Layer-wise Instance Binding for Regional and Occlusion Control in Text-to-Image Diffusion Transformers

Supplementary Material

This supplementary material is organized as follows:

- Sec. A details optional modules of LayerBind, including the vital block selection for the Hard-Binding and the implementation details of the Layer Blending mechanism.
- Sec. B provides detailed introductions of the experimental setup, including the strategy for layout parsing, the dataset construction, and the evaluation metrics.
- Sec. C provides supplementary experiment analyses, including efficiency and further module ablations.
- Sec. D further discusses applications of LayerBind, including the implementation details for composited image editing and its compatibility with external visual adapters.
- Sec. E address the limitations of LayerBind. We showcase representative failure cases and discuss potential solutions for repair.
- Sec. F provides more visualization of generation results.

A. Optional Modules of LayerBind

A.1. Vital Block Selection for Hard-Binding

As discussed in prior work [34], "modality competition" can cause instances to be suppressed by strong background contexts. To address this, we leverage the observation that different DiT blocks exhibit varying sensitivities to text [1, 47]. We suggest that forcing "text-dominant" blocks to focus exclusively on their primary strength (semantic injection) offers a natural, minimally disruptive intervention for instance initialization.

To identify these vital blocks, we empirically analyze attention maps recorded during generation. Using segmentation tools on multiple runs (20 prompts, 5 seeds), we extract foreground masks and calculate the response intensity of foreground tokens (as queries) to different token types, averaged over the first 20% of steps. The results (Fig. 10) reveal a consistent trend in both FLUX and SD3.5: while foreground self-attention remains high across all layers, responses to background versus text tokens diverge significantly. Early blocks are visually dominant, whereas mid-to-late blocks become progressively more text-responsive. Based on these findings, the selection is twofold:

1. We always select Layer 0, uniquely critical for establishing initial semantic binding [1, 47].
2. We select the 2 most text-responsive blocks from early-mid stages and the 6 most text-responsive blocks from late stages.

This empirical selection balances semantic injection strength with maintaining image generation quality. Specifically, we select blocks $[0, 15, 18, 42, 45, 48, 50, 53, 54]$ for FLUX (partially aligning with [1, 47]) and $[0, 11, 14, 19, 21, 24, 29, 32, 34]$ for SD3.5. Restricting Hard Binding to these vital blocks maximizes semantic injection while minimizing disruption to overall quality, as applying it to excessive blocks degrades performance.

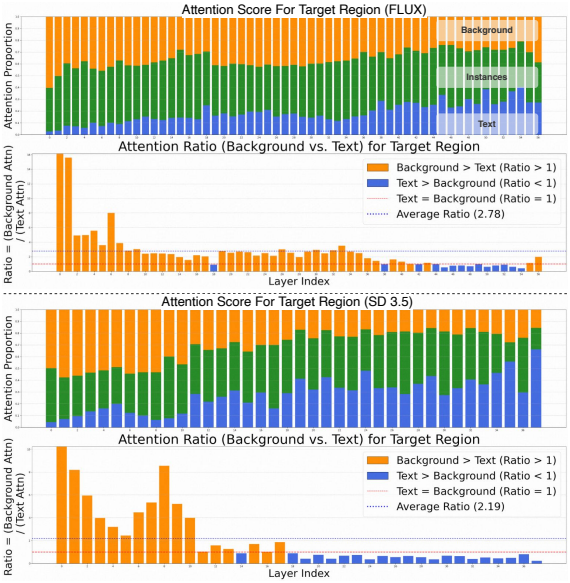

Figure 10. Attention response weights of foreground to background, self, and text tokens across different attention blocks (in FLUX and SD3.5).

A.2. Layer Blending Module

As described in Sec. 4.2, we composite occluding layers using a foreground-aware alpha mask $\alpha^{(i)}$. Since the branch $B^{(i)}$ and global latent I share the same background context, we can estimate this mask directly from their difference. First, we compute a robust saliency map Z by normalizing the difference using the local background variance (σ_{bg}), estimated from the Median Absolute Deviation (MAD) of a surrounding background area with the estimated region:

$$Z = \text{Smooth}\left(\left\| \frac{B^{(i)} - I[idx^{(i)}]}{\sigma_{\text{bg}} + \epsilon} \right\|_2^{\gamma}\right) \tag{13}$$

where $I[idx^{(i)}]$ is the global latent in the instance region and γ is a correction factor, default with 0.9. This coarse saliency map Z is then refined into a spatially smooth alpha mask by solving a Screened Poisson equation via an itera-

tive solver:

$$\alpha_{k+1} = \frac{1}{4+\lambda}\left(\sum_{p\in\mathcal{N}(\alpha_k)} \alpha_p + \lambda Z\right) \tag{14}$$

where λ balances the data term Z against the smoothness term. After convergence, we apply an optional Otsu's thresholding (a classic, unsupervised algorithm for automatic foreground thresholding) to explicitly separate the foreground region. Finally, we perform morphological reconstruction to fill any internal holes, ensuring the instance appears contiguous and enhancing the blend's realism.

