Data-Driven Loss Functions for Inference-Time Optimization in Text-to-Image

Supplementary Material

A. Comparison with Additional Methods

A.1. Comparison with COMPASS

While our method demonstrates strong performance across various models and benchmarks, fine-tuning methods like COMPASS [55] can achieve higher accuracy in specific in-distribution settings, though often at the cost of degrading other alignment capabilities. For instance, on the GenEval benchmark with SD 1.4, COMPASS achieves a spatial accuracy of 43–46% compared to our 36% (Table 1). We attribute this to the nature of fine-tuning, which modifies model weights and can lead to specialization on the training distribution. A more fine-grained analysis suggests this specialization may be narrow. On T2I-CompBench, which uses stricter geometric criteria, our method outperforms COMPASS for all models (e.g., 0.278 vs. 0.254 on SD 1.4); additionally, Table 6 shows that COMPASS's advantage on SD 1.4 is largely driven by a significant performance gap on the "left of" relation (59.2% vs. 40.8%), while performance on other relations is more comparable.

Importantly, COMPASS's specialization appears to come at a cost to generalization, particularly in complex, out-of-distribution scenarios (Tables 2, 3). This is most evident when handling prompts with multiple spatial relations—a compositional challenge not seen during training. As shown in Table 3, our method's performance degrades gracefully, whereas COMPASS's accuracy drops sharply. For instance, on prompts with three objects and two relations involving OOD categories, our method achieves 28% accuracy compared to COMPASS's 12%. The performance gap widens with complexity: for four objects and three relations with OOD categories, our method maintains 19% accuracy, while COMPASS drops to just 5%. This highlights the advantage of our test-time optimization in maintaining compositional structure and degrading gracefully under complexity.

This trade-off extends to other capabilities beyond spatial reasoning. Our method better preserves the original aesthetics and style of the base model, as demonstrated in the qualitative comparison in Fig. 8. We quantify this observation in Table 5, which shows that images generated with our method have a higher visual similarity to the base model's generations using both DINO [39] and CLIP [43] similarities (e.g., on FLUX.1-dev: 0.863 vs. 0.775 for DINO). Furthermore, as shown in Table 7, COMPASS's improvements in spatial control are accompanied by degradation in other areas: on SD2.1, color accuracy drops from 0.85 to 0.71, and counting accuracy falls from 0.44 to 0.20. In contrast, our test-time approach is applied selectively only when spatial relations are present in the prompt, thereby preserving the base model's performance on these dimensions.

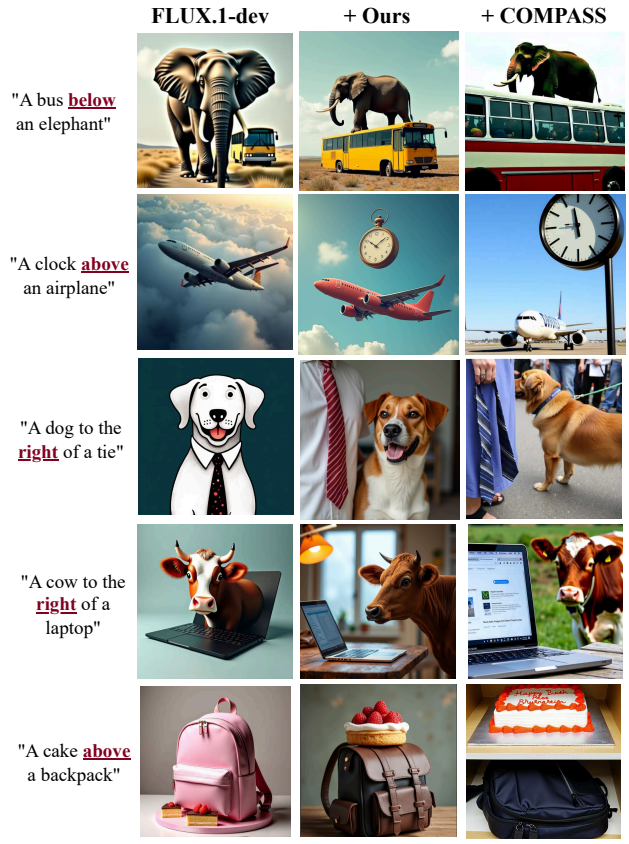

Figure 8. Comparing COMPASS [55] to our method, using FLUX.1-dev [28] as the base model.

Base Model	Methods	DINO ↑	CLIP ↑
Flux-dev			
	Base vs. COMPASS	0.7751	0.7721
	Base vs. Ours	**0.8625**	**0.8767**
SD 1.4			
	Base vs. COMPASS	0.7135	0.7530
	Base vs. Ours	**0.7360**	**0.8012**

Table 5. Our approach generates images that are closer to the base model. In Figure 8 we see that qualitatively style and aesthetics are closer to the base model.

A brown dog to the right of a fluffy teddy bear in a living room filled with wooden shelves, a patterned rug on the floor, soft cushions on the sofa, and warm light streaming in from a wide window

A blue backpack below a frosted cake in a dining room where a long wooden table is set with plates and glasses, curtains sway gently beside the window, and a chandelier hangs overhead casting soft light

A silver laptop to the left of a spotted cow in a farmyard with a wooden fence enclosing the animals, a red barn standing tall nearby, hay scattered on the ground, and wide fields opening up behind the scene

Figure 9. **Qualitative results with dense prompts.** Our method preserves the intended spatial relations even when objects are described with attributes and embedded in detailed scene contexts.

	FLUX.1-dev		SD 1.4	
Relation	**Ours**	**COMPASS**	**Ours**	**COMPASS**
above	61.5%	63.5%	30.8%	39.4%
below	67.0%	54.5%	33.9%	38.4%
left of	60.5%	56.6%	40.8%	**59.2%**
right of	54.6%	67.6%	40.7%	40.7%

Table 6. Per-relation accuracy comparison. While FLUX.1-dev results are comparable between methods, in SD 1.4 COMPASS shows notably higher accuracy for "left of" (59%) compared to other relations.

