Animator-Centric Skeleton Generation on Objects with Fine-Grained Details

Supplementary Material

In this supplementary material, we provide a detailed introduction to the construction of our proposed dataset, including the preparation of the rigging ground truth of the mesh in Sec. 1. In Sec. 2, we elaborate on the semantic understanding model's details, covering the design of the network and output postprocessing. Finally, in Sec. 3, we include additional experimental results.

1. Dataset Construction Details

Starting from over 150K rigged assets, we construct a metadata table for each instance, including per-skeleton statistics such as the legality of leaf-joint bounding boxes (`is_leaf_joint_bbox_legal`), the number of connected joint trees (`fixed_joint_tree_num`), the number of valid joints after cleaning (`fixed_joint_num`), the total number of skeleton joints (`skeleton_num`), an AABB overlap score between skeleton and mesh (`aabb_box_iou`), the rig category (`skinning_category_en`), whether additional auxiliary bones are present (`contain_additional_joints`), and optional manual topology-quality labels (`topo_level`). Based on this metadata, we apply a two-stage filtering strategy. If no manual labels are available, we keep only samples that satisfy: (i) leaf-joint bounding boxes are geometrically valid (`is_leaf_joint_bbox_legal` = `True`); (ii) the skeleton is a single connected tree (`fixed_joint_tree_num` = 1); (iii) the cleaned joint count exceeds a minimum complexity threshold (`fixed_joint_num` $>$ 5); (iv) the total number of joints remains within a reasonable range (`skeleton_num` $\leq$ 400); (v) the skeleton-to-mesh alignment is sufficiently good (`aabb_box_iou` $>$ 0.2); (vi) the asset is not a partial human fragment only (`skinning_category_en` $\neq$ "human part"); and (vii) a valid file path or remote URL is available. When manual topology-quality labels are present, we first select all models annotated as high-quality (`topo_level` = "high") that also satisfy the tree and joint-count constraints above, and then complement them with unlabeled models that pass the same rule-based filter. The union of these two subsets forms the final pool of valid skeleton–mesh pairs used by our experiments.

On top of this cleaned set, we build balanced train/test partitions with a category-aware sampler: for training, we always keep all non-humanoid assets and humanoid assets containing additional auxiliary joints, and then randomly subsample the remaining simpler humanoid rigs at a matched ratio. The number of vertices or joints can optionally sort the resulting indices to improve load balancing across workers. For testing, we use the same filtering rules but disable resampling to ensure that each valid instance appears exactly once in the evaluation split. Finally, we obtain our high-quality rigging dataset, consisting of $81,142$ training samples and $1,491$ test samples.

2. Semantic Understanding Model Details

Joint semantic prediction. Fig. 1 shows the whole pipeline of our semantic understanding model. Given a skeleton with K joints, we denote joint positions by $\mathbf{J} \in \mathbb{R}^{K\times 3}$ and the kinematic tree by a parent index array $\mathbf{p} \in \{-1, 0, \ldots, K-1\}^K$, where $\mathbf{p}_i = -1$ indicates the root joint. For human characters we use a joint-only semantic model f_θ^{human}, whereas for animals we use a joint–mesh model f_θ^{animal} that additionally takes vertex positions $\mathbf{V} \in \mathbb{R}^{N\times 3}$ as input. From the parent array we construct a skeleton adjacency matrix $\mathbf{A} \in \{0,1\}^{K\times K}$ by connecting each joint to its parent (and optionally symmetrizing and adding self-connections). The human semantic predictor then computes per-joint logits

$$\mathbf{L}^{\text{human}} = f_\theta^{\text{human}}\big(\mathbf{J}, \mathbf{A}, \mathbf{p}\big) \in \mathbb{R}^{K\times C_{\text{H}}}, \tag{1}$$

where C_{H} is the number of human joint semantic classes. Similarly, the animal semantic predictor takes joint positions, adjacency and mesh vertices as input,

$$\mathbf{L}^{\text{animal}} = f_\theta^{\text{animal}}\big(\mathbf{J}, \mathbf{A}, \mathbf{V}\big) \in \mathbb{R}^{K\times C_{\text{A}}}, \tag{2}$$

with C_{A} animal semantic classes. In both cases, we obtain class probabilities via a softmax over the semantic dimension,

$$\mathbf{P}_{i,c} = \frac{\exp(\mathbf{L}_{i,c})}{\sum_{c'=1}^{C}\exp(\mathbf{L}_{i,c'})}, \quad i = 1, \ldots, K, \tag{3}$$

