Supplementary Material for
Hi-LASSIE: High-Fidelity Articulated Shape and Skeleton Discovery from Sparse Image Ensemble

Chun-Han Yao[1*] Wei-Chih Hung[2] Yuanzhen Li[3] Michael Rubinstein[3]
Ming-Hsuan Yang[134] Varun Jampani[3]

[1]UC Merced [2]Waymo [3]Google Research [4]Yonsei University

We present the implementation details, model analyses, and additional results of Hi-LASSIE in this supplementary document. Given the 3D nature of our results, we also provide a short video to describe our framework with illustrations and visual results. We encourage readers to see the supplementary video for better visualization of our 3D reconstructions.

1. Implementation Details

We implement the Hi-LASSIE framework in PyTorch [8]. All model parameters are updated by an Adam optimizer [4]. The 3D surfaces are rendered at a resolution of 384×384 for shape optimization. Given an image ensemble with 30 images, the overall optimization takes roughly 15 minutes on a single GTX 1080 GPU. Note that the random seed initialization does not affect our optimization outputs much since the most sensitive parameters (e.g. camera, initial pose, part shapes) are not randomly initialized.

1.1. Notation table

Table 1. **Notations.** For the key variables in Hi-LASSIE, we list the symbol, variable name, state space, and whether the variable is instance-specific or shared within an animal class. Generally, we use i for part indexing and j for instance indexing.

Symbol	Variable name	State space	Instance-specific
$\bar{R}_i$	Resting part rotation (w.r.t. parent joint)	$\mathbb{R}^{3\times3}$	
s_i	Part scaling (w.r.t. bone length)	$\mathbb{R}$	
$\mathcal{F}_i$	Shared part MLP	MLP	
$\pi = (R_0, t_0)$	Camera viewpoint	$(\mathbb{R}^{3\times3}, \mathbb{R}^3)$	✓
R_i	Part rotation (w.r.t. parent joint)	$\mathbb{R}^{3\times3}$	✓
$\mathcal{F}_i^{\Delta}$	Instance part MLP	MLP	✓

1.2. DINO-ViT feature extraction and clustering

Similar to [1, 9, 10], we extract the *key* from the last layer of a pre-trained DINO-ViT [2] network as 2D semantic features. Likewise, we extract the *class* tokens and use their average attention map as a saliency estimation. To perform high-resolution optimization, we crop and resize the input images to 1024×1024 for feature extraction, resulting in a feature/saliency map of size 128×128. We then cluster the features of salient image patches by an off-the-shelf K-means algorithm, using number of clusters $c = 8$. Finally, we obtain a pseudo ground-truth object silhouette $\hat{M}$ by thresholding the minimum distance to cluster centroids and applying dense CRF filtering [5]. Fig. 1 shows some example results of DINO feature clustering.

*Work done as a student researcher at Google.

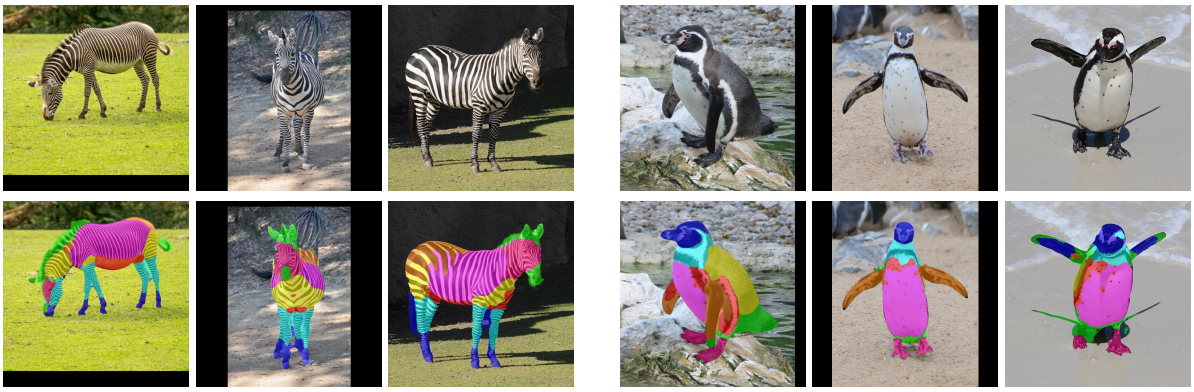

Figure 1. **DINO-ViT feature clusters.** We show the example images (top) and segmentation masks of 8 semantic clusters (bottom). Although the DINO features provide a dense correspondence between images, they are sometimes noisy due to self-occlusions or ambiguous texture (*e.g.*, zebra's head and legs or penguin wings). Instead of directly using these clusters as 3D parts, we find that jointly considering geometric and semantic cues as symmetry information (*e.g.*, left and right legs) is more robust.