Model	**Position ↑**	**Count ↑**	**Color ↑**
Vanilla SD2.1	0.07	0.44	0.85
COMPASS	**0.51**	0.20	0.71
Learn-to-Steer (Ours)	**0.54**	0.44*	0.85*

Table 7. GenEval results comparing fine-tuning (COMPASS) vs. test-time optimization (Ours) on SD2.1. COMPASS results are from their paper. *Our method preserves the vanilla model's performance on non-spatial tasks since we only apply steering when spatial relations are present in the prompt.

A.2. Comparison with Layout-guided Methods

Table 8 compares *Learn-to-Steer* with layout-guided approaches, such as RAGD [10], 3DIS [11], and MIGC [58]. Results for these methods are sourced from RAGD [10], where GenEval's performance was evaluated using MLLM-generated layouts. Comparisons against MLLM-driven pipelines are inequitable due to their massive parameter counts and undisclosed training data.

B. Additional Ablation Studies

The inputs to the relation classifier are the cross-attention maps of two objects, alongside the current denoising timestep. We analyzed how different aggregation strategies for the attention maps affect performance. Specifically, we compared three approaches: **(1) Mean (Global):** Computing the mean across all layers and heads; **(2) Mean (Per-Layer):** Computing the mean across heads within each layer, then concatenating the layer outputs; and **(3) Full Concatenation:** Retaining all maps by concatenating all layers and heads into a single feature vector.

While Table 9 demonstrates that the "Mean (Global)" strategy achieves the highest alignment score (0.78), we find that it comes at a cost to image quality, consistently yielding over-constrained results with "dull" or neutral backgrounds (See Fig. 10). Similarly, the "Mean (Per-Layer)" strategy, which scores slightly lower than our method (0.69 vs. 0.70), also exhibits inconsistent visual quality, occasionally suffering from the same background degradation. In contrast, the "Full Concatenation" strategy provides the most balanced

Method	**G. Pos. ↑**
3DIS† [11]	0.72
MIGC† [58]	0.76
RAGD† [10]	**0.80**
Learn-to-Steer (Ours)	0.61

Table 8. Comparison of spatial relation scores from the GenEval benchmark (*G. Pos.*) against layout-guided methods. † results are reported from [10]. Note that unlike *Learn-to-Steer*, these methods rely on explicit layout inputs.

Aggregation Variant	**G. Pos. ↑**
Full Concatenation (Ours)	0.7
Mean: Per-Layer	0.69
Mean: Global (All layers/heads)	**0.78**

Table 9. Ablation results for different attention aggregation strategies on FLUX.1-dev (GenEval validation set).

Figure 10. **Impact of Aggregation on Visual Aesthetics.** Comparing our chosen method (Full Concatenation) against the Mean Aggregation baselines. While Mean Aggregation forces the objects into position, it frequently results in loss of background detail and vividness. Our method preserves the visual richness of the base model.

setup, offering strong adherence to the spatial prompt while maintaining high visual aesthetics. Therefore, we decided to use this strategy for our final method.

C. Additional Implementation Details

C.1. Adapting FOR's handcrafted spatial loss to Flux-based models

We adapted FOR's [26] handcrafted spatial loss as follows: For each pair of objects, we first aggregated all cross-attention maps across all layers and heads into a single attention map per object. Subsequently, we applied the manual spatial loss function on the two aggregated maps in exactly the same manner as in the official implementation. To ensure a fair comparison, we ran a hyperparameter search over the scale factor of the step size used during test-time optimization to find the optimal setup.

C.2. Benchmarks

To ensure a fair comparison, we normalize all relation names to {*"above", "below", "to the left of", "to the right of"*} across benchmarks. All baselines we reproduced use this normalized set. We will release normalized test sets for both benchmarks.

C.3. OOD Evaluation Details

Our out-of-distribution (OOD) evaluation tests generalization on 200 prompts across 16 object categories unseen by both our method and COMPASS [55]. We identified 6 categories from T2I-CompBench vocabulary that were out-of-distribution for both methods, and used ChatGPT to generate 10 additional diverse categories, ensuring no training data overlap.

C.4. Extending GenEval to an Open-Set of Categories and Multiple Relations

The current GenEval benchmark is constrained for only recognizing the 80 COCO categories, and can only verify a single pair of relations. To facilitate an open set of object categories, we replaced the object detector with CLIPSeg [35], a zero-shot object segmentation approach. To evaluate multiple relations, we generated metadata files in GenEval's expected structure, where each prompt is paired with a list of required objects along with their pairwise relations, verify each relation, and indicate success only when all relations are correctly generated in an image. To verify that our automatic evaluation is reliable, we compared its results against manual evaluation conducted by one of the authors on a mixed set of in-distribution and OOD categories (Table 10). The scores align closely across all settings.

Objects	Relations	Manual Eval.	Automatic Eval.
3	2	0.29	0.28
4	3	0.12	0.12
5	3	0.07	0.10

Table 10. Comparison of automatic evaluation vs. manual evaluation of multiple spatial relations.

C.5. Object Categories for Multi-Relation Evaluation

For the multi-relation evaluation, similar to the single-relation case, we tested on both in-distribution and out-of-distribution (OOD) object categories. The specific objects used were:

In-distribution categories (16 objects): backpack, bowl, cat, skateboard, bottle, cake, tie, vase, sheep, toothbrush, teddy bear, suitcase, mouse, knife, chair, and umbrella.

Out-of-distribution categories (16 objects): teapot, corgi, furby, robot, turtle, rabbit, butterfly, key, pig, bee, tiger, hammer, monkey, razor, snake, and stapler.

C.6. Extending Our Method to Diagonal Spatial Relations

Since there is no labeled data for diagonal relations (top-left, top-right, bottom-left, bottom-right) readily available,

we generated training samples by inferring relations directly from GQA bounding boxes. For each image, we examined all object pairs and computed their relative offsets and angles. Pairs whose center displacement fell within diagonal sectors were labeled accordingly, while cases with large overlaps were discarded.