B. Extended Evaluation Details

B.1. LLM-based Layout Parser

As outlined in Sec. 4.1, consistent with prior works [8, 24, 55, 59], we employ an LLM to generate layout plans when precise user-specified layouts are unavailable. However, LayerBind has specific input requirements that generic parsers neglect: a decoupled background prompt and an explicit occlusion order. To address this, we design a specialized prompt engineering strategy using a state-of-the-art LLM (e.g., GPT-5-mini [37]), as visualized in Fig. 11. The parsing process leverages Chain-of-Thought (CoT) prompting and is divided into two distinct stages:

1. **Spatial reasoning:** The LLM first engages in a planning phase to analyze the input caption. It deduces the overall scene structure, ensuring logical spatial relationships and reasonable object placement before committing to coordinates.
2. **Structured output:** We constrain the LLM to output a JSON. Crucially, this schema requires the model to explicitly generate a `"background_prompt"` for context isolation and assign an `"order"` index to each instance based on its layer index, thereby rigorously defining the occlusion hierarchy.

To ensure stability and adherence to this schema, we utilize in-context learning by providing few-shot examples (as shown at the bottom of Fig. 11), enabling the model to learn the correct patterns for layout generation.

Furthermore, this parser serves as the foundation for our benchmark construction. Although prior works [8, 24, 55, 59] have utilized layout generation for evaluating T2I-CompBench, none have publicly released their ground-truth layout annotations. This absence prevents a fair, unified comparison across different methods. Therefore, we use this standardized parser to re-annotate the dataset, ensuring a consistent evaluation ground for all baselines.

Layer Parser Prompt (Simplified)

You are a layout annotator for text-to-image generation.
Goal: From an input text prompt, produce ONLY a JSON object with:
- planning, rewritten prompt,
- background prompt,
- instances list: instances with isolated captions, 1024×1024 bounding boxes, and depth order.
Rules (Detailed rules are attached after each item):
1. Plan the layout: ...
2. Rewrite the caption: ...
3. Instance extraction: ...
4. Generate detailed caption for each instance: ...
5. Relative depth annotation: ...
6. Bounding box layout annotation: ...
7. Generate background prompt for overall scene: ...
Input: <INPUT_PROMPT>
OUTPUT JSON schema:

```
{
 "planning": "string",
 "rewritten_prompt": "string",
 "background_prompt": "string",
 "elements": [
 {
  "region_prompt": "string",
  "layout": [x_top, y_top, x_bottom, y_bottom],
  "order": 1
 }...]
}
```

<In-Context Examples>

Figure 11. The simplified prompt template used for our LLM-based Layout Parser. Specific rules are defined for each part of the parsing, and these rules are made comprehensible to the model through In-Context Examples.

B.2. Dataset Construction

We evaluate LayerBind on two primary tasks: Occlusion Control and General T2I Alignment. Our prompts are primarily sourced from T2I-CompBench [21]. However, as this benchmark lacks publicly available layout annotations, we re-annotated the data using our proposed Layout-Parser to ensure a unified input setting. The dataset construction consists of two parts:

- **Layout-annotated T2I-CompBench:** We utilize the *color*, *shape*, *texture*, *2D-spatial*, *numeracy*, and *complex* subsets for general alignment evaluation. For the 3D-spatial subset, which focuses on occlusion, we applied strict filtering. We employed the Layout-Parser to generate bounding boxes and layer orders, followed by manual verification to remove invalid cases (e.g., where regions are fully occluded or exhibit ambiguous 3D relationships). This resulted in 800 high-quality prompts for occlusion evaluation. The sample counts for other subsets remain consistent with the original benchmark.
- **BindBench (Complex Occlusion):** The original T2I-CompBench-3D is limited to spatial relationships between only two objects, which is insufficient for evaluat-

ing complex occlusion. Notably, LaRender [58] proposed the RealOcc dataset to address similar issues; however, it contains only 60 prompts and is currently not publicly accessible. To enable a more comprehensive evaluation, we construct **BindBench**. We reuse instance categories from T2I-CompBench but recombined them to form complex scenes featuring 3 to 5 overlapping objects. After applying our Layout-Parser for annotation and conducting rigorous manual filtering, we obtain 200 challenging prompts specifically designed to benchmark multi-instance occlusion control.

