A. Eval3D Qualitative Analysis

Geometric Consistency: Fig. 8 illustrate the geometric artifacts identified by our proposed geometric consistency metric. By comparing the image-based normals with the rendered geometric normals, the metric effectively highlights texture-geometry mis-alignments. For instance, see the misalignment between the corresponding texture map and the underlying geometry in the taco fillings, the eggs on the nest, the ramen bowl and the salmon examples of Fig. 8. **Such misalignment are evident when comparing the "Algorithm generation" column with the "Algorithm rendered normal" column.** Among the evaluated text-to-3D algorithms, ProlificDreamer exhibits the most significant texture-geometry misalignments. While MVDream's generations are aesthetically appealing, they sometimes display noisy, non-smooth geometries with holes and cavities, such as the flower pot in the first row (right) of Figure 8.

Structural Consistency: Fig. 18 qualitatively highlights the structural consistency metric. The first column of the figure shows renderings of the generated 3D assets at $0°, 90°, 180°$ and $270°$. The next two columns present novel view predictions by the zero123 algorithm, using the $0°$ and $90°$ renderings of the generated assets, respectively.

The second and third examples demonstrate faulty generations that suffer from the well-known Janus issue (multi-head reconstructions), where all four viewpoints (in the second and third rows) contain the heads of a chimpanzee and an orange cat, respectively. **While the text-based 3D generation algorithm struggled, the image-based novel-view synthesis algorithm successfully identified the failure by making multi-view consistent predictions (columns 2, 3).**

Since Zero123 relies on one of the renderings of the generated asset as a reference viewpoint, it can potentially fail to generate consistent predictions if the reference viewpoint contains artifacts or is confusing (e.g., row 3, column 2, generated using the $90°$ rendering of the generated 3D asset in column 1). Rows 1 and 4 showcase examples where good generations are well-aligned with the proposed structural consistency metric, demonstrating qualitative alignment between algorithm generations and zero123 predictions.

Semantic Consistency: Fig. 19 demonstrates our proposed semantic consistency metric. Similar to the geometric consistency metric, the semantic consistency metric is computed directly in the 3D geometric space (as the percentage of semantically consistent mesh vertices), allowing us to localize artifacts in the 3D space. The figure side-by-side compares the generated textured asset with the underlying geometry in the first two columns, while the last column depicts the semantic inconsistency map (bright regions represent mesh vertices with high standard deviation of the back-projected DINO features from neighboring viewpoints).

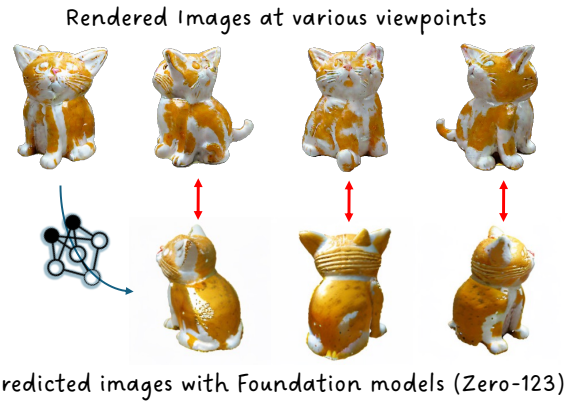


Figure 7. **Structural consistency:** We compare the rendered images with those predicted by Zero-123 [29]. A structurally coherent object should maintain consistent appearance across different viewpoints, allowing one to predict its appearance from another angle. If the predictions & renderings differ significantly, it likely indicates an issue. In this figure, generated cat has multiple faces. Since a normal cat only has one face, results from Zero-123 [29] show noticeable inconsistencies, allowing us to localize the structural incoherence.

The proposed metric effectively localizes artifacts such as Janus issues (multiple dog noses in row 1, multiple pigeon faces in row 4), extraneous geometry (monkey's hand in row 2), noisy texture-geometry alignment (in row 3), and arbitrary floaters in the final row of the figure. Moreover, since DINO features strongly correlate with the semantic class of the underlying image, faulty 3D generations whose semantic interpretation changes with viewpoints are often highlighted by the proposed metric (see Magic3D generation in row 2 of Fig. 21, which has a low semantic consistency score of 55.2%).

Text-3D Alignment: The alignment of the input text to the generated 3D asset is crucial as it demonstrates how well the 3D generation algorithm follows the input instructions. In Fig. 9, we highlight the text-3D alignment of several 3D generations. While some of these generations are aesthetically appealing, they fail to satisfy the user's textual instructions (e.g., the first and second generations in the first row). In other cases, certain objects or entities mentioned in the input prompt are missing (e.g., the missing shovel in row 2, column 1, and only two apples generated in column 2).

Despite demonstrating strong text-3D alignment overall, the generated frog with a mechanical heart (row1, column 3) displays an example where SDS-optimized generations can find unexpected or undesirable ways to satisfy the text prompt.