and derive an initial semantic label per joint by $\hat{y}_i = \arg\max_c \mathbf{P}_{i,c}$. Depending on the mode, we either directly use this maximum-likelihood prediction ("`ORG_NAME`"), or feed $\mathbf{P}$ into our rule-based optimization pipeline ("`OPTIMIZED_NAME`") detailed below to obtain refined labels $\tilde{y}_i$. Finally, we optionally reorder joints according to a canonical semantic hierarchy $\mathcal{H}$ (e.g., pelvis $\rightarrow$ spine $\rightarrow$ head, shoulder $\rightarrow$ upper arm $\rightarrow$ lower arm $\rightarrow$ hand), by performing a depth-first traversal of the kinematic tree guided by $\tilde{y}_i$. This produces a semantically ordered skeleton $(\tilde{\mathbf{J}}, \tilde{\mathbf{p}}, \tilde{\mathbf{y}})$ and enables deterministic naming of each joint via a lookup table $\mathcal{M}$ (e.g., pelvis, spine_01, thigh_L, shoulder_R), augmented with numerical suffixes for semantics that are allowed to appear multiple times.

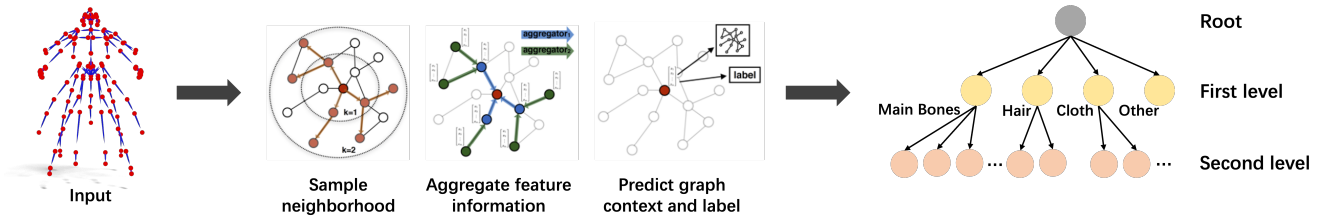

Figure 1. Our semantic understanding model. Given a skeleton as input, a GraphTransformer predicts a multi-level semantic label for each joint.

Semantic label post-processing. To obtain robust and anatomically consistent joint semantics, we apply a multi-stage post-processing pipeline to the raw logits $\mathbf{P}$ and initial labels $\hat{\mathbf{y}}$. We first exploit the fact that some semantic types (e.g., pelvis, spine root, head) should appear at most once, while others (e.g., ribs, tail segments, fingers) may appear multiple times. Let $\mathcal{C} = \{0, \dots, C-1\}$ denote all semantic classes, and $\mathcal{C}_{\text{multi}} \subset \mathcal{C}$ the set of classes allowed to repeat (given by `JointNumDict` for humans and animals). We define the unique set $\mathcal{C}_{\text{uniq}} = \mathcal{C} \setminus \mathcal{C}_{\text{multi}}$ and enforce that each $c \in \mathcal{C}_{\text{uniq}}$ is assigned to at most one joint by keeping, for each such class, only the joint with highest probability $\mathbf{P}_{i,c}$ and marking all other occurrences as invalid (-1). For all invalid joints, we then reassign a label by selecting the most probable class among $\mathcal{C}_{\text{multi}}$. On top of this generic assignment, we introduce several anatomy-aware refinements. For humans, we explicitly enforce a root semantic at the kinematic root, correct pelvis semantics by identifying the joint whose children include spine and both thighs, and apply prior chains for arms and legs (e.g., hand $\rightarrow$ lower arm $\rightarrow$ upper arm $\rightarrow$ clavicle, toe $\rightarrow$ foot $\rightarrow$ lower leg $\rightarrow$ upper leg) via a function process_priori_semantic_chain that replaces inconsistent chain labels with a target sequence of semantics. Finger semantics are refined by first extracting bone-chain paths from the kinematic tree, grouping hand-side chains attached to the left or right hand, and then assigning finger types (thumb to little finger) based on chain length and spatial arrangement. We further enforce intra-chain and inter-chain consistency for human additional bones by (i) making all joints on a given auxiliary chain share the same semantic label and (ii) encouraging chains that share the same parent to have consistent labels when they belong to a predefined set of auxiliary categories. For animals, we apply an analogous process with species-agnostic priors: we first obtain a common prediction using the unique/multi-label scheme, then enforce a root semantic label, and correct primary limb chains for forelegs and hind legs with pre-specified semantic sequences. After these refinements, we run a final unique-label correction to guarantee that all classes in $\mathcal{C}_{\text{uniq}}$ appear at most once in the skeleton. If the "`OPTIMIZED_NAME`" mode is selected, the resulting labels $\tilde{\mathbf{y}}$ are then used to reorder joints according to the standard semantic hierarchy $\mathcal{H}$ via recoder_joints_by_semantic_order$(\mathbf{J}, \mathbf{p}, \tilde{\mathbf{y}})$, and joint names are instantiated by mapping each semantic label to a canonical name and, for multi-occurring labels, appending an index-based suffix (e.g., `tail_01`, `tail_02`). This post-processing ensures that the final semantics are structurally consistent, anatomically plausible, and canonically ordered across heterogeneous skeletons.

