PLANYOURPIXEL: STRUCTURED VISUAL PLANNING FOR FINE-GRAINED T2I GENERATION CONTROL AND FAILURE LOCALIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent advancements in text-to-image (T2I) generation have improved visual quality, yet fine-grained control remains a fundamental challenge. Current layout-conditioned models struggle with numerical reasoning, attribute binding, and spatial grounding. These failures stem from the inherent ambiguity of natural language, where abstract prompts result in underspecified layouts. Even with LLM-based planning, the absence of an underlying scene logic causes models to assign coordinates before resolving the semantic relationships between objects, leading to floating items, incorrect spatial positioning, and attribute leakage. To bridge this gap, we introduce PlanYourPixel (PYP), an agent-based framework that utilizes specialized roles to decompose natural text into a structured scene representation. By resolving semantic constraints into a verifiable plan, PYP provides a unified foundation for two applications: layout-guided generation for image synthesis and automated atomic question generation for fine-grained failure localization. We evaluated our paradigm on PYP-Bench, a diagnostic suite of 1,800 prompts and corresponding layouts. Our results demonstrated that PYP-guided image generation consistently outperforms standard layout baselines. PYP generated 12,503 questions and formed a diagnostic framework that uncovers critical failure patterns that aggregate metrics typically conceal. Our code and data are located at `https://figshare.com/s/f64f422eda57b012b983`

1 INTRODUCTION

Modern Text-to-Image (T2I) models [19; 18; 13] can generate highly realistic images from natural language prompts. However, visual quality alone does not ensure scene-level adherence to the objects, attributes, and relationships described in the prompt. This remains particularly challenging for prompts involving multiple objects, fine-grained attributes, and spatial relations. Such prompts often leave visual and geometric structure implicit, requiring models to infer how linguistic descriptions should translate into a coherent scene.

Layout-guided generation reduces this gap by separating spatial planning from image synthesis. However, its effectiveness depends directly on the quality of the predicted layout. Existing methods either predict bounding boxes directly with LLMs [17; 4; 6; 20] or learn spatial arrangements from layout distributions [9; 14; 21]. In both cases, methods often proceed from extracted objects and coarse relations directly to layout generation, without explicitly resolving scene context, relation directionality, object geometry such as aspect ratio and relative scale, or finer spatial constraints such as alignment and contact.These layout errors can propagate to image synthesis, where inadequate box geometry may cause attribute binding failures, while incorrect placement can lead to spatial relation failures. As illustrated in Figure 1, a baseline layout misplaces the plane relative to the girl, while the PYP layout preserves the intended right-of relation.

A similar limitation appears in evaluation. Existing benchmarks assess compositional capabilities such as probing multiple objects, attributes, and spatial relations [10; 8; 15; 23], but often stop at skill-level assessment. This can hide distinct errors within the same skill. As shown in Figure 1, an attribute failure may arise from generating the wrong property or binding it to the wrong object, while a spatial failure may result from an incorrect object arrangement. Fine-grained diagnosis

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

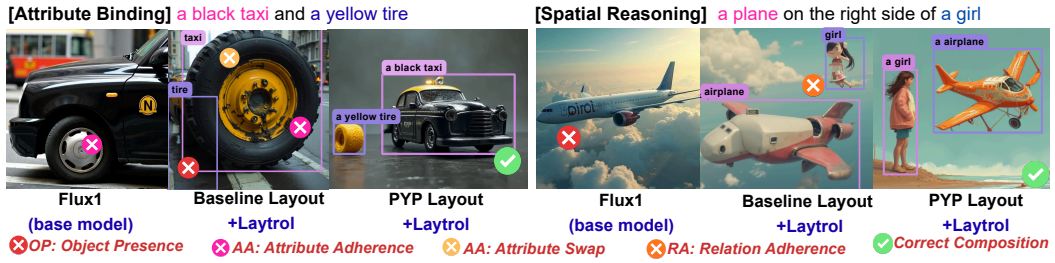

Figure 1: Qualitative comparison of our work PYP and the LaYouSyn baseline [21] on prompts from T2I-CompBench [10]. For each prompt, we compare (1) the Flux1 base model, (2) its L2I variant, Laytrol [11], conditioned on the LaYouSyn layout, and (3) Laytrol conditioned on the PYP layout. PYP better preserves prompt adherence; colored × markers show failures localized by PYP-VQA.