1.3. 3D skeleton and camera viewpoint initialization

We further exploit DINO features to roughly initialize camera viewpoints and identify symmetry plane for 3D skeleton estimation. Assuming that most animal images are taken from a small range of elevation and mostly vary in azimuth angles, we propose to roughly cluster them into side and front views by the left-right symmetry information in DINO features. As shown in Fig. 2, we first sort the images based on their left-right symmetry of DINO clusters. Then, we determine whether a given reference image is side or front view to identify the symmetry plane and initialize 3D joints. For instance, the symmetry plane is $x = 0$ if the image has symmetric features (front view) and $z = 0$ if the image features are asymmetric (side view).

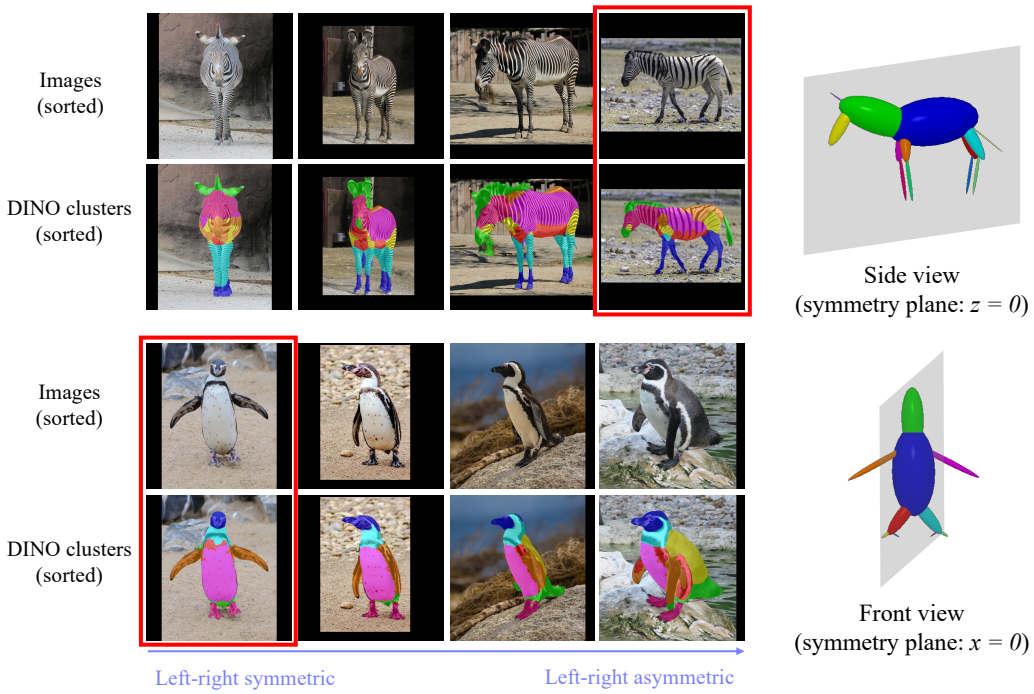

Figure 2. **Exploiting DINO clusters for skeleton and symmetry plane initialization.** Based on the left-right symmetry information in DINO clusters, we determine whether a given reference image (marked in red) is side or front view and initialize the symmetry plane and 3D joints accordingly.

Our skeleton discovery assumes a single reference image where most skeletal parts are visible, which is accessible in all animal image ensembles of our interest. We show additional results using different reference images in Fig. 3, demonstrating that Hi-LASSIE is generic and robust to various initial poses or self-occlusions as long as all the skeletal parts are visible to some extent. We observe that these reference images lead to slightly different part configurations, with minimal affects on final quantitative metrics.