3. Experiments

3.1. Quantitative Results

We retrain two relevant baselines [5] and [6], on our dataset, and report the full results in Tab. 1. Across different evaluation metrics, our method consistently and significantly outperforms all baselines. Across all quantitative metrics — Precision, Recall, Accuracy, F1-Score, and the three Chamfer-Distance–based measures (CD-J2J, CD-J2B, CD-B2B) — our method achieves consistently superior performance compared to the retrained baselines [4–6]. Under a strict spatial threshold of $\tau = 0.01$, our model not only yields more accurate joint predictions but also exhibits substantially better geometric alignment at both the joint and bone levels. These results collectively demonstrate that our method captures fine-grained skeletal structures more reliably than existing approaches.

3.2. Additional Qualitative Results

In this section, we present additional visual results, including comparisons with baselines, demonstrations of density control, zero-shot generalization of our method on the ArticulationXL [4] dataset, further generalization across diverse character types on real-human and AnimeRig [5] datasets, as well as results from more complex editing tasks.

3.2.1. Additional Comparison Results

Fig. 2 provides qualitative comparisons on our dataset, offering further visual support for the findings discussed in the main paper. Across a wide variety of object categories and structural configurations, our method consistently generates clean, well-organized, and semantically meaningful

	Mode	Precision ↑	Recall ↑	Accuracy ↑	F1_Score ↑	J2J ↓	J2B ↓	B2B ↓
UniRig [7]	wo.train	0.105	0.066	0.078	0.077	0.038	0.031	0.026
Puppeteer [3]		0.168	0.086	0.106	0.105	0.046	0.038	0.033
MagicArticulate [4]	retrain	0.712	0.701	0.697	0.707	0.044	0.034	0.032
DRiVE [5]		0.596	0.633	0.592	0.593	0.053	0.042	0.046
ARMO [6]		0.718	0.708	0.704	0.713	0.043	0.039	0.036
Ours		**0.745**	**0.731**	**0.729**	**0.730**	**0.036**	**0.027**	**0.025**

Table 1. Joint prediction results on the test set.

skeletons. In contrast to baseline approaches, which often produce incomplete, noisy, or structurally inconsistent bone layouts, our results exhibit strong robustness, clearer hierarchical organization, and better alignment with the underlying geometry. These observations reinforce the advantages of our approach in handling diverse and complex shapes.

3.2.2. More Controllable Generation Results

Density control: Fig. 3 demonstrates that our model supports explicit and fine-grained control over the density of the generated skeleton. As the target bone count increases from low to high levels, the model not only adds new joints in a structurally coherent manner but also prioritizes generating auxiliary bones—such as hair and cloth. This behavior closely mirrors practical rigging workflows, where higher-density skeletons are typically achieved by enriching auxiliary structures rather than altering the primary kinematic chain. The results highlight the model's ability to scale structural complexity while maintaining anatomical plausibility and functional consistency.

Main bones editing: To thoroughly demonstrate the controllability of our main bones conditional generation, we perform a data augmentation procedure: starting from the originally extracted main bones, we randomly insert two additional joints along existing bone segments to construct a modified main bones input. This augmented skeleton is then used as a conditioning signal to evaluate the model's ability to generate auxiliary bones. As shown in Fig. 4, our method faithfully preserves the structure of the provided main bones while producing plausible and well-organized auxiliary bones.

We further compute the bone-to-bone chamfer distance between the input main bones and the generated main bones, achieving an average value below 10^{-4}. This extremely small discrepancy verifies that our model can strictly adhere to the main bones constraint. Such controllability is highly practical for animators, enabling them to define a coarse template and automatically obtain high-quality auxiliary bones on top of it.

3.2.3. Zero-Shot Results on ArticulationXL

To evaluate the generalization capability of our model, we directly perform zero-shot inference on the open-source dataset ArticulationXL [4], using the model trained solely on our dataset. Results across different categories are presented in Fig. 5 and Fig. 6. It is important to note that the ground-truth skeletons in the dataset ArticulationXL, which are sampled from ObjaverseXL [1], are not always highly accurate. As shown in the first, second, and fourth rows of Fig. 5 and the first, second, and sixth rows of Fig. 6, our method generates skeletons with substantially richer details—such as skirt structures and tails—and provides more precise joint localization around areas like the knees. Due to the imperfect quality of the ground truth, we report qualitative comparisons only rather than quantitative metrics. We hope that both our dataset and method can contribute meaningfully to the advancement of this research community.

3.2.4. Real-Human and AnimeRig Results

To further demonstrate the generalization capability of our method, we additionally evaluate it on a collection of real-human meshes curated following the procedure of [2]. To ensure a fair comparison, we retrain MagicArticulate on our dataset and use it as the primary baseline. Compared with the results produced by MagicArticulate shown in Fig. 7, our method generates skeletons with noticeably higher accuracy—particularly around critical articulation regions such as elbows and knees. Moreover, our approach can reliably produce finger-level bones that adapt to different poses, further demonstrating its superior semantic understanding and structural precision.

