisons in Fig. 6 of the main paper.

We also conducted additional experiments with different bank sizes, including $K = 2$, 10, 60, 100, 500, and report the PCK scores on PASCAL [1] in Tab. 3. The quality grows with K; we pick $K = 60$ as a good trade-off with the computational cost.

1.5. More Visualizations from 3D-Fauna

We show more visualization results of 3D-Fauna on a wide variety of animals in Figure 7, Figure 8 and Figure 9, including horse, weasel, pika, koala and so on. Note that our model produces these articulated 3D reconstructions from just a single test image in feed-forward manner, without even knowing the category labels of the animal species. With the articulated pose prediction, we can also easily animate the reconstructions in 3D. More visualizations are presented at `https://kyleleey.github.io/3DFauna/`.

1.6. Failure Cases and Limitations

Despite promising results on a wide variety of quadruped animals, we still recognize a few limitations of the current method. First, we only focus on quadrupeds which share a similar skeletal structure. Although this covers a large number animals, including most mammals as well as many reptiles, amphibians and insects, the same assumption will not hold for many other animals in nature. Jointly estimating the skeletal structure and 3D shapes directly from raw images remains a fundamental challenge for modeling the entire biodiversity. Furthermore, for some fluffy animals that are highly deformable, like cats and squirrels, our model still struggles to reconstruct accurate poses and 3D shapes, as shown in Fig. 4.

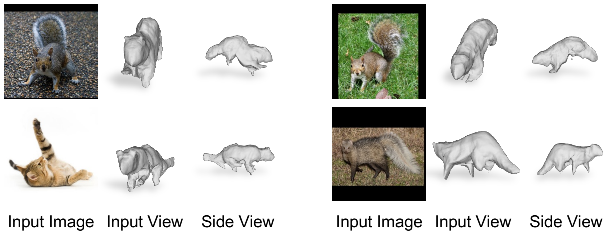

Figure 4. **Failure Cases.** For fluffy and highly deformable animals in challenging poses, our model still struggles in predicting the accurate poses and shapes.

Another failure case is the confusion of left and right legs, when reconstructing images taken from the side view, for instance, in the second row of Fig. 7. Since neither the object mask nor the self-supervised features [8] can provide sufficient signals to disambiguate the legs, the model would ultimately have to resort to the subtle appearance cues, which still remains as a major challenge. Finally, the current model still struggles at inferring high-fidelity appearance in a feed-forward manner, similar to [11], and hence, we still employ a fast test-time optimization for better appearance reconstruction (within seconds). This is partially due to the limited size of the dataset and the design of the texture field. Leveraging powerful diffusion-based image generation models [9] could provide additional signals to train a more effective 3D appearance predictor, which we plan to look into for future work.

2. Additional Technical Details

2.1. Modeling Articulations

In this work, we focus on quadruped animals which share a similar quadrupedal skeleton. Here, we provide the details for the bone instantiation on the rest-pose shape based on a simple heuristic, the skinning model, and the additional bone rotation constraints.

Adaptive Bone Topology. We adopt a similar quadruped heuristic for rest-pose bone estimation as in [11]. However, unlike [11] which focuses primarily on horses, our method needs to model a much more diverse set of animal species. Hence, we make several modifications in order for the model to adapt to different animals automatically. For the 'spine', we still use a chain of 8 bones with equal lengths, connecting the center of the rest-pose mesh to the two most extreme vertices along z-axis. To locate the four feet joints, we do not rely on the four xz-quadrants as the feet may not always land separately in those four quadrants, for instance, for animals with a longer body. Instead, we locate the feet based on the distribution of the vertex locations. Specifically, we first identify the vertices within the lower 40% of the total height (y-axis). We then use the center of these vertices as the origin of the xz-plane and locate the lowest vertex within each of the new quadrants as the feet joints. For each leg, we create a chain of three bones of equally length connecting the foot joint to the nearest joint in the spine.

