SAOR: Single-View Articulated Object Reconstruction

Supplementary Material

A. Additional Results

Qualitative Results. In Fig. A3 and Fig. A4 we present additional qualitative results on various animal categories all generate using our SAOR models that are trained on multiple categories. We provide additional results showing full 360-degree predictions for multiple different categories on the project website: mehmetaygun.github.io/saor.

Part Consistency. We also compared SAOR's surface estimates with A-CSM [7] in Fig. A1. Unlike A-CSM, our method does not use any 3D parts or 3D shape priors but is still able to capture finer details like discriminating left and right legs. A-CSM groups left and right legs as a single leg while their reference 3D template has left and right legs as a separate entity. Moreover, it mixes left-right consistency if the viewpoint changes.

Without Depth. We also demonstrate examples from a variant of our model that was trained *without* using relative depth map supervision in Fig. A2. We observe that this model is still capable of estimating detailed 3D shapes with accurate viewpoints and similar textures as the full model. However, the model trained without depth maps tends to produce wider shapes compared to the full model. Quantitative results for our model without relative depth are available in Table 2 in the main paper.

Limitations. We showcase some failure cases of our method in Fig. A5. Our method fails when the animal is captured from the back, as there is insufficient data available from that angle in the training sets. Note, methods such as [13] partially address this by using alternative training data that includes image sequences from video. Furthermore, when there is also partial visibility (e.g., only the head is visible), our method produces less meaningful results as our architecture does not explicitly model occlusion.

Part Ablations. We conducted an additional ablation experiment on the number of parts used for horses. Results are provided in Table A1. Notably, the PCK scores do not significantly vary with different numbers of parts. Therefore, for all other experiments, we used 12 parts.

Number of Parts	6	12	24
PCK	43.8	44.9	44.1

Table A1. Keypoint transfer results on Pascal horses [2] where the number of parts are varied.

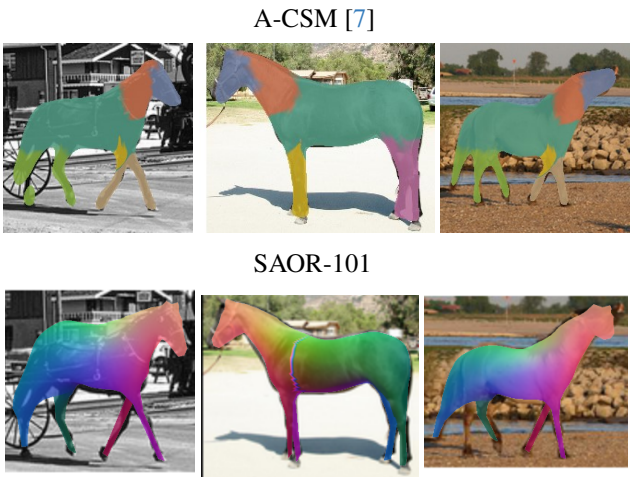

Figure A1. Comparison with A-CSM [7] on horses using example images from their paper. Even though A-CSM uses a 3D template with pre-defined fixed parts, it still maps left and right legs to the same leg in the template and the legs are not consistent across viewpoints (i.e., the part assignment is different in the top row depending on whether the horse is facing left or right. In contrast, despite not using any 3D object priors at training time, our method is much more consistent in its assignment. However, it does mistake one of the left legs for the horse's tail in the final column.

B. Additional Implementation Details

B.1. Data Pre-Processing

When constructing our training datasets, we run a general-purpose animal detector [1] and eliminate objects if any of the following criteria hold: i) the confidence of the detection is less than 0.8, ii) the minimum side of the bounding box is less than 32 pixels, iii) the maximum side of the bounding box is less than 128 pixels, and iv) there is no margin greater than 10 pixels on all sides of the bounding box.

We then automatically extract segmentation masks using Segment Anything Model [6] with the detected bounding box. We automatically estimate the relative monocular depth using the transformer-based Midas [9, 10], using their Large DPT model.

To obtain cluster centers for the balanced sampling step in Section 3.3 in the main paper, we resize the estimated segmentation masks to 32×32, and cluster the 1024-dimensional vectors into 10 clusters using a Gaussian mixture model in all of our experiments. Visualization of cluster centers of various animals can be found in Fig. A6.