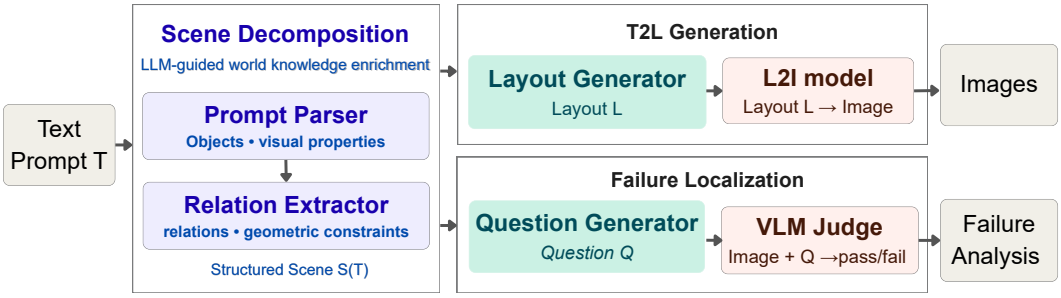

Figure 2: **PYP workflow.** A text prompt is decomposed into a scene representation, which is then shared across two branches: constraint-aware T2L generation and fine-grained failure localization.

reveals what failed and where, providing actionable feedback, that can guide prompt or layout refinement to generate better images or support targeted model fine-tuning.

We introduce *PlanYourPixel* (PYP), a structured visual planning framework centered on scene decomposition. As shown in Figure 2, PYP decomposes a text prompt into a structured scene representation enriched using LLM-guided world knowledge. Compared to existing approaches, our scene representation explicitly captures detailed object properties, such as aspect ratios, and inter-object geometric constraints, while leveraging LLM world knowledge to infer implicit scene context and spatial requirements. This shared layout representation supports two complementary applications. A *constraint-aware T2L planner* uses the structured scene to guide layout generation, while *PYP-VQA* converts the same scene requirements into targeted questions for failure-localized evaluation. By sharing the scene representation across generation and evaluation, PYP provides a common basis for coherent layout planning and fine-grained diagnosis of prompt-adherence failures. Our main contributions are as follows:

- We introduce **PlanYourPixel (PYP)**, an agent-based framework for grounded layout generation. PYP enriches object-level properties and converts inter-object relations into explicit pairwise geometric constraints that can guide and validate box placement.
- We introduce **PYP-VQA**, a failure-localization method, and **PYP-Bench**, a benchmark containing 1,800 prompts and 12,503 atomic binary questions across seven failure types. Each question is paired with a prompt-grounded expected answer, enabling fine-grained diagnosis of visual reasoning failures.
- Experiments show that PYP improves numerical and spatial layout accuracy over baseline layout planners and yields much improved downstream prompt adherence across multiple L2I backbones. PYP-VQA localizes distinct failures across T2I and L2I models, shows stronger agreement with human judgments and exposed systematic weaknesses in state-of-the-art (SOTA) T2I models.

2 PYP FRAMEWORK

PlanYourPixel (PYP) is an agent-based framework for layout-guided text-to-image (T2I) generation and fine-grained failure localization. Given a prompt T, sequential role-based agents construct an explicit scene representation $S(T)$, with each agent handling a distinct reasoning task. This makes the scene structure explicit and verifiable. The same representation supports both generation and evaluation, as shown in Figure 3.