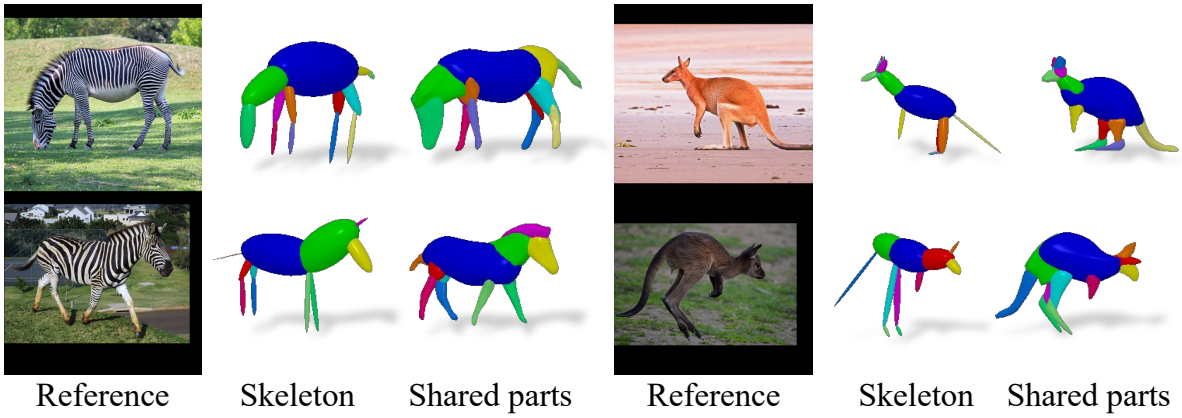

Figure 3. **3D Skeleton and parts discovered from different reference images.**

1.4. Frequency-decomposed neural surfaces

For each neural part surface, we uniformly sample a set of surface points $X = \{(x, y, z)|x^2 + y^2 + z^2 = 1\}$ from a unit sphere and decode their deformation in the canonical space through an MLP network. Unlike LASSIE [10] which uses the MLP network proposed in NeRS [11], we integrate the multiplicative layers similar to BACON [7] into our part MLPs to decompose the surfaces in the frequency space. Specifically, each layer encodes a surface point x via positional encoding (PE) as: $\text{PE}_i(x) = \sin(\omega_i x + \phi_i)$, where $i = 0, ..., L$ (number of layers $L = 9$ in our experiments). The PE frequency is pre-defined as $\omega_i = 0.25\pi \cdot i$, which increases with layer index i. After deforming the shared/instance part surfaces in the canonical space, we then scale, translate, and rotate them by their corresponding bone transformations to obtain an overall articulated shape. Note that we find symmetric surface points at resting pose to symmetrize the final surface deformation and texture. The shape reconstruction and regularization losses are applied on the outputs of all layers. The frequency-decomposed part surfaces are visualized in Fig. 4.

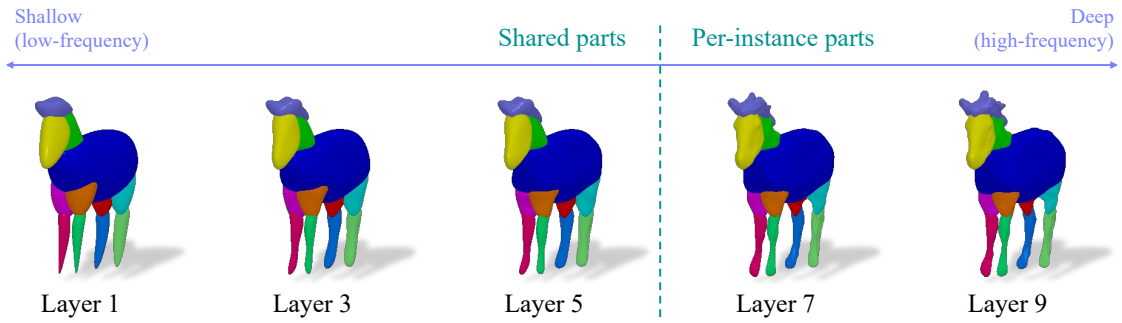

Figure 4. **Frequency decomposition of part surfaces.** We visualize the neural surface outputs of intermediate layers in our part MLPs. The outputs of shallow layers contain the low-frequency base shape, and the deep layers produce high-frequency deformations. When performing instance-specific optimization, we freeze layers 1-6 (shared part MLPs) and only fine-tune the rest (instance part MLPs).

1.5. Optimization pseudo-code

We provide the pseudo-code below to clarify our optimization process. The overall optimization include 5 main stages: 1) camera, 2) pose, 3) shape, 4) all parameters, and 5) per-instance deformation. The parameters are optimized using the corresponding losses in each stage. In each iteration, we first update the surface feature MLPs, then use the updated features to optimize 3D surfaces, forming an EM-style optimization. This EM-style optimization can progressively refine the 3D surface features (semantics) as well as pose and shape fitting (geometry).