To comprehensively assess LayerBind's performance in terms of layout precision, occlusion accuracy, semantic consistency, and generation quality, we employ the following metrics:

B.3. Evaluation Metrics

1. UniDet-Depth (Relative Depth Accuracy). We utilize the official evaluation script from T2I-CompBench [21]. This metric employs the UniDet depth estimation model to predict the depth map of the generated image. It then computes the average depth value within the ground-truth bounding boxes of instance pairs to determine if the generated depth order matches the input condition.

2. CLIP Score (Semantic Alignment). We assess text-image alignment at both global and local levels using the CLIP ViT-L/14 model:

- **CLIP-G (Global Consistency):** We calculate the cosine similarity between the whole image embedding and the full scene text embedding. This measures how well the overall image captures the global prompt, reflecting both object existence and reasonable spatial composition.
- **CLIP-L (Regional Fidelity):** To evaluate whether specific instances are generated with sufficient regional detail, this metric calculates the similarity between the embedding of the cropped instance region (defined by the layout box) and its corresponding regional prompt.

3. $\mathbf{L}_{Acc/VQA}$ (Fine-grained Layout Faithfulness). Following [59, 60], which implement VLM-based metrics to evaluate layout faithfulness, we introduce L_{Acc} to measure layout generation accuracy and L_{attr} for fine-grained evaluation of regional attribute fidelity. Both metrics are based on VQAScore [32] and implemented with Qwen2.5-VL [2]. Specifically, L_{Acc} is computed by questioning each region with the template: "This image contains {object class}?", and L_{attr} is computed by questioning with the detailed regional prompt. These two metrics are evaluated on the more complex BindBench dataset and offer more discriminative comparisons than the CLIP-L.

4. $\mathbf{O}_{VQA}$ (Perceptual Occlusion Success). While depth estimation suffices for simple object pairs, it struggles with the complex, multi-instance occlusions found in our BindBench. To address this, we employ the VQAScore [32] as a perceptual metric to develop the O_{VQA} metric. Similar to $L_{Acc/VQA}$, we utilize Qwen2.5-VL [2] as a visual judge. We feed the generated image and a query regarding the occlusion relationship into the MLLM (as illustrated in Fig. 12). The model's accuracy serves as a proxy for the human-perceived success rate of occlusion control.

Figure 12. Illustration of leveraging QWen-VL2.5 [2] as a VQA judge for evaluating 3D spatial relationships. It can accurately perceive spatial relationships and score the image content on whether it satisfies the question.

Model	1	2	3	4	5	6
FLUX	18%	31%	45%	60%	76%	89%
SD3.5	24%	39%	55%	73%	92%	107%

Table 4. LayerBind's additional inference time when inputting different numbers of regions. Each region occupies 25% of the image tokens (e.g., 1024 tokens). The inference cost of LayerBind increases linearly with the number of additional tokens.

5. HPS v2 (Generation Quality). Layout control methods often risk degrading image quality (e.g., introducing unnatural lighting or artifacts). To quantify this trade-off, we report the Human Preference Score v2 (HPS) [49]. Trained on large-scale human choices, HPS correlates well with human aesthetic judgments. A higher HPS indicates that LayerBind effectively preserves the high-fidelity generation capabilities of the base DiT model.

C. Extended Experiment Analysis

C.1. Efficiency Analysis

In terms of run-time efficiency, the introduction of instance branches inevitably increases the total token count, leading to additional computational overhead. In Table 1, we initially reported the inference overhead for standard generation scenarios (based on BindBench cases, involving approximately 40–50% additional tokens). To more comprehensively evaluate LayerBind's inference efficiency, Table 4 details the overheads of varying loads, ranging from 1 to 6 input regions (corresponding to a token count increase of 25% to 150%). LayerBind's overhead scales linearly with the number of additional tokens, avoiding the quadratic computational explosion often associated with extended token sequences in Transformers. This efficiency is attributed to two key design choices: (1) branch tokens

are active only during the limited early initialization steps, and (2) our local update calculation employs an block-wise strategy rather than full-sequence calculation. In summary, LayerBind maintains a highly practical trade-off, achieving precise control with manageable computational costs.

Figure 13. The illustration of the effectiveness of the proposed LSN and naive regional prompting [4] strategies. Without layer-wise updates, errors such as concept blending and failure in occlusion control may occur.