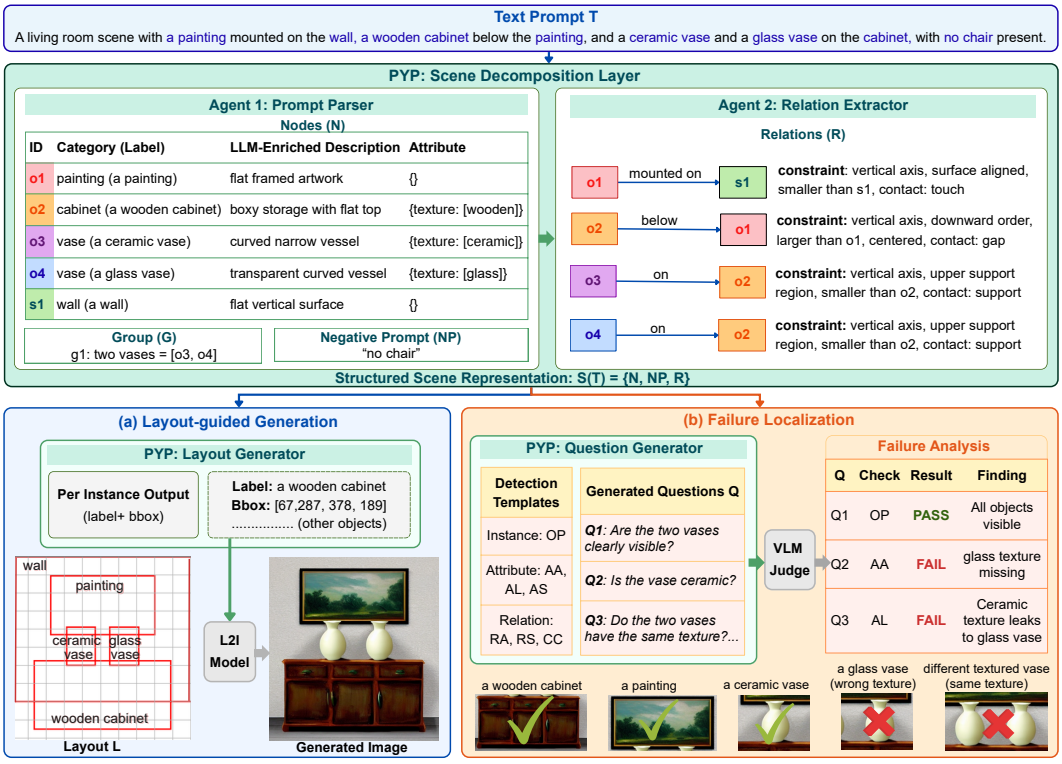

Figure 3: **Overview of the PYP framework.** Given a text prompt T, the Scene Decomposition Layer extracts objects, attributes, and relations to form a structured scene representation $S(T)$. This representation supports two tasks: (a) Layout-guided Generation, where the PYP Layout Generator produces a scene layout L to guide any L2I model and (b) Failure Localization, where a VLM judge answers atomic visual queries Q to identify fine-grained errors in the generated image.

2.1 SCENE DECOMPOSITION

This module transforms an unstructured text prompt T into a structured scene element set $S(T) = \{N, NP, R\}$. Serving as a unified blueprint for downstream tasks, this shared representation extracts explicit textual details and leverages real-world knowledge to enrich geometric constraints, resolving inherent spatial ambiguities.

Prompt Parser. The Prompt Parser scans the text prompt T to extract core entities and establish the initial node set (N). It represents each node as a dictionary (see Figure 3, *Agent 1: Prompt Parser*), containing a unique identifier, category, label, LLM-enriched description, and a nested attribute dictionary with prompt-specified properties (e.g., color, texture, and shape). Beyond the explicit text, the agent enriches each description with visual cues that are useful for layout planning but may be implicit in the prompt. As illustrated in Figure 3, the cabinet is described as *boxy with a flat top*, providing a suitable support surface for the vases, while the glass vase is described as transparent. Such descriptions also capture cues such as orientation, aspect ratio, and shape proportions. The parser categorizes entities into two types: (a) **Objects** (id: $o\#$), which represent distinct, countable entities.The agent decomposes plurals and vague quantities into individual nodes with unique IDs, e.g., o_3: vase and o_4: vase; (b) **Stuff** (id: $s\#$), which represents uncountable, non-bounded surfaces such as walls or floors that serve as physical support or spatial media, e.g.,

s_1: wall. To preserve quantity information, related object nodes are mapped under a shared group identifier to form quantity groups (G). Finally, explicit exclusion statements are compiled into a negative prompt string (NP)for negative grounding and hallucination control.