Algorithm 1 Hi-LASSIE multi-stage optimization

Parameters: camera viewpoints π^j, part scaling $\{s_i\}$, resting part rotation $\{\bar{R}_i\}$, part rotation $\{R_i\}^j$, shared part MLPs $\{\mathcal{F}_i\}$, instance part MLPs $\{\mathcal{F}_i^{\Delta}\}^j$ (i: part index, j: instance index).
Losses: silhouette loss $\mathcal{L}_{sil}$, part silhouette loss $\mathcal{L}_{part}$, semantic consistency loss $\mathcal{L}_{sem}$, part rotation loss $\mathcal{L}_{rot}$, symmetry loss $\mathcal{L}_{sym}$, Laplacian regularization $\mathcal{L}_{lap}$, surface normal loss $\mathcal{L}_{norm}$.

1: **Stage 1 - camera:** Optimize π^j for all j using $\mathcal{L}_{sem}$ until convergence.
2: **Stage 2 - pose:** Optimize π^j, $\{s_i\}$, $\{\bar{R}_i\}$, $\{R_i\}^j$ for all i, j using $\mathcal{L}_{sem}$, $\mathcal{L}_{rot}$, $\mathcal{L}_{sym}$ until convergence.
3: **Stage 3 - shape:** Optimize $\{\mathcal{F}_i\}$ for all i using $\mathcal{L}_{sil}$, $\mathcal{L}_{part}$, $\mathcal{L}_{sem}$, $\mathcal{L}_{lap}$, $\mathcal{L}_{norm}$ until convergence.
4: **Stage 4 - all:** Optimize all parameters using all losses until convergence.
5: **Stage 5 - instance:** Optimize $\{\mathcal{F}_i^{\Delta}\}^j$ for an instance j using $\mathcal{L}_{sil}$, $\mathcal{L}_{part}$, $\mathcal{L}_{sem}$, $\mathcal{L}_{lap}$, $\mathcal{L}_{norm}$ until convergence.

Algorithm 2 EM-style semantic and geometric optimization

Parameters: features MLPs $\{Q_i\}$.
Losses: semantic consistency loss $\mathcal{L}_{sem}$.

1: **repeat**
2: **E-step**: Update $\{Q_i\}$ by sampling 3D surface points, projecting them onto each image, and averaging the corresponding image features.
3: **M-step**: Optimize neural surfaces using the updated $\{Q_i\}$ in $\mathcal{L}_{sem}$ (Eq. 5 in manuscript). Note that the M-step also involves updating other parameters with different losses depending on the optimization stage.
4: **until** end of optimization stage

1.6. Dataset statistics and code licenses

In our experiments, we make use of the publically available Pascal-part [3] (http://roozbehm.info/pascal-parts/pascal-parts.html) and LASSIE [10] datasets. Each image ensemble contains $n = 30$ images ($n = 16$ for Pascal-Part sheep). Note that the total number of source-target pairs for keypoint or part transfer evaluation is $n \times (n-1)$ since we exhaustively use every image as source or target. For implementation and evaluation, we also utilize the released source code or models of the following methods:

- A-CSM [6]: https://github.com/nileshkulkarni/acsm/blob/master/LICENSE (Apache License 2.0)
- 3D Safari [12]: https://github.com/silviazuffi/smalst/blob/master/LICENSE.txt (MIT-License)
- NeRS [11]: https://github.com/jasonyzhang/ners/blob/main/LICENSE (BSD 3-Clause License)
- DINO-ViT [2]: https://github.com/facebookresearch/dino/blob/main/LICENSE (Apache License 2.0)
- DINO clustering [1]: https://github.com/ShirAmir/dino-vit-features/blob/main/LICENSE (MIT-License)

2. Ablative Studies

2.1. 3D skeleton: user annotation v.s. automatic discovery

In Fig. 5, we compare our self-discovered 3D skeleton of zebra against the manually-annotated skeleton in LASSIE [10]. Note that both skeleton share similar bone structures, and yet our skeleton provides better initialization of resting pose, simple part shapes (length and thickness), and 3D surface features since it is estimated from a reference image in the ensemble.