C.2. Layer-wise Nursing vs. Regional Prompting

A straightforward alternative to our Layer-wise Semantic Nursing (LSN) is standard regional prompting [4], which injects semantics without explicit layer-wise compositing. We observe that for layouts with spatially disjoint instances (i.e., no complex occlusion), Regional Prompting effectively refines instance details. However, as illustrated in Fig. 13, it fails in two critical scenarios:

- Concept Blending: Without the explicit isolation, attention leakage may occur between regions. This can lead to concept blending, where semantics blend across boundaries even without explicit visual overlap.
- Occlusion Failure: During the nursing phase, the initialized latent representations often lack sufficiently distinct semantic boundaries to maintain occlusion naturally. Without layer-wise updates to strictly enforce visibility ordering, standard regional prompting fails to ensure that the foreground robustly overwrites the background. Consequently, the pre-established occlusion relationships degrade or vanish in the final output.

Therefore, we conclude that the proposed LSN is a more robust strategy, essential for maintaining both semantic and the correct occlusion order.

C.3. Effect of η_1 and η_2

The selection of η_1 and η_2 is critical for LayerBind's performance. First, η_1 is the decisive factor for generation success. As illustrated in Fig. 9, an excessively low η_1 fails to impose sufficient spatial constraints, while an overly high value causes significant stylistic or layout over-decopled between the instance and the background (Fig. 13). As shown in Fig. 14, $\eta_1 = 0.25$ serves as a robust default for both FLUX and SD3.5, though fine-tuning within $[0.1, 0.3]$ can further optimize specific cases. Regarding η_2, its role includes semantic detail refinement with layout maintenance.

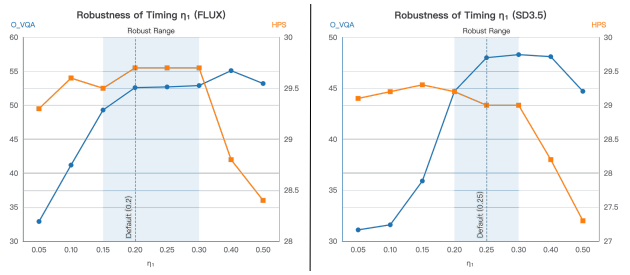

Figure 14. The performance is stabilized across a robust η_1 range, while fine-tuning η_1 can further optimize specific cases.

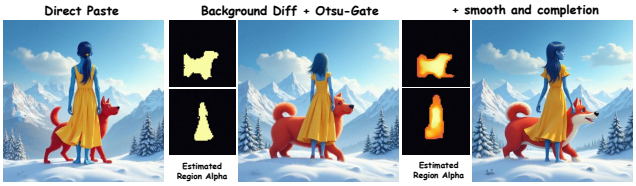

Figure 15. The visualization of the estimated alpha mask and the effectiveness of different branch blending strategies.

For simple instances where structural preservation is the priority, $\eta_2 = 0.5$ suffices. However, for instances with complex attribute details, we recommend increasing η_2 to 0.7 to ensure faithful semantic details.

C.4. Effectiveness of Branch Blending

The Branch Blending mechanism works as shown in Fig. 15, only applied to the occluded instances region, which is an optional step for enhancing the instance edge quality. We found that in most cases, as long as the bottom-layer instances have sufficient unoccluded parts, direct paste can maintain the occlusion, while further blending strategies further improve the generation quality of both the instances and the overall scene.

D. Extended Applications

D.1. Implementation of Composited Image Editing

As illustrated in Fig. 8, LayerBind supports multi-region, multi-instruction composited image editing. This capability stems from LayerBind's inherent ability to spawn and merge instance branches onto any arbitrary background generation trajectory. Specifically, the editing pipeline proceeds as follows:

- First, we obtain the denoising trajectory of the image. For synthesized images, we can directly use the original prompt and initial noise; for real images, inversion methods [45] can be used to retrieve the denoising trajectory.
- Then, a key distinction from our standard layout control (which branches at pure noise $t = T$) is the timing of branch creation. For editing tasks, to ensure optimal structural consistency with the original image, we instan-