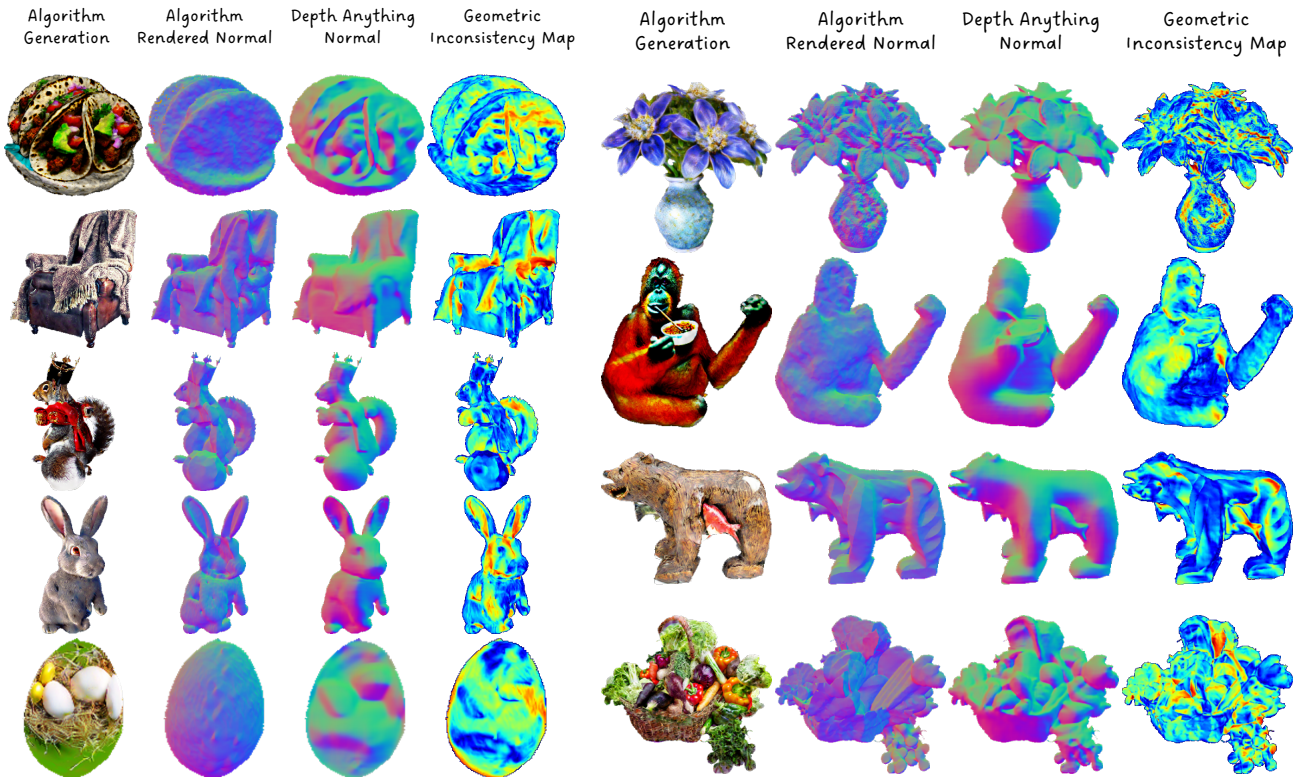


Figure 8. **Geometric Consistency Metric** evaluates texture-geometry alignment by comparing geometrically-rendered normals with image-based Depth-Anything normals. We back-project the consistency estimates onto the 3D mesh to localize 3D artifacts (missing ramen bowl, missing salmon, incorrect vegetable geometry). Bright greenish-red (or cyan) spots = high inconsistency regions , darker-blue regions = consistent areas (we use jet color mode).

B. Text-3D Comparison using Eval3D

We compare text-3D generations from multiple algorithms for two prompts in Fig. 20 and Fig. 21. For both the prompts, Magic3D [28] and DreamFusion [41] generate overly smooth and simplified geometries, which compromise the aesthetic appeal of the assets. This gets reflected quantitatively in their high geometric consistency scores and qualitatively in the corresponding geometric inconsistency maps as shown in the second column (bright spots indicate higher inconsistency, while darker-blue regions indicate lower error).

For each algorithm's generation in both figures, the semantic consistency metric effectively localizes artifacts in the 3D space. For example, ProlificDreamer exhibits Janus issues, Magic3D shows semantically confusing renderings of relatively thin geometries, and DreamFusion displays low texture-geometry alignment in its blurry reconstructions. Finally, MVDream completely fails to satisfy the text prompt in the first figure, generating a Michelangelo-style human instead of a dog.

C. Eval3D Failure Cases

While the proposed 3D evaluation metrics align better than current alternatives and offer an interpretable, fine-grained solution to 3D evaluation, there are still some failure cases that warrant future investigation. Fig. 10 demonstrates two such failure cases for semantic consistency metric.