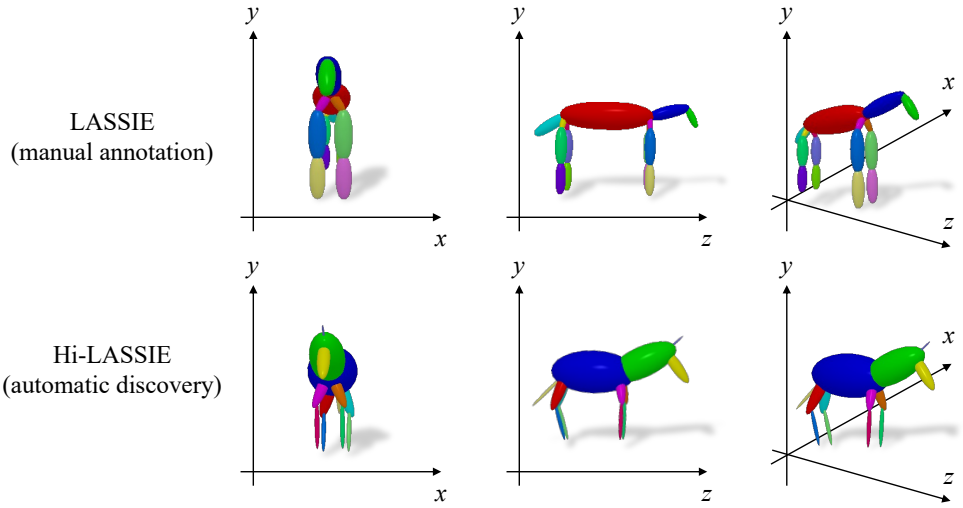

Figure 5. **Visual comparisons of manually annotated and automatically discovered 3D skeletons.** Our skeleton provides a good class-specific initialization of 3D pose and shape, while LASSIE [10] skeleton contains straight leg bones and uniform part shapes.

2.2. Shared v.s. per-instance parts

We show the qualitative comparisons between shared parts and per-instance parts in Fig. 6, which demonstrate that our frequency-based decomposition can effectively preserve the 3D part priors learned from all instance while allowing instance-specific deformation.

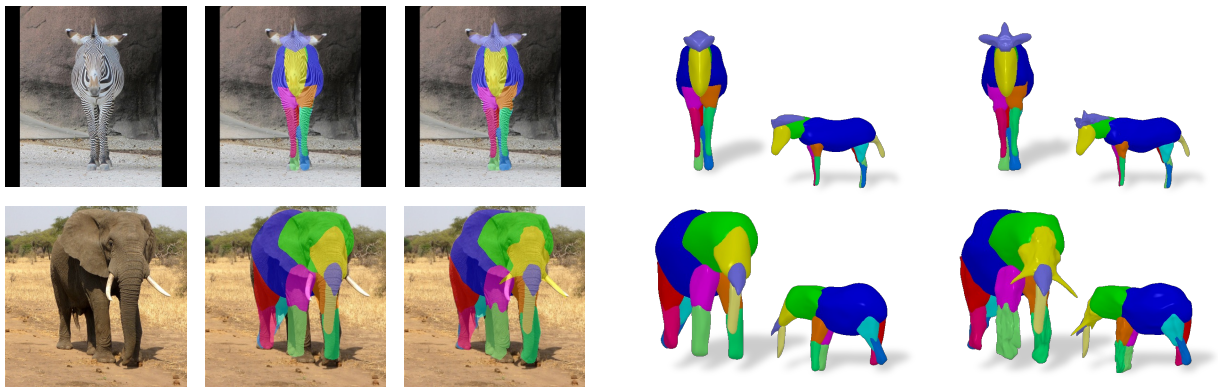

Figure 6. **Visual comparisons of shared and instance parts.** Our instance-specific optimization can produce high-fidelity 3D details to fit the input images for faithfully.

3. Additional Results

3.1. Animation via pose interpolation

Our skeleton-based representation enables part manipulation like texture transfer or pose transfer between instances. In Fig. 7, we show some examples of realistic animation generated by pose interpolation between two instances.

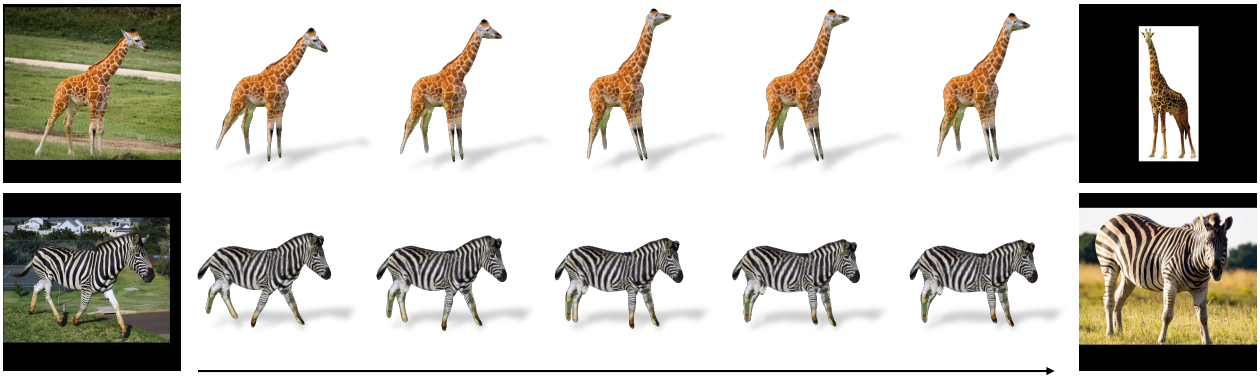

Figure 7. **Animation via pose interpolation.** We show the gradual transformation from source to target poses.