B.2. Architecture

We use a ResNet-50 [3] as our image encoder f_{enc} in our CUB[12] experiments and the smaller ResNet-18 in

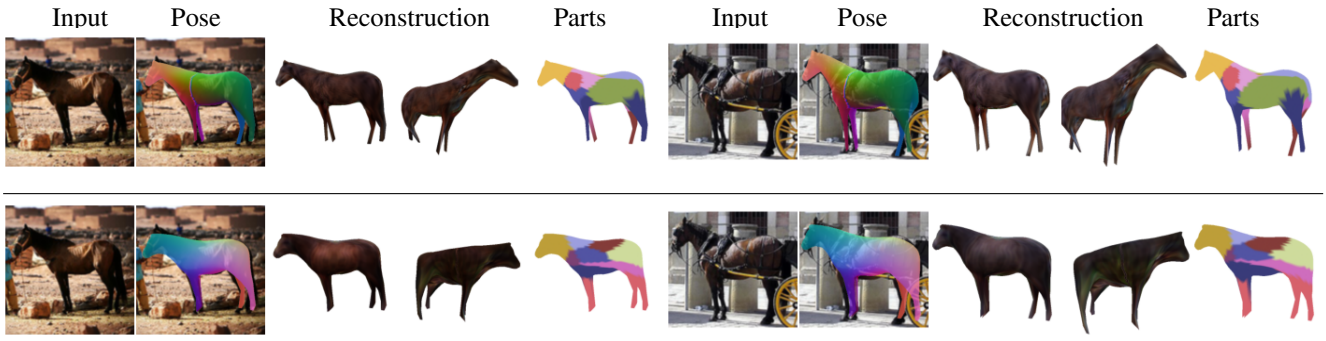

Figure A2. Comparison of models trained with relative depth supervision (top) and without (bottom). Our model trained without depth also estimates detailed 3D shapes with the correct viewpoint. However, the 3D predictions are marginally worse as the model without depth produces slightly wider 3D shapes. Please note that part assignment and pose orientation are different since the two models started from different random initializations.

Layer	Input	Output	Dim
Linear (3,512)	S°	l_x	$N \times 512$
Linear (512,512)	ϕ_{im}	l_z	1×512
2 $\times$ Linear (512,128)	$l_x + l_z$	L	$N \times 128$
Linear (128,3)	l	D	$N \times 3$

Table A2. Architecture details of our Deformation Net f_d.

Layer	Input	Output	Dim
Linear (3,512)	S°	l_x	$N \times 512$
Linear (512,512)	ϕ_{im}	l_z	1×512
Linear (512,128)	$l_x + l_z$	L	$N \times 128$
Linear (128,128)	L	L	$N \times 128$
Linear (128,K)	L	W	$N \times K$
K $\times$ Linear (512, 9)	ϕ_{enc}	π	$K \times 9$

Table A3. Architecture details of our Articulation Net f_a. K is the number of parts and N is the number of vertices, π is camera parameters.

Layer	Input	Output	Dim
Linear (512,512)	ϕ_{im}	L	$512 \times 1 \times 1$
Upsample	L	L_{up}	$512 \times 4 \times 4$
Upsample + Conv2D	L_{up}	L_{up}	$256 \times 8 \times 8$
Upsample + Conv2D	L_{up}	L_{up}	$128 \times 16 \times 16$
Upsample + Conv2D	L_{up}	L_{up}	$64 \times 32 \times 32$
Upsample + Conv2D	L_{up}	L_{up}	$32 \times 64 \times 64$
Upsample + Conv2D	L_{up}	L_{up}	$16 \times 128 \times 128$
Conv2D	L_{up}	T	$3 \times 128 \times 128$

Table A4. Architecture details of our Texture Net f_t.

Layer	Input	Output	Dim
1 $\times$ Linear (512,128)	ϕ_{im}	L	128
C $\times$ Linear (128,6)	L	$\boldsymbol{r}_p, \boldsymbol{t}_p$	128
Linear (128,C)	L	$\boldsymbol{\alpha}$	128

Table A5. Architecture details of our Pose Net f_p. C is the number of cameras, and $\boldsymbol{\alpha}$ are the associated scores for each camera [13].

quadruped animal experiments. This is in contrast to much larger ViT-based backbones used in other work [13]. We initialize these encoders from scratch, i.e., no supervised or self-supervised pre-training is used. The architecture details are presented in the following tables: deformation network f_d in Table A2, articulation network f_a in Table A3, texture network f_t in Table A4, and pose network f_p in Table A5.

