A. Implementation Details

To help reproduce our results, we include a comprehensive report of hyperparameters and the model architectures used in this work in Table 6.

The `Roberta-base` encoder [29] in layout predictor consists of 12 layers and 12 heads, with a hidden dimension size of 768. We fine-tune the model on GPT-synthetic dataset with relative position objective and MS-COCO dataset with absolute position objective for 100 epochs. The training batch size is 64, and the learning rate of encoder starts at 1e-6 and decays to 1e-8. The learning rate of the GMM output layer starts at 4e-5 and decays to 1e-8. We use the pre-trained `ViT-B/32` [35] to calculate CLIP similarities. For the diffusion model, we adopt the pre-trained `stable-diffusion-v1-4` [40] and stick with the default parameters. When optimizing the combination weights of cross-attention, the initial value is set to $1/N$, where N is the number of objects. The weight is projected to $[-1, 2]$ after each gradient descent step to avoid extreme values.

B. Details of GPT-synthetic Dataset and Dataset Statistics

In this section, we detail the process of creating the GPT-synthetic dataset and report the statistics of each dataset.

The GPT-synthetic dataset contains the 80 object categories in MS-COCO [25], and each description contains 2-5 objects and 1-4 relations. To create a text description with N objects and M relations, N objects are first sampled without replacement from the same MS-COCO super-category (*e.g.,* N objects from furniture), so that they are more likely to appear together in the same scene in real world. A color attribute is randomly assigned to each object with probability 0.5, and the assigned color is randomly sampled from a pre-defined list of colors. Among the N objects, M pairs are then sampled without replacement and randomly assigned a spatial relation from "left of," "right of," "above," and "below." We consider these four relations because they can be easily and reliably measured by comparing the center position, and we additionally check the relations to ensure no contradiction exists (*e.g.,* A is above B, B is above C, and C is above A). With specified objects and relations, GPT3 [4] is prompted to generate a sentence that mentions all objects and relations, given 5 demonstration examples. We specifically instruct GPT3 to generate diverse sentences. Table 7 shows the complete instruction, a sample demonstration, and a query that is used to generate a sentence. In practice, we manually write 20 demonstrations and randomly sample 5 for each generation.

	2 objects	3 objects	4 objects	5 objects
1 relation	200	50	0	0
2 relations	0	50	50	0
3 relations	0	0	50	50
4 relations	0	0	0	50

Table 4. Statistics of GPT-synthetic dataset.

Table 4 shows the number of text descriptions for the GPT-synthetic dataset. For the other two datasets, VSR contains 500 descriptions, and each description involves two objects and one spatial relation. MS-COCO contains 500 descriptions, including 200 descriptions with two objects, 150 descriptions with three objects, and 150 descriptions with four objects. There are no explicit spatial relations in MS-COCO descriptions. In general, GPT-synthetic contains the most complex text descriptions in terms of the number of objects and spatial relations.

	MS-COCO		VSR		GPT-synthetic	
	Global CLIP	Local CLIP	Global CLIP	Local CLIP	Global CLIP	Local CLIP
VANILLA-SD	0.2890	0.2357	**0.3071**	0.2412	**0.3006**	0.2243
COMPOSABLE-DIFFUSION	0.2892	**0.2397**	0.2948	0.2394	0.2886	0.2327
STRUCTURE-DIFFUSION	0.2870	0.2339	0.2972	0.2395	0.2912	0.2396
PAINT-WITH-WORDS	**0.2902**	0.2391	0.2974	0.2394	0.2961	**0.2418**
Ours	0.2892	0.2375	0.3029	**0.2415**	0.2944	0.2403

Table 5. CLIP Similarity of our method and baselines.

C. CLIP Similarity

We additionally use CLIP similarity [35] to measure how well the generated image aligns with the text description. Specifically, we consider CLIP similarity at two different granularities. **Global CLIP score** calculates the CLIP similarity between the whole text description and the whole image. On the other hand, **local CLIP score** calculates the similarity between the local object description and its corresponding bounding box in the image, if the object is detected. We use the local description defined in Sec. 3.2, *e.g.,* "A photo of a black mailbox."

The performance is presented in Table 5. We observe that different methods have very close global and local CLIP scores, which may be ascribed to the limited capability of vision-and-language models when text descriptions consist of multiple objects [59]. Based on this observation, the CLIP similarity is not sufficient to indicate which method performs better in our experiments, and we leverage the subjective evaluation in Sec. 4 for a more informative analysis.

Meanwhile, we also analyze why our CLIP scores are lower than baselines. We identify two possible reasons: ❶ Local CLIP scores only count **detected** objects but vanilla-SD misses objects more often. If we assign 0 CLIP score to undetected objects, it yields scores of 0.106 (vanilla-SD) *vs* 0.133 (ours). ❷ During optimization, the objective function combines local and global CLIP scores, where we emphasize the local part (5 times weighted). This leads to higher local scores but lower global scores.

Module	Attribute	Value
Layout Predictor	Model checkpoint	`Roberta-base` [29]
	Layers	12
	Heads	12
	Hidden dimension d	768
	Training batch size	64
	Training epoch	100
	Learning rate	$1e-6 \rightarrow 1e-8$ for transformer layer $4e-5 \rightarrow 1e-8$ for GMM
Diffusion Model	Model checkpoint	`stable-diffusion-v1-4` [40]
	Sampling steps	50
	Sampling variance	0.0
	Resolution	512×512
	Latent channels	4
	Latent down-sampling factor	8
	Conditional guidance scale	7.5
Attention Optimization	Checkpoint for CLIP loss	`ViT-B/32` [35]
	γ	5
	Optimizer	Adam [20]
	Learning rate	0.05
	λ_i initialization	$1/N$, where N is object numbers

Table 6. Hyperparameters and model architectures used in this paper.