To further evaluate the generalization ability of our model, we additionally test on cases from the AnimeRig dataset [5] that contain challenging structures such as skirts, hair, and other fine-grained appendages. We again compare against the strongest baseline — MagicArticulate retrained on our dataset — and present the results in Fig. 8. As shown, our method generates auxiliary bones that better correspond to skirt folds, hair strands, and other geometric details, whereas the baseline either misses these structures or produces inaccurate joint placements. These results demonstrate the robustness of our approach across diverse input shapes and complex geometric variations.

References

[1] Matt Deitke, Ruoshi Liu, Matthew Wallingford, Huong Ngo, Oscar Michel, Aditya Kusupati, Alan Fan, Christian Laforte, Vikram Voleti, Samir Yitzhak Gadre, et al. Objaverse-xl: A universe of 10m+ 3d objects. *Advances in Neural Information Processing Systems*, 36:35799–35813, 2023. 3

[2] Hsuan-I Ho, Lixin Xue, Jie Song, and Otmar Hilliges. Learning locally editable virtual humans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21024–21035, 2023. 3

[3] Chaoyue Song, Xiu Li, Fan Yang, Zhongcong Xu, Jiacheng Wei, Fayao Liu, Jiashi Feng, Guosheng Lin, and Jianfeng Zhang. Puppeteer: Rig and animate your 3d models. *arXiv preprint arXiv:2508.10898*, 2025. 3

[4] Chaoyue Song, Jianfeng Zhang, Xiu Li, Fan Yang, Yiwen Chen, Zhongcong Xu, Jun Hao Liew, Xiaoyang Guo, Fayao Liu, Jiashi Feng, et al. Magicarticulate: Make your 3d models articulation-ready. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 15998–16007, 2025. 2, 3

[5] Mingze Sun, Junhao Chen, Junting Dong, Yurun Chen, Xinyu Jiang, Shiwei Mao, Puhua Jiang, Jingbo Wang, Bo Dai, and Ruqi Huang. Drive: Diffusion-based rigging empowers generation of versatile and expressive characters. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 21170–21180, 2025. 2, 3

[6] Mingze Sun, Shiwei Mao, Keyi Chen, Yurun Chen, Shunlin Lu, Jingbo Wang, Junting Dong, and Ruqi Huang. Armo: Autoregressive rigging for multi-category objects. *arXiv preprint arXiv:2503.20663*, 2025. 2, 3

[7] Jia-Peng Zhang, Cheng-Feng Pu, Meng-Hao Guo, Yan-Pei Cao, and Shi-Min Hu. One model to rig them all: Diverse skeleton rigging with unirig. *ACM Transactions on Graphics (TOG)*, 44(4):1–18, 2025. 3

Figure 2. Comparison of skeleton generation results on our test set, and $*$ indicates the method is directly inferred with a publicly available checkpoint.