In Fig. 10, the semantic consistency metric successfully localizes artifacts for both ProlificDreamer (regions with Janus issues, i.e., intersecting multiple heads) and MVDream (regions with poor texture-geometry alignment due to dog fur-like texture). However, the quantitative semantic consistency score for ProlificDreamer fails to align with human judgment for this particular prompt (while overall Prolificdreamer-Human alignment using semantic consistency metric equals 63%). We hypothesize that this is because the artifact is localized to a relatively thin region of semantic confusion where multiple heads intersect, thereby affecting the metric value less significantly. Another cause of failure is the occlusion of geometric parts at certain viewpoints, which potentially leads to different semantic interpretations from those viewpoints, resulting in higher semantic inconsistency values for such vertices (MVDream's dog

Figure 9. **Text-3D Alignment** analyzes how well the generated 3D assets align with user text instructions. We leverage the open-sourced LLaVA model to estimate text-3D alignment. The examples above showcase various scenarios: complete failure of text-3D alignment (first two prompts), missing objects (e.g., the shovel in the fourth example, two apples in the fifth example), and perfect alignment for the last column prompts.

nose).

Fig. 10 (right) highlights a particular scenario where a Gaussian-splatting-based algorithm generates relatively noisy and potentially out-of-distribution textures, making it difficult for the DinoV2 model to derive any meaningful semantic interpretation from the corresponding renderings.

D. Experimental Details

D.1. Training Details

We evaluated six text-to-3D and two image-to-3D algorithms for Eval3D benchmark. For all algorithms, we leveraged code and implementations from the Threestudio project [18] (or its extensions). Note that Threestudio's implementations have some differences from the original works. We refer readers to the "Notable differences from the paper" sections corresponding to each algorithm on the Threestudio project page. All our experiments were done on NVIDIA A40 GPUs. Most compute cost are spent on generating 3D assets for benchmarking, and the compute cost varies for different generation methods, ranging from 0.3 GPU hours (for DreamFusion [41]) to 6 GPU hours (for ProlificDreamer [56]). For evaluation, after rendering the mesh and 120 RGB images, the majority of compute cost is spend on running the foundation models, including LLaVA-7B [27], DINOv2 [39], and Stable-Zero123 [1]. Altogether, when run parallelly, these processes take about 5 minutes per 3D asset.

D.2. Hyper-parameter Selection

We follow standard practices by normalizing the object within $[-1, 1]$, sampling cameras around it, and rendering 256×256 images. Foundation models like DINO are robust to varying resolutions. The threshold for each metric is determined using a hold-out validation set, and the number of views is selected empirically.

D.3. Mesh Extraction and Vertex Visibility

We use marching cubes or marching tetrahedra to extract meshes from the learned density or tetrahedral fields (in Magic3D). While this process may introduce mesh artifacts, we find that, in practice, it has a limited effect on the semantic consistency metric compared to inconsistencies caused by other factors, such as the Janus effect. We only consider vertices that are visible from at least five viewpoints. Vertex visibility can be determined by rendering the mesh via rasterization. This helps avoid labeling spurious vertices (which exhibit low DINO variance due to being visible in very few frames) as consistent vertices.

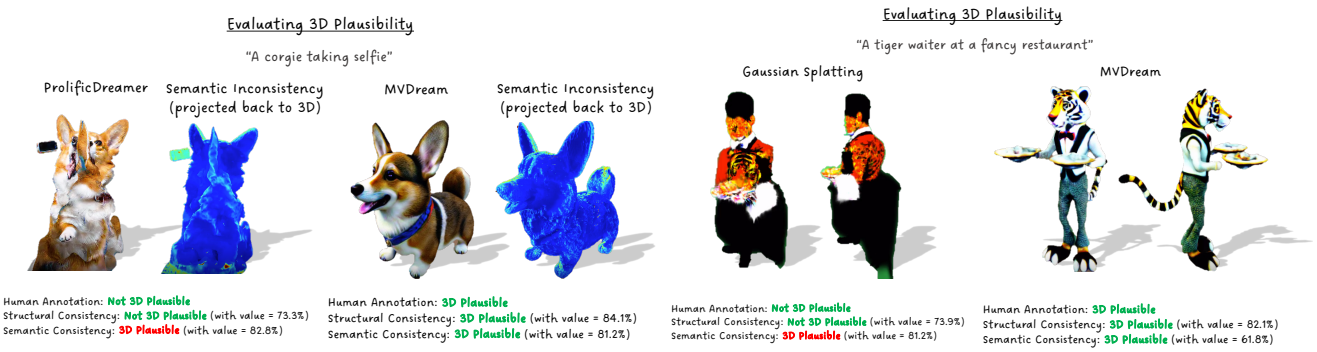


Figure 10. **Eval3D Semantic Consistency Failure Cases** Potential Reasons for failures – Artifacts (intersection of multiple faces) being localized to very thin regions; occlusion of geometric structures for certain viewpoints, making the overall geometry's semantic interpretation ambiguous; noisy gaussian splatting structures being too OOD for DINO model. (Zoom-in for best visualization)

D.4. Design of Structural Consistency Metric

Based on the observation that 3D generation algorithms typically perform well under small viewpoint shifts but struggle with significant changes, we evaluate every 90° to balance computational cost and effectiveness. The camera is positioned at a 70° elevation with varying azimuth angles to fully capture the top and front viewpoints. Bottom viewpoints are excluded, as they are often planar and semantically simpler, allowing us to conserve computation. To aggregate the scores across different inputs (*e.g.*, 0° vs. 90°), we experimented with both `max` and `average`. We find that while Zero123 is strong, it is not perfect and may perform worse on some input images due to noise. Taking the `max` enhances robustness to this variability.

