ARTICRITIC: A VISION–LANGUAGE CRITIC FOR GENERATED ARTICULATED OBJECTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Despite recent advances in articulated object generation, systematic evaluation of generated assets remains limited. We introduce **ArtiCritic**, a training-free, vision-language model (VLM)-assisted framework that combines three-stage inspection with evidence-guided adjudication. To address the limitations of holistic VLM judgments in visual localization and fine-grained reasoning, ArtiCritic decomposes evaluation into localized checks of intrinsic validity, part completeness, and articulation plausibility. To accommodate differences in modeling granularity, we propose **cross-granularity structural matching (CGSM)**, which localizes reference part instances in candidate renderings, associates them with URDF links, and assesses reference-supported fixed attachments and independent motion requirements. Each check produces defect candidates with supporting evidence and confidence estimates. A final VLM adjudicator reassesses these candidates, rejects unsupported claims, and consolidates confirmed defects into an overall verdict with repair suggestions. Experiments on manually annotated articulated assets show that ArtiCritic can identify defective assets and provide targeted diagnostic feedback.

1 INTRODUCTION

Recent progress in articulated-object generation has simplified the creation of interactive 3D assets. As generation scales up, systematic automatic inspection becomes essential to verify whether these assets contain all required components and support plausible motion.

Current approaches to articulated-object generation broadly fall into code-based generation and geometry-based modeling. Code-based methods rely on execution checks, geometric tests, and visual critics, yet they may overlook missing functional components or faulty motion artifacts outside the scope of predefined tests and visual evidence Le et al. (2025). Geometry-based modeling is typically evaluated using geometric similarity, articulation errors, and physical consistency metrics; these metrics alone cannot deliver a complete diagnosis of part-level and motion-related requirements for individual assets Lei et al. (2023). Human curation remains essential in practice: Articraft Zhou et al. (2026) filters generated assets via manual ratings, whereas Artiverse adopts human model selection together with multi-stage verification of part and motion annotations Iliash et al. (2026). These limitations motivate a unified automatic inspection workflow for evaluating individual generated assets.

To this end, we present **ArtiCritic**, which takes reference images, optional category text, and a candidate asset in Unified Robot Description Format, and returns a pass/fail verdict, supporting evidence, and repair suggestions. Our starting point is that reliable inspection requires explicit coverage and localized evidence. We organize the workflow around *intrinsic validity*, *part completeness*, and *articulation plausibility*. Each stage proposes issues and records their locations, evidence, and confidence. A final vision-language model (VLM) reviews these candidates in the context of the whole object. This structure directs VLM reasoning toward specific observations, while part and joint inventories organize which requirements must be checked.

Two challenges arise in this workflow. First, inspection requires detailed reference part and joint inventories. We combine Semantic-SAM masks Li et al. (2023) with PartGLEE and VLPart proposals Li et al. (2024); Sun et al. (2023): masks provide boundaries, proposals guide labeling, and

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.



Figure 1: **Validating generated articulated objects.** From left to right, we illustrate a missing cabinet handle, a locked lamp-head joint, and a toilet lid rotating about a misplaced front hinge.

the VLM assigns labels and fuses fragments while preserving distinct part instances. For articulation, OPD Jiang et al. (2022) supplies motion proposals that a VLM combines with the part inventory to infer independently required motions. Reference-supported fixed attachments are assessed separately. Second, reference parts and generated links often differ in granularity. Our **Cross-Granularity Structural Matching (CGSM)** allows one-to-many and many-to-one correspondences. It associates localized part instances with supporting URDF links and checks reference-supported attachments and independent motion requirements across these correspondences.

To summarize, our contributions are:

- We present a systematic automatic inspection framework for generated articulated assets, producing verdicts, localized evidence, and repair suggestions.
- We develop reference-inventory construction and CGSM to fully capture part and joint requirements and validate them across heterogeneous modeling granularities.
- We demonstrate the effectiveness of ArtiCritic via a comprehensive suite of evaluations.

2 Related Work

2.1 Generation of Articulated Objects

Advances in articulated-object generation broaden the supply of interactive assets, increasing the need for scalable and systematic validation. Existing approaches to articulated-object generation broadly fall into two groups: geometry-based modeling and learned synthesis, and code-based generation. The former includes articulating static meshes through part decomposition and kinematic inference, exemplified by Articulate AnyMesh Qiu et al. (2025), with extensions incorporating motion constraints, functional completion, scene context, and explicit instructions Xu et al. (2026); Zhao et al. (2026); Yang et al. (2025); Li et al. (2026a). Other approaches jointly model shape and articulation through diffusion or structured representations, often incorporating physical constraints Lei et al. (2023); Luo et al. (2025); Kreber & Stueckler (2025); Chen et al. (2025); Wang et al. (2026a); Li et al. (2026c); Wu et al. (2026). SPARK He et al. (2026) further combines VLM-guided reconstruction with generated open-state images to refine articulation. Complementing these approaches, code-based methods express assets as editable programs, progressing from scene construction with visual feedback Hu et al. (2024) to articulation programming from reconstructed or retrieved parts and text-driven geometry design Zhao et al. (2025); Le et al. (2025); Gao et al. (2026). Articraft Zhou et al. (2026) supports scalable synthesis through a dedicated SDK, while aDSL Wang et al. (2026b) integrates construction and verification through explicit hierarchies and spatial constraints. These advances motivate systematic validation beyond executability and visual plausibility. ArtiCritic particularly targets code-based generation, where language models can translate localized defect reports and corrective suggestions into direct code revisions.