Using these labels, we constructed cross-attention training samples following the same procedure as for the four basic relations (above, below, left, right). For a given relation (e.g., *"top-left"*), we inverted the image twice: once with the correct relation and once with an incorrect relation sampled uniformly from the remaining options (e.g., *"bottom-left"*). This shows that our dual-image inversion strategy can naturally extend to more complex spatial relations such as diagonals.

We then trained a diagonal relation classifier in the same way as for the four basic relations, with one exception: as a quick prototype of the aggregation module, we computed the average across all cross-attention maps before forwarding the representation to the classification module.

C.7. Potential limitations of dual-inversion

Our dual inversion strategy has an underlying assumption that the inversion process succeeds, and generates well-behaving attention maps even for the incorrect prompt. While we did not experiment with inverting very complex prompts, inversion it is likely to be less stable in these cases. Another limitation arises when inverting images that contain multiple instances of the same object (2 cats and a dog). Such images and prompts are known to be less stable.

C.8. Dense Prompt Construction

To create dense prompts, we asked ChatGPT [38] to enhance base prompts from the GenEval dataset, and generate an enriched variant. Each variant preserved the original spatial relation, added an attribute to every object, placed the objects in a realistic location (e.g., kitchen, living room, park), and included a vivid description of the location with detailed context.

C.9. Architecture Details

First we process each set of attention maps individually, using a transformer based aggregation module that maps each set of maps $L \times H \times h \times w$ to one single $h \times w$ map per image, where L, H are the number of cross attention layers and heads and h, w are the attention map dimensions. Then we process these two maps jointly to predict the relation class.

Aggregation Module. For a given object, the aggregator receives all $L \times H$ attention maps of size $h \times w$. Each map is treated as a separate token in a sequence of length $L \cdot H$. We then project each token from $h \cdot w$ dimensions to a shared embedding dimension $d = 256$. This yields a sequence length $L \cdot H$ with a feature dimension of d, to which we add standard positional encodings [51] in $\mathbb{R}^d$ that signifies the spatial location. A timestep embedding $\phi(t) \in \mathbb{R}^d$ is also added to all tokens, in the input to every transformer layer, allowing the aggregator to adapt to the current denoising stage. A four-layer, four-head transformer then processes this sequence to exchange information across different layers and heads. To aggregate the transformer output from $\mathbb{R}^{L \cdot H, d}$ to $\mathbb{R}^d$, we use another four-layer, single-head transformer. For the second transformer, we learn a query vector in $\mathbb{R}^{1 \times d}$, while the key and value are projected from the output of the first transformer ($\mathbb{R}^{L \cdot H, d}$). Finally, a linear layer maps the output in $\mathbb{R}^d$ to $\mathbb{R}^{h \cdot w}$ reshaped to an aggregated map $\mathbf{A}_{\text{agg}} \in \mathbb{R}^{h \times w}$. The same aggregator (shared weights) processes both subject and object maps.

In practice, to process SD-based maps we use two-layer, two-head transformers instead of four-layer, four-head.

Classification Module. The two aggregated maps are stacked into a two-channel image, flattened, projected to $d = 256$ dimensions, tagged with standard positional encodings and passed through a four-layer eight-head Transformer. Finally, the transformer's output is concatenated with a low-dimension timestep embedding (16-d) and fed to an MLP that returns logits over the relation classes. The complete architecture is trained jointly end-to-end.

In practice, to process SD-based maps we use two-layer, four-head transformers in the classification module.

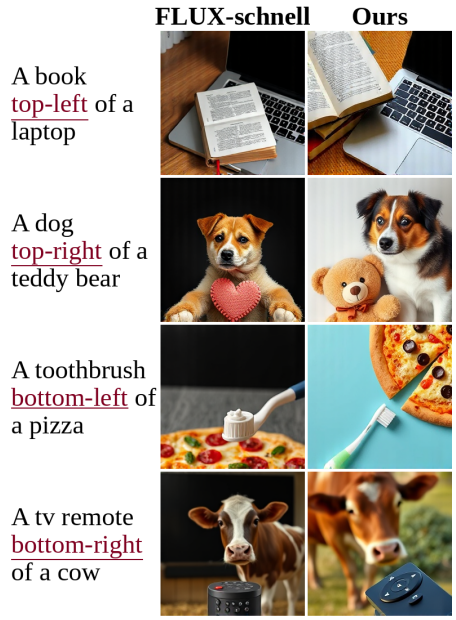

Figure 11. **Qualitative comparison of diagonal spatial relations.** The base model often misplaces objects or ignores the diagonal specification, while our method better adheres to the intended spatial layout.

C.10. Other details

Relation Extraction. Relation triplets are extracted from the prompt using SpaCy [21] a natural lang. parsing library, with some hard-coded exceptions, like "fire hydrant".

Classifier Training. For training the classifier we used a batch size of 64, with AdamW optimizer [34], a weight decay of 0.05, and a learning rate of $5 \cdot 10^{-5}$ for Flux-based models and 0.0001 for SD-based models. We used a ReduceOnPlateu scheduler with a factor of 0.5 and patience 5. We trained the classifier for a maximum of 200 epochs in Flux-based models and 150 for SD-based models, with early stopping patience of 10 epochs.

Test-Time Optimization and Inference. We optimize the first 50% of the denoising steps. In FLUX.1-schnell, we only optimize the initial noise. The step size α is 5 for SD2.1 and Flux-based models, and 7.5 for SD1.4 and number of optimization iterations per step are 15.

We use the guidance scales suggested by HuggingFace [24], specifically 3.5 for FLUX.1-dev, 0 for FLUX.1-schnell, and 7.5 for SD based models.

Image Resolution. We generate images at different resolutions depending on the base model: 256×256 for FLUX.1-schnell and 512×512 for FLUX.1-dev, SD2.1, and SD1.4.

