Camera Control for Text-to-Image Generation via Learning Viewpoint Tokens

Supplementary Material

A. Code and Training Details

Our project is implemented using Python and PyTorch. We build much of the implementation upon the source code released by Harmon [42]. Training follows a cosine-annealed schedule with 1% linear warmup and gradient clipping (norm 1.0). We use an MLP of 3 layers with a hidden dimension of 1024 and an output dimension the same as the token embeddings.

B. More Nano Banana Results

Figure 11 shows more results from Nano Banana with different prompts we tried. We ask Gemini to describe the camera position of the 3D rendering to generate the first two camera prompts. We write the third camera prompt.

C. Training Dataset

Camera Settings We use a focal length of 35mm and Blender's default sensor size of 36 mm, resulting in an FOV of 54.4.

Object selection and normalization. We only include objects with semantically unambiguous front-facing orientations (e.g., the front of a vehicle, the face of an animal, the interactive side of furniture) and normalize the scale of each object to fit in a square bounding box of side length 1.

Rendered Dataset. The full set of 3,111 objects is rendered at 120 random viewpoints each against transparent backgrounds, providing dense viewpoint coverage (373,320 total images). We generate captions for all objects to enable text-conditioned generation.

Photorealistic Augmented Dataset. From the 3,111 objects, we select 800 diverse, highest-quality assets for photorealistic augmentation. Each object is rendered from 20 random viewpoints and processed through Gemini 2.5 Flash Image [9] model with the rendered image and an image editing prompt: *"Using the provided image, maintain the {object_name}'s location, pose, and head/gaze direction; remove all 3D rendering cues, polygon edges, and flat surfaces; transform the {object_name} into a new photorealistic {object_name} with the following NEW features: {desc_text}; and inpaint the transparent background with {background} so that the {object_name} appears organically integrated into the scene with correct relative size, lighting, shadows, atmospheric perspective, and natural interaction with the environment."*

We generate 3-5 detailed object descriptions per asset and curate 30 background prompts categorized by context (on land, on water, in air). During augmentation, we randomly sample object-background combinations to produce diverse, realistic appearances with varied environments. We filter the results to remove images with incorrect object pose, prompt misalignment (e.g., object scale incorrect for scene depth, background angle mismatched with object viewpoint), or physical implausibilities (e.g., floating objects), yielding 6,559 high-quality augmented images. Figure 14 shows examples of failures. Figure 15 shows more examples of rendered training images and training images augmented to include backgrounds. Table 8 shows the captions for the images.

Captions. For the rendered images, we use the captions generated for the 3,111 objects as the text prompt, appended with a viewpoint token. For the photorealistic augmented images, we use the combination of the detailed object description and the background augmentation prompts as the text prompt, appended with a viewpoint token. The detailed descriptions and the background augmentation prompts are the same as the ones used for the Gemini 2.5 Flash Image when creating the augmented image.

D. Testing Dataset

For evaluation of viewpoint accuracy and CLIP similarity, we use the 11 test objects (easy set) from Compass Control [27] and introduce 26 additional objects (**diverse set**) spanning broader categories: common animals (dog, cat, horse, cow, rabbit), rare animals (okapi, red panda, shoebill), vehicles (car, motorcycle, fighter jet, helicopter, buggy, snowmobile, gundam), furniture (chair), people (girl, woman, boy, man, elderly, Santa Claus, skeleton), and mythical creatures (phoenix, unicorn, mermaid). Notably, 11 of the additional objects (okapi, red panda, shoebill, buggy, snowmobile, gundam, Santa Claus, skeleton, phoenix, unicorn, mermaid) do not appear in our training data, testing generalization to unseen categories. For the diverse set (26 objects), we generate three descriptive phrases to test fine-grained prompt following. By combining 37 objects and background prompts, we have 555 unique prompt-object pairs. For each combination, we sample 10 random viewpoints, totaling 5,550 test samples.

E. More Examples of Overfitting by Compass Control

Figure 12 shows more examples of Compass Control overfitting to its training objects. For example, when it is asked to generate Santa Claus, it generates a shoe with Santa Claus appearance. Figure 13 further illustrates the distribution of these overfitting cases. Specifically, we exam-

A photo of an SUV on a winding country road with green fields, trees, and distant mountains under a sunny sky. angled diagonally to show its rear and left-side from a rear three-quarter view (approx. 220° azimuth), with the camera slightly below eye level (10 degrees). It occupies approximately 60% of the image width, positioned in the center slightly to the left.