2.2 Evaluation of Articulated Objects

Evaluation of generated 3D objects has progressed from multi-view quality and text alignment He et al. (2023) to VLM-based comparison and human-aligned scoring Wu et al. (2024); Zhang et al. (2025a); Maiti et al. (2025). More diagnostic approaches, such as Eval3D Duggal et al. (2025),

localize inconsistencies through foundation-model probes, with further assessment of parts and materials Zhang et al. (2025b). Complementary benchmarks evaluate geometric programs, CAD reconstruction, and part-level structure Li et al. (2026b); Doris et al. (2026); Yang et al. (2026), while 3DHarnessBench Liu et al. (2026) examines active inspection and tool use in 3D-to-code reconstruction. These efforts extend evaluation beyond global appearance, but articulated objects also require checking joint structure and behavior under motion.

Several generation and animation systems partially address these requirements through visual feedback on geometry and articulation Le et al. (2025); Gao et al. (2026). ArtiMo Zou et al. (2026) further uses trajectory cues to guide localized motion diagnosis and repair. Beyond visual critique, Articraft Zhou et al. (2026) combines built-in validation with LLM-authored tests of geometric and pose-dependent relations. aDSL Wang et al. (2026b) pairs an Image Critic for checklist-guided visual inspection with a Code Critic that checks reported defects against the program. Although complementary, these checks do not jointly cover all three dimensions considered here. ArtiCritic unifies intrinsic validity, part completeness and articulation plausibility. It grounds candidate defects in localized evidence and confidence estimates for VLM adjudication, producing an overall verdict and targeted feedback.

3 METHOD

ArtiCritic evaluates generated articulated assets through three complementary stages. **Intrinsic validity** checks syntax, mesh integrity, and potential self-intersections during motion. **Part completeness** identifies missing or mismatched functional components through reference-grounded correspondences. **Articulation plausibility** examines whether the joint structure and motion parameters satisfy reference requirements. Each stage produces defect candidates with localized targets, supporting evidence, and confidence estimates, which a final VLM adjudicator reassesses to determine the verdict.

3.1 INSPECTION PIPELINE

Inspection begins with reference-independent *intrinsic validity* checks. A URDF parser validates syntax, link-joint connectivity, joint definitions, and limits, while mesh checks identify missing or unusable geometry. Motion sweeps flag potential self-intersections. These deterministic checks provide high-confidence evidence for structural and geometric defect candidates.

Next, *part completeness* assesses reference-required functional components, instance counts, and layout. *Articulation plausibility* derives motion requirements independently of the candidate and checks whether its structure supports them. Both stages use the reference inventories (Section 3.2) and CGSM (Section 3.3) to match functional instances to candidate geometry and assess their connections.

Finally, the evidence-guided VLM adjudication and recheck process reviews each defect candidate using its target, supporting reference and candidate evidence, and confidence estimates. Confidence indicates evidence strength rather than imposing a fixed acceptance threshold. The adjudicator can exempt intentional contact or nesting, such as buttons within housings or pistons within cylinders, and dismiss unsupported extra-joint claims caused by incomplete reference inventories. Parser-confirmed structural failures are retained directly. Visual defects require evidence of the affected component, the error, and its consequence for completeness or functionality. A consistency recheck resolves contradictions between decisions and explanations. The asset receives `FAIL` if any actionable defect remains, and `PASS` otherwise.

3.2 REFERENCE PART AND JOINT INVENTORIES

Reliable reference inventories support both subsequent part matching and joint inference. However, neither open-vocabulary part detectors nor direct VLM predictions consistently capture all visible functional components. We therefore combine segmentation masks with detector proposals. Semantic-SAM Li et al. (2023) first produces region masks $\mathcal{M} = \{m_k\}_{k=1}^{K}$, which provide spatial boundaries but may fragment functional parts and lack reliable semantic labels.