Bone Rotation Prediction. Similar to [11], the viewpoint and bone rotations are predicted separately using different networks. The viewpoint ξ_1 is predicted via a multi-hypothesis mechanism, as discussed in Sec. 2.2. For the bone rotations $\xi_{2:B}$, we first project the middle point of each *rest-pose* bone onto the image using the predicted viewpoint, and sample its corresponding local feature from the feature map using bilinear interpolation. A Transformer-based [10] network then fuses the global image feature, local image feature, 2D and 3D joint locations as well as the bone index, and produces the Euler angle for the rotation of each bone. Unlike [11], we empirically find it beneficial to add the bone index on top of other features instead of concatenation, which tends to encourage the model to separate the legs with different rotation predictions.

Skinning Weights. With the estimated bone structure, each bone b except for the root has the parent bone $\pi(b)$. Each vertex $V_{\text{ins},i}$ on the shape V_{ins} is then associated to all the bones by skinning weights w_{ib} defined as:

$$\begin{aligned} w_{ib} &= \frac{e^{-d_{ib}/\tau_s}}{\sum_{k=1}^{B} e^{-d_{ik}/\tau_s}}, \quad \text{where} \\ d_{ib} &= \min_{r\in[0,1]} ||V_{\text{ins},i} - r\tilde{\mathbf{J}}_b - (1-r)\tilde{\mathbf{J}}_{\pi(b)}||_2^2 \end{aligned} \tag{1}$$

is the minimal distance from the vertex $V_{\text{ins},i}$ to each bone b, defined by the rest-pose joint location $\tilde{\mathbf{J}}_b$ in world coordinates. The τ_s is a temperature parameter set to 0.5. We then use the *linear blend skinning equation* to pose the vertices:

$$\begin{aligned} V_i(\xi) &= \left(\sum_{b=1}^{B} w_{ib} G_b(\xi) G_b(\xi^*)^{-1}\right) V_{\text{ins},i}, \\ G_1 = g_1, \quad G_b &= G_{\pi(b)} \circ g_b, \quad g_b(\xi) = \begin{bmatrix} R_{\xi_b} & \mathbf{J}_b \\ 0 & 1 \end{bmatrix}, \end{aligned} \tag{2}$$

where the ξ^* denotes the bone rotations at rest pose.

Bone Rotation Constraints. Following [11], we regularize the magnitude of bone rotation predictions by $\mathcal{R}_{\text{art}} = \frac{1}{B-1}\sum_{b=2}^{B} ||\xi_b||_2^2$. In experiments, we find a common failure mode where instead of learning a reasonable shape with appropriate leg lengths, the model tends to predict excessively long legs for animals with shorter legs and bend them away from the camera. To avoid this, we further constrain the range of the angle predictions. Specifically, we forbid the rotation along y-axis (side-way) and z-axis (twist) of the lower two segments for each leg. We also set a limit to the rotation along y-axis and z-axis of the upper segment for each leg as $(-10°, 10°)$. For the body bones, we further limit the rotation along the z-axis within $(-6°, 6°)$.

2.2. Viewpoint Learning Details

Recovering the viewpoint of an object from only one input image is an ill-posed problem with numerous local optima in the reconstruction objective. Here, we adopt the multi-hypothesis viewpoint prediction scheme introduced in [11]. In detail, our viewpoint prediction network outputs four viewpoint rotation hypotheses $R_k \in SO(3), k \in \{1,2,3,4\}$ within each of the four xz-quadrants together with their corresponding scores σ_k. For computational efficiency, we randomly sample one hypothesis at each training iteration, and minimize the loss:

$$\mathcal{L}_{\text{hyp}}(\sigma_k, \mathcal{L}_{\text{rec},k}) = (\sigma_k - \texttt{detach}(\mathcal{L}_{\text{rec},k}))^2, \tag{3}$$

where `detach` indicates that the gradient on reconstruction loss is detached. In this way, σ_k essentially serves as an estimate of the expected reconstruction error for each hypothesis k, without actually evaluating it which would otherwise require the expensive rendering step. During inference time, we can then take the `softmax` of its inverse to obtain the probability p_k of each hypothesis k: $p_k \propto \exp(-\sigma_k/\tau)$, where the temperature parameter τ controls the sharpness of the distribution.