Neutral Relation Class. We included a fifth, "neutral" class during training. This class captures non-directional relationships (described by "is") and acts as a background category. We do not use this neutral class during generation.

D. Additional Qualitative Results

Figure 14 provides comprehensive comparisons across all four base models on GenEval prompts, demonstrating consistent improvements in spatial accuracy. Figure 13 (Appendix), demonstrate how our method overcomes the three main failure modes: incorrect object placement, entity neglect, and object fusion.

We perform extensive comparisons on both GenEval (Figures 17, 18, 19, and 20) and T2I-CompBench (Figures 21, 22, 23, and 24).

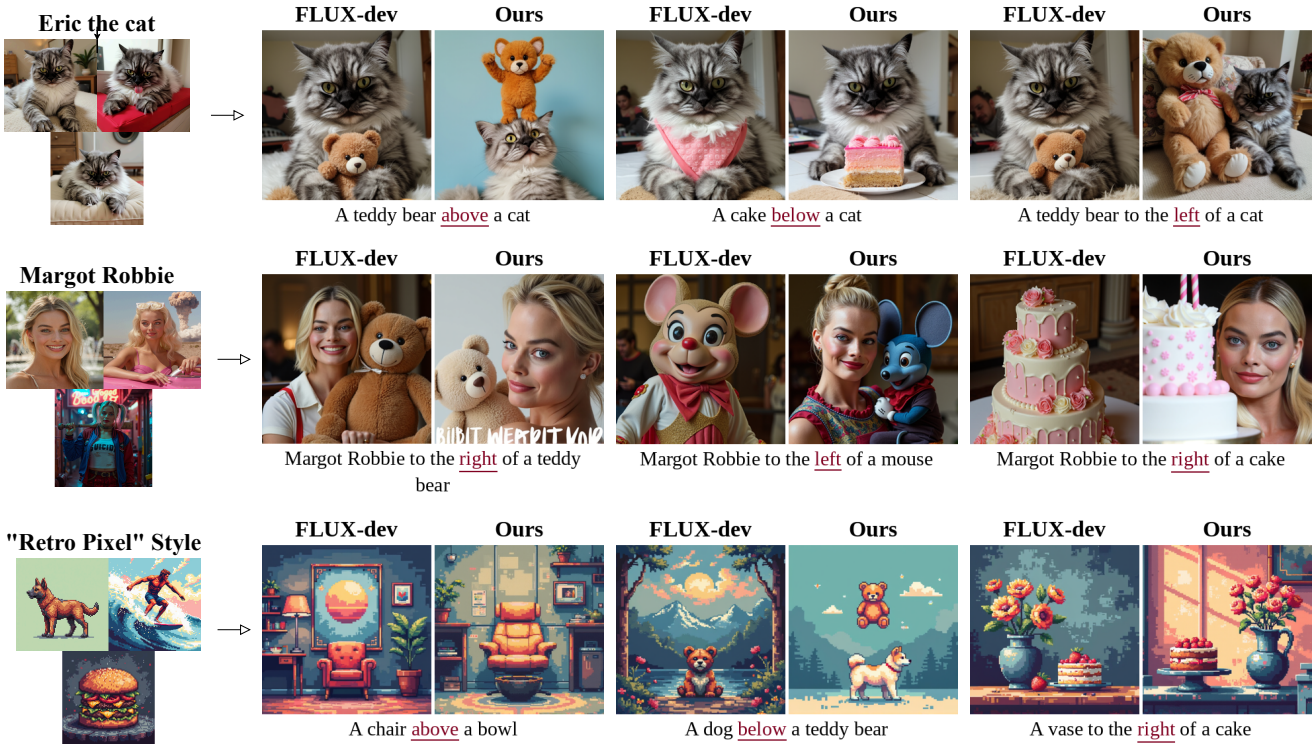

Figure 12. **More qualitative results of zero-shot transfer to personalized LoRAs of both subject and style.**

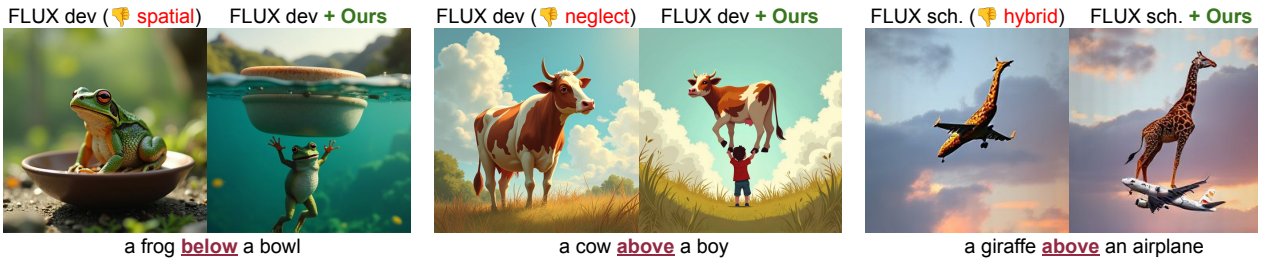

Figure 13. **Common base model failures.** Our method addresses common text-to-image generation failures, including incorrect object placement (left), object neglect (middle), and fused, chimeric hybrids (right).