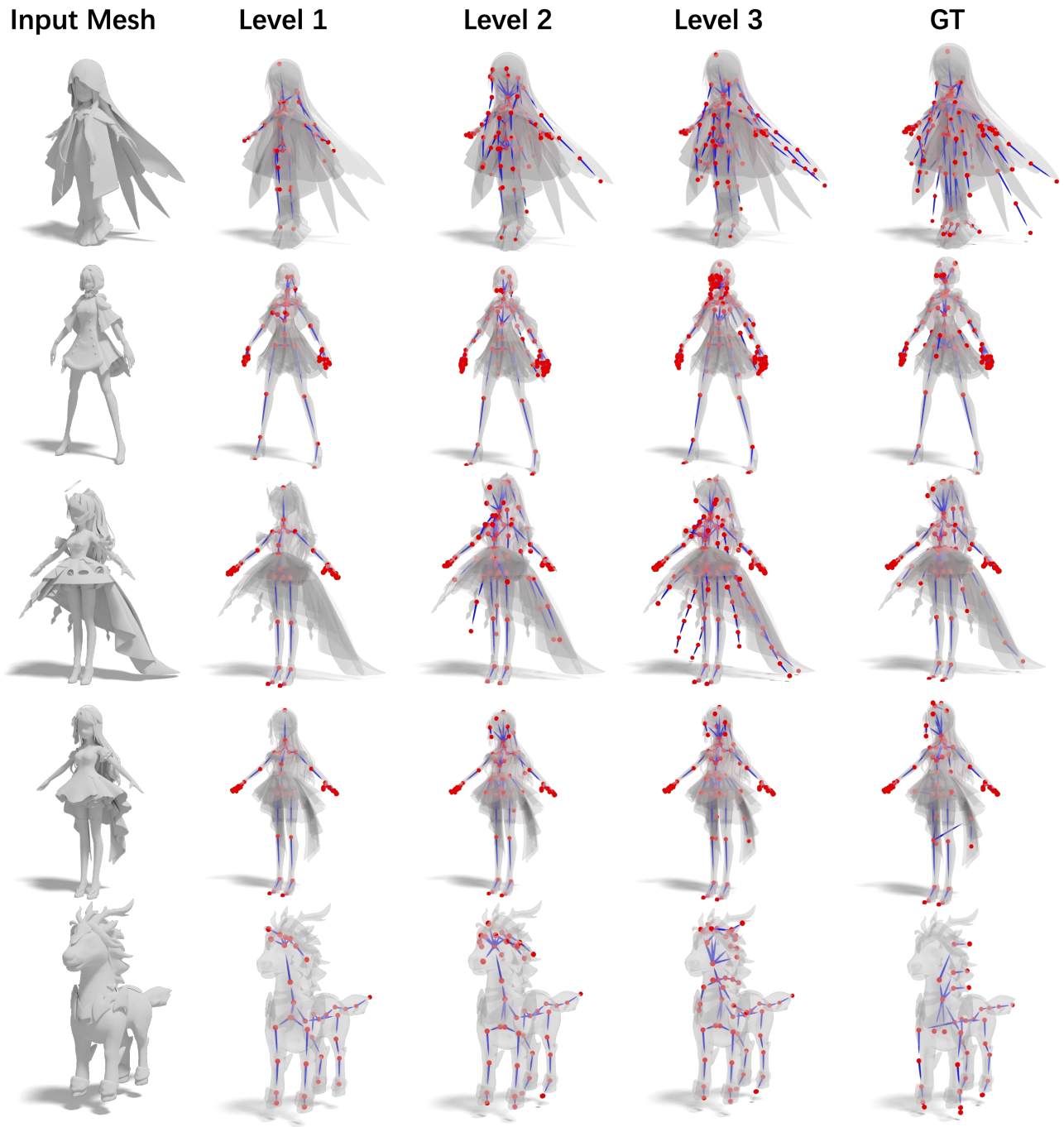

Figure 3. Density control results. From Level 1 to Level 3, our method progressively increases the number of generated bones while maintaining structural plausibility. The model not only produces different bone densities on demand, but also naturally adds auxiliary bones as the bone count increases, resulting in more detailed and expressive rig structures.