B.3. 3D Evaluation Details

For 3D quantitative evaluation, we used the Animal3D dataset [14]. The dataset includes pairs of input images with their corresponding 3D models, which are estimated via optimizing the SMAL [17] model. Moreover, the 3D models are manually verified to eliminate poorly estimated shapes. We used the test split of the dataset for the horse, cow, and sheep categories. As there is no global pose alignment between our predictions and the dataset, we run the ICP algorithm to align them. We optimize rotation, $R \in \mathcal{R}^3$, translation $T \in \mathcal{R}^3$, and global scale $s \in \mathcal{R}^1$ with the Adam optimizer [5] using L1 norm as our alignment objective. We also follow the same alignment steps for the MagicPony [13] baseline.

B.4. Training Losses

Here we describe the training losses from the main paper in more detail. The appearance loss is a combination of an RGB and perceptual loss [16]. $\mathcal{L}_{appr} = \lambda_{rgb}\mathcal{L}_{rgb} + \lambda_{percp}\mathcal{L}_{percp}$. These terms are defined below,

$$\mathcal{L}_{rgb} = ||\sum_{i,j} I_{i,j} - \hat{I}_{i,j}||_2, \quad (1)$$

Figure A3. Additional qualitative results for our SAOR approach on various different animal categories. Note that the part assignment displays the part with the highest probability for each vertex, but in practice, the articulation for each vertex can be explained by a linear combination of multiple parts.

Figure A4. Additional qualitative results for our SAOR approach on various different animal categories. Note that the part assignment displays the part with the highest probability for each vertex, but in practice, the articulation for each vertex can be explained by a linear combination of multiple parts.

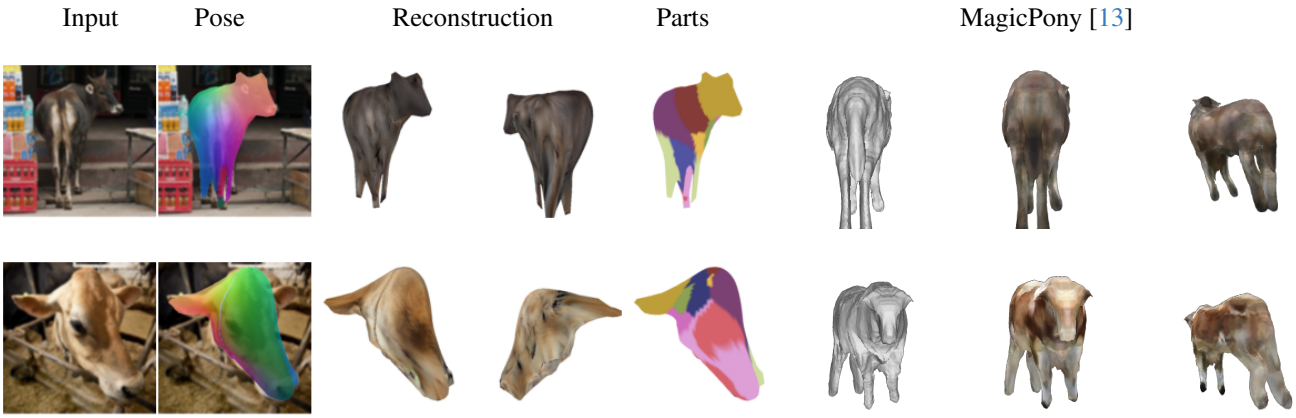

Figure A5. Failure cases on cows. On the left we see SAOR-101 predictions (estimated pose, original viewpoint reconstruction, different view, and estimated parts). On the right we display MagicPony [13] (original viewpoint reconstruction, textured reconstruction, different view). When the pose is very different than the typical ones present in the training set (top) or there is too much occlusion (bottom) our method fails to produce a sensible shape estimate. For the first example, MagicPony fails to capture the articulation of the head, and for the second occluded example it predicts an average template shape with the wrong pose.