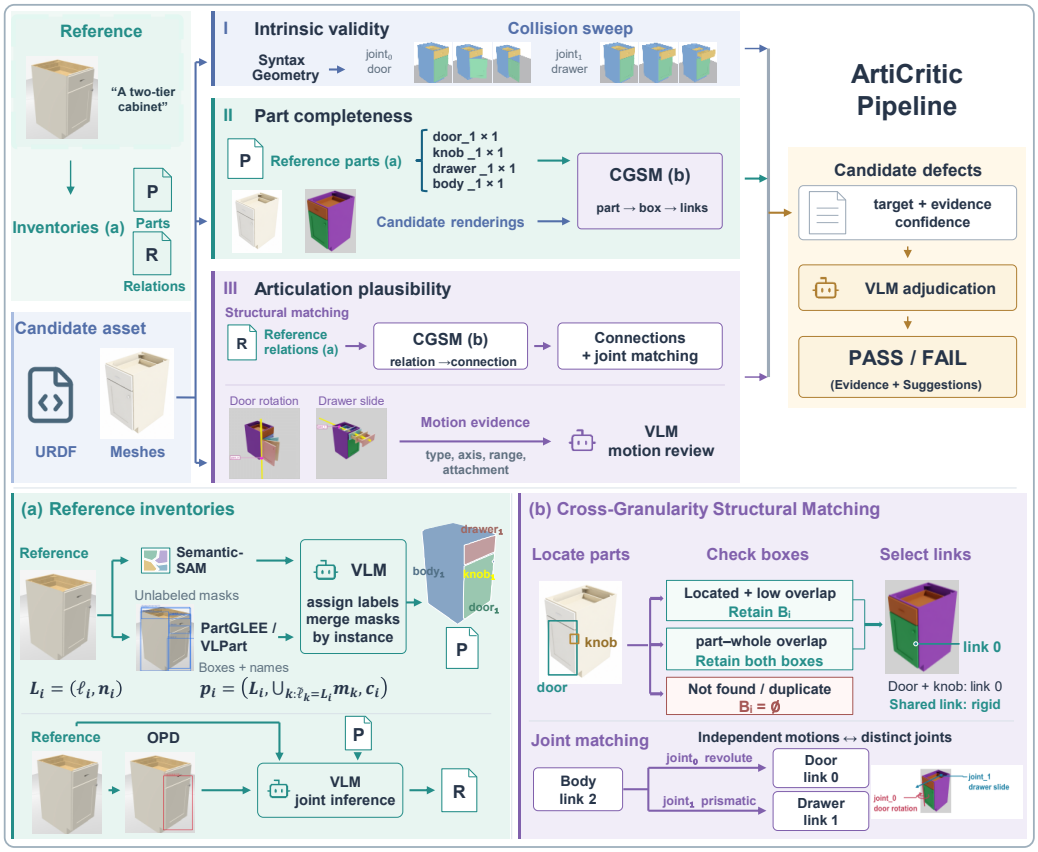

Figure 2: **Overview of ArtiCritic**. Articulated assets are evaluated through intrinsic validity, part completeness, and articulation plausibility. **(a)** Reference part and relation inventories guide evaluation. **(b)** Cross-Granularity Structural Matching (CGSM) aligns these inventories with candidate surfaces, links, and joint structures. Candidate defects undergo VLM adjudication to produce a final decision with evidence and suggestions.

To supply semantic guidance, we use the bounding boxes, part labels, and confidence scores predicted by PartGLEE Li et al. (2024) and VLPart Sun et al. (2023). PartGLEE serves as the primary detector: proposals from the two models are merged when their bounding-box overlap exceeds a threshold, retaining PartGLEE's output even when their labels disagree. Unmatched proposals are retained as additional evidence. The resulting proposal collection is

$$\mathcal{D} = \{d_j\}_{j=1}^{N}, \qquad d_j = (b_j, \ell_j, s_j), \tag{1}$$

where b_j is a bounding box, ℓ_j is a semantic part name, and s_j is the detector score. The index j identifies a proposal, not a final part instance.

For each mask m_k, the VLM examines its region in the reference image together with the detector proposals $\mathcal{D}$. It adopts the semantic name of a compatible proposal d_j or, if none is suitable, infers a label directly from visual evidence:

$$\ell_k = \begin{cases} \ell_j, & \text{if the VLM selects } d_j \text{ for } m_k, \\ \ell_k^{\mathrm{VLM}}, & \text{otherwise.} \end{cases} \tag{2}$$

Alongside label assignment, the VLM groups mask fragments belonging to the same physical component. Each resulting instance receives an index within its semantic category, giving each mask an instance-aware label $\tilde{\ell}_k = (\ell_k, n_k)$. Masks sharing the same full label are merged, while distinct instances with the same semantic name retain different indices. For example, fragments of the upper drawer receive $(\text{drawer}, 1)$, whereas those of the lower drawer receive $(\text{drawer}, 2)$. The same

grouping and indexing procedure applies to components identified by the VLM without detector support.

After labeling, masks sharing the same instance-aware label are merged into a single part region. Let $\{L_i\}_{i=1}^{P}$ denote the distinct instance-aware labels. Each reference part is represented as

$$p_i = \left(L_i, \bigcup_{k:\, \tilde{\ell}_k = L_i} m_k,\, c_i \right), \tag{3}$$

where c_i denotes the associated confidence. For example, two fragments labeled $(\text{drawer}, 1)$ are combined into one drawer region. These records form the reference part inventory, providing explicit component identities and locations for subsequent joint inference.

We next construct a reference joint inventory by combining local motion proposals with reasoning over these part instances. OPD Jiang et al. (2022) first detects openable regions and predicts candidate motion types, axes, and origins. We associate these regions with reference parts through spatial overlap, attaching local motion cues to identifiable components. The VLM then receives the reference image, part inventory, and OPD proposals to determine which components should move independently and what motion types they require.

For each potential moving component, the VLM examines its visible attachment and functional relationship to surrounding parts, using OPD predictions as supporting evidence. This distinguishes independent motion from motion inherited from an attached component: two drawers require separate sliding motions, whereas handles rigidly attached to them require no additional joints. The VLM also adds visually supported motion requirements missed by OPD. When the reference does not clearly establish a connection or distinguish between motion types, the corresponding hypothesis receives lower confidence and may retain multiple plausible types.