A photo of an SUV on a winding country road with green fields, trees, and distant mountains under a sunny sky. angled diagonally to show its rear and left-side from a rear three-quarter view with the camera slightly below eye level. It occupies approximately 60% of the image width, positioned in the center slightly to the left.

A photo of an SUV on a winding country road with green fields, trees, and distant mountains under a sunny sky. Taken by a Camera 140 degrees to the left of the front view. With the SUV in the center slightly to the left, taking around 1/2 of the image.

Figure 11. **More Nano Banana results** with different camera prompts [9].

Table 7. **Viewpoint regressor accuracy** across the test split of four datasets. We report mean and median angular errors (degrees) and radius error (normalized).

Dataset	Azimuth↓		Elevation↓		Radius↓		Yaw↓		Pitch↓	
	Mean	Median	Mean	Median	Mean	Median	Mean	Median	Mean	Median
ControlNetPlus Augmentation	4.16	2.12	2.99	2.17	0.043	0.031	0.498	0.380	0.512	0.390
Rendered Dataset	2.65	1.65	2.25	1.64	0.031	0.022	0.434	0.361	0.430	0.350
Photorealistic Dataset	11.14	4.41	5.40	4.24	0.084	0.061	0.911	0.635	0.958	0.684
Compass Control Training Set	9.54	5.14	–	–	–	–	–	–	–	–

ine Compass Control's outputs for the Santa Claus, rabbit, and dolphin prompts in our test set and identify the mismatches—cases where the generated object does not match the prompt. In these failures, Compass Control often produces shapes resembling objects from its training set (lions, ostriches, teddy bears, shoes, and sofas), instead of the object named in the test prompt. As a comparison, our results follow the prompts for both categories included in our training set (e.g., rabbit and dolphin) and novel categories not included in our training set (e.g., Santa Claus).

F. Viewpoint Regressor

The regressor we use in evaluation is built on a pretrained ResNet-34 [11] backbone appended with three linear layers with ReLU [1] activation. The regressor outputs a 6-dimensional vector representing the viewpoint: $[\sin(\theta_{\text{az}}), \cos(\theta_{\text{az}}), \theta_{\text{el}}, r, \theta_{\text{pitch}}, \theta_{\text{yaw}}] \in \mathbb{R}^6$. We normalize the $[\sin(\theta_{\text{az}}), \cos(\theta_{\text{az}})]$ component to have a norm of 1. We train the network to estimate the pose of objects using a range of data with known poses generated by (i) ControlNetPlus [45] provided Canny edge maps of the rendered 37 testing objects (ii) rendered dataset (iii) photorealistic augmented dataset, and (iv) Compass Control training dataset.

Figure 12. **Compass Control vs. ours.** Each example shows three images: **Left:** 3D ground truth rendering, **Middle:** Compass Control [27], **Right:** Our method. The comparison demonstrates our model's improved viewpoint control and generalization compared to Compass Control.

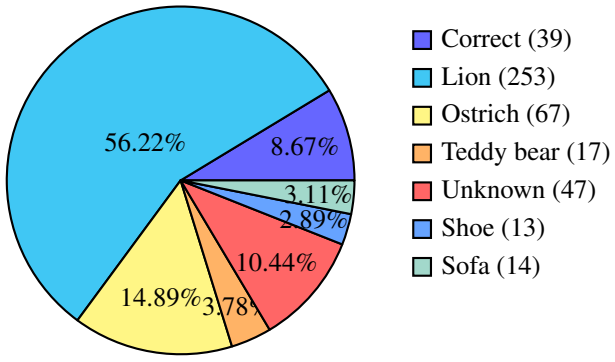

Figure 13. **Compass Control overfitting distribution.** Rendered images for novel test objects categorized as "correct", similar to a training object, or "unknown" (150 images each for Santa Claus, rabbit, and dolphin).