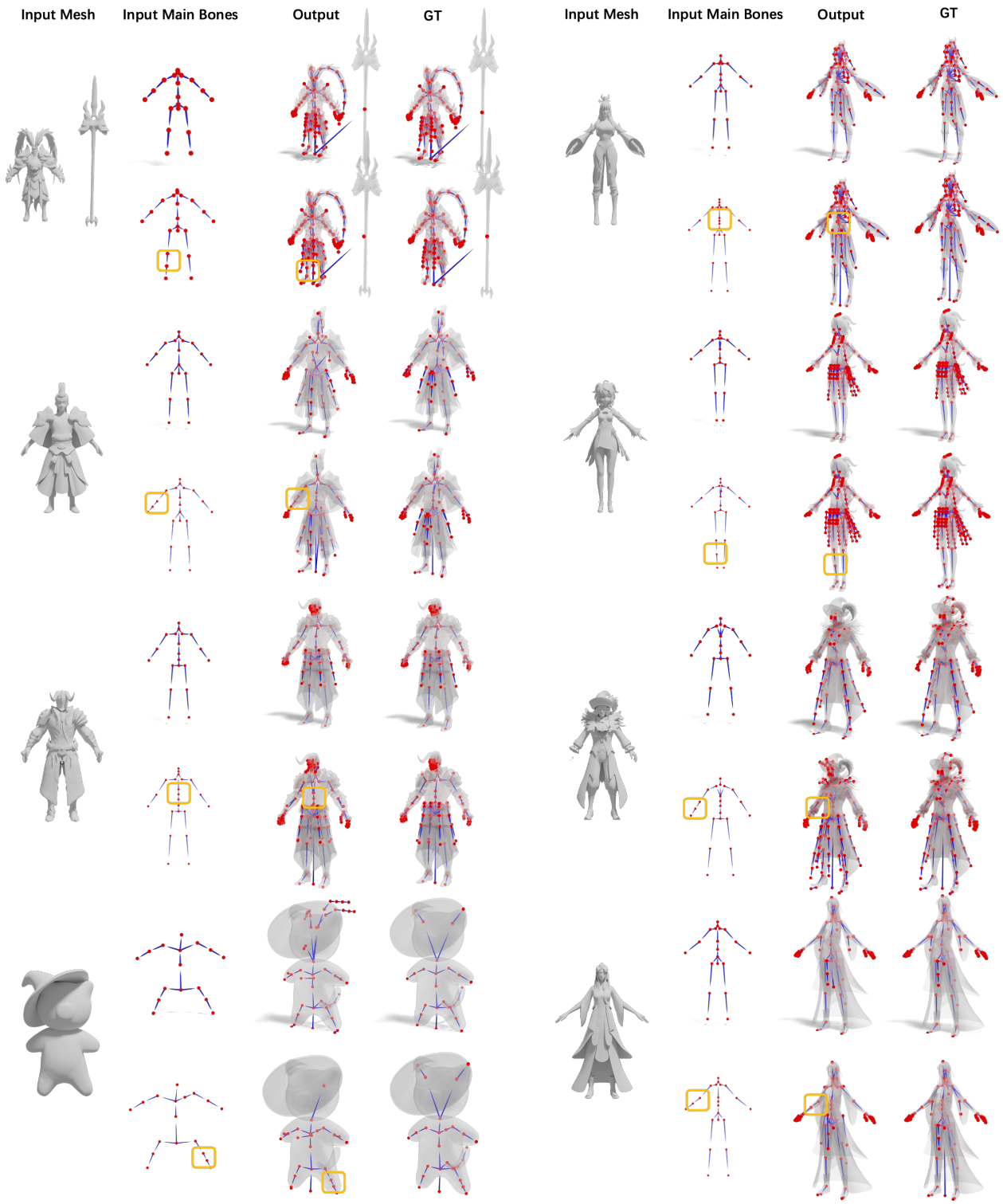

Figure 4. For the same mesh input, our model can generate high-quality auxiliary bones conditioned on different user-specified main bones inputs, while preserving the main bones themselves. Regions where the main bones changes are highlighted with yellow boxes.

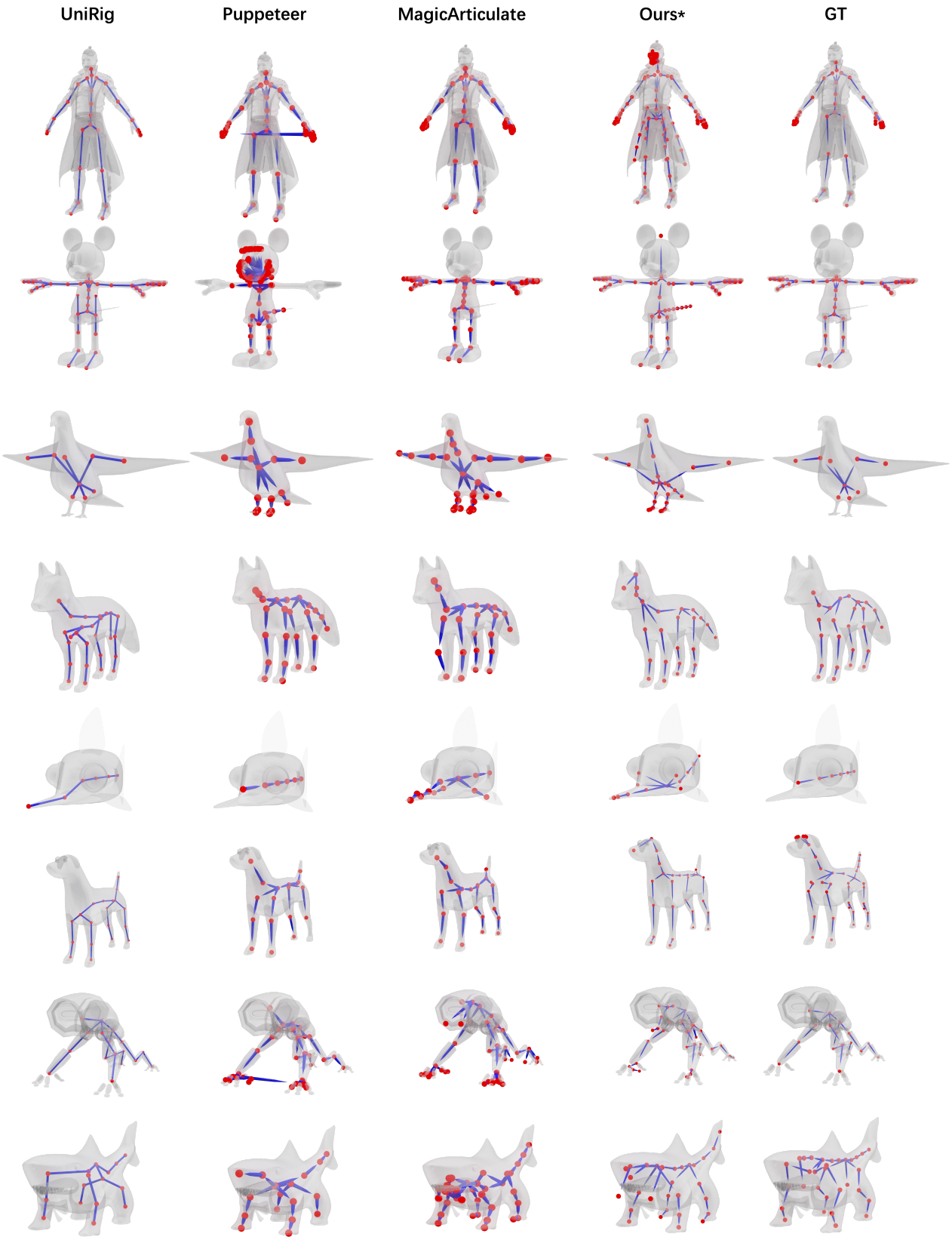

Figure 5. Comparison of skeleton generation results on ArticulationXL, and $*$ indicates our method is trained on our dataset.