The joint inventory records expected independent motions, with one entry per distinct moving-part instance. Each entry contains the component identity or image location, plausible motion types, intended function, supporting evidence, and confidence. Multiple plausible types express uncertainty about the same motion requirement. Fixed attachments, such as a handle moving with its drawer, do not create additional motion entries; CGSM assesses these attachments using the reference image and part inventory (Section 3.3). Both inventories are constructed independently of the candidate, preserving requirements for components or motions absent from it.

3.3 Cross-Granularity Structural Matching

Cross-Granularity Structural Matching (CGSM) matches reference part instances to candidate URDF links without requiring one-to-one correspondence. A door and its handle may share one link, while a single functional component may span several links. CGSM accommodates these differences by establishing part and link correspondences and then checking whether the candidate provides the required motions.

We first select a candidate rendering with a viewpoint close to the reference image and obtain the image with link annotation. Given both images and the reference part inventory, the VLM locates each reference instance in the candidate and predicts a bounding box. A box is set to empty if no corresponding component is found or its overlap with a previously retained box exceeds an IoU threshold. Overlapping boxes are retained when visual evidence supports a hierarchical part-whole structure. We record whether an empty box results from unsuccessful localization or rejected duplication.

For each retained box, the VLM selects the corresponding links from a rendering at the same viewpoint, where links have distinct colors and explicit URDF names. It provides the color-to-link association, while the VLM determines which links represent the localized part. We record

$$M(p_i) = (B_i, \mathcal{L}_i, c_i), \qquad B_i = \varnothing \;\Rightarrow\; \mathcal{L}_i = \varnothing, \tag{4}$$

where B_i is the retained box, $\mathcal{L}_i$ contains the selected link names, and c_i denotes matching confidence. Boxes distinguish visible instances even when they share a link, while link sets accommodate parts represented by multiple links.

These correspondences support motion and attachment checks. For each reference relation $r_{ij} = (p_i, p_j, \tau_{ij})$, where τ_{ij} specifies the expected connection type, the VLM selects candidate joints using the matched parts, URDF topology, and rendered evidence. Joint types and reuse across independent requirements are checked programmatically, while motion renderings support judgments of direction, range, and attachment.

The VLM also identifies fixed attachments supported by the reference image and part inventory. The matched parts must share a single link or be represented by links connected only through fixed joints. Violations become defect candidates, while ambiguous attachments receive lower confidence for final adjudication.

In Figure 2(b), the door and its knob are localized in separate boxes, and their supported part-whole overlap is retained. Both map to link 0, satisfying their rigid attachment. Separate links joined by a fixed joint would also satisfy this relation, whereas an unintended movable connection would generate a defect candidate. The door–body and drawer–body motion requirements are matched to revolute joint 0 and prismatic joint 1, respectively. These independent motions require distinct joints, even when other parts share a link. Missing correspondences and incompatible connections are forwarded with localized evidence and confidence for final adjudication.

4 EXPERIMENTS

We evaluate whether ArtiCritic distinguishes acceptable articulated assets from defective ones and whether its diagnostics explain failures beyond executable validity.

4.1 EXPERIMENTAL SETUP

Our evaluation set contains 150 articulated assets from three source groups: 37 generated by SPARK He et al. (2026) (**Source-1**), 20 produced by an Articraft-based workflow Zhou et al. (2026) (**Source-2**), and 93 drawn from PartNet-Mobility Xiang et al. (2020) and Artiverse Iliash et al. (2026) (**Source-3**). The first two groups use web-sourced reference images and assess defect detection across different generation paradigms. Source-3 contains 22 unmodified assets and 71 corrupted variants, enabling controlled evaluation of injected defects such as removed parts and associated joints or altered joint limits. The dataset focuses on storage furniture and microwaves, supplemented by a small random sample of other articulated objects from the source datasets.

We manually annotate all assets using three separate criteria. (i) Part completeness: the candidate must contain the required visible components with the correct instance counts. Missing components, reference-inconsistent extra components, and count mismatches are labeled as errors. Extraneous components are not counted as errors when the ambiguity arises from incomplete reference views. (ii) Joint completeness: each independent motion supported by visible articulation cues must have a corresponding movable connection. A missing connection or a rigid attachment that prevents the required motion constitutes an error. (iii) Motion plausibility: existing joints must produce plausible motion in the rendered sequences. Incorrect motion types, directions, or ranges, as well as detachment or unintended interpenetration, are labeled as errors. Only defects supported by the reference images or motion evidence are counted. An asset is labeled `PASS` when no error is identified under these criteria and `FAIL` when at least one error is confirmed, yielding 50 `PASS` and 100 `FAIL` samples for asset-level evaluation.

Development and test sets are disjoint. All annotations are finalized before evaluation, and no test assets are used for pipeline tuning. All baselines use the same VLM.

4.2 QUANTITATIVE COMPARISON

Table 1 reports overall accuracy and the accuracy on the different sources. For a critic used to screen generated assets, FAIL detection is more consequential than aggregate accuracy: a false PASS leaves a defective asset uncorrected. The PASS column checks that higher failure sensitivity does not come only from rejecting everything.

Articraft and aDSL both adopt their open-source inspection pipelines. For GPT-6-Astra, we feed the assets and reference images into the model and require it to follow our pipeline to evaluate candidate

Table 1: Comparison of accuracy on the manually annotated dataset from different sources. FAIL denotes manually labeled defective assets, and PASS denotes acceptable assets.