2.3. Mask Discriminator Details

To sample another viewpoint and render the mask for the mask discriminator, we randomly sample an azimuth angle and rotate the predicted viewpoint by that angle. For conditioning, the detached input base embedding $\tilde{\phi}$ is concatenated to each pixel in the mask along the channel dimension, similar to CycleGAN [17]. In practice, we also add a gradient penalty term in the discriminator loss following [7, 16].

2.4. Network Architectures

We adopt the architectures in [11] except the newly introduced Semantic Base Shape Bank and mask discriminator. For the SBSM, we add a modulation layer [3, 4] to each of the MLP layers to condition the SDF field on the base embeddings $\tilde{\phi}$. To condition the DINO field, we simply concatenate the embedding to the input coordinates to the network. The mask discriminator architecture is identical to that of GIRAFFE [7], except that we set input dimension as $129 = 1 + 128$, accommodating the 1-channel mask and the 128-channel shape embedding. We set the size of the memory bank $K = 60$. In practice, to allow bank to represent categories with diverse kinds of shapes, we only fuse the value tokens with top 10 cosine similarities.

Parameter	Value/Range
Optimiser	Adam
Learning rate on prior and bank	1×10^{-3}
Learning rate on others	1×10^{-4}
Number of iterations	800k
Enable articulation iteration	20k
Enable deformation iteration	500k
Mask Discriminator iterations	(80k, 300k)
Batch size	6
Loss weight λ_{m}	10
Loss weight λ_{im}	1
Loss weight λ_{feat}	$\{10, 1\}$
Loss weight λ_{Eik}	0.01
Loss weight λ_{def}	10
Loss weight λ_{art}	0.2
Loss weight λ_{hyp}	$\{50, 500\}$
Loss weight λ_{adv}	0.1
Image size	256×256
Field of view (FOV)	$25°$
Camera location	$(0, 0, 10)$
Tetrahedral grid size	256
Initial mesh centre	$(0, 0, 0)$
Translation in x- and y-axes	$(-0.4, 0.4)$
Translation in z-axis	$(-1.0, 1.0)$
Number of spine bones	8
Number of bones for each leg	3
Viewpoint hypothesis temperature τ	$(0.01, 1.0)$
Skinning weight temperature τ_{s}	0.5
Ambient light intensity k_a	$(0.0, 1.0)$
Diffuse light intensity k_d	$(0.5, 1.0)$

Table 4. **Training details and hyper-parameter settings.**

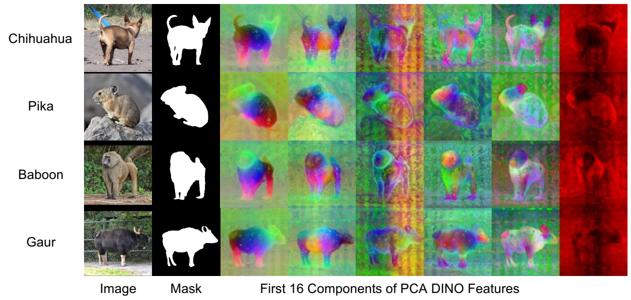

Figure 5. **Data Samples.** We show some samples of our training data. Each sample consists of the RGB image, automatically-obtained segmentation mask, and the corresponding 16-channel PCA feature map.

2.5. Hyper-Parameters and Training Schedule

The hyper-parameters and training details are listed in Tab. 4. We train the model for 800k iterations on a single NVIDIA A40 GPU, which takes roughly 5 days. In particular, we set λ_{feat}=10, and λ_{hyp}=50 at the start of training. After 300k iterations we change the values to λ_{feat}=1, λ_{hyp}=500. During the first 6k iterations, we allow the model to explore all four viewpoint hypotheses by randomly sampling the four hypotheses uniformly, and gradually decrease the chance of random sampling to 20% while sampling the best hypothesis for the rest 80% of the time. To save memory and computation, at each training iteration, we only feed images of the same species in a batch, and extract one base shape by averaging out the base embeddings. At test time, we just directly use the shape embedding for each individual input image.