For the Compass Control training dataset, we only have the annotation for the θ_{az}; therefore, we only backpropagate loss on the $[\sin(\theta_{\text{az}}), \cos(\theta_{\text{az}})]$ output. We hold out 10% of each subset for validation and measure azimuth estimation errors of $4.16°$ on images of type (i), $2.64°$ for type (ii), $11.14°$ for type (iii), and $9.53°$ for type (iv). See Tab. 7 for more details.

(a) Incorrect scale (b) Viewpoint mismatch (c) Object floating

Figure 14. **Excluded augmented images.** (a) Object scale incorrect for scene depth, (b) background viewpoint mismatch, (c) object floating without grounding.

G. More Qualitative Examples

Figures 16 and 17 show examples of different camera parameters with the prompt "A photo of a red sports car in a national reserve in a snowy landscape". Figures 18 and 19 present additional qualitative results on object categories not included in our training data. Figure 20 provides further examples for categories that are part of our training set.

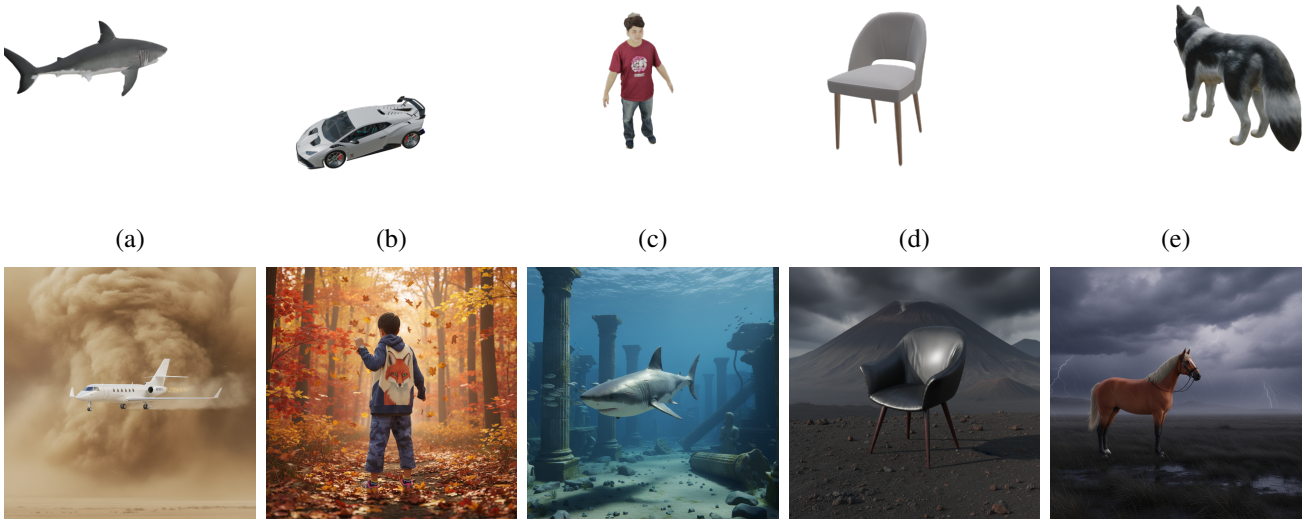

Figure 15. **Training dataset examples.** Top row: rendered dataset. Bottom row: photorealistic augmented dataset.

Table 8. **Text prompts** for images in Figure 15, listed left-to-right, top-to-bottom.

ID	Prompt
(a)	A 3D model of a great white shark with dark gray back. The shark has white underside, pointed snout, gill slits, and powerful tail fin.
(b)	A 3D rendered white sports car with large rear wing. The car features aggressive aerodynamics, air intakes, and track-focused modifications.
(c)	A boy in burgundy t-shirt with graphic and gray jeans. He has dark hair and wears dark sneakers, standing casually.
(d)	A modern dining chair with beige fabric and curved open backrest. The chair has wooden legs and a distinctive circular cutout in the backrest.
(e)	A 3D model of a husky with black and white fur. The dog has pointed ears and curled tail with blue eyes, standing.
(f)	Object: A business jet with private registration, rear-mounted engines, and standard wing tips. Background: A sky filled with swirling dust during a distant sandstorm.
(g)	Object: A boy with brown skin and graphic print t-shirt underneath. Background: An autumn forest with colorful, falling leaves.
(h)	Object: A shark with dorsal fin, cream underbelly, and robust body. Background: A submerged ancient city ruin.
(i)	Object: An armchair with padded armrests, low profile back, and tapered wooden legs. Background: A volcanic landscape with dark ash.
(j)	Object: A horse wearing a bridle with dark legs. Background: A plain under a dramatic, stormy sky.