Relation Extractor. Using the parsed node set, the Relation Extractor identifies explicit interactions and spatial dependencies between entities (see Figure 3, *Agent 2: Relation Extractor)*. It represents each relation as a directed edge from a subject to an object through a predicate label (e.g., $o_2 \rightarrow$ below $\rightarrow o_1$). However, the predicate alone does not fully specify the intended geometry. For example, in Figure 3, the prompt states that the cabinet is *below* the painting, but does not indicate how large the gap should be, whether the objects should be center-aligned, or how their relative sizes should differ. PYP therefore uses the LLM to enrich each relation with explicit geometric constraints that resolve these missing spatial details before layout generation. The resulting relation-level plan provides the Layout Generator with clear guidance for object placement, scale, alignment, and contact (see Appendix A.1.2).

2.2 Application 1: Layout-Guided Generation

As shown in Figure 3(a), PYP uses the shared structured scene representation $S(T) = \{N, NP, R\}$ to construct a layout L for image synthesis. The resulting layout is supplied together with the input prompt T to an L2I model as spatial guidance.

Layout Generation. The Layout Generator agent converts the structured scene representation into object-level layout specifications. Each object is represented as $[\text{label}, \text{bbox}]$, where $\text{bbox} = [x, y, w, h]$ specifies its position and extent on a 512×512 canvas. Object dimensions are initialized from the object type and metadata in N, while the relations and geometric constraints in R determine relative size, ordering, alignment, and contact.
The agent maps these qualitative constraints into coordinates in three stages. First, it adjusts object dimensions to satisfy relative-size constraints while preserving the intended proportions. Second, it maps relational constraints onto the dominant spatial axis: vertical relations constrain y-coordinates, horizontal relations constrain x-coordinates, and alignment constraints regulate the orthogonal axis when needed. For example, in Figure 3(a), the cabinet is placed below the painting, while the two vases are positioned within the upper support region of the cabinet. Third, the agent enforces the specified contact mode at the relevant boundary of the bounding box. A *gap* constraint maintains separation, *touch* aligns adjacent boundaries, and *support* anchors the supported object to the upper boundary of its supporting object. The generated layout is then checked against the specified size, ordering, alignment, and contact constraints, and inconsistent coordinates are revised before rendering. The complete set of supported geometric constraints is provided in Appendix A.1.3.

2.3 Application 2: Failure Localization

As shown in Figure 3(b), PYP uses the structured scene representation $S(T) = \{N, NP, R\}$ to generate targeted visual queries for fine-grained evaluation. These queries cover seven failure categories across object presence, attribute binding, and relation-level consistency.

Question Generation for Failure Localization. The Question Generator converts $S(T)$ into a set of atomic binary questions $Q = \{q_1, q_2, \ldots, q_m\}$. Each failure category is associated with a parameterized question template, summarized in Table 1. The generator selects the applicable templates and fills their slots using the objects, attributes, and relations extracted in N and R. Each instantiated question is paired with its expected Yes/No answer. For example, in Figure 3(b), the presence of the two vase instances triggers the multi-instance Object Presence template in the first row in Table 1, producing *"Focusing on the two vases: are they all clearly visible?"* with an expected answer of `Yes`. The same scene representation also provides the object attributes and relations needed to instantiate attribute and relation-specific queries (see the rest of the table). A VLM judge answers each generated question for the synthesized image, and its response is compared with the expected answer. A mismatch identifies the corresponding failure category and the involved object or relation, enabling localized diagnosis rather than only an image-level quality score.

3 PYP BENCHMARK

We introduce *PYP-Bench*, which aims to evaluate image generation with fine-grained failure localization beyond the aggregate scores reported by existing benchmarks [10]. It contains 1,800 prompts from T2I-CompBench and 12,503 pyp-generated diagnostic questions. As summarized in Table 1, the questions span seven failure categories. Attribute-binding prompts use Categories 1–4, object-relation prompts use Categories 1 and 5–7, and complex-composition prompts use all seven.

Table 1: Failure categories in PYP benchmark with detection templates and expected answers (GT).