Method	All assets (accuracy)	FAIL assets	PASS assets	Source-1	Source-2	Source-3
Articraft	59.3%	63.0%	52.0%	45.9%	65.0%	63.4%
aDSL	70.0%	76.0%	58.0%	59.5%	65.0%	75.3%
GPT-6-Astra	82.0%	84.0%	78.0%	81.1%	70.0%	84.9%
ArtiCritic	**88.0%**	**92.0%**	**80.0%**	**89.2%**	**80.0%**	**89.2%**

assets from these dimensions: URDF syntax, geometric quality, motion-induced collision detection, part and joint completeness, and the correctness of joint motion. As shown in the table, the inspections of Articraft and aDSL are relatively incomplete. They primarily validate their own generated assets, which may limit their effectiveness on out-of-domain assets. GPT-6-Astra performs better but still misses defects that ArtiCritic identifies. ArtiCritic achieves higher defect recall while retaining the highest PASS accuracy among the compared methods, indicating that its gains do not arise solely from more frequent rejection.

Because missed defects undermine asset screening, we further analyze FAIL detection by manually annotated primary defect type. Table 2 reports results for missing or extra parts, missing required joints, and abnormal joint motion. Other geometric-quality issues are excluded from this breakdown.

Table 2: The FAIL assets are annotated by defect types to compare each method's prediction accuracy across different error categories.

Primary defect	Articraft	aDSL	GPT-6-Astra	ArtiCritic
Missing / extra parts	60.6%	66.7%	75.8%	**87.9%**
Missing required joints	53.3%	83.3%	90.0%	**96.7%**
Abnormal joint motion	75.0%	78.1%	**90.6%**	**90.6%**
All FAIL assets	63.0%	76.0%	84.0%	**92.0%**

Articraft's code-based tests may miss defects outside the generated checks, whereas aDSL's visual critique provides stronger coverage of part and joint errors. GPT-6-Astra performs well across categories, but ArtiCritic further improves detection of missing or extra parts and missing joints. These gains are consistent with explicit reference inventories and CGSM, which check instance completeness and required structural relationships before final adjudication. ArtiCritic matches GPT-6-Astra on abnormal motion, where assessment relies more directly on VLM interpretation of motion renderings. This pattern suggests that structured reference requirements and localized evidence complement general multimodal reasoning.

4.3 Qualitative Comparison

We focus the qualitative comparison on GPT-6-Astra, the strongest baseline in our quantitative evaluation. Articraft and aDSL integrate inspection into generator-specific refinement loops, so standalone evaluation may not fully reflect their intended capabilities.

The top row of Figure 3 shows three assets accepted by Astra despite missing reference components. ArtiCritic identifies these omissions, including a cabinet's central carved panel and two missing legs in a six-legged reference modeled with only four. The reference inventory combines segmentation masks and detector proposals to capture fine details and distinguish individual instances. CGSM maps these instances to candidate geometry, exposing missing components and count mismatches even when the corresponding semantic category is present.

The bottom row shows articulation defects overlooked by Astra in a tower crane, desk lamp, and monitor stand. Their candidates lack the expected hook translation, lamp-head rotation, or screen adjustment. ArtiCritic derives these requirements from the reference inventories independently of the candidate. CGSM checks whether the matched components support the required relative motion, while motion evidence assesses the behavior of existing joints. This distinction enables localized detection of missing motion capabilities even when other joints function correctly.



Figure 3: Qualitative comparison with the GPT-6-Astra examiner. All assets in the figure are accepted by Astra. The top row marks missing components, and the bottom row marks missing joints.

4.4 Ablation Experiment

This ablation isolates how the reference part inventory is constructed. We compare four variants under the same reference images and VLM setting: (i) PartGLEE and VLPart used together, (ii) direct VLM prediction of boxes and part names, (iii) Semantic-SAM masks followed by VLM labeling, and (iv) our full fusion of detector proposals, semantic masks, and VLM assignment and fusion. The four outputs are shown side by side in Figure 4.

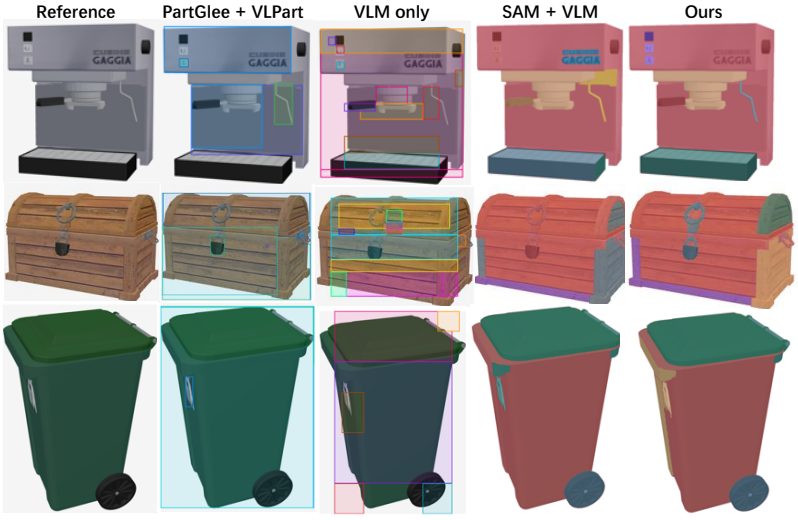

Figure 4: Reference-inventory ablation. Columns compare the reference, detector-only proposals from PartGLEE+VLPart, direct VLM prediction, Semantic-SAM followed by VLM labeling, and our proposal-guided mask fusion. Rows show a coffee machine, a wooden chest, and a waste bin.