Figure 16. **Generation results across azimuth and elevation.** Columns represent azimuth angles (10° to 325°) and rows represent elevation angles (0° to 45°). All examples use a fixed camera radius of 1.5 with pitch = 0° and yaw = 0°. All examples use the same seed 42.

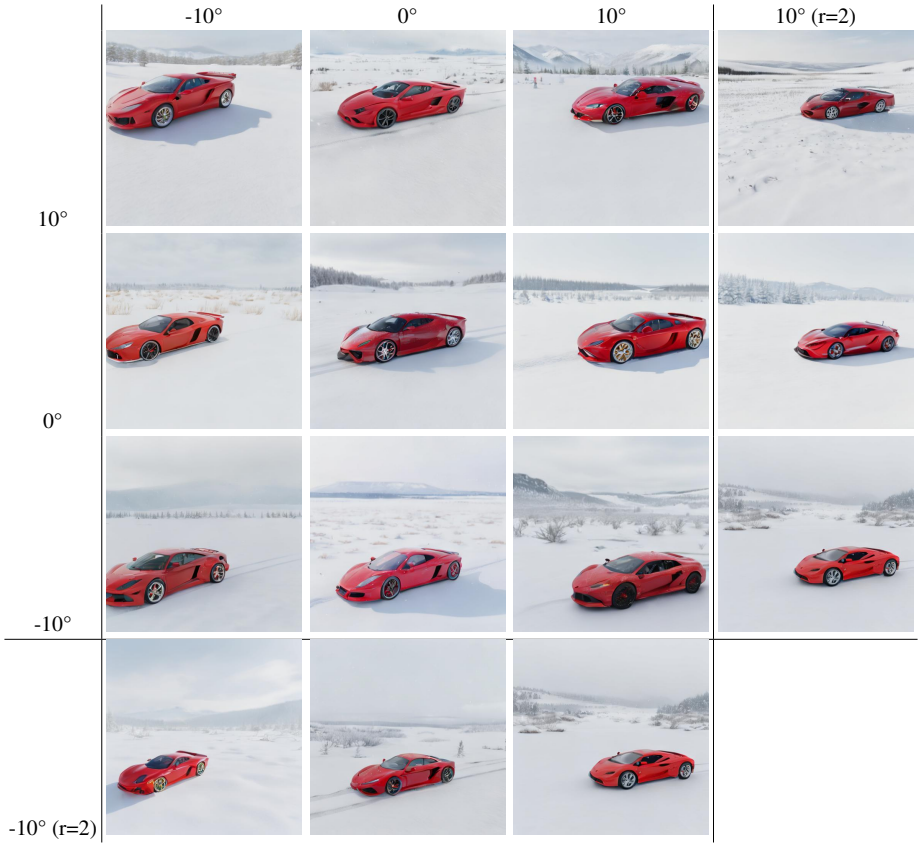

Figure 17. **Generation results across pitch, yaw, and radius.** Main 3×3 grid shows radius=1.5, extra row and column show radius=2.0. All examples use a fixed azimuth = 55° and elevation = 15°. All examples use the same seed 42.