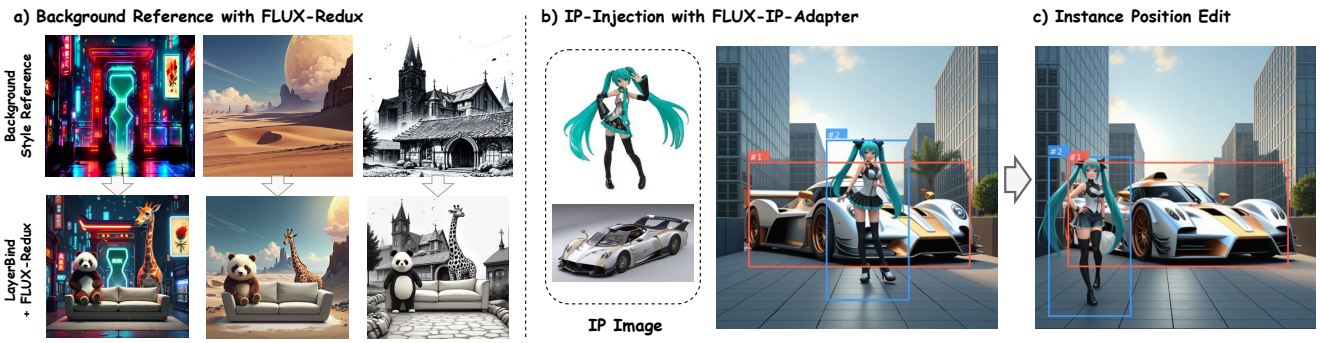

Figure 16. The proposed LayrBind can be integrated with external visual adapters. a) With the FLUX-Redux plugin, LayerBind can directly generate the background by referencing an existing image. b) With IP-adapter [56], LayerBind can inject visual references into regions. c) LayerBind supports instance position editing.

tiate the edit branches after the first denoising step rather than at initialization.

- Finally, each edit region undergoes an independent branch guided by its specific instruction via Eq. 3, functioning analogously to parallel multi-region image inpainting. For fusion, we employ a larger initialization ratio ($\eta_1 \approx 0.4$-0.5). At this stage, the edited content and structure are firmly established; completing the subsequent standard denoising process yields the final edited result.

We plan to provide more extensive analysis and examples of this application in future iterations of this work.

D.2. Compatibility with External Adapters

Since LayerBind relies solely on attention mechanisms, it is inherently compatible with most external adapters. We exemplify this capability in Fig. 16 using two popular tools: the FLUX Redux adapter and IP-Adapter. First, LayerBind can utilize the Redux adapter as a substitute for the textual background prompt. This allows the global scene style to be initialized directly from a reference image while maintaining LayerBind's structural control. Then, LayerBind integrates seamlessly with IP-Adapter. By injecting visual priors from the IP-Adapter into specific instance regions, LayerBind simultaneously governs both the spatial layout and the specific visual reference.

D.3. Generation with Transparent Instances

Fig. 17 showcases some generation results with transparent instances/reflection, LayerBind can naturally handle transparent object generation without occlusion scenarios. When there is occlusion, since transparent objects are difficult to estimate region alpha using background differences, we directly use direct paste to handle the occluded regions. We found that under the subsequent nursing mechanism, LayerBind can also handle occlusion relationships between transparent instances.

Figure 17. Visualization of generating transparent instances.

D.4. Implementation of Position Editing

As shown in Fig. 16 (c), by making the implementation adjustments, LayerBind can handle instance position editing. The Branch mechanism of LayerBind is built on a shared noisy latent of the regions; therefore, modifying the region coordinates will change the generation trajectory. If it wishes to preserve the originally generated instance features during position editing, alpha estimation should be performed on the instance region before blending it into the new region, rather than directly changing the region coordinates of the instance.

D.5. Complex Scene Generation (>10 Instances)

In Fig. 18, we present the results of using LayerBind to generate more instances of complex layouts. We find that when the input layout itself is spatially logical (i.e., all objects are correctly positioned in the background in a logically reasonable manner), LayerBind is able to accurately handle complex arrangements with more than 10 instances. However, with such a large number of instances, most of the input cases are counterfactual. Since LayerBind is a training-free method that only uses the context sharing ability inherent in the model itself, it struggles to handle generation scenarios outside of its training data distribution, which can lead to artifacts or binding failures.

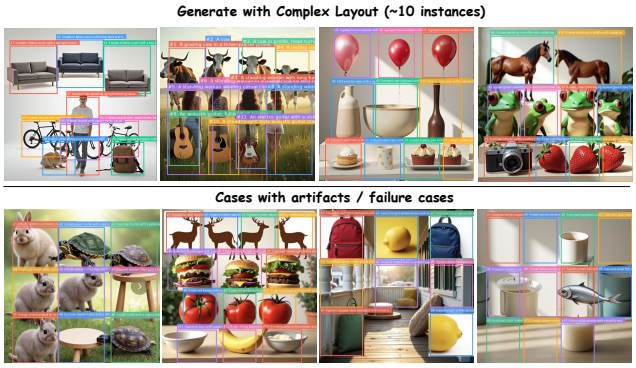

Figure 18. LayerBind successfully handles rational complex spatial relationships, while failing in counterfactually incorrect layout arrangements.