Detector-only predictions locate some parts reliably but miss subtle or uncommon components, while direct VLM predictions exhibit inaccurate boxes and omissions. Semantic-SAM masks improve coverage but can still be mislabeled or incorrectly merged. In the coffee machine example, SAM+VLM merges the rinse tube with its background. Our method uses the PartGLEE proposal to distinguish this component, directing the VLM toward local proposal and mask matching.

The full method combines detector localization, mask boundaries, and VLM reasoning to label missed parts and merge fragments of the same instance. Across the three examples, it produces more complete inventories with more precise localization, particularly for small components. These

qualitative results suggest that proposal-guided matching helps the VLM resolve local ambiguities and improves the reference inventories supplied to CGSM.

4.5 CGSM Evaluation

We evaluate CGSM's tolerance to valid link decompositions and sensitivity to motion violations using two original assets and four modified variants (Figure 5). For the cabinet, each drawer handle either shares the drawer link, connects through a fixed joint, or connects through an unintended revolute joint. CGSM accepts the first two representations, while the pipeline flags the third for introducing relative motion that violates the required rigid attachment.

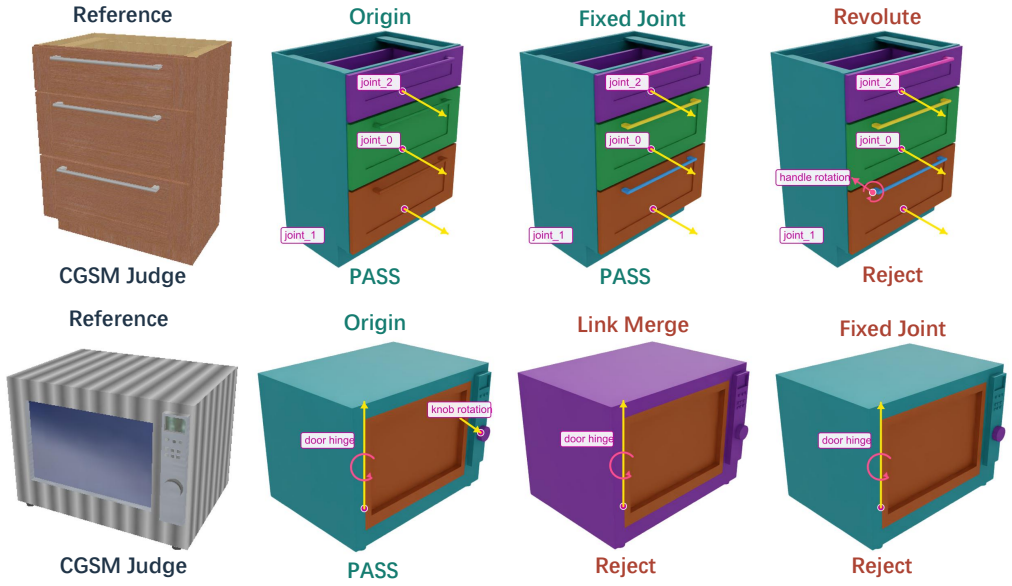

Figure 5: Three variants for the fixed handle and the microwave rotary knob. We annotate different components in the URDF as well as the introduced unintended joint.

We also present another microwave case, where the rotary control has three configurations: kept as an independently actuated link, merged into the body link, or retained with its joint changed to a fixed joint. Only the first configuration passes inspection. Although the second and third variants differ in modeling granularity, both place the rotary knob and the microwave cabinet within the same rigid group defined by CGSM. Both variants should be rejected as defective assets.

CGSM introduces no false alarms on the two legal variants and can localize the source of defects. The storage-furniture case demonstrates that a functional rigid group may span multiple links as long as fixed joints maintain the attachment between components. The microwave case illustrates that visible geometry alone is insufficient for judgment: merging or locking the rotary knob eliminates the required relative degree of freedom, and such cases will be flagged as defects.

5 Conclusion

In summary, we introduced ArtiCritic, an automated inspection framework for generated articulated objects that combines deterministic checks with VLM reasoning to assess intrinsic validity, part completeness, and articulation plausibility. To improve judgment accuracy, the framework directs the VLM toward localized evidence and targeted questions rather than relying on holistic image understanding alone. Within this process, segmentation masks and detector proposals guide the construction of reference part and joint inventories, while cross-granularity structural matching aligns functional instances and their relations with different link decompositions. Extensive experiments support the effectiveness of these designs. Together, they establish a systematic inspection protocol that provides localized diagnoses and actionable feedback for articulated object generation.

REFERENCES

Honghua Chen, Yushi Lan, Yongwei Chen, and Xingang Pan. Artilatent: Realistic articulated 3d object generation via structured latents. In *Proceedings of the SIGGRAPH Asia 2025 Conference Papers*, pp. 1–11, 2025.