E. Eval3D Benchmark

Here we provide more details about the Eval3D benchmark beyond what covered in the main paper experiments section. Figure 12 contains the statistics calculated from these scene graphs. The left histogram shows the number of prompts that have a certain number of entities. The right shows the statistics of the total number of semantic elements in a scene graph. Figure 13 illustrates the frequencies of each semantic element type. We also give the sub-category for entity and attribute. Here "Entity whole" is the major objects (e.g., "horse"), and "Entity part" is part of a major object (e.g., "saddle on a horse").

E.1. Human Annotations

Eval3D is annotated by 10 computer vision graduate students who have been well-trained to do the task. We strictly follow their institutions' rules during training and annotation. All annotators help for free and we really appreciate their effort. The major potential participation risk is that some 3D assets are so low-quality that they might be visually disturbing to humans. We collect annotations for 160 prompts on 6 models and 4 evaluation criteria. The annotation guideline is as follows:

Geometric Consistency: For each text prompt, we show annotators videos of RGB and rendered surface normals, placed side-by-side for all six algorithms. The annotator needs to give each 3D asset a score between 0–9, focusing on the *texture alignment with the surface normal*. We only use the scores for pairwise comparison. Ties are allowed. The annotation interface is shown in Fig. 17. The inter-annotator agreement, in terms of pairwise agreement (i.e. the probability that both annotators believe a 3D asset is better than the other), is 86.8%.

Semantic and Structural Consistency: For each 3D asset, we ask the annotator to judge its structural consistency or 3D plausibility by showing then a rendered RGB video. We educate them about artifacts like Janus issues (corgi in Fig. 18), fluctuating semantics (Fig. 19), arbitrary/incorrectly rendered novel views (Fig. 18). The annotator chooses among four options: *yes*, *uncertain yes*, *uncertain no*, and *no*. The annotation interface is shown in Fig. 17. We compute the inter-annotator agreement in two ways: (1) excluding the examples where annotators answer "uncertain yes" or "uncertain no", then the pairwise agreement between annotators is 97.2%. (2) if we consider "yes" and "uncertain yes" as one class, "no" and "uncertain no" as the other class, then the agreement is 83.1%.

Aesthetics: We use the same annotation template as for geometric consistency. For each text prompt, we show rendered RGB videos of 3D assets of all six algorithms. The annotator needs to rank the generations based on *whether a 3D asset is aesthetically pleasing, containing sharp, natural, vivid, bright, and high-resolution textures.* Ties are allowed.

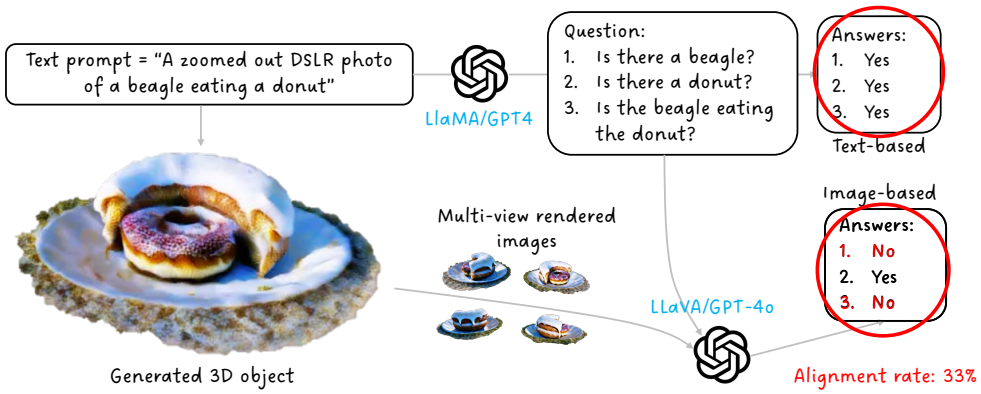


Figure 11. **Illustration of Eval3D Text-3D alignment pipeline.**

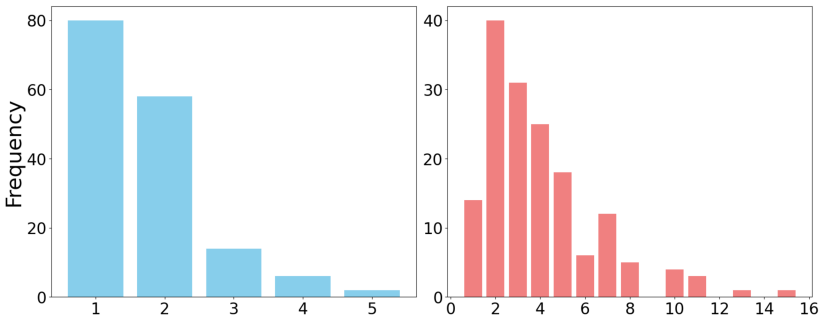


Figure 12. **Statistics of prompts in Eval3D Benchmark Left:** The number of entities in a prompt. **Right:** Number of semantic elements in a Prompt.