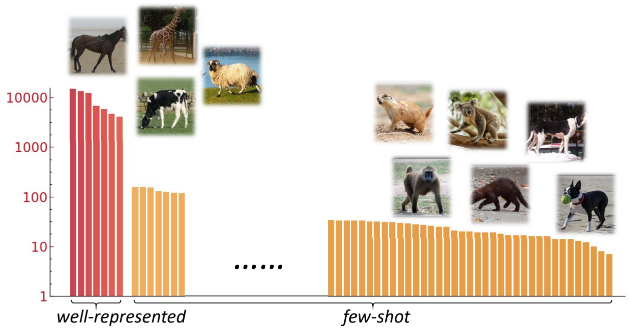

Figure 6. **Species Distribution.** We show the distribution of different animal species in our training dataset, including well-represented species with thousands of images and rare species with less than 100 images.

2.6. Data Pre-Processing

We use off-the-shelf segmentation models [5, 6] to obtain the masks, crop around the objects and resize the crops to a size of 256 × 256. For the self-supervised features [8], we randomly choose 5k images from our dataset to compute the Principal Component Analysis (PCA) matrix. Then we use that matrix to run inference across all the images in our dataset. We show some samples of different animal species in Fig. 5. It is evident that these self-supervised image features can provide efficient semantic correspondences across different categories. Note that masks are only for supervision, our model takes the raw image shown on the left as input for inference.

2.7. Species Size Distribution

We show a plot of the distribution of different species in our dataset below, including 7 well-represented categories (red) and 121 few-shot categories (orange). To balance the training, we duplicate the samples of few-shot categories to match the size of the rest. Many examples in Fig. 4 of the main paper and Fig. 7 in fact belong to the few-shot categories, such as koala, fisher and prairie dog.

References

[1] Mark Everingham, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. *IJCV*, 2015. 3

[2] Angjoo Kanazawa, Shubham Tulsiani, Alexei A Efros, and Jitendra Malik. Learning category-specific mesh reconstruction from image collections. In *ECCV*, 2018. 3

[3] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *CVPR*, 2020. 4

[4] Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Alias-free generative adversarial networks. *NeurIPS*, 2021. 4

[5] Alexander Kirillov, Yuxin Wu, Kaiming He, and Ross Girshick. Pointrend: Image segmentation as rendering. In *CVPR*, 2020. 5

[6] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *ICCV*, 2023. 5

[7] Michael Niemeyer and Andreas Geiger. GIRAFFE: Representing scenes as compositional generative neural feature fields. In *CVPR*, 2021. 4

[8] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023. 3, 5

[9] Yawar Siddiqui, Justus Thies, Fangchang Ma, Qi Shan, Matthias Nießner, and Angela Dai. Texturify: Generating textures on 3d shape surfaces. In *ECCV*, 2022. 3

[10] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *NeurIPS*, 2017. 4

[11] Shangzhe Wu, Ruining Li, Tomas Jakab, Christian Rupprecht, and Andrea Vedaldi. Magicpony: Learning articulated 3d animals in the wild. In *CVPR*, 2023. 1, 2, 3, 4

[12] Jiacong Xu, Yi Zhang, Jiawei Peng, Wufei Ma, Artur Jesslen, Pengliang Ji, Qixin Hu, Jiehua Zhang, Qihao Liu, Jiahao Wang, et al. Animal3d: A comprehensive dataset of 3d animal pose and shape. In *ICCV*, 2023. 1, 2, 3

[13] Yuxiang Yang, Junjie Yang, Yufei Xu, Jing Zhang, Long Lan, and Dacheng Tao. Apt-36k: A large-scale benchmark for animal pose estimation and tracking. *NeurIPS*, 2022. 1, 2, 3

[14] Chun-Han Yao, Wei-Chih Hung, Yuanzhen Li, Michael Rubinstein, Ming-Hsuan Yang, and Varun Jampani. Lassie: Learning articulated shapes from sparse image ensemble via 3d part discovery. *NeurIPS*, 2022. 1, 2

[15] Chun-Han Yao, Wei-Chih Hung, Yuanzhen Li, Michael Rubinstein, Ming-Hsuan Yang, and Varun Jampani. Hi-lassie: High-fidelity articulated shape and skeleton discovery from sparse image ensemble. In *CVPR*, 2023. 1, 2