Anna C Doris, Jacob Thomas Sony, Ghadi Nehme, Era Syla, Amin Heyrani Nobari, and Faez Ahmed. Cadbench: A multimodal benchmark for ai-assisted cad program generation. *arXiv preprint arXiv:2605.10873*, 2026.

Shivam Duggal, Yushi Hu, Oscar Michel, Aniruddha Kembhavi, William T Freeman, Noah A Smith, Ranjay Krishna, Antonio Torralba, Ali Farhadi, and Wei-Chiu Ma. Eval3d: Interpretable and fine-grained evaluation for 3d generation. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13326–13336. IEEE, 2025.

Yipeng Gao, Yunhao Ge, Peilin Cai, Daniel Seita, and Laurent Itti. Lam: Language articulated object modelers. 2026.

Yumeng He, Ying Jiang, Jiayin Lu, Yin Yang, and Chenfanfu Jiang. Spark: Sim-ready part-level articulated reconstruction with vlm knowledge. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7233–7243, 2026.

Yuze He, Yushi Bai, Matthieu Lin, Wang Zhao, Yubin Hu, Jenny Sheng, Ran Yi, Juanzi Li, and Yong-Jin Liu. T^3Bench: Benchmarking current progress in text-to-3d generation. *arXiv preprint arXiv:2310.02977*, 2023.

Ziniu Hu, Ahmet Iscen, Aashi Jain, Thomas Kipf, Yisong Yue, David A Ross, Cordelia Schmid, and Alireza Fathi. Scenecraft: An llm agent for synthesizing 3d scenes as blender code. In *ICML*, pp. 19252–19282, 2024.

Denys Iliash, Jiayi Liu, Egor Fokin, Qirui Wu, Ali Mahdavi Amiri, Manolis Savva, and Angel X Chang. Artiverse: A diverse and physically grounded dataset for articulated objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8932–8942, 2026.

Hanxiao Jiang, Yongsen Mao, Manolis Savva, and Angel X Chang. Opd: Single-view 3d openable part detection. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXIX*, pp. 410–426. Springer, 2022.

Jens U Kreber and Joerg Stueckler. Guiding diffusion-based articulated object generation by partial point cloud alignment and physical plausibility constraints. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 3206–3214. IEEE, 2025.

Long Le, Jason Xie, William Liang, Hung-Ju Wang, Yue Yang, Yecheng Jason Ma, Kyle Vedder, Arjun Krishna, Dinesh Jayaraman, and Eric Eaton. Articulate-anything: Automatic modeling of articulated objects via a vision-language foundation model. In *International Conference on Learning Representations*, volume 2025, pp. 17578–17602, 2025.

Jiahui Lei, Congyue Deng, William B Shen, Leonidas Guibas, and Kostas Daniilidis. Nap: Neural 3d articulated object prior. *Advances in Neural Information Processing Systems*, 36:31878–31894, 2023.

Feng Li, Hao Zhang, Peize Sun, Xueyan Zou, Shilong Liu, Jianwei Yang, Chunyuan Li, Lei Zhang, and Jianfeng Gao. Semantic-sam: Segment and recognize anything at any granularity. *arXiv preprint arXiv:2307.04767*, 2023.

Junyi Li, Junfeng Wu, Weizhi Zhao, Song Bai, and Xiang Bai. Partglee: A foundation model for recognizing and parsing any objects. In *European Conference on Computer Vision*, pp. 475–494. Springer, 2024.

Ruining Li, Yuxin Yao, Matt Zhou, Chuanxia Zheng, Christian Rupprecht, Joan Lasenby, Shangzhe Wu, and Andrea Vedaldi. Instruct-particulate: Scaling feed-forward 3d object articulation with kinematic control. *arXiv preprint arXiv:2606.14699*, 2026a.

Wenyi Li, Renkai Luo, Yue Yu, Huan-ang Gao, Mingju Gao, Li Yuan, Chaoyou Fu, and Hao Zhao. Benchmarking phd-level coding in 3d geometric computer vision. *arXiv preprint arXiv:2603.30038*, 2026b.

Zhe Li, Xiang Bai, Jieyu Zhang, Zhuangzhe Wu, Che Xu, Ying Li, Chengkai Hou, and Shanghang Zhang. Urdf-anything: Constructing articulated objects with 3d multimodal language model. *Advances in Neural Information Processing Systems*, 38:94974–95002, 2026c.

Ling Liu, Bingchen Gong, Amal Dev Parakkat, and Maks Ovsjanikov. 3dharnessbench: Probing agentic 3d-to-code capabilities of frontier vision-language models. *arXiv preprint arXiv:2609.06535*, 2026.

Rundong Luo, Haoran Geng, Congyue Deng, Puhao Li, Zan Wang, Baoxiong Jia, Leonidas Guibas, and Siyuan Huang. Physpart: Physically plausible part completion for interactable objects. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 12386–12393. IEEE, 2025.

Shalini Maiti, Lourdes Agapito, and Filippos Kokkinos. Gen3deval: Using vllms for automatic evaluation of generated 3d objects. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18552–18562. IEEE, 2025.