Instruction	Given several objects, write a sentence that describes the given objects. Additionally, if location relation between objects is specified, the sentence needs to contain sufficient information that reveals the relation. Try to generate sentences as diverse as possible and DO NOT simply state the object locations.
Demonstrations	Objects: silver car, green motorcycle, blue bus, yellow truck Relation: silver car right of blue bus, yellow truck left of blue bus Sentence: The blue bus was driving along the road, with a silver car positioned to its right and a yellow truck overtaken and left behind on the left side of the bus, while a green motorcycle zoomed past on the opposite lane.
Query	Objects: red sandwich, yellow carrot, brown hot dog, green cake Relation: yellow carrot right of green cake Sentence:

Table 7. Complete instruction and example demonstration used to generate the GPT-synthetic Dataset.

D. Layout Predictor Analyses

In this section, we provide more details about the layout predictor and analyze how the layout predictor affects our method from the following two aspects.

Layout predictor implementation details We detail our layout predictor in Figure 6. Given an input text, the positions of each object are inferred with the following three steps: ❶ We extract noun phrases as objects using spaCy. ❷ We feed the sentence to a RoBERTa encoder. The outputs at corresponding tokens will be used as object representations. ❸ A prediction head outputs K mixture means $\{\boldsymbol{\mu}_{ik}\}_{k=1}^{K}$ for object i using its object representation, from which the left/rightmost mean is taken to compute the loss.

We also perform experiments to evaluate the performance of the layout predictor. Specifically, we are interested in ❶ whether spaCy is able to identify objects mentioned in the input text, and ❷ whether layout predictions are consistent with the spatial relations mentioned in the text. In our GPT-synthetic dataset, it turns out spaCy successfully extracts 96.1% objects, and the predicted locations correctly reflect 89.4% of spatial relations. Therefore, the layout predictor is mostly accurate and can be used to provide a good estimation of the object's position.

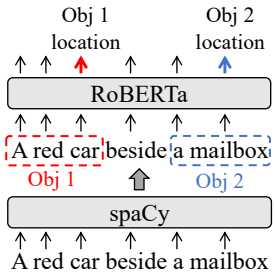


Figure 6. Layout predictor.

Layout predictor helps synthesize correct spatial relations We first demonstrate the position of each object predicted by our layout predictor and further show how the predicted position helps diffusion models generate objects with correct spatial relations. Fig. 7 illustrates two examples. Given the text description, the first column demonstrates images synthesized directly by vanilla stable diffusion model, where some objects are mislocated (*e.g.,* the spoon and bowl) and missed (*e.g.,* sandwich). The second column shows the predicted position of each object by our layout predictor, where it correctly locates objects according to the specified spatial relations (*e.g.,* the apple is placed beneath the sandwich).

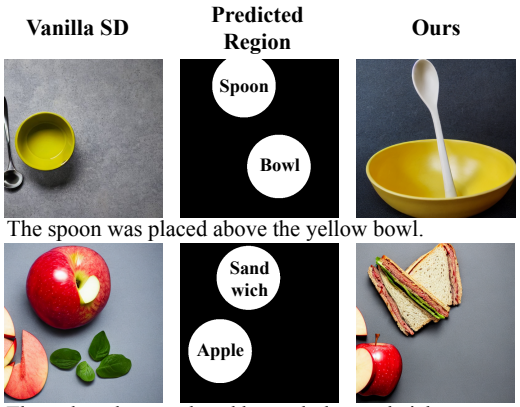


Figure 7. Layout predictor helps generate objects at correct locations. First column: Images generated by vanilla stable diffusion model. Second column: Pixel region generated by our layout predictor. Third column: Images generated by out method.

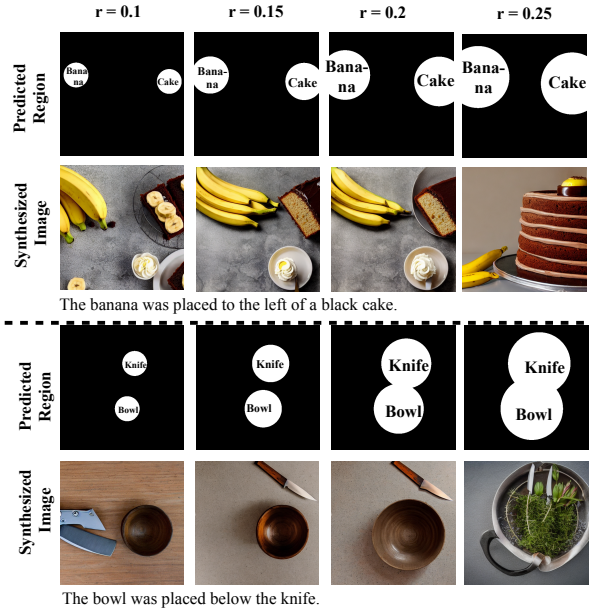


Figure 8. Example images generated by our method with different radii. For each panel, the top row is the predicted regions with different radii, and the bottom row is the generated images.

Finally, the last column shows images can be generated following the predicted layout, thus locating objects at correct position.