[16] Yunzhi Zhang, Shangzhe Wu, Noah Snavely, and Jiajun Wu. Seeing a rose in five thousand ways. In *CVPR*, 2023. 4

[17] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, 2017. 4

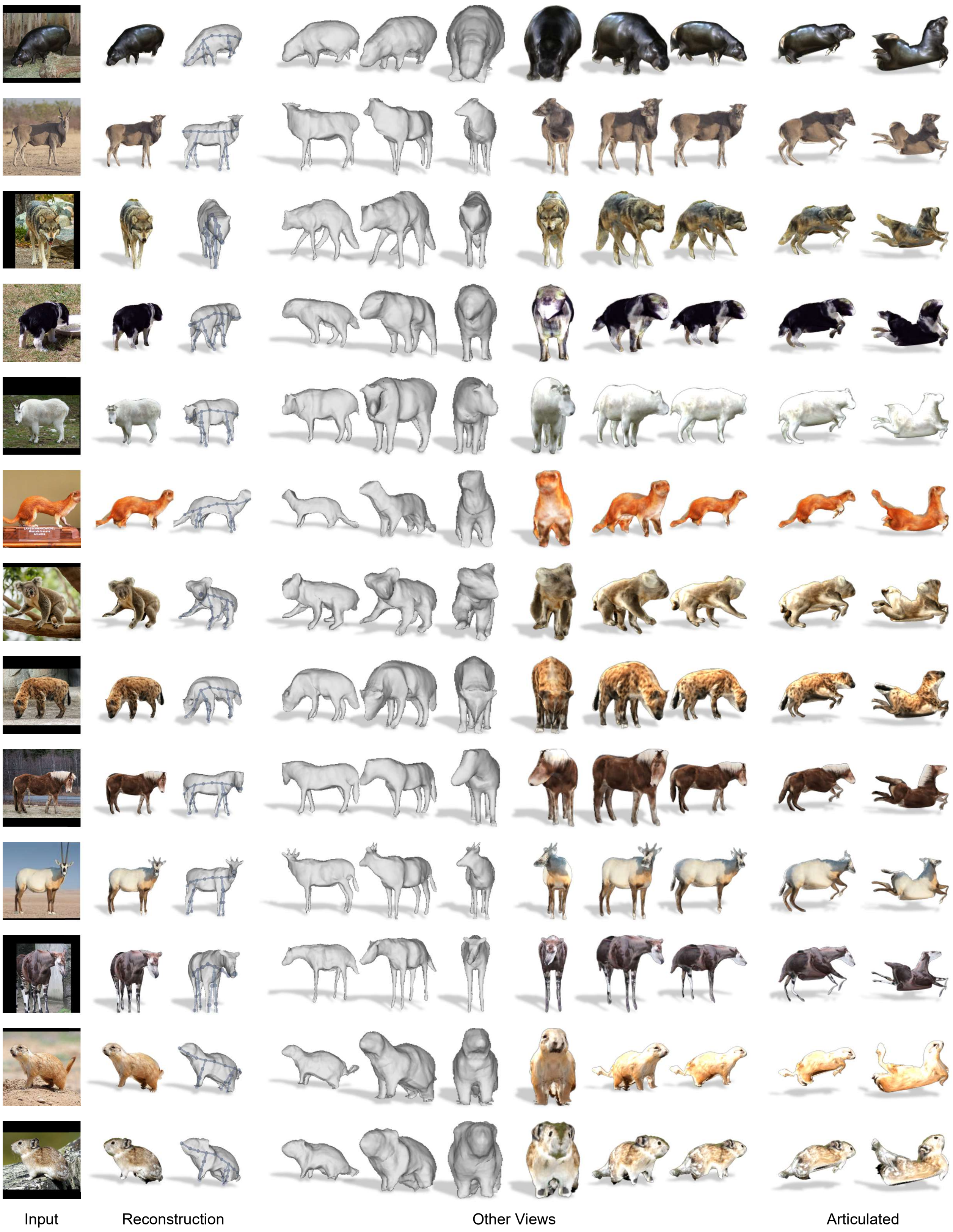

Figure 7. **Single Image 3D Reconstruction.** Given a single image of any quadruped animal at test time, our model reconstructs an articulated and textured 3D mesh in a feed-forward manner without requiring category labels, which can be readily animated.