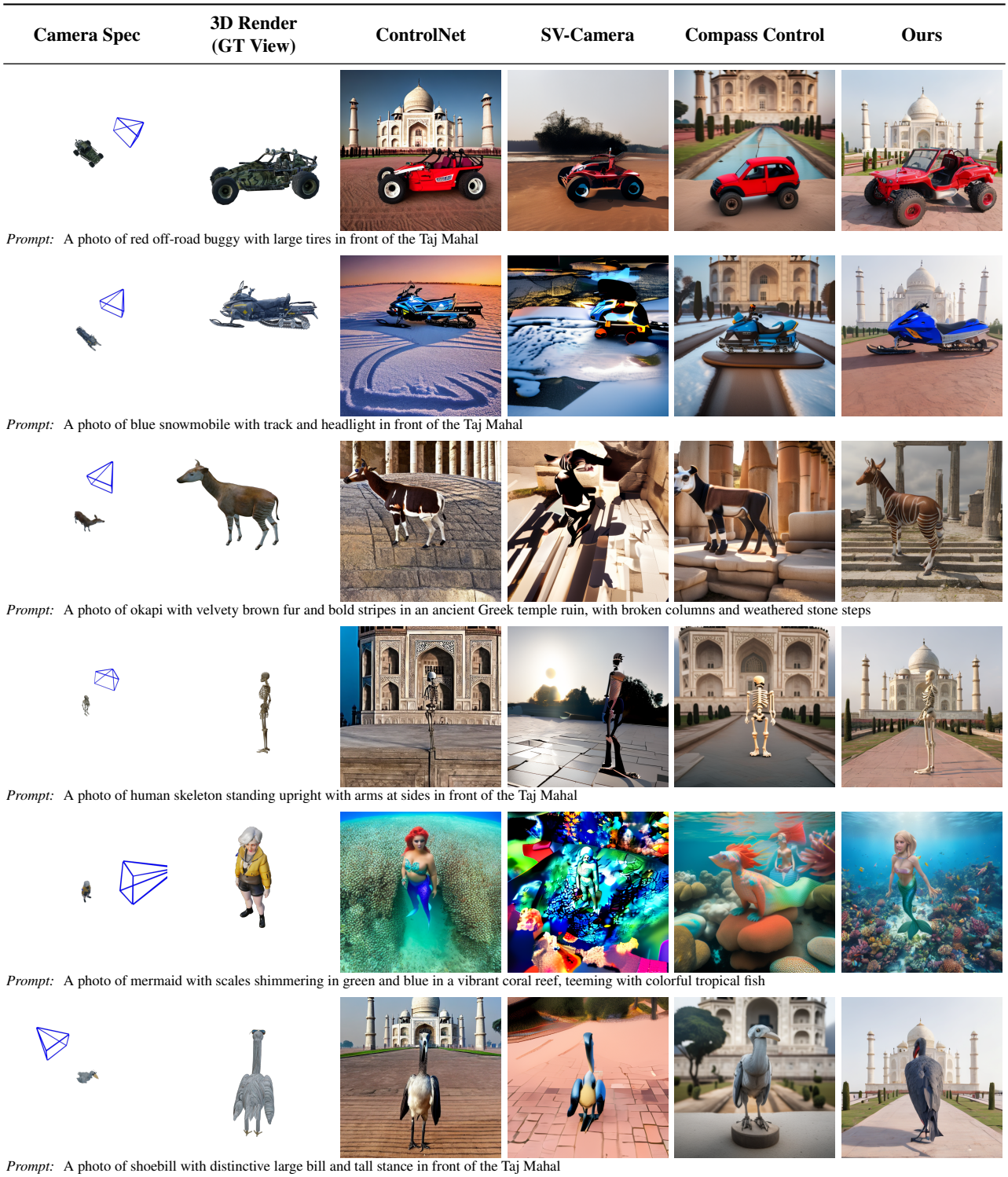

Figure 18. **More qualitative comparisons (Part 1).** Each row pair shows images (top) and the corresponding prompt (bottom). The first two columns display the camera frustum (3D illustration) and a ground truth 3D rendering from a similar 3D object to the prompt. The remaining columns show results from different methods: ControlNet [50], Stable-Virtual-Camera [52], Compass Control [27], and our method. The object types in the prompts are not included in our training dataset.

Figure 19. **More qualitative comparisons (Part 2).** Each row pair shows images (top) and the corresponding prompt (bottom). The first two columns display the camera frustum (3D illustration) and a ground truth 3D rendering from a similar 3D object to the prompt. The remaining columns show results from different methods: ControlNet [50], Stable-Virtual-Camera [52], Compass Control [27], and our method. The object types in the prompts are not included in our training dataset.