3.2. 2D comparisons on LASSIE images

To demonstrate that Hi-LASSIE can produce faithful and detailed 2D parts, we qualitatively compare the 2D fitting results on LASSIE [10] images in Fig. 8.

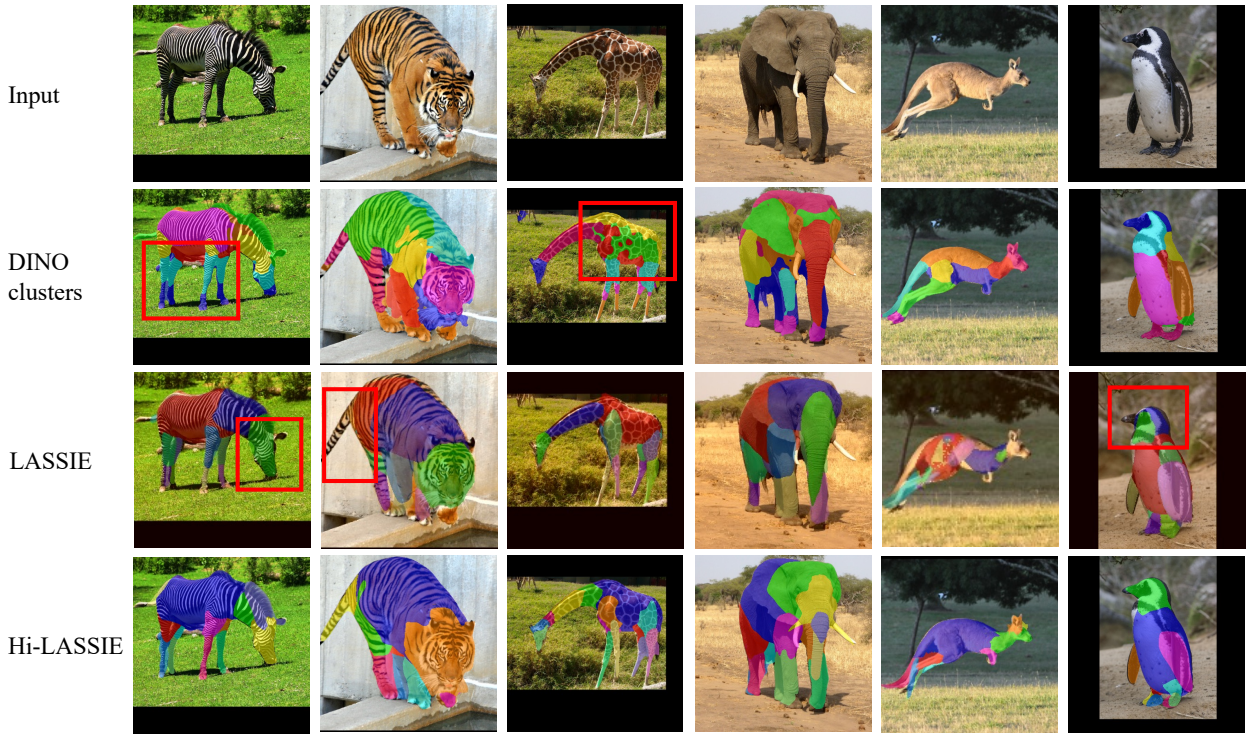

Figure 8. **2D visual results on LASSIE [10] images.** Unlike DINO clusters, Hi-LASSIE can clearly separate the semantically similar parts (*e.g.*, 4 legs) and is more robust to noisy features. Moreover, our results better align with the object boundaries compared to LASSIE [10].

3.3. 3D comparisons on Pascal-Part images

In Fig. 9, we show some qualitative results on the horse and cow images from Pascal-Part dataset [3]. Compared with A-CSM [6] and LASSIE [10], our results are more detailed and faithful to the input images while requiring less user input.