Layout predictor does not restrict the object size Once the position of each object is predicted, we define the pixel region of each object as a circle centered at the predicted coordinate with a radius r. We demonstrate two examples for images generated with different radii r in Fig. 8. Generally, the object size increases as the radius increases. However, the circular region does not strictly bound the object, and objects can go beyond the region (*e.g.,* the cake in the first row is larger than predicted region). Moreover, the generation results are not highly sensitive to the choice of r, as shown by the similar outputs of $r = 0.15$ and $r = 0.2$. We thus fix $r = 0.2$ in our experiments. However, we also would like to highlight the weak observed correlation between the image quality and the radius r. We provide two examples in Figure 9. In these experiments, for small objects (knives), applying a larger radius has more chance to negatively influence the quality; while the opposite holds for large objects (person).

	Object Recall	SPRel Precision
Ours	65.1%	75.0%
Ground truth position	66.3%	78.1%
No absolute position obj	62.7%	68.2%
No relative position obj	63.5%	58.6%
Soft pixel region	64.3%	69.8%

Table 8. Ablation study for layout predictor on VSR dataset.

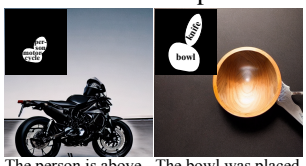


Figure 9. Small & large region.

E. Ablation Study for Layout Predictor

Using ground truth location in place of layout predictor In Sec. 4.1, we report the results based on our layout predictor. We now investigate the performance when ground truth location is used in place of the layout predictor. Specifically, we use the center of the ground truth bounding box as the object position, and we use the same radius $r = 0.2$ to construct the pixel region. We evaluate the performance on VSR dataset since it contains the ground truth bounding box information. As can be observed in Table 8, providing ground truth location boosts the performance, especially for spatial relation precision, since the pixel region is guaranteed to preserve the correct spatial relations. We also evaluate the performance of PAINT-WITH-WORDS baseline when the ground truth position is given. It achieves 64.7% object recall and 58.3% SPRel precision, which is worse than the counterpart of our method.

Training layout predictor with only absolute or relative position objective Our layout predictor is jointly trained with both absolution and relative position objectives (Sec. 3.3). We now explore our method's performance when the predictor is trained with only one of the objectives. The results are shown in Table 8. We observe that both settings lead to performance drop, indicating that both objectives are critical for an effective layout predictor. Moreover, removing relative position objective leads to significant performance degradation on SPRel precision, which demonstrates the importance of the objective.

Hard versus Soft Threshold on Pixel Region We use a hard threshold to get the pixel region in Sec. 3.4. Here we

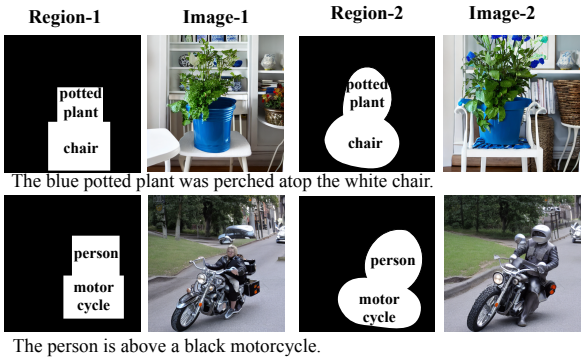


Figure 10. Example images generated from user provided region. The regions are shown in the first and third columns, and corresponding images are shown in the second and fourth columns.

explore a different strategy that produces a soft pixel region for an object. Specifically, we expand the pixel region of an object to the whole image but assign a smaller weight for pixels that are further away from the object. Formally, the output of the attention layer at time t becomes

$$\begin{aligned} \boldsymbol{O}(t) = &\sum_{i=1}^{N} \lambda_{it} \boldsymbol{G}_i \odot \text{Attention}(\boldsymbol{Q}, \boldsymbol{K}_{\boldsymbol{L}_i}, \boldsymbol{V}_{\boldsymbol{L}_i}) \\ &+ \left(1 - \sum_{i=1}^{N} \lambda_{it} \boldsymbol{G}_i\right) \odot \text{Attention}(\boldsymbol{Q}, \boldsymbol{K}_{\boldsymbol{D}}, \boldsymbol{V}_{\boldsymbol{D}}), \end{aligned} \tag{10}$$

where $\lambda_{it}, \boldsymbol{Q}, \boldsymbol{K}_{\boldsymbol{L}_i}, \boldsymbol{V}_{\boldsymbol{L}_i}, \boldsymbol{K}_{\boldsymbol{D}}, \boldsymbol{V}_{\boldsymbol{D}}$ are defined in Sec. 3.4, and $\boldsymbol{G}_i$ is a soft pixel region matrix for object O_i, with $\boldsymbol{G}_i(x,y) = g\left((x,y); \boldsymbol{C}_i, \sigma^2 \boldsymbol{I}\right) / g\left(\boldsymbol{C}_i; \boldsymbol{C}_i, \sigma^2 \boldsymbol{I}\right)$, where $g\left((x,y); \boldsymbol{C}_i, \sigma^2 \boldsymbol{I}\right)$ is the probability density of a 2D Gaussian distribution with mean $\boldsymbol{C}_i$ and covariance matrix $\sigma^2 \boldsymbol{I}$ at point (x,y), $\boldsymbol{C}_i$ is the center coordinate of the object, and σ is a hyperparameter. Intuitively, the combination weight of an object decreases as the pixel moves away from the object center, and the weights are normalized so that the object center has combination weight λ_{it}. The performance of this strategy is shown in Table 8, where it achieves a slightly worse performance compared to the hard threshold region.