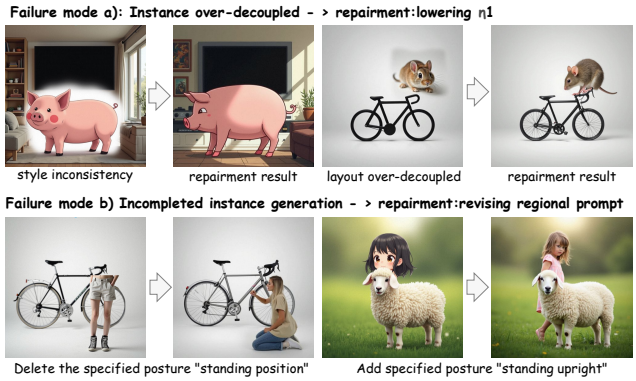

Figure 19. Illustration of typical failure cases and repair measures.

E. Limitations

Fig. 19 illustrates representative failure cases encountered by LayerBind, which can be summarized as follows:

- **Instance-Background over-decoupling:** In some results, we observe a stylistic or structural detachment between the instance and the background. This typically arises when η_1 is set too high, leaving the subsequent nursing phase insufficient capacity to restore global harmony. Reducing η_1 (sacrificing a degree of rigid layout control) effectively mitigates this issue.
- **Incomplete instance generation:** The successful instance generation is dependent on the alignment between its regional prompt and its spatial location. For example, as shown in the figure, depending on the position of the human, adjusting their poses in prompts can avoid incomplete generation.

Beyond the visualized cases, we note that LayerBind struggles with "Dense Layout" scenarios common in traditional Layout-to-Image benchmarks [48, 50, 59, 64]. This limitation stems from our core design: decoupling background and instance generation makes it challenging to maintain holistic consistency in highly cluttered scenes. Nevertheless, we maintain that LayerBind's primary strength lies in *customized generation* scenarios as a training-free controller, rather than dense scene synthesis. Future work could explore integrating LayerBind's mechanics with model fine-tuning strategies to achieve stronger global coherence while retaining precise regional and occlusion control.

F. More Visualizations

Finally, we show cases of more visualization results on the evaluated datasets, in Figs. 20, 21, and 22.