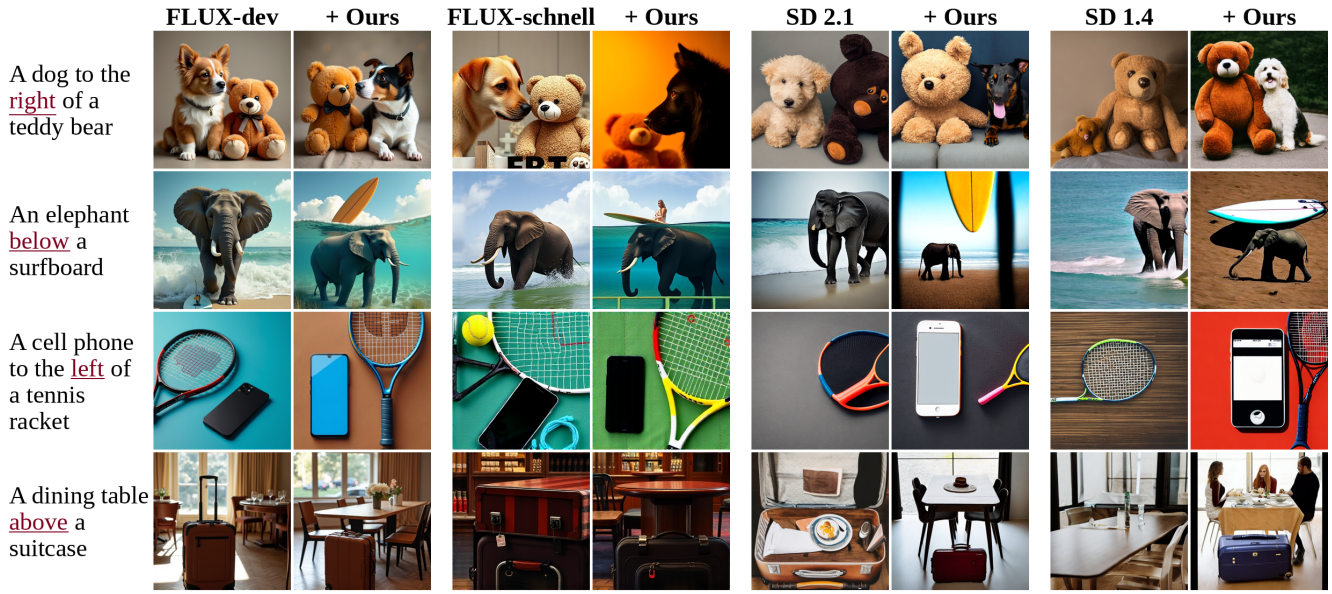

Figure 14. **Comparisons with all four base models**. Prompts were taken from GenEval [15]. Each pair uses the same seed.

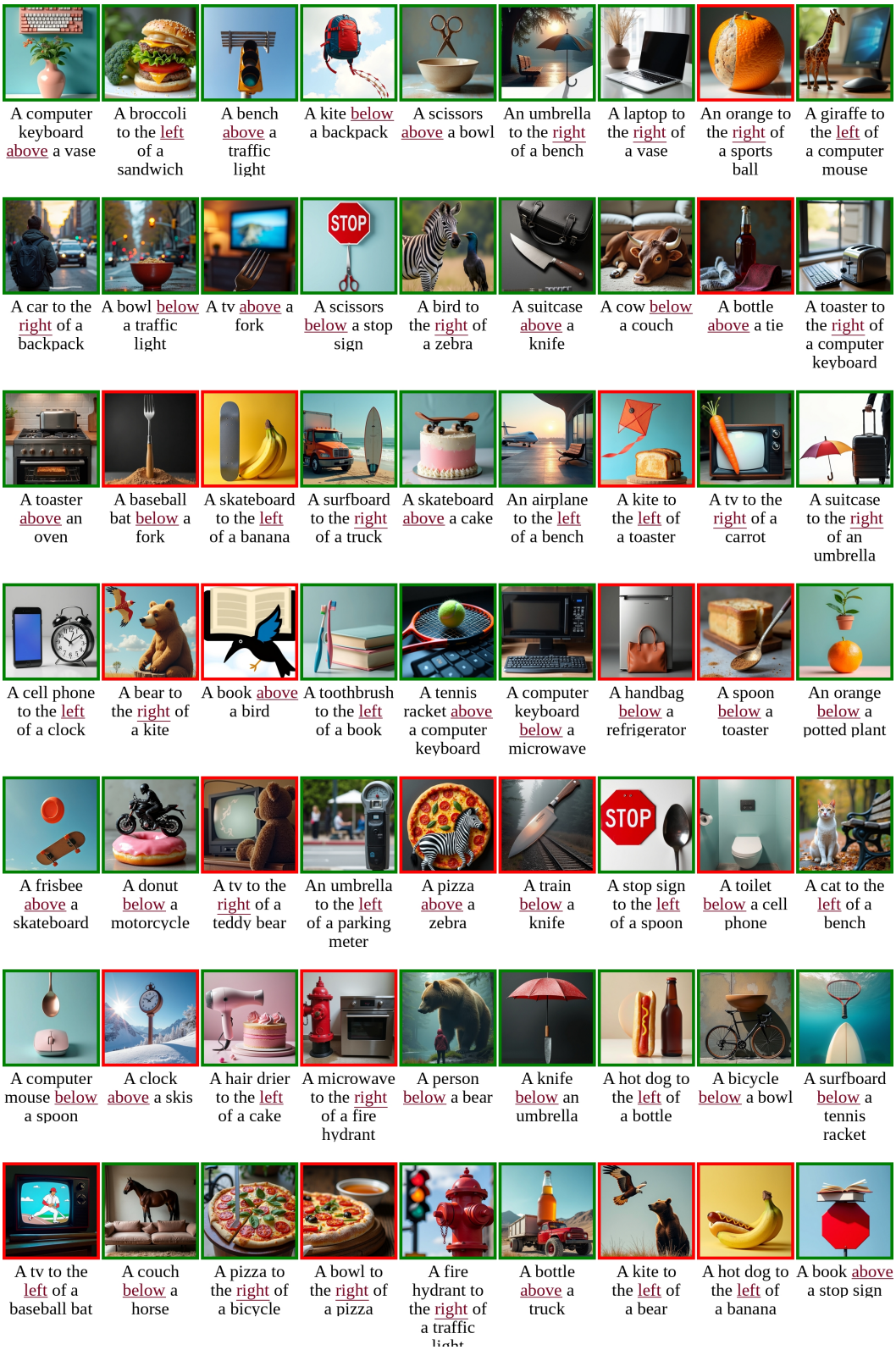

Figure 15. **Uncurated image generation results** from FLUX.1-dev [28] with *Learn-to-Steer*, using validation prompts from GenEval [15]. Green boxes mark spatially-aligned images (per GenEval), while red boxes mark misaligned ones.