Finally, we show examples in Fig. 10 where user provided region (possibly irregular) is given to our method. The generated images largely follow the provided layout, which demonstrates that our method can be adapted for image generation with better user interaction.

F. Performance on Uncommon Combinations

To further test if our method can generate high-fidelity images for novel text descriptions, we demonstrate the performance of our method and baselines on a dataset that contains uncommon scenes. This uncommon synthetic dataset consists of 100 text descriptions, and it differs from the GPT-synthetic dataset in Sec. 3.3 from two aspects. (1) When sampling objects for a description, we remove the constraint that objects need to belong to the same super category. Sampling without this constraint can thus produce rare object pairs (*e.g.,* objects from food and vehicle can occur in the same description). (2) We manually check generated samples and only keep the ones that are unlikely to appear in real life. The result is demonstrated in Table 9. We observe that our method achieves the best result in terms of object recall and spatial relation precision, indicating that our method can better generalize to novel text descriptions. Some visual examples can be found in Fig. 5 and Fig. 11.

	Object Recall	SPRel Precision
VANILLA-SD	39.8%	52.6%
COMPOSABLE-DIFFUSION	30.1%	50.8%
STRUCTURE-DIFFUSION	40.1%	51.6%
PAINT-WITH-WORDS	41.2%	54.7%
Ours	**42.4%**	**59.6%**

Table 9. Performance on text descriptions that contain uncommon object pairs, object-attribute pairs, and spatial relations.

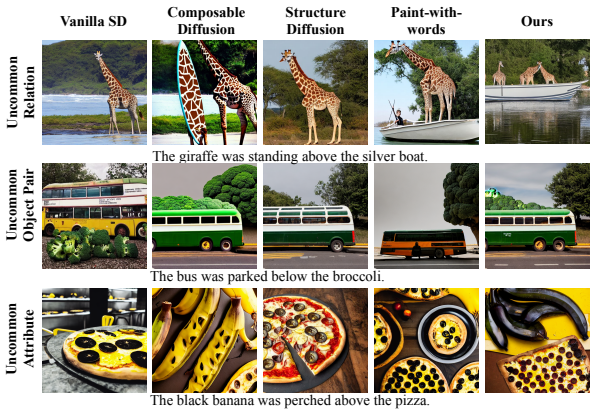


Figure 11. Example images generated by our method and baselines on uncommon relations, object pairs, and attributes.

G. More Examples and Failure Cases

In this section, we provide more example images from our method and baselines. Then, we analyze the potential failure cases of our method.

More Examples We provide more example images from our method and baselines in Fig. 14. The results are consistent with Fig. 3, where our method generates images with high object, attribute, and spatial fidelities.

Failure Cases We present two failure cases in Fig. 12. For each example, we first show the predicted pixel region by our layout predictor and the corresponding generated image (left two columns). We observe that these predicted positions tend to be at the edge of the image, which reduces the region of the object. We hypothesize that this will lead to insufficient attention to the corresponding object, and thus the object cannot be successfully synthesized (*e.g.,* the cell

phone in the first row). We further demonstrate in the right two columns that moving pixel regions inside the image can resolve these failure cases, where the missing objects can be synthesized. Future work may consider adding the constraint that the predicted object center cannot locate at the edge of the image.

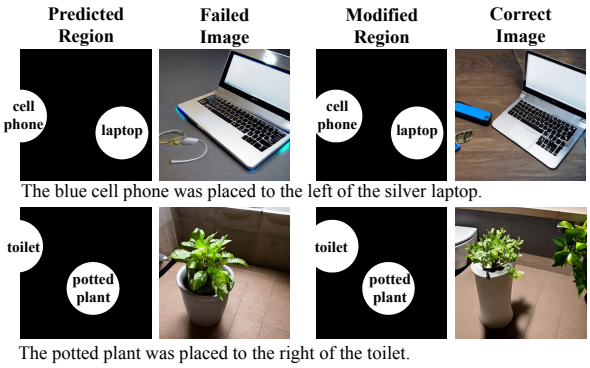


Figure 12. Failure Cases. The first two columns show the predicted region and the synthesized image, where some objects are missing. The last two columns demonstrate that modifying the pixel region can resolve the problem.

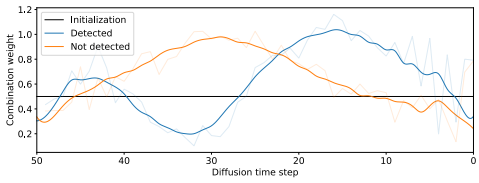


Figure 13. $\boldsymbol{\lambda}$ distribution before and after optimization

H. Details of Subjective Evaluation

In this section, we provide further details of our subjective evaluation. We compare with all baselines on MS-COCO, VSR, and GPT-synthetic datasets. For each dataset, we randomly sample 25 text descriptions for evaluation. We evaluate on Amazon Mechanical Turk, and 85 workers participate in our study. During the subjective evaluation, the workers are asked four questions: (1) (*Object Fidelity*) Does the image contain all objects mentioned in the text? (2) (*Attribute Fidelity*) Are all synthesized objects consistent with their characteristics described in the text (*e.g.,* color and material)? (3) (*Spatial Fidelity*) Does the image locate all objects at the correct position such that the spatial relations in the text are satisfied (if an object in the relationship is missing, it is considered as an incorrect generation)? and (4) (*Overall*) Which image in the pair has higher fidelity with the text and has better quality? For the first three questions, we present the participant with a single image generated by one method, and ask the participant to rate the image using a score of 0, 1, or 2, where 2 denotes all objects/attributes/relations are correct and 0 denotes none of them is correct. For the last question, participant will see a pair of images, where one of them is generated by our method and the other one is generated by one baseline. The participant is then asked to select the better image in terms of overall fidelity and quality. The subjective evaluation interface is shown in Fig. 15 and Fig. 16. The subjective evaluation results are shown in Table 1. We also provide all generated images by our method and baselines in Figures 17, 18, and 19.