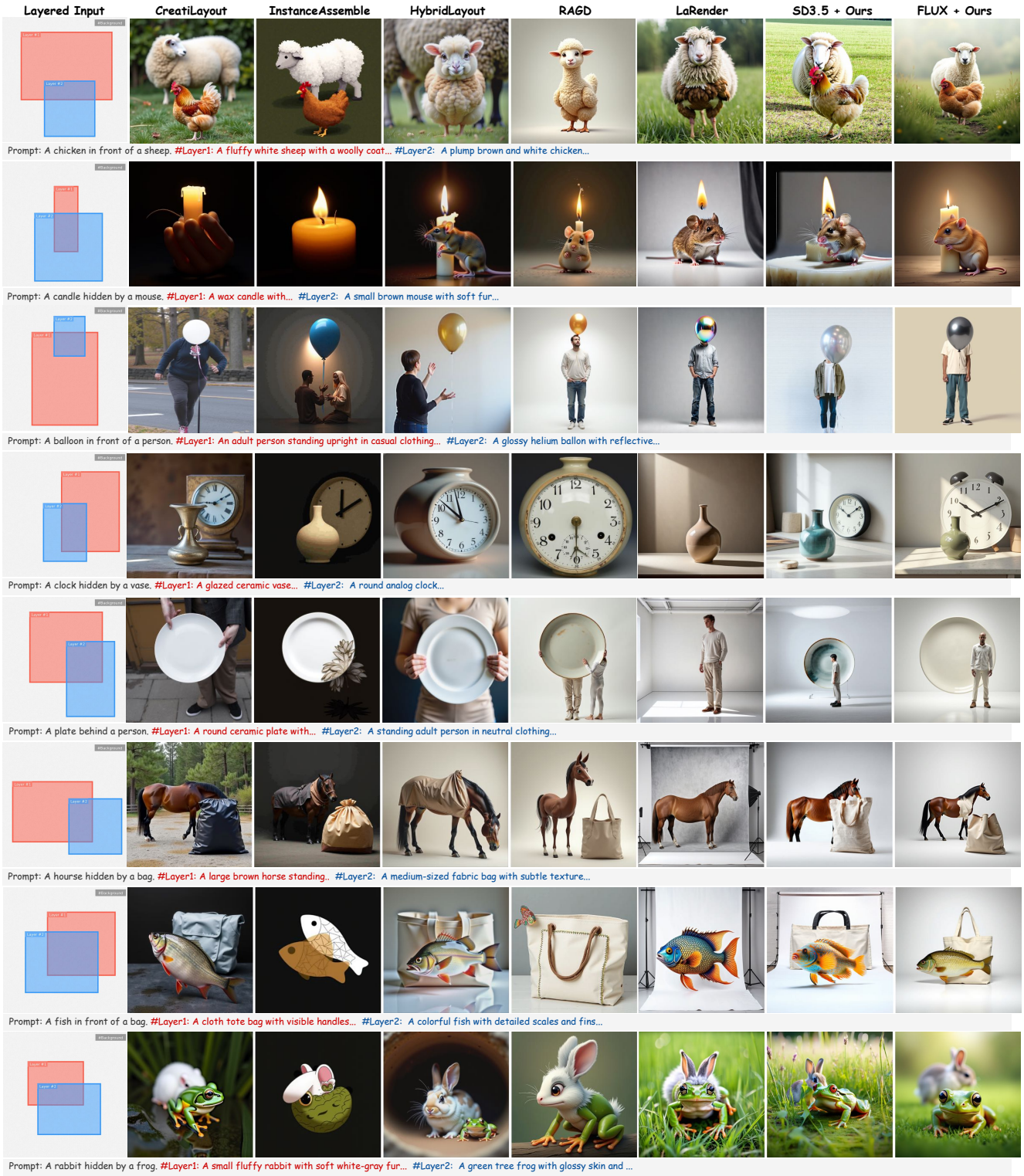

Figure 20. Visualization of occlusion control abilities on T2ICompBench-3D dataset.

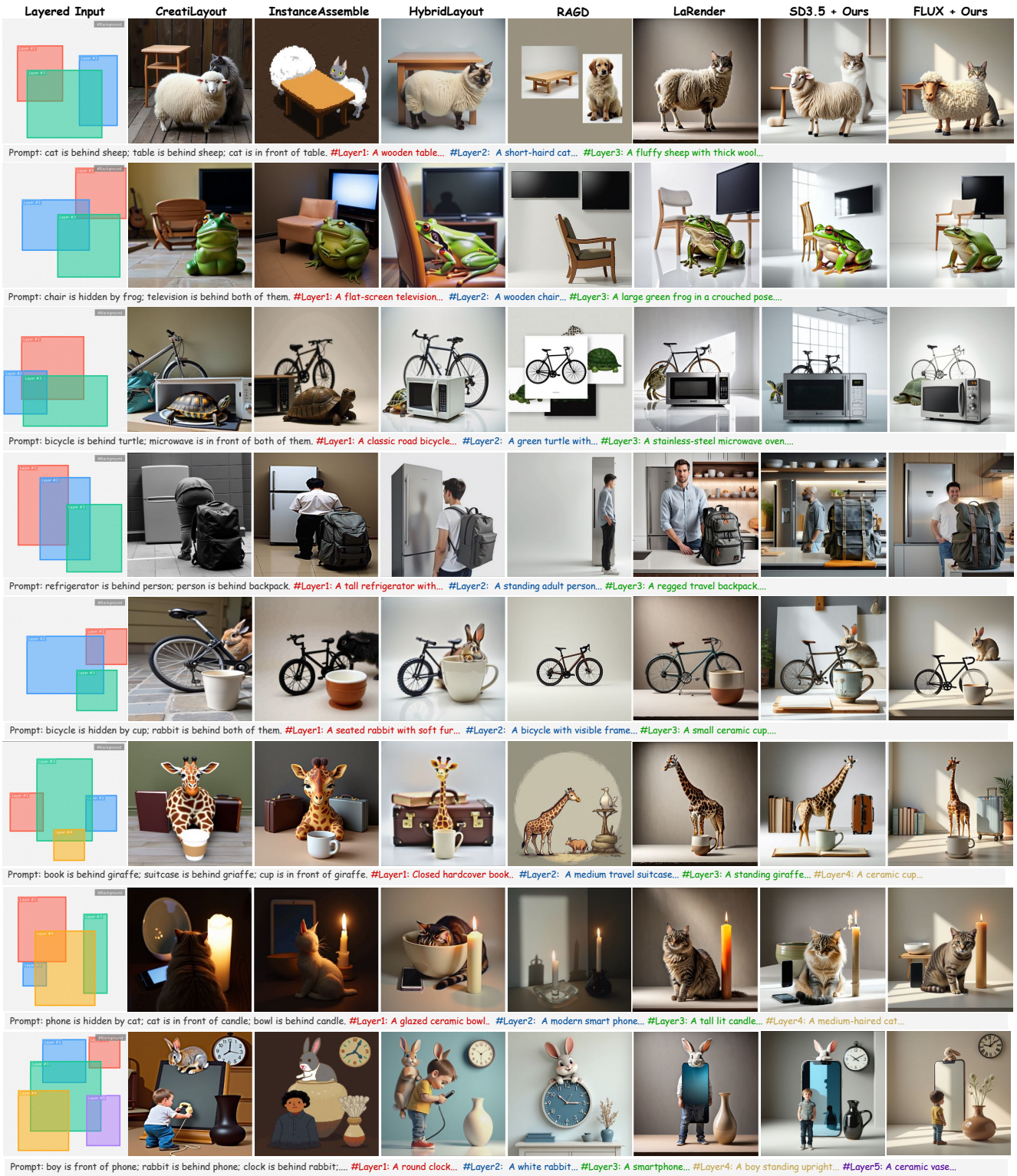

Figure 21. Visualization of occlusion control abilities on BindBench dataset.