Figure 16. **Handling Multiple Relations.** As opposed to the base model, our method is capable of generating complex scene structures that contain multiple relations in a single prompt.

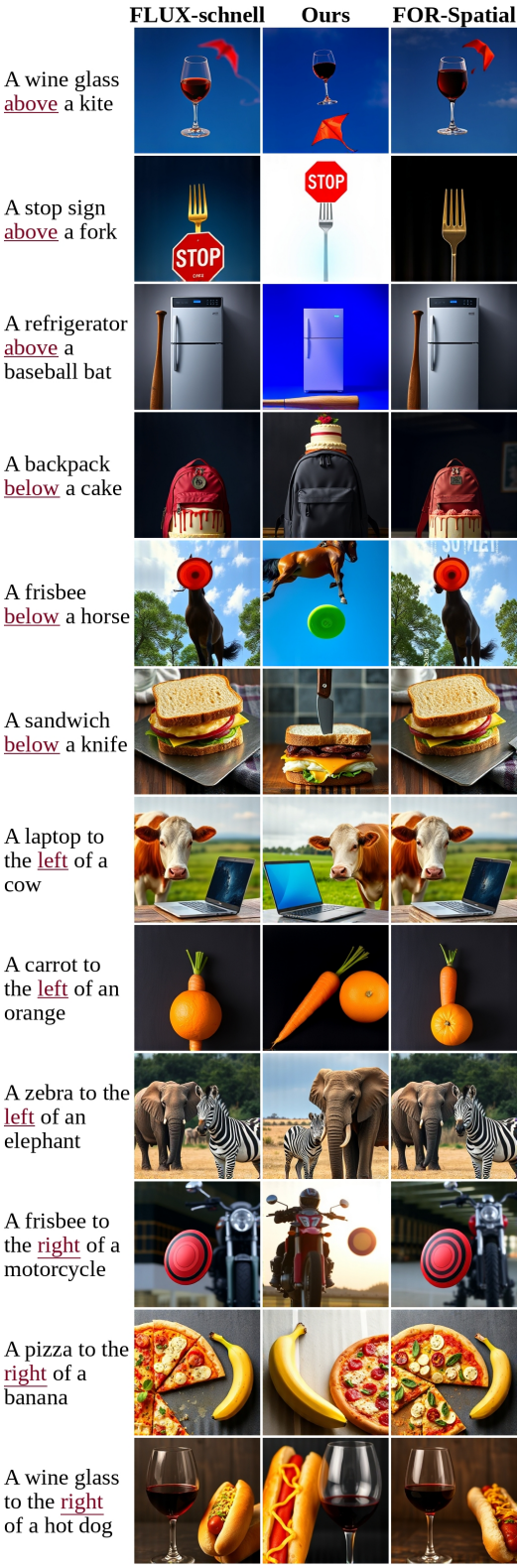

Figure 17. Qualitative comparison using FLUX.1-schnell [28] with prompts from the GenEval [15] benchmark. For each prompt, the same seed is used for all methods.