I. Analysis of Optimized Combination Weights across Denoising Process

Recall that at inference time, each identified object is associated with a learnable coefficient λ_{it}. These coefficients serve as combination weights in Equation (8) during the image synthesis process and are dynamically optimized across every object and denoising step. To better demonstrate the outcomes of this optimization, we further visualize the optimized λ_{it} in Figure 13. In the figure, the coefficients are averaged across 100 input texts, and the two curves show the objects that are detected or not detected in the synthesized images respectively. We observe that after optimization, $\lambda_i t$ tends to increase in general. Besides, for those undetected objects, the final combination weights tend to be larger in the initial denoising steps. We hypothesize that at these initial denoising steps, our algorithm focuses more on the objects that are hard to be synthesized.

J. Performance on different number of denoising steps

In previous experiments, the denoising steps are fixed to 50. To further test if our method can generalize to different number of denoising steps, we perform experiments to measure the performance of our method as well as `Vanilla-SD` under 30 and 70 denoising steps. The result is shown in Table 10. These results, as well as the experiment in Table 1 and Table 2, indicate that our method consistently outperform baseline at different denoising steps.

	30 steps		70 steps	
	Object Recall	SPRel Precision	Object Recall	SPRel Precision
`Vanilla-SD`	40.7%	52.2%	42.8%	55.2%
Ours	46.5%	55.6%	48.4%	61.3%

Table 10. Performance with different denoising steps.

References

[59] Tristan Thrush, Ryan Jiang, Max Bartolo, AmanpreetSingh, Adina Williams, Douwe Kiela, and Candace Ross. Winoground: Probing vision and language models for visio-linguistic compositionality. In *CVPR*, 2022.