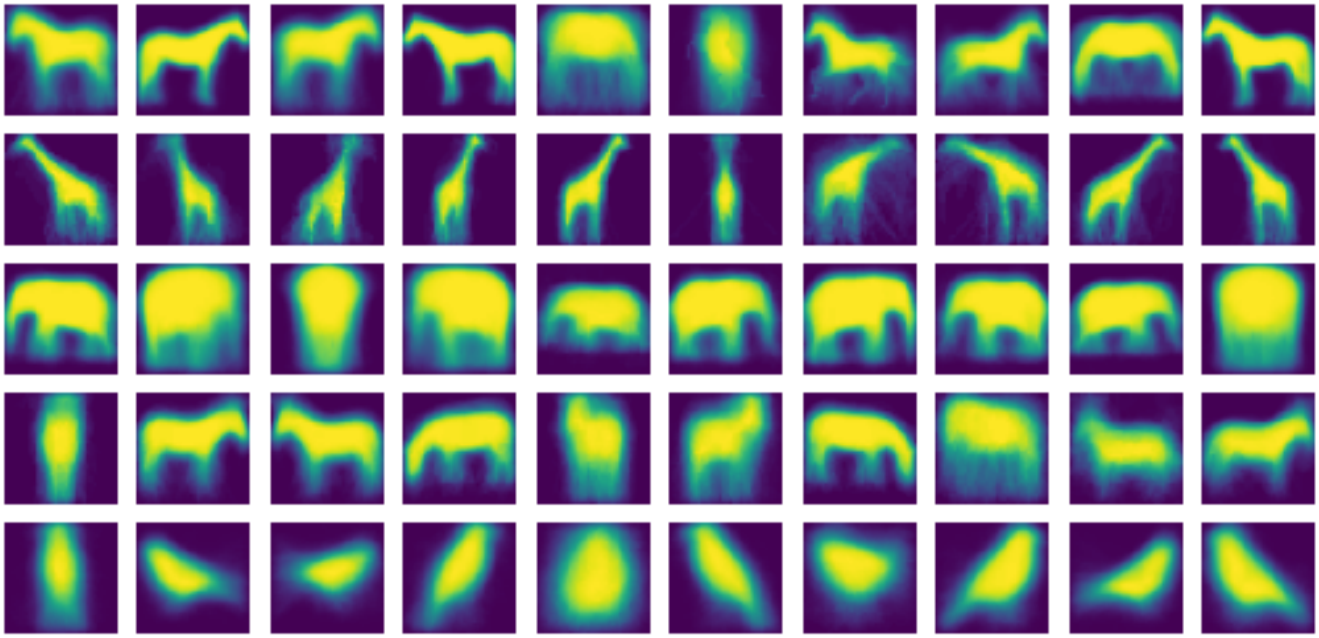

Figure A6. Visualization of the cluster centers obtained from estimated silhouettes of various animal categories used in our balanced sampling. We observe that these cluster centers broadly capture the dominant viewpoints of each object category. Top to bottom: horse, giraffe, elephant, zebra, and bird.

$$\mathcal{L}_{percp} = ||\phi_p(I_{i,j}) - \phi_p(I_{i,j})||_2, \tag{2}$$

where ϕ_p is a function that extracts features from different layers of the VGG-16 [11] network.

The mask loss is calculated based on the difference between the automatically generated ground truth segmentation mask M and the estimated mask $\hat{M}$ derived from our predicted 3D shape,

$$\mathcal{L}_{mask} = \lambda_{mask} \sum_{i,j} ||M_{i,j} - \hat{M}_{i,j}||_2. \tag{3}$$

Likewise, the depth loss is computed using the automatically generated relative depth D and the estimated depth $\hat{D}$ from the predicted shape,

$$\mathcal{L}_{depth} = \lambda_{depth} \sum_{i,j} ||D_{i,j} - \hat{D}_{i,j}||_2. \tag{4}$$

Our swap loss is a combination of the RGB and mask loss between the input image I and swapped image I^{sw},

$$\mathcal{L}_{swap} = \lambda_{swap} \left[\mathcal{L}_{mask}(I, I^{sw}) + \mathcal{L}_{rgb}(I, I^{sw}) \right]. \quad (5)$$

Finally, we also employ part regularization on the part assignment matrix W to encourage equal-sized parts,

$$\mathcal{L}_{part} = \lambda_{part} \sum_{k}^{K} \left((\sum_{i}^{N} W_{i,k}) - N/K \right)^2 \quad (6)$$

where N is the number of vertices in the mesh and K is the number of parts. We also apply 3D regularization on the 3D shape, $\mathcal{L}_{smooth} = \lambda_{smooth} \sum LS$, where L is the laplacian of shape S and $\mathcal{L}_{normal}$ which is defined below,

$$\mathcal{L}_{normal} = \lambda_{normal} \sum_{\mathbf{n_i}, \mathbf{n_j} \in \mathbf{\Omega}} 1 - \frac{\mathbf{n_i}.\mathbf{n_j}}{||\mathbf{n_i}||.||\mathbf{n_j}||} \quad (7)$$

Here, n_i, n_j are normals of neighbor faces. And the smoothness regularization is defined as $\lambda_{smooth}\mathcal{L}_{smooth} = ||LV||$, where L is the Laplacian operator on the vertices. The final regularization term is defined as,

$$\mathcal{L}_{reg} = \lambda_{part}\mathcal{L}_{part} + \lambda_{smooth}\mathcal{L}_{smooth} + \lambda_{normal}\mathcal{L}_{normal}. \quad (8)$$

We note the weights used in our experiments for each loss in Table A6.