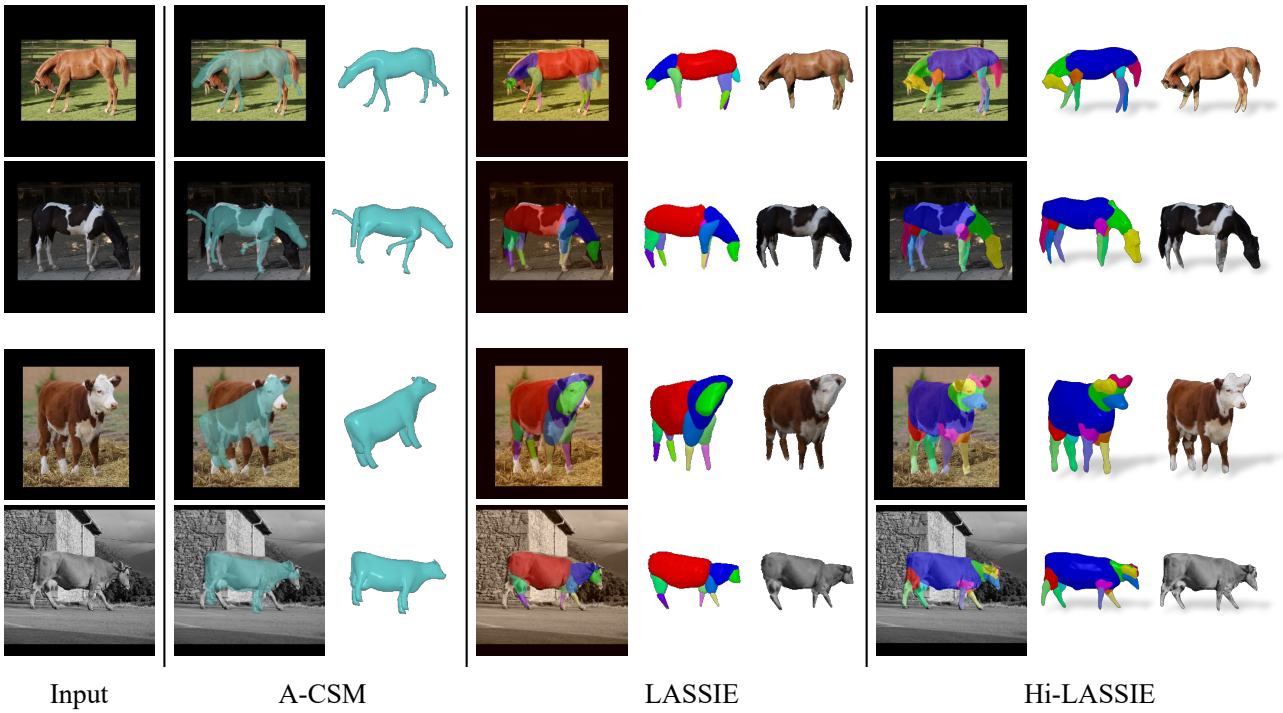

Figure 9. **Visual comparisons on Pascal-Part [3] images.** A-CSM [6] can produce detailed 3D shapes by leveraging a mesh template and large-scale training images. However, the outputs often contain irregular poses or mis-align with the images. LASSIE [10] generates consistent 3D parts from the given skeleton, but misses the instance-specific shape details. By comparison, Hi-LASSIE results are high-quality and faithful to the images.

3.4. Failure cases

Finally, we show some failure cases in Fig. 10 and 11. Since Hi-LASSIE only uses the self-supervisory DINO features as image-level supervision, its performance is sensitive to incomplete (occlusion/truncation) or noisy (ambiguous appearance) features. Another limiting factor is the articulation diversity within the given image ensemble. For instance, if an image ensemble contains only few images or little camera/pose variations, it may be challenging to 1) choose a good reference image for skeleton discovery and 2) learn detailed 3D shapes from multiple views.

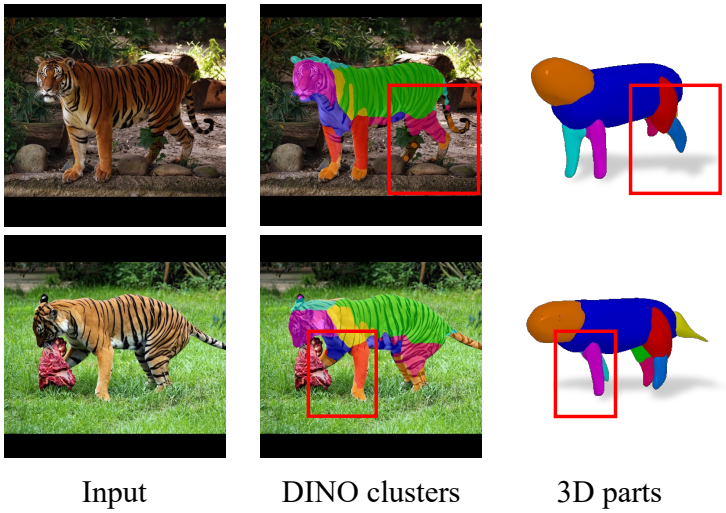

Figure 10. **Failure cases caused by occlusions/truncation.** Our 3D outputs tend to be inaccurate when the pseudo ground-truth silhouettes are incomplete due to occlusions or truncation.