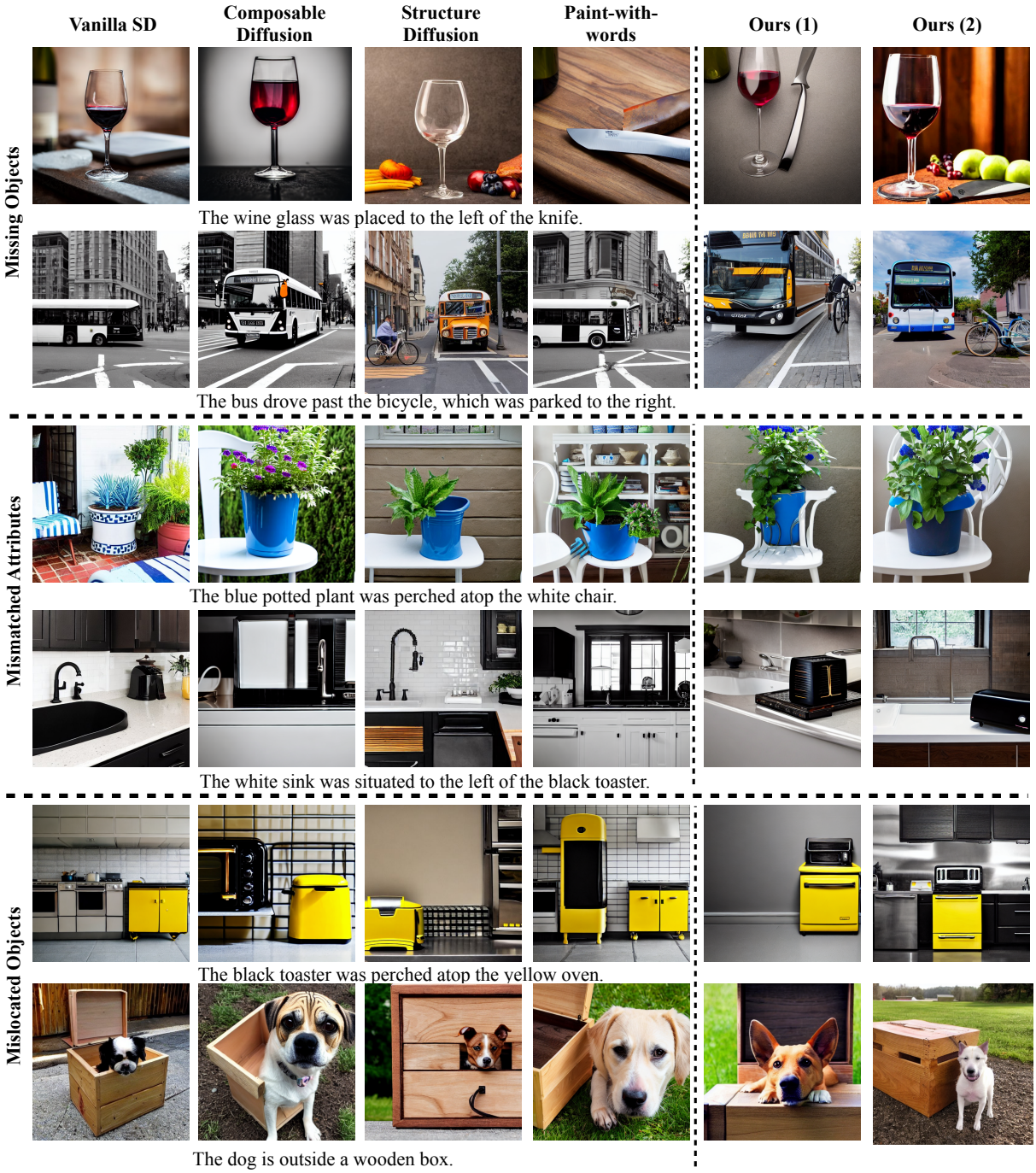


Figure 14. **Example images generated by our method and baselines.** Typical errors of baselines include missing objects, mismatched attributes, and mislocated objects. Ours (1)/(2) show the results with two different random seeds.

Instructions:

Please read the instructions carefully. Failure to follow the instructions may lead to rejection of your results. Your task will involve evaluating whether target objects have been successfully synthesized using AI models. First, you will see a text description that outlines the objects to be generated (e.g., "The bed is below the black cat."). Then you will see an image, which is generated based on the provided text by an AI algorithm. You will then be asked to evaluate if the generated image contains all the objects mentioned in text. You will use a scoring system ranging from 0 to 2, where 0 indicates all objects are incorrect or missing, 1 means some objects are incorrect or missing, and 2 means all objects are successfully generated. Notice that you should **only** rate if the objects are synthesized or not; you should disregard their inconsistencies with text description such as colors or relative positions (e.g., left/right).

Example: We provide an example to help you understand how to evaluate the generated results. The text description is "**The bed is below the black cat.**"

We can observe that cat is successfully generated, but the bed is not. Therefore, this example is partially correct, and you should rate score 1. Again, notice that it synthesizes a white cat while the text says "black cat", but you should ignore the inconsistencies of color and position.

Question:

The text description is "**The motorcycle is parking to the right of a bus.**" Does this image contain the objects mentioned in the text description? Rate the generation results from 0 (all objects missed) to 2 (all objects are generated).

☐ 0
☐ 1
☐ 2

Figure 15. Instructions and an example question of the subjective evaluation on Amazon Mechanical Turk. The goal is to evaluate whether the generated images contain all specified objects (object fidelity). The interfaces for attribute fidelity and spatial fidelity are similar.

Instructions:

Please read the instructions carefully. Failure to follow the instructions will lead to the rejection of your results. In this task, you will be asked to judge and compare the quality of two AI-generated images. Specifically, you will first see a text description, which describes the desired content we want to generate (e.g., "The bed is below the white cat."). Then you will see two images, which are generated based on the provided text by different AI algorithms. You will then be asked to evaluate which image better follows the text description. When evaluating, you should consider the following aspects: (1) Does the synthesized image contain all objects mentioned in the text? (2) Does each object in the image follow the text description? (3) Does the image preserve the correct spatial relations mentioned in the text? (4) Does the image look real and natural? Then, you will choose the better image from the two candidate images.

Example: We provide an example to help you understand how to evaluate the generated results. The text description is "**The bed is below the white cat.**"

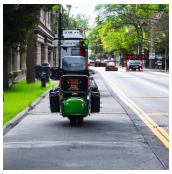
(A)

(B)

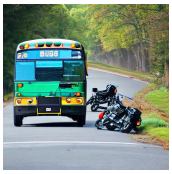
You will evaluate based on the above criteria. First, it is important that the edited images faithfully synthesize the two objects "bed" and "cat" in the image. All methods generate a cat in the image. However, method A fails to generate high quality "bed", while method B generates a better bed. Second, both methods try to generate a white-colored cat. Third, both methods preserve the correct spatial relation that the cat is above a bed. Finally, the cat in method B looks more natural. Considering all the above analysis, method B is better.

Question:

The text description is "**The motorcycle is parking to the right of a bus.**" Which image in the pair has higher fidelity with the text and has better quality? Please give an overall evaluation based on the above criteria.



(A)



(B)

☐ (A)
☐ (B)

Figure 16. Instructions and an example question of the subjective evaluation on Amazon Mechanical Turk. The goal is to compare two images generated by baselines and our method.