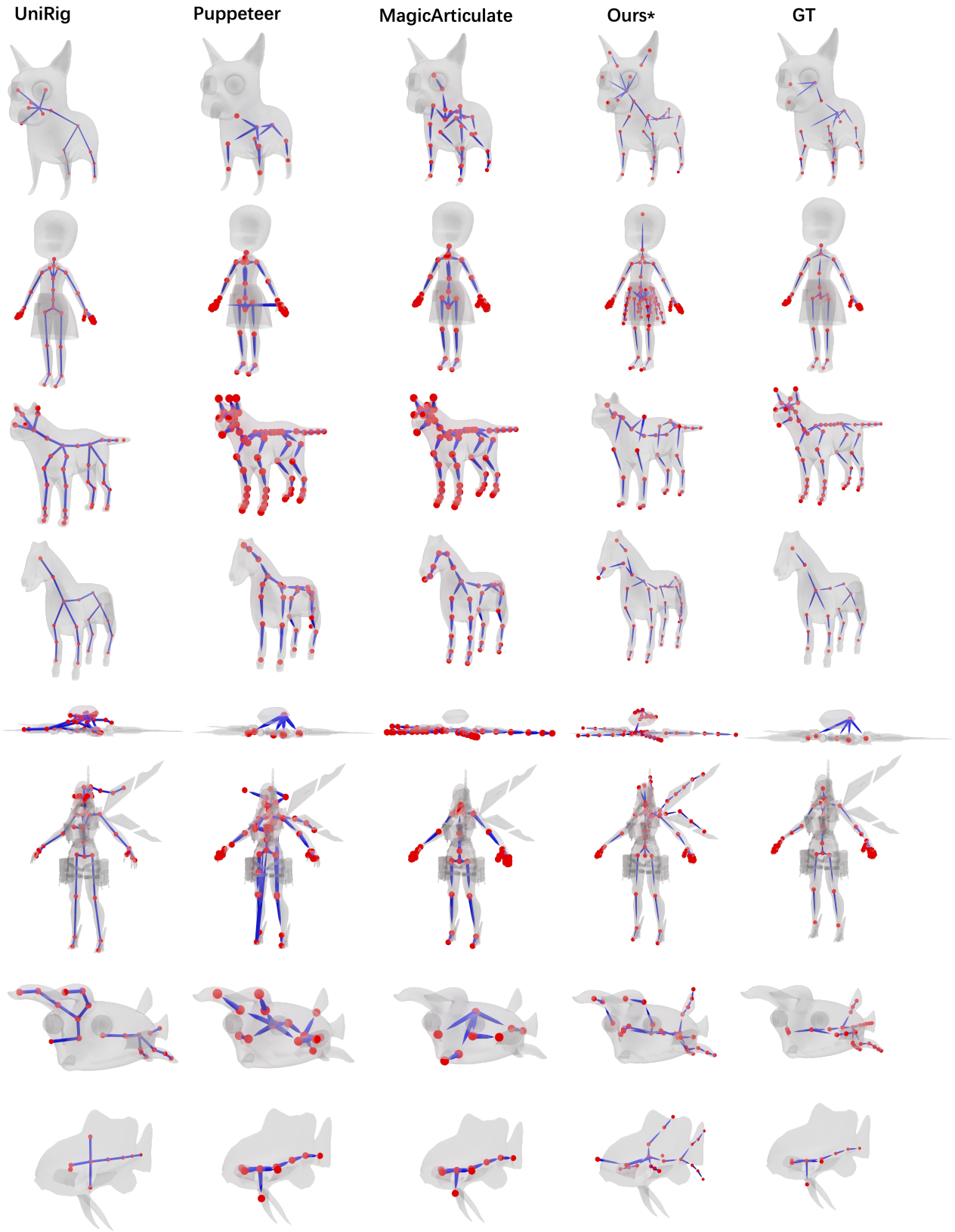

Figure 6. More skeleton generation results on ArticulationXL, and $*$ indicates our method is trained on our dataset.