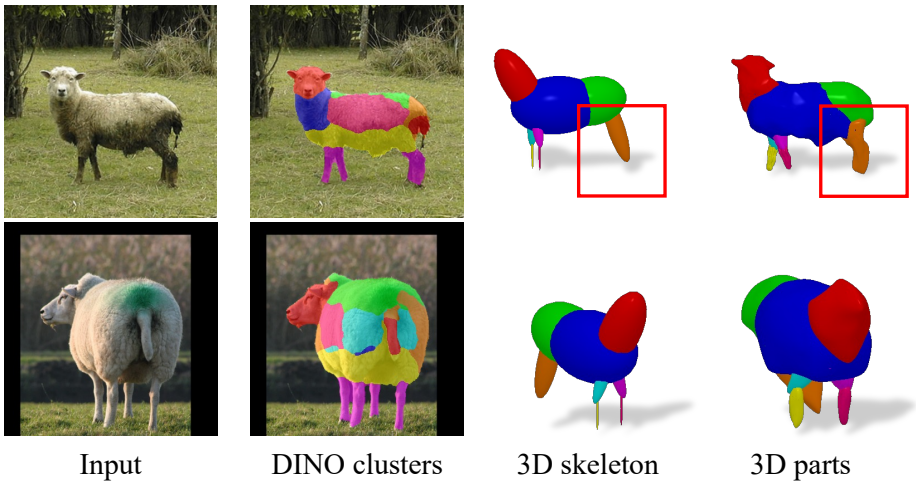

Figure 11. **Failure cases caused by limited image ensemble or ambiguous camera viewpoint.** The sheep image ensemble in our experiments contains 16 images with limited articulation variation, making it difficult to discover a good reference skeleton or learn detailed 3D shapes. As a result, Hi-LASSIE produces undesired 3D skeleton and inaccurate camera viewpoint estimations.

References

[1] Shir Amir, Yossi Gandelsman, Shai Bagon, and Tali Dekel. Deep ViT features as dense visual descriptors. *arXiv preprint arXiv:2112.05814*, 2021. 1, 4

[2] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, pages 9650–9660, 2021. 1, 4

[3] Xianjie Chen, Roozbeh Mottaghi, Xiaobai Liu, Sanja Fidler, Raquel Urtasun, and Alan Yuille. Detect what you can: Detecting and representing objects using holistic models and body parts. In *CVPR*, pages 1971–1978, 2014. 4, 7

[4] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 1

[5] Philipp Krähenbühl and Vladlen Koltun. Efficient inference in fully connected CRFs with Gaussian edge potentials. *NeurIPS*, 24, 2011. 1

[6] Nilesh Kulkarni, Abhinav Gupta, David F Fouhey, and Shubham Tulsiani. Articulation-aware canonical surface mapping. In *CVPR*, pages 452–461, 2020. 4, 7

[7] David B Lindell, Dave Van Veen, Jeong Joon Park, and Gordon Wetzstein. Bacon: Band-limited coordinate networks for multiscale scene representation. In *CVPR*, pages 16252–16262, 2022. 3

[8] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. PyTorch: An imperative style, high-performance deep learning library. In *NeurIPS*, pages 8024–8035, 2019. 1

[9] Narek Tumanyan, Omer Bar-Tal, Shai Bagon, and Tali Dekel. Splicing ViT features for semantic appearance transfer. *arXiv preprint arXiv:2201.00424*, 2022. 1

[10] 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. *arXiv preprint arXiv:2207.03434*, 2022. 1, 3, 4, 5, 6, 7

[11] Jason Zhang, Gengshan Yang, Shubham Tulsiani, and Deva Ramanan. NeRS: Neural reflectance surfaces for sparse-view 3D reconstruction in the wild. *NeurIPS*, 34, 2021. 3, 4

[12] Silvia Zuffi, Angjoo Kanazawa, Tanya Berger-Wolf, and Michael J Black. Three-D Safari: Learning to estimate zebra pose, shape, and texture from images "in the wild". In *ICCV*, pages 5359–5368, 2019. 4