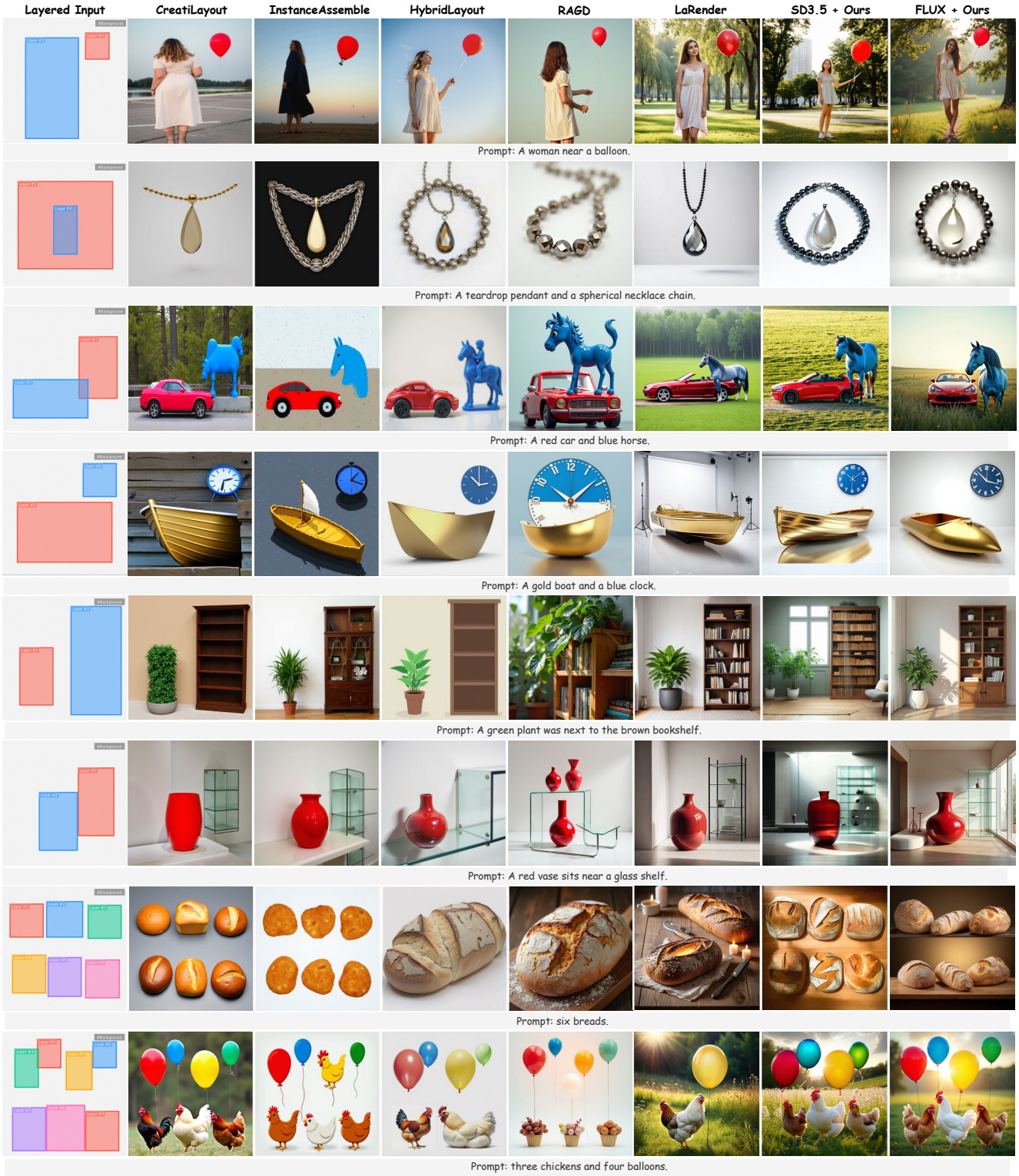

Figure 22. Visualization of T2I alignment tasks on T2ICompBench dataset.