Xiaowen Qiu, Jincheng Yang, Yian Wang, Zhehuan Chen, Yufei Wang, Tsun-Hsuan Wang, Zhou Xian, and Chuang Gan. Articulate anymesh: Open-vocabulary 3d articulated objects modeling. *arXiv preprint arXiv:2502.02590*, 2025.

Peize Sun, Shoufa Chen, Chenchen Zhu, Fanyi Xiao, Ping Luo, Saining Xie, and Zhicheng Yan. Going denser with open-vocabulary part segmentation. *arXiv preprint arXiv:2305.11173*, 2023.

Penghao Wang, Siyuan Xie, Hongyu Yan, Xianghui Yang, Jingwei Huang, Chunchao Guo, and Jiayuan Gu. Artllm: Generating articulated assets via 3d llm. *arXiv preprint arXiv:2603.01142*, 2026a.

Rui-Huan Wang, Si-Tong Wei, Jia-Qi He, Heng-Yi Wei, Baoquan Chen, and Peng-Shuai Wang. adsl: Agentic 3d creation via joint agent-program design. *arXiv preprint arXiv:2608.17975*, 2026b.

Tong Wu, Guandao Yang, Zhibing Li, Kai Zhang, Ziwei Liu, Leonidas Guibas, Dahua Lin, and Gordon Wetzstein. Gpt-4v (ision) is a human-aligned evaluator for text-to-3d generation. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 22227–22238. IEEE, 2024.

Zhuangzhe Wu, Yue Xin, Chengkai Hou, Minghao Chen, Yaoxu Lyu, Jieyu Zhang, and Shanghang Zhang. Urdf-anything+: End-to-end generation for simulation-ready articulated assets. *arXiv preprint arXiv:2603.14010*, 2026.

Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, Li Yi, Angel X. Chang, Leonidas J. Guibas, and Hao Su. SAPIEN: A simulated part-based interactive environment. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

WenBo Xu, Liu Liu, Li Zhang, Dan Guo, and RuoNan Liu. Motionanymesh: Physics-grounded articulation for simulation-ready digital twins. *arXiv preprint arXiv:2603.12936*, 2026.

Yikang Yang, Zhanpeng Hu, Youtian Lin, Mengqi Zhou, Jingxi Xu, Feihu Zhang, Jiaheng Liu, and Yao Yao. P3d-bench: Benchmarking mllms for parametric 3d generation and structural reasoning. *arXiv preprint arXiv:2606.11152*, 2026.

Yixuan Yang, Luyang Xie, Zhen Luo, Zixiang Zhao, Tongsheng Ding, Mingqi Gao, and Feng Zheng. Artiworld: Llm-driven articulation of 3d objects in scenes. *arXiv preprint arXiv:2511.12977*, 2025.

Yuhan Zhang, Mengchen Zhang, Tong Wu, Tengfei Wang, Gordon Wetzstein, Dahua Lin, and Ziwei Liu. 3dgen-bench: Comprehensive benchmark suite for 3d generative models. *arXiv preprint arXiv:2503.21745*, 2025a.

Yuhan Zhang, Long Zhuo, Ziyang Chu, Tong Wu, Zhibing Li, Liang Pan, Dahua Lin, and Ziwei Liu. Hi3deval: Advancing 3d generation evaluation with hierarchical validity. *arXiv preprint arXiv:2508.05609*, 2025b.

Mandi Zhao, Yijia Weng, Dominik Bauer, and Shuran Song. Real2code: Reconstruct articulated objects via code generation. In *International Conference on Learning Representations*, volume 2025, pp. 668–686, 2025.

Mingrui Zhao, Sai Raj Kishore Perla, Kai Wang, Sauradip Nag, Duc Anh Nguyen, Jiayi Peng, Ruiqi Wang, Angel X Chang, Manolis Savva, Ali Mahdavi-Amiri, et al. Functionalization via structure completion and motion rectification. *arXiv preprint arXiv:2605.18010*, 2026.

Matt Zhou, Ruining Li, Xiaoyang Lyu, Zhaomou Song, Zhening Huang, Chuanxia Zheng, Christian Rupprecht, Andrea Vedaldi, and Shangzhe Wu. Articraft: An agentic system for scalable articulated 3d asset generation. *arXiv preprint arXiv:2605.15187*, 2026.

Chunyu Zou, Peng Dai, Yi-Hua Huang, Ze Yuan, Jingwei Huang, Yeming Yao, and Xiaojuan Qi. Artimo: Agent-driven articulated mesh animation. *arXiv preprint arXiv:2608.20699*, 2026.

AI USE STATEMENT

In this work, we used generative AI tools to assist in designing and refining research methodology and experimental plans, developing and polishing research hypotheses, implementing the proposed algorithm, and interpreting experimental results.

This work does not involve constructing mathematical statements, providing key steps for mathematical proofs, or assisting in proof writing.

In addition, generative AI tools were used to summarize and analyze existing literature, explore research directions, identify research gaps, brainstorm ideas, and revise the wording and structure of the manuscript. All AI-assisted outputs have been manually reviewed. During manuscript preparation, we iteratively revised AI-generated drafts to ensure that the wording and structure faithfully reflect the intended method. We also adjusted pseudocode presentation, citation formatting, and figure captions to maintain consistency across the manuscript and figures. We take full responsibility for all final content of this work, including text, claims and assets produced with generative AI assistance.