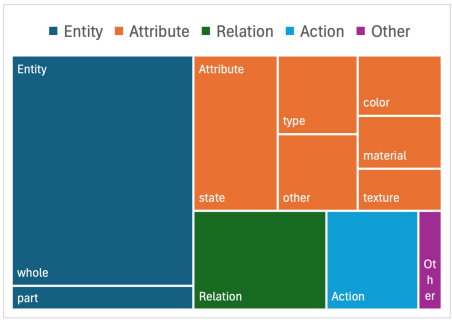


Figure 13. **Frequencies of each type of semantic element in the Eval3D scene graphs.**

The interface is illustrated in Fig. 16. The inter-annotator agreement is 83.8%.

Text-3D Alignment: We adopt the annotation template of previous text-to-image evaluation works [10, 59] and evaluate RGB videos instead of images. For each 3D asset, We ask annotators to answer a series of questions generated automatically from the text prompt, as discussed in [10]. The annotators perform multiple-choice video question answering. They can also mark a question as *unreasonable* in case they believe the question is not reasonable. The annotation interface is shown in Fig. 15. We compute a score for each 3D asset by counting how many questions have been answered "yes". The inter-annotator agreement, in terms of pairwise comparison between 3D assets, is 95.4%.

Note on computing automatic evaluations' alignment with human. For geometric consistency, aesthetics, and text-3D alignment, we compute evaluation metrics' alignment with humans using the same way as we compute the inter-annotator agreement, i.e. pairwise comparison agreement. For semantic and structural consistency, humans annotate "yes" "no" while the automatic evaluation gives a continuous value. We process the automatic evaluation by finding a threshold to divide its scores into two classes. For all evaluation metrics, we report the maximum value of human alignment given all possible thresholds. For Eval3D, the threshold for structural consistency is 75.8%; for semantic consistency it is 63.3%.

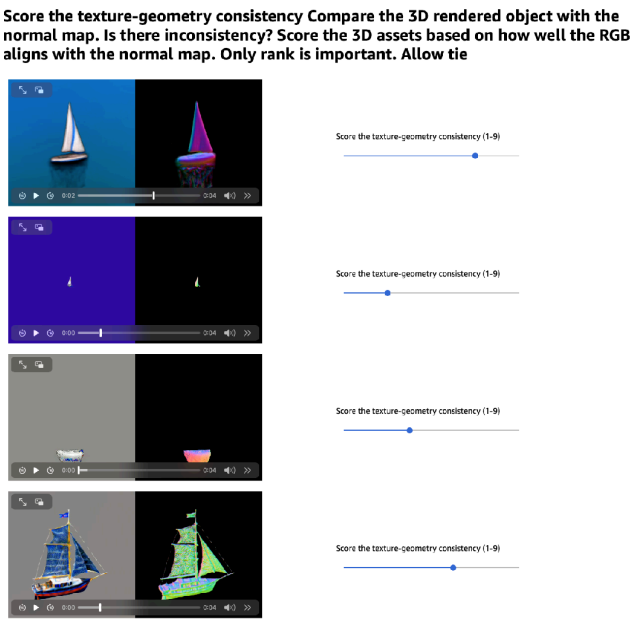


Figure 14. **Geometric Consistency Annotation Interface:** For each prompt, we show RGB & normal map videos of the assets generated by all six 3D generation models. We only show 4 of them here.

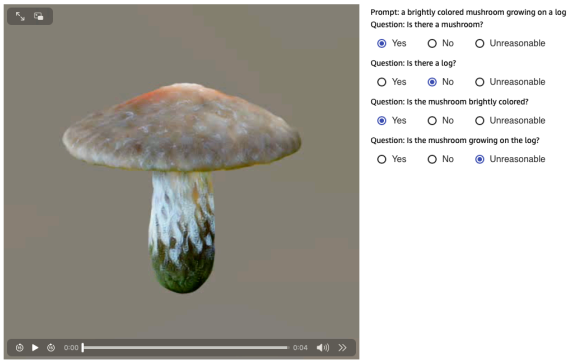


Figure 15. **Text-3D Alignment Annotaiton Interface.**

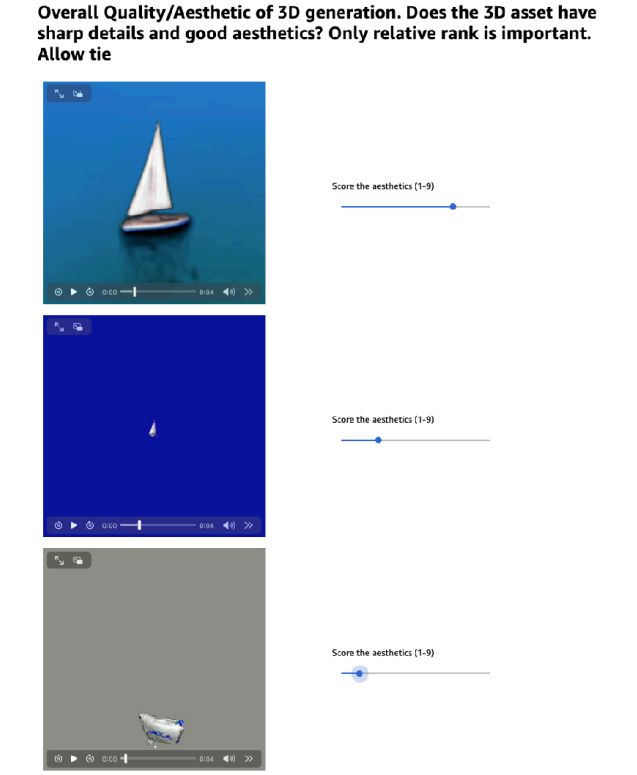


Figure 16. **Aesthetics Annotation Interface:** For each prompt, we show the RGB videos of the assets generated by all six 3D generation models. We only show 3 of them here.