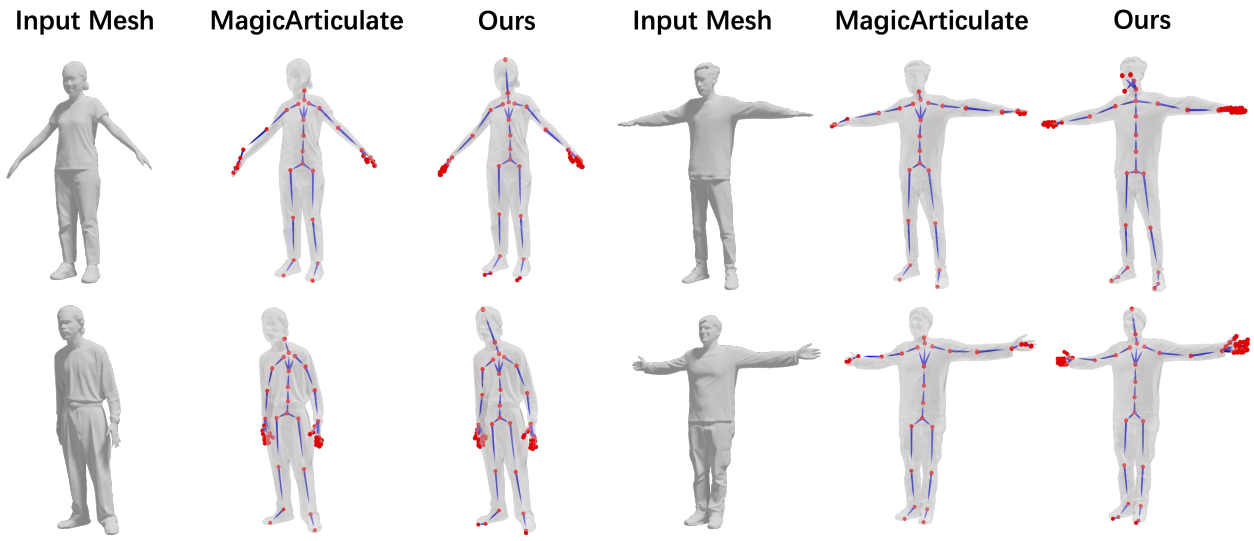

Figure 7. Comparison of skeleton generation results on Real-Human data.

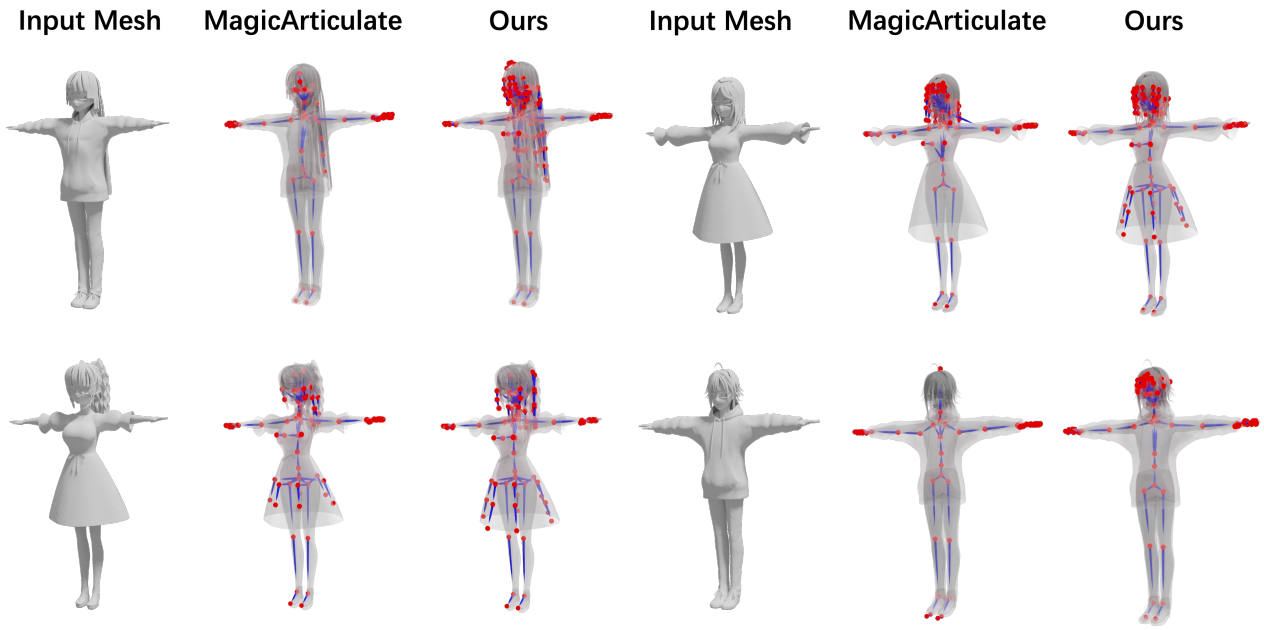

Figure 8. Comparison of skeleton generation results on AnimeRig.