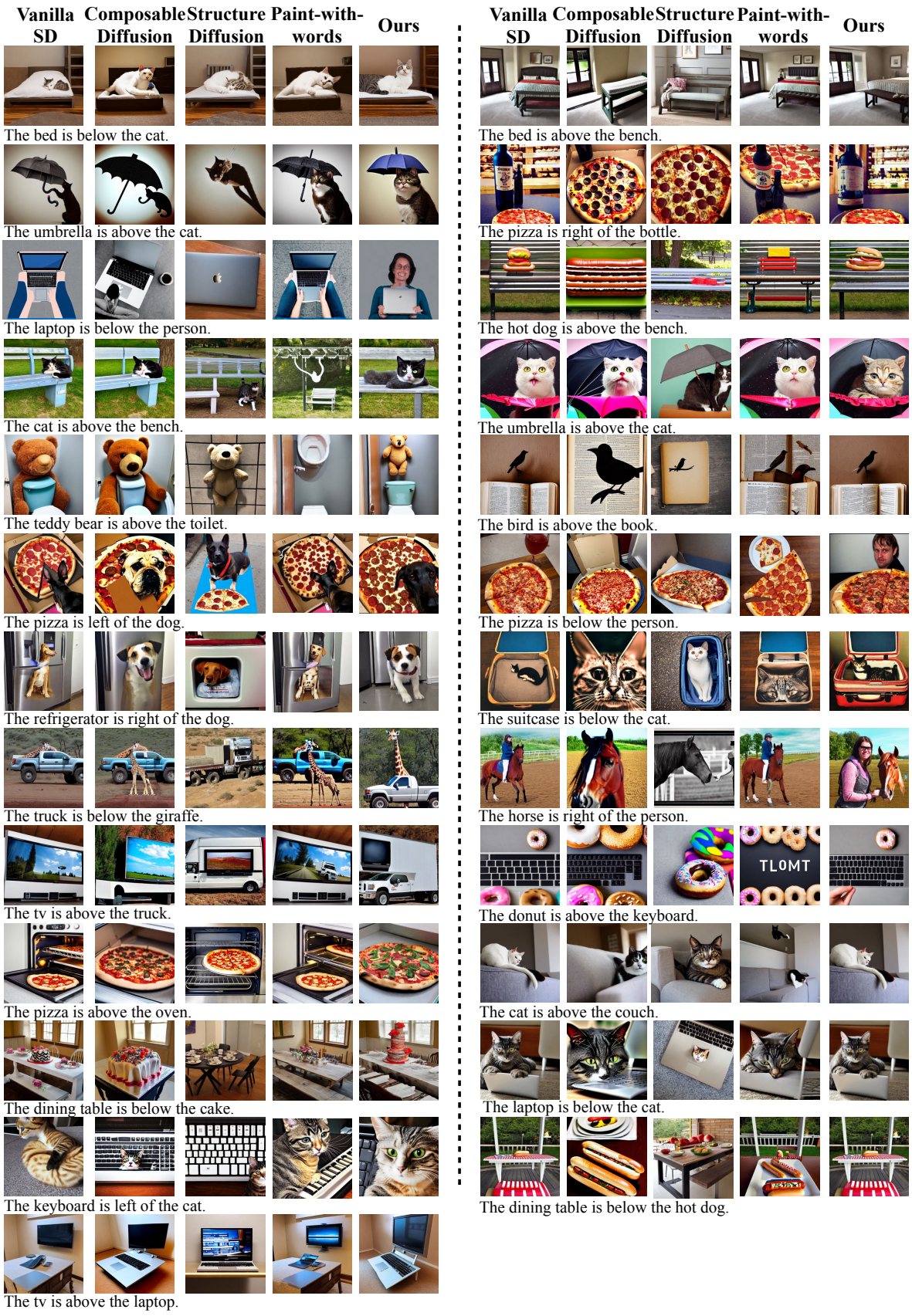


Figure 17. Generated images for subjective evaluation on VSR dataset.