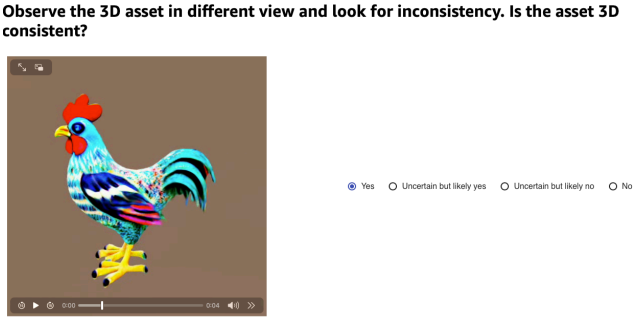


Figure 17. **Structural & Semantic Consis. Annotation Interface.**

Figure 18. **Structural Consistency** measures overall 3D plausibility by comparing (via Dreamsim) the text-based 3D asset renderings with predictions from the image-based novel view synthesis algorithm, Zero123 (i.e., comparing column 1 with columns 2 and 3). The middle rows highlight faulty generations with Janus issues, while the remaining rows showcase multi-view consistent generations.

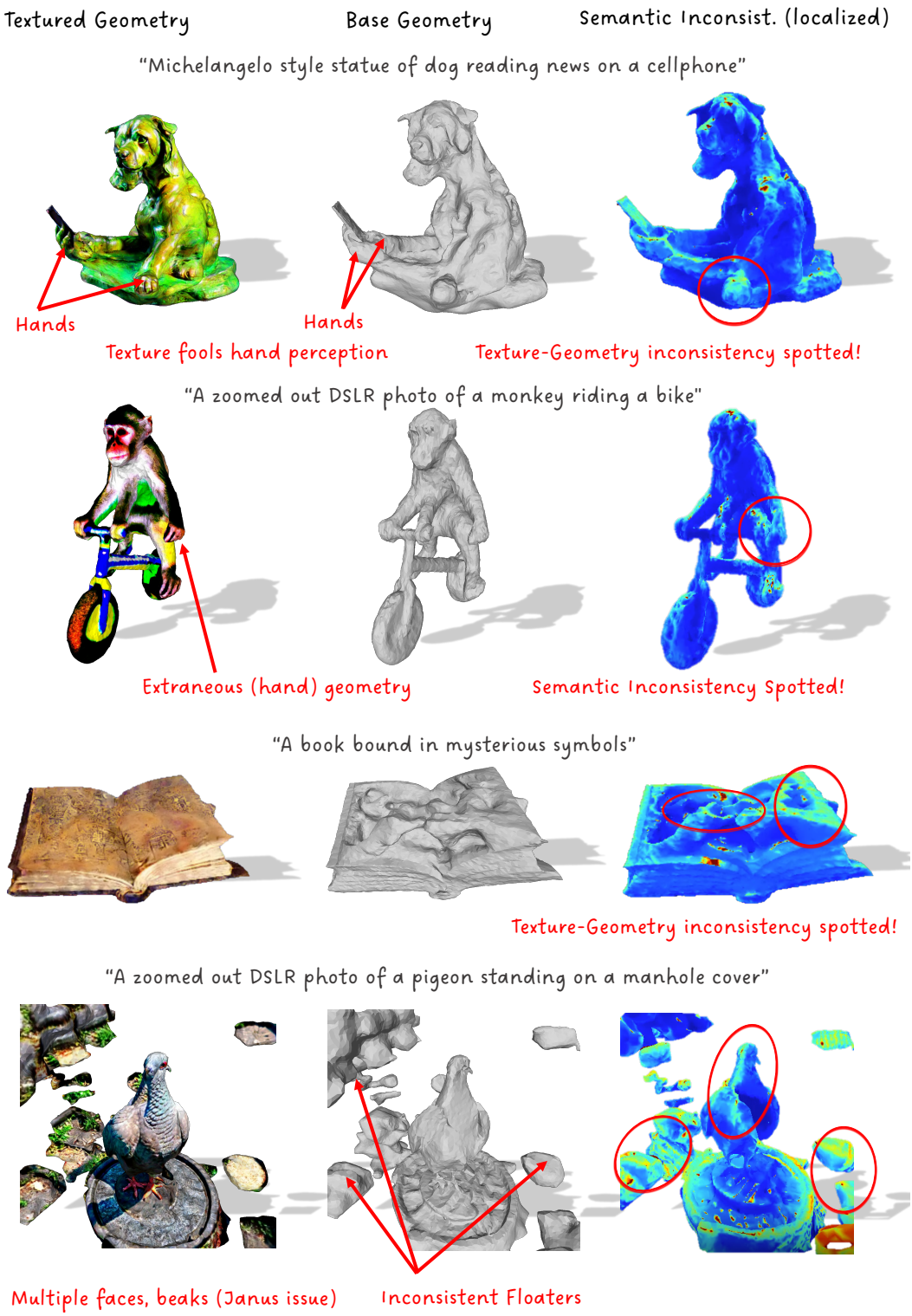
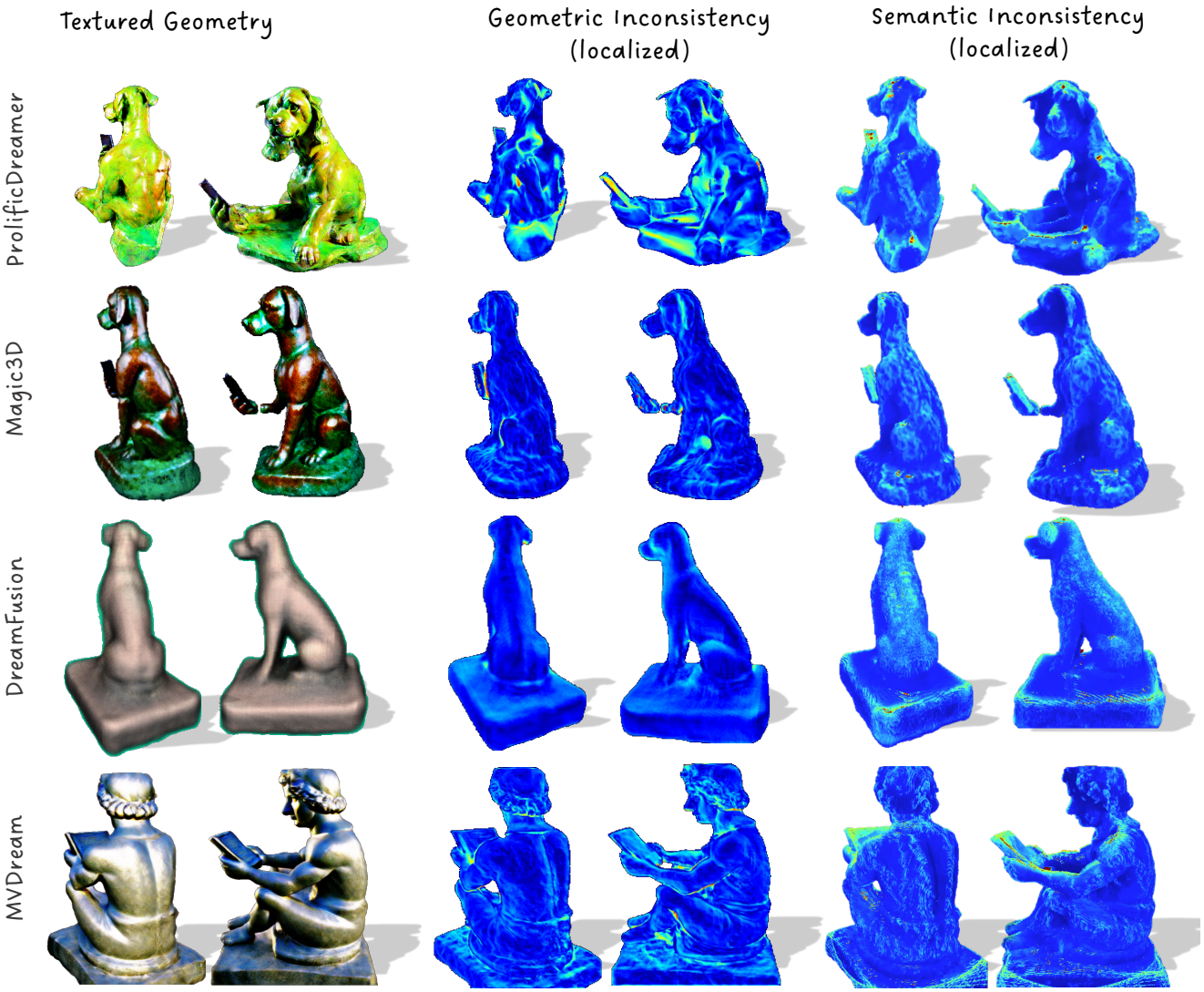