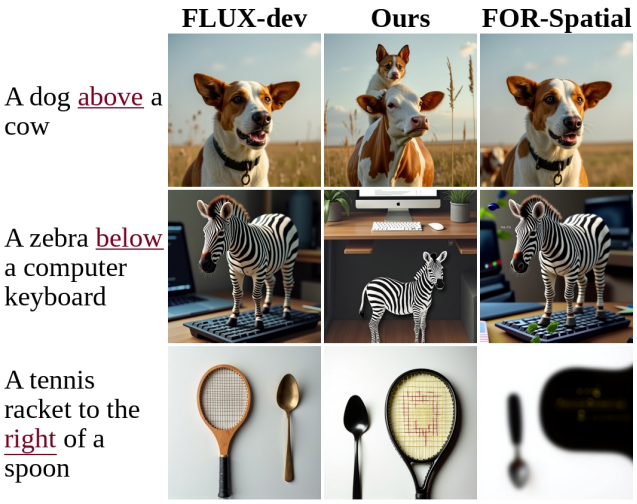

Figure 18. FLUX.1-dev on the GenEval benchmark

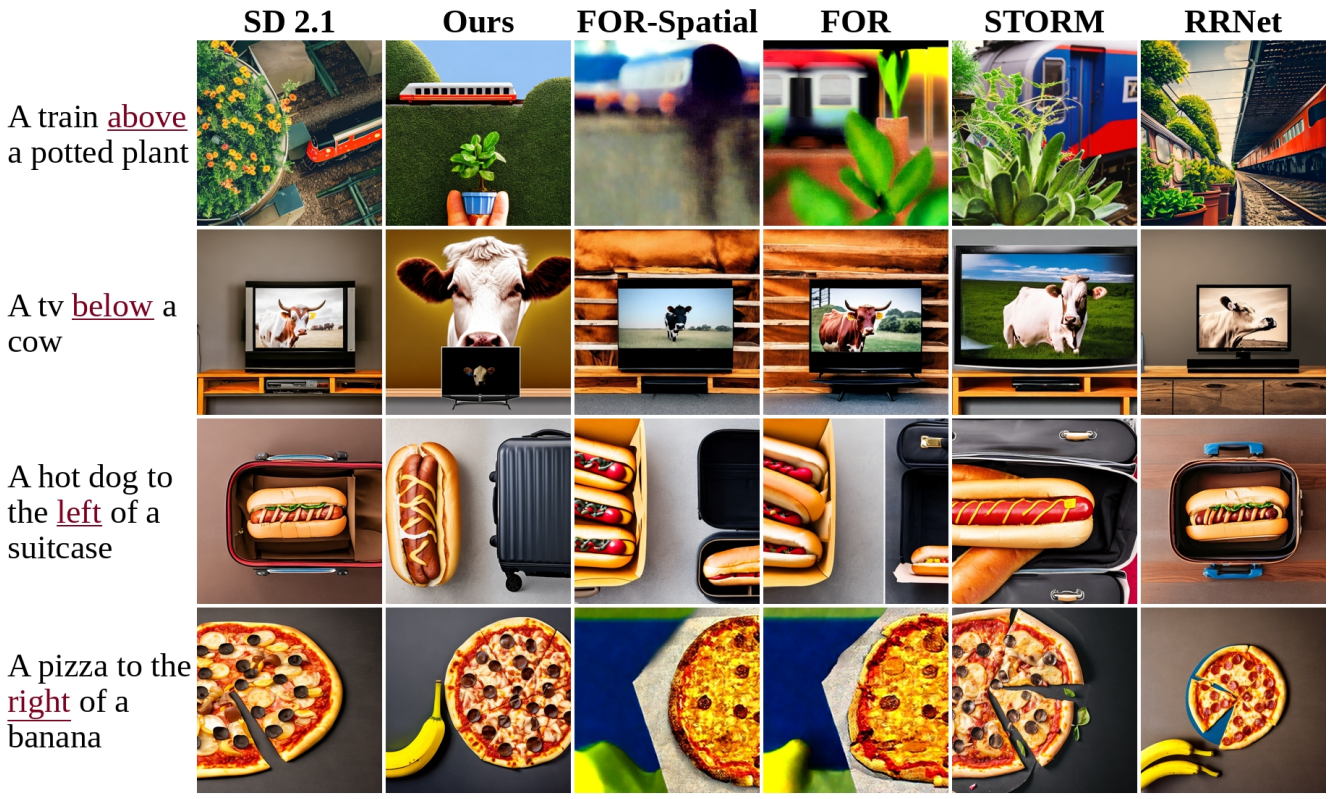

Figure 19. SD 2.1 on the GenEval benchmark

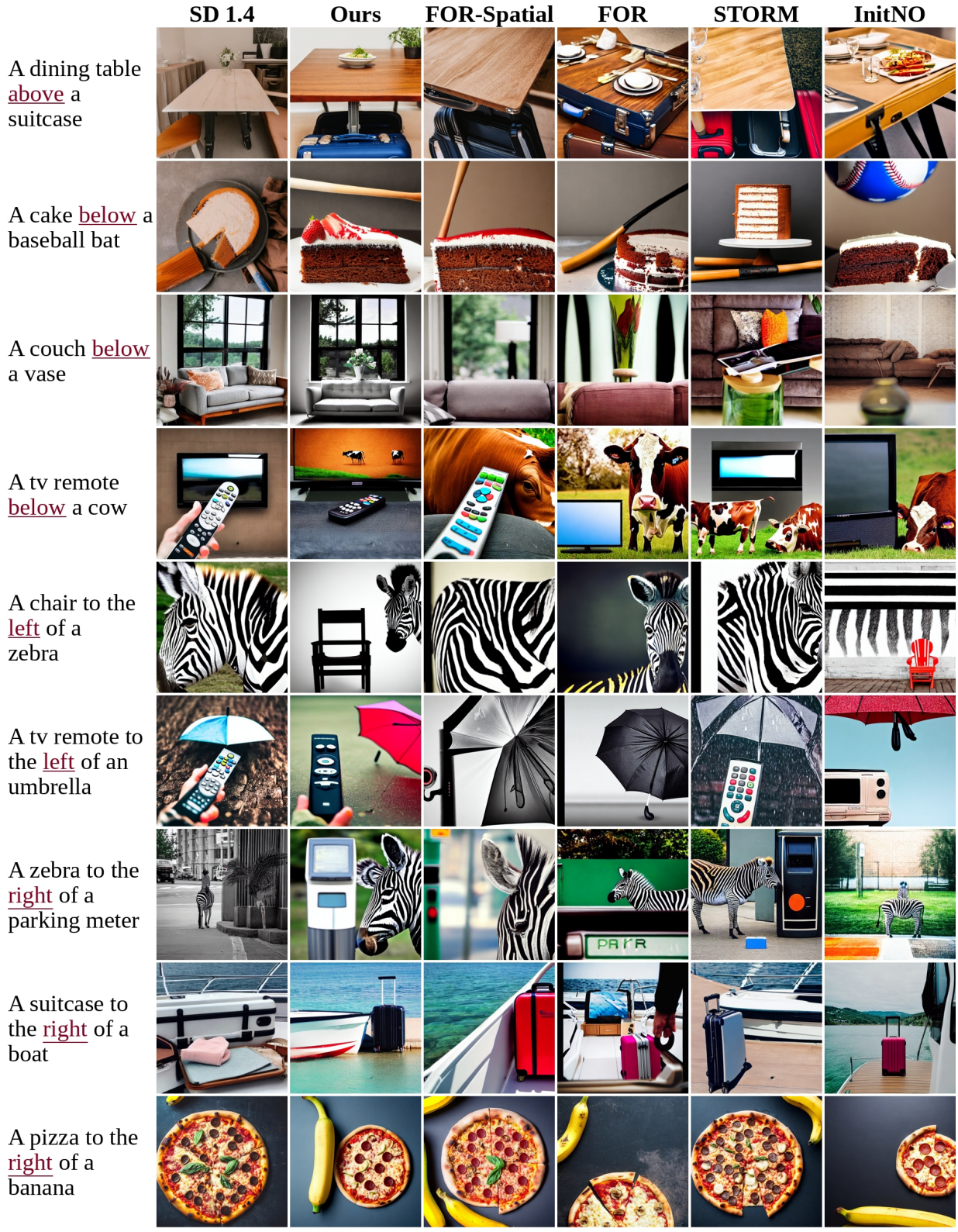

Figure 20. SD 1.4 on the GenEval benchmark