B.5. Training

In our experiments, we trained two different models: SAOR-101 and SAOR-Birds. The bird model is trained from scratch on CUB [12] for 500 epochs. In the first 100 epochs we only learn deformation, and then enable articulation afterwards.

The SAOR-101 model is trained in two steps. We first train the model using only Horse data from LSUN [15] then finetune it on all 101 animal categories downloaded from the iNaturalist website [4]. In a similar fashion to the SAOR-Birds model, we only learn deformation in the first 100 epochs, then allow articulation for about 300 epochs on horse data. Finally, fine-tune the model on all categories on iNaturalist data for 150 epochs. We utilize Adam [5] with a fixed learning rate for optimizing our networks. We note the hyperparameters used in Table A6.

Our simplified swap loss leads to easy hyper-parameter selection compared to Unicorn [8]. For instance, in their swap loss term, the following parameters need to be decided: i) feature bank size, ii) minimum and maximum viewpoint difference, and iii) number of bins to divide samples in the feature bank depending on the viewpoint. Moreover, they need to do multistage training where they increase the latent dimensions for the shape and texture codes to obtain similar shapes during training. Here the number of stages and the dimension of latent codes in each stage are also hyperparameters. In our method, we eliminated all of these hyperparameters. Moreover, as we do not use all of the hypotheses cameras to estimate loss during a forward pass as in [13] and as a result of our simplified swap loss, model training is six times faster than Unicorn, as they use six cameras during training, for the same number of epochs.

Parameter	**Value/Range**
Optimization	
Optimizer	Adam
Learning Rate	1e-4
Batch Size	96
Epochs	500
Image Size	128×128
Mesh	
Number of Vertices	2562
Number of Faces	5120
UV Image Size	$64 \times 128 \times 3$
Number of Parts	12
Initial Position	(0,0,0)
Camera	
Translation Range	(-0.5, 0.5)
Azim Range	(-180,180)
Elev Range	(-15, 30)
Roll Range	(-30, 30)
FOV	30
Number of Cameras	4
Loss Weights	
λ_{rgb}	1
λ_{percp}	10
λ_{mask}	1
λ_{depth}	1
λ_{swap}	1
λ_{smooth}	0.1
λ_{normal}	0.1
λ_{part}	1
λ_{pose}	0.05

Table A6. Training hyperparameters.

References

[1] Sara Beery, Dan Morris, and Siyu Yang. Efficient pipeline for camera trap image review. In *Data Mining and AI for Conservation Workshop at KDD*, 2019. 1

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

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 1

[4] iNaturalist. iNaturalist. www.inaturalist.org, accessed 8 November 2023. 6

[5] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 2, 6

[6] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. In *ICCV*, 2023. 1

[7] Nilesh Kulkarni, Abhinav Gupta, David Fouhey, and Shubham Tulsiani. Articulation-aware canonical surface mapping. In *CVPR*, 2020. 1

[8] Tom Monnier, Matthew Fisher, Alexei A Efros, and Mathieu Aubry. Share with thy neighbors: Single-view reconstruction by cross-instance consistency. In *ECCV*, 2022. 6

[9] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. *ICCV*, 2021. 1

[10] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *PAMI*, 2022. 1

[11] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *ICLR*, 2015. 5

[12] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011. 1, 6

[13] Shangzhe Wu, Ruining Li, Tomas Jakab, Christian Rupprecht, and Andrea Vedaldi. Magicpony: Learning articulated 3d animals in the wild. In *CVPR*, 2023. 1, 2, 5, 6

[14] 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. 2

[15] Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. LSUN: Construction of a Large-scale Image Dataset using Deep Learning with Humans in the Loop. *arXiv:1506.03365*, 2015. 6

[16] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. 2

[17] Silvia Zuffi, Angjoo Kanazawa, David W Jacobs, and Michael J Black. 3d menagerie: Modeling the 3d shape and pose of animals. In *CVPR*, 2017. 2