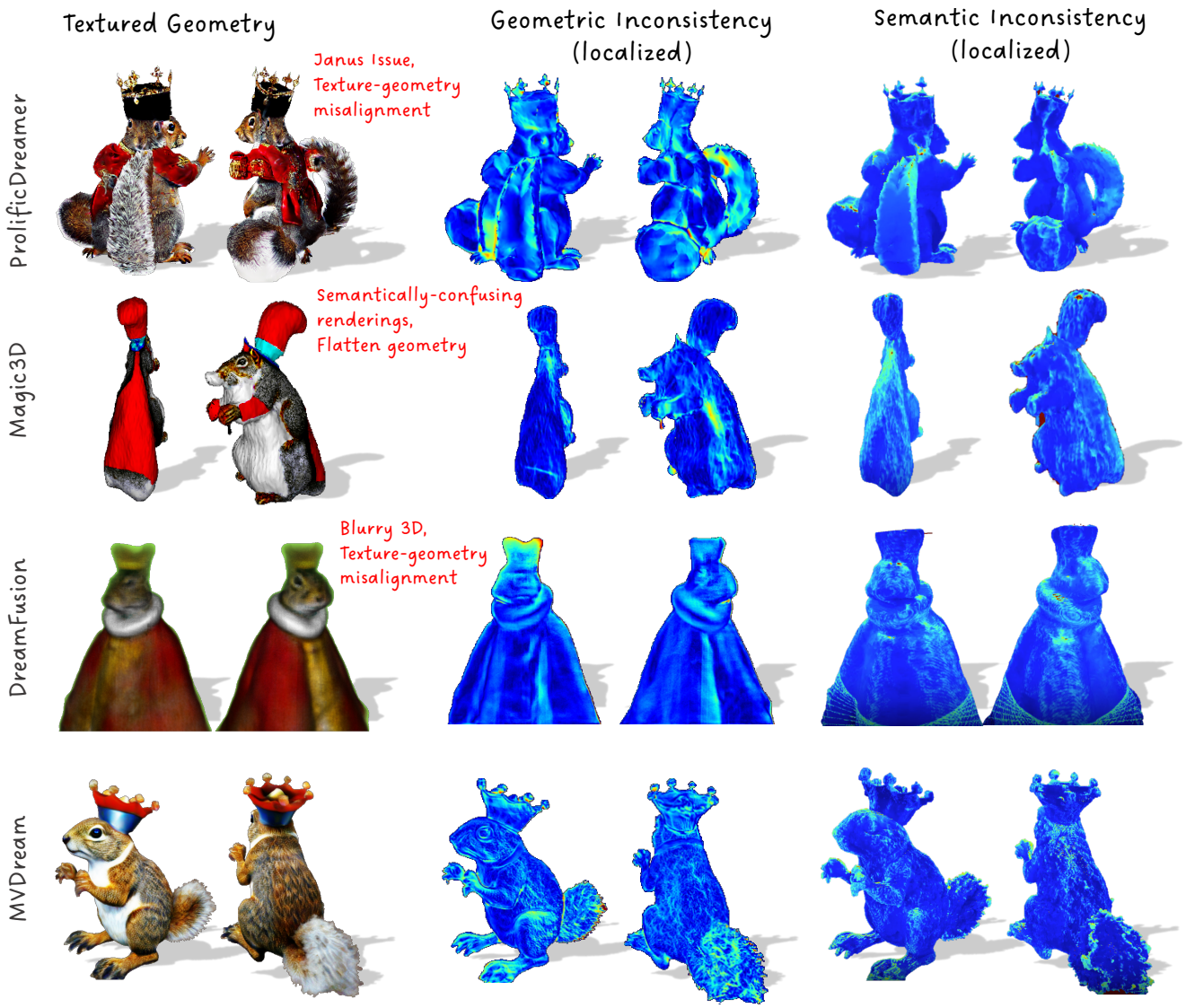
Figure 19. **Semantic Consistency** leverages the DinoV2 foundational model to measure the multi-view semantic consistency of each mesh vertex. We showcase various scenarios that could lead to multi-view semantic confusion, such as Janus issues, extraneous geometry, incorrect texture-geometry alignment, and generated floaters. Incorrect texture-geometry alignment will lead to incorrect back-projection of 2D Dino features onto 3D points, leading to high inconsistency.



"Michelangelo style statue of dog reading news on a cellphone"

Approach	Geometric Consistency	Semantic Consistency	Structural Consistency	Aesthetics	Text-3D Alignment
ProlificDreamer	83.1	75.8	79.8	56.5	**66.7**
Magic3D	97.7	71.95	**81.5**	53.5	**66.7**
DreamFusion	**98.0**	72.0	75.9	12.0	33.3
MVDream	91.9	**80.0**	79.4	**88.1**	16.7

Figure 20. **Text-to-3D Generation Comparison**: Magic3D and DreamFusion generate geometrically consistent but overly smooth and simpler geometries that lack aesthetic appeal. MVDream fails miserably to align with this particular prompt, while ProlificDreamer has noticeable localized artifacts in both geometric and semantic inconsistency maps.



"A squirrel dressed like Henry VIII king of England"

Approach	Geometric Consistency	Semantic Consistency	Structural Consistency	Aesthetics	Text-3D Alignment
ProlificDreamer	82.2	75.0	81.9	65.1	**66.7**
Magic3D	**97.3**	55.2	79.9	40.3	**66.7**
DreamFusion	94.1	74.5	**84.8**	1.36	**66.7**
MVDream	87.1	**80.0**	84.5	**84.1**	**66.7**

Figure 21. **Text-to-3D Generation Comparison**: Magic3D and DreamFusion generate geometrically consistent but overly smooth and simpler geometries that lack aesthetic appeal. Magic3D fails miserably on the semantic consistency metric for this prompt due to semantically confusing renderings from certain viewpoint. Both MVDream and ProlificDreamer exhibit noticeable artifacts in the geometric consistency map for this prompt – ProlificDreamer shows issues with texture-geometry alignment, while MVDream suffers from noisy generation.