Figure 21. FLUX.1-dev on the T2I-CompBech benchmark

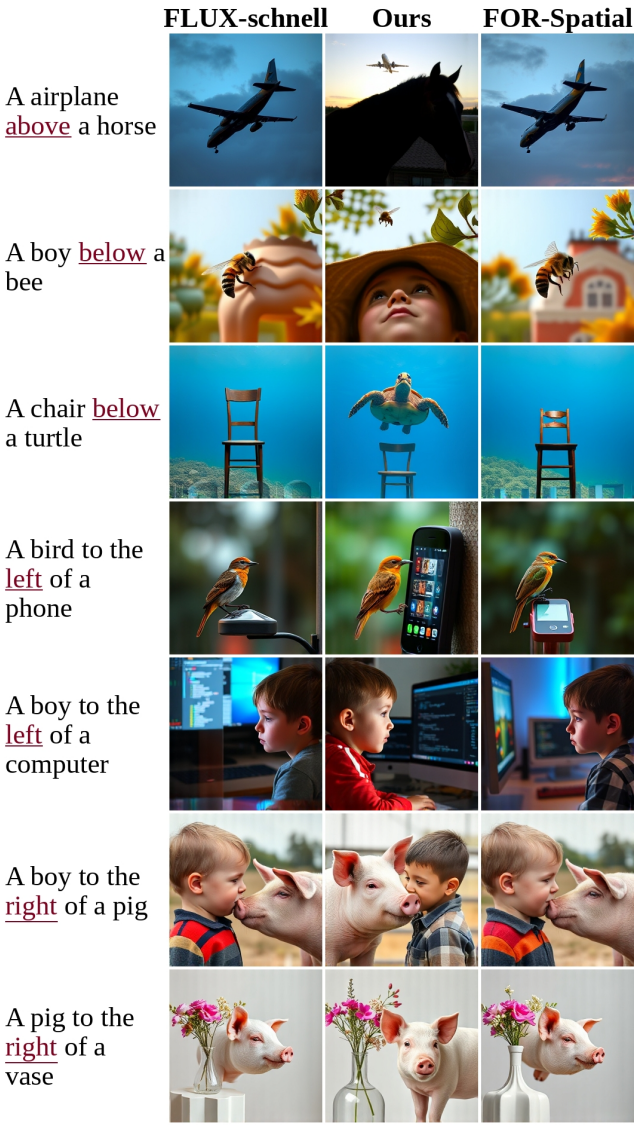

Figure 22. FLUX.1-schnell on the T2I-CompBech benchmark

Figure 23. SD 2.1 on the T2I-CompBech benchmark

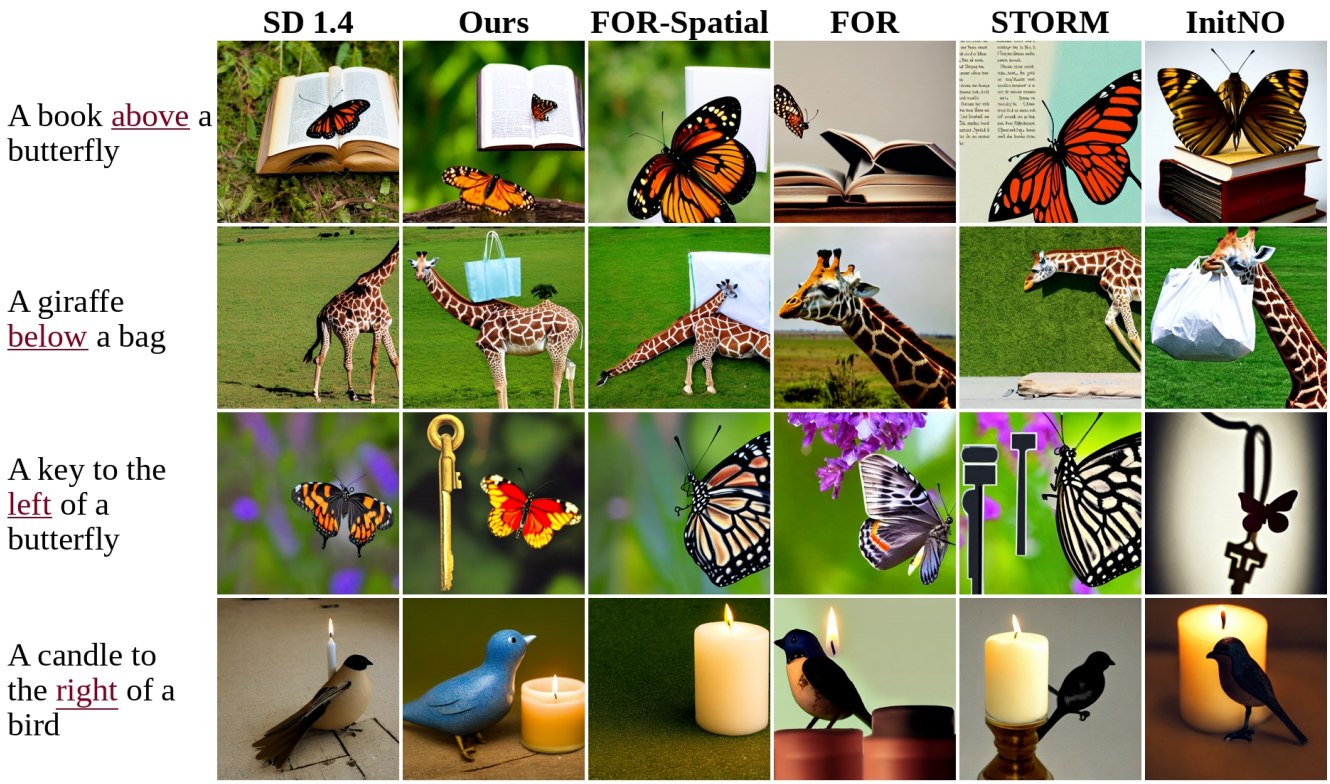

Figure 24. SD 1.4 on the T2I-CompBech benchmark