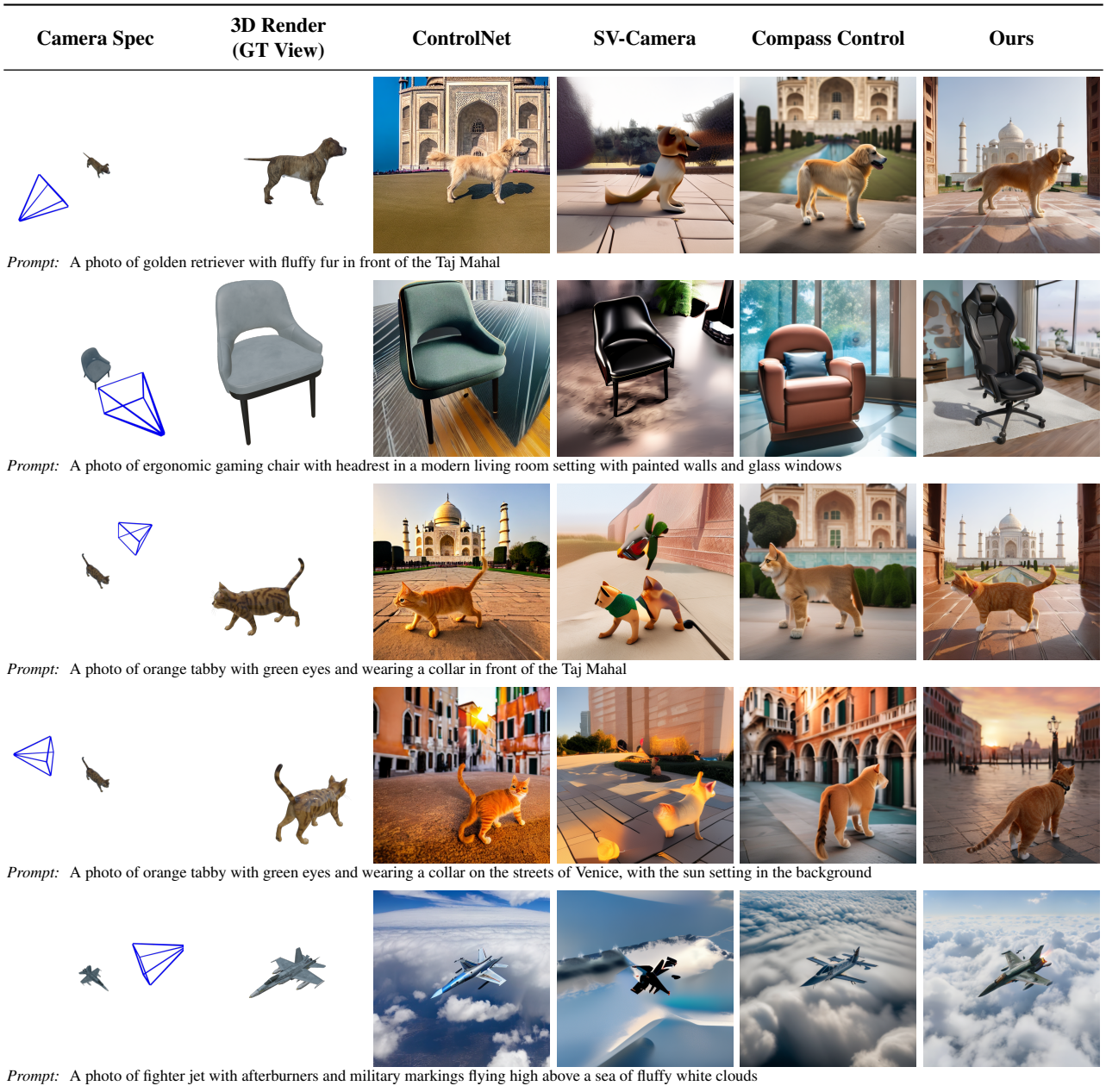

Figure 20. **More qualitative comparisons (Part 3).** Each row pair shows images (top) and the corresponding prompt (bottom). The first two columns display the camera frustum (3D illustration) and a ground truth 3D rendering from a similar 3D object to the prompt. The remaining columns show results from different methods: ControlNet [50], Stable-Virtual-Camera [52], Compass Control [27], and our method. The object types in the prompts are included in our training dataset.