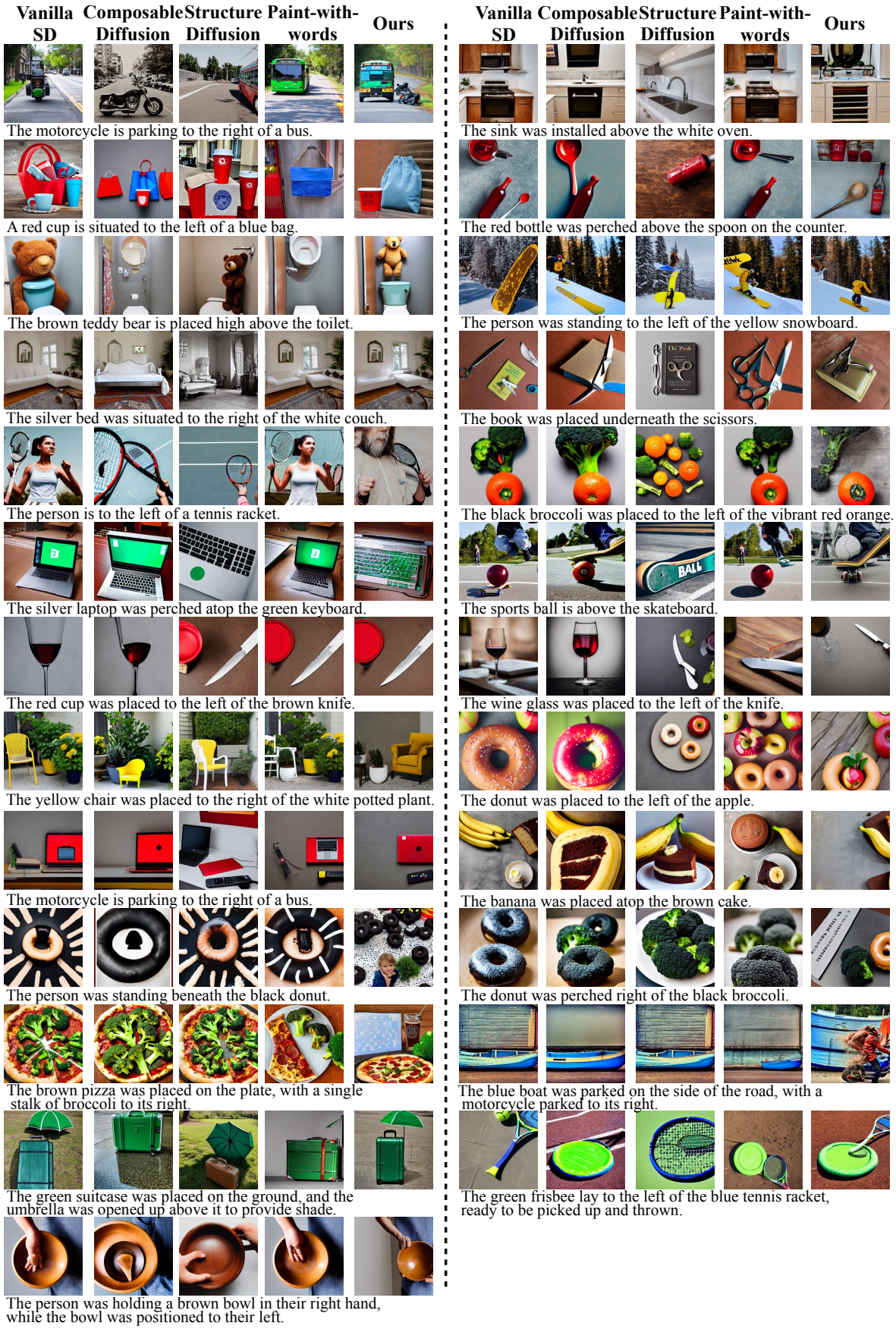


Figure 18. Generated images for subjective evaluation on GPT-synthetic dataset.

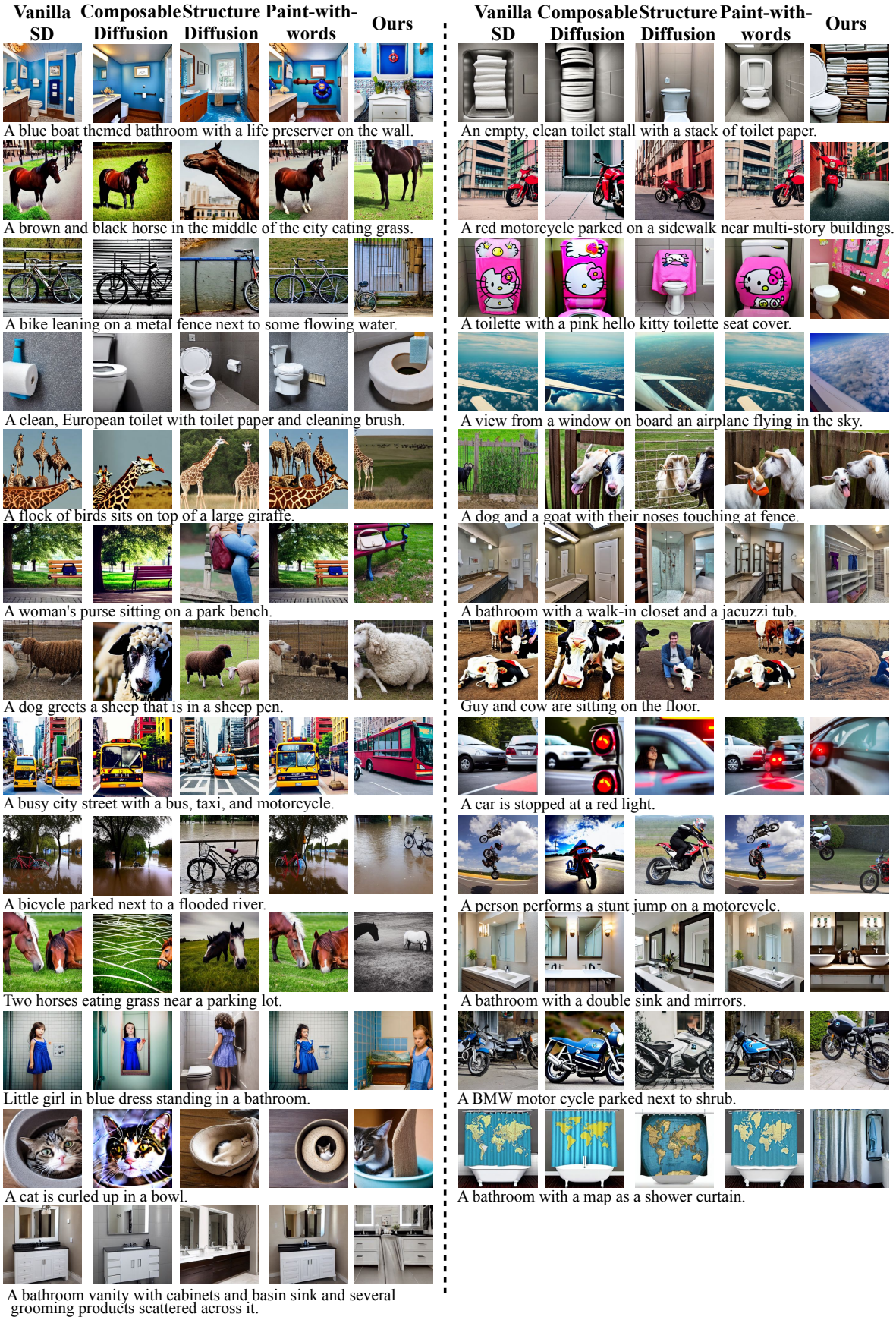


Figure 19. Generated images for subjective evaluation on MS-COCO dataset.