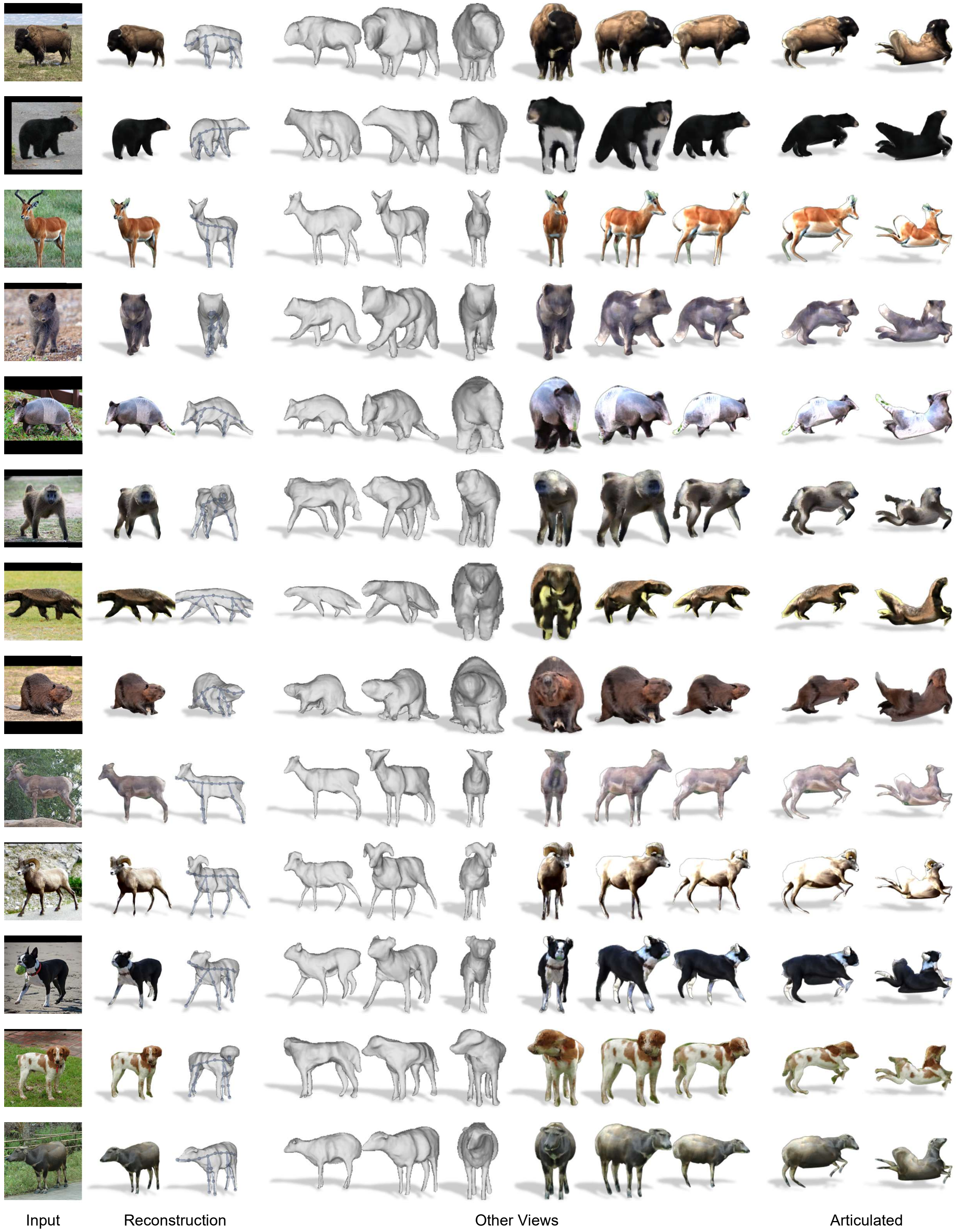

Figure 8. **Single Image 3D Reconstruction.** Given a single image of any quadruped animal at test time, our model reconstructs an articulated and textured 3D mesh in a feed-forward manner without requiring category labels, which can be readily animated.

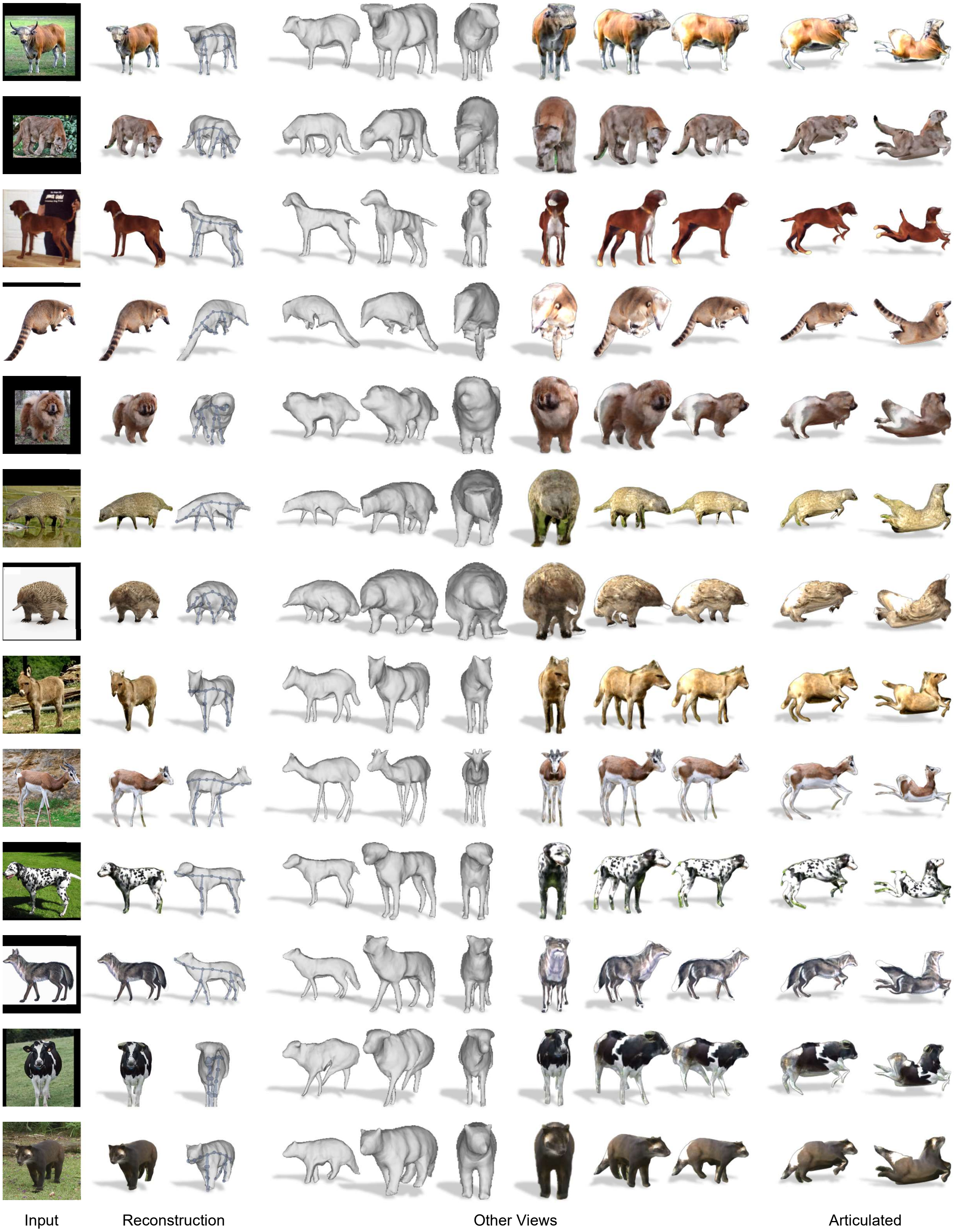

Figure 9. **Single Image 3D Reconstruction.** Given a single image of any quadruped animal at test time, our model reconstructs an articulated and textured 3D mesh in a feed-forward manner without requiring category labels, which can be readily animated.