ID	Localized Failure	Detects	Detection Template	GT
1. OP	Object Presence	Missing objects or incorrect object count	**Single:** "Focusing on the <object>: is it clearly visible?" **Multiple:** "Focusing on the <count> <objects>: are they all clearly visible?"	Yes
2. AA	Attribute Adherence	Missing or wrongly assigned attributes	"Focusing on the <object>: is the <skill> <attribute_value>?"	Yes
3. AS	Attribute Swap	Attribute assigned to the wrong object	"Focusing on the <target_object>: is the <skill> <reference_attribute>?"	No
4. AL	Attribute Leakage	Attribute unintentionally shared across objects	"Focusing on the <object_A> and <object_B>: do they have the same <skill>?"	No
5. RA	Relation Adherence	Incorrect spatial or semantic relation	"Focusing on the <subject> and the <object>: is the <subject> <relation> the <object> in the image?"	Yes
6. RS	Relative Size	Incorrect scale relationship between related objects	"Focusing on the <subject> <relation> the <object>: is their relative size consistent with the scene and camera perspective?"	Yes
7. CC	Contact Consistency	Incorrect physical contact, separation, or containment	**Support:** "Does the image show the <subject> physically supported by or resting on the <object>?" **Gap:** "Are the <subject> and the <object> clearly separated without unintended physical contact?" **Occupied:** "Does the image show the <subject> clearly contained or enclosed within the interior space of the <object>?"	Yes

4 EVALUATION

We conducted a comprehensive evaluation of PlanYourPixel (PYP) across three dimensions: the quality of our agent-based layout generation, a comparison between layout-conditioned and text-only image generation, and a fine-grained failure analysis using the PYP benchmark.

Implementation Details. We use Gemma 4 31B for all PYP agents, with temperature 1.0, top-$p = 0.95$, and top-$k = 64$ across all experiments. The Prompt Parser uses structured prompts with in-context examples, the Relation Extractor uses chain-of-thought prompting, and the Layout Generator uses in-context examples with explicit task instructions. The template-based Question Generator populates the diagnostic templates in Table 1 from the structured scene representation.

4.1 RQ1: CAN PYP GENERATE BETTER LAYOUT DESIGNS COMPARED TO EXISTING FRAMEWORKS?

Dataset. We evaluate PYP on the NSR-1K dataset [4], which consists of 738 spatial and 262 numerical reasoning prompts and layout annotations.

Baselines. PYP is training-free, and we compare it against both training-based planners (*Layout-Transformer [9], Ranni [5], finetuned Llama-3.1-8B [22], LaYouSyn [21]*) and training-free frameworks (*LLM-Grounded Diffusion [17], LLM-Blueprint [7], LayoutGPT [4]*).

Evaluation Metrics. We evaluate PYP on NSR-1K benchmark for numerical and spatial reasoning over generated layouts [4]. Numerical reasoning is measured using precision, recall, and accuracy

Table 2: Baseline comparison on NSR-1K, where PYP outperforms training-based and training-free methods on complex layouts.

Method	Numerical Reasoning			Spatial Reasoning
	Prec. ↑	Rec. ↑	Acc. ↑	Acc. ↑
Training-based methods				
LayoutTransformer* [9]	75.70	61.69	22.26	6.36
Ranni* [5]	56.23	83.28	40.80	53.29
Llama-3.1-8B (finetuned)*	79.33	93.36	70.84	86.64
LaYouSyn (GRIT pretraining) [21]	77.62	**99.23**	95.14	92.58
Training-free methods				
LLMGroundedDiffusion (GPT-4o-mini)* [17]	84.36	95.94	89.94	72.46
LLM Blueprint (GPT-4o-mini)* [7]	**87.21**	67.29	38.36	73.52
LayoutGPT (Llama-3.1-8B) [4]	78.61	84.01	71.71	75.40
LayoutGPT (GPT-4o-mini) [4]	73.82	86.84	77.51	92.01
PYP (ours)	81.29	98.92	**95.96**	**93.29**

Note: Metrics are computed on generated layouts. * Results reported in LayoutGPT [4] and LaYouSyn [21].

over object categories and counts. Following LayoutGPT[4], precision measures the fraction of predicted objects matched by the ground truth. Recall measures the fraction of ground-truth objects recovered by the prediction. For accuracy, an image receives 1 when both the predicted category and count exactly match the ground truth; otherwise, it is 0. Spatial accuracy measures whether each predicted object pair satisfies the ground truth relation, inferred from bounding box centers among *left*, *right*, *above*, and *below*.

RQ1 Results

As shown in Table 2, PYP achieves the highest accuracy on NSR-1K benchmark [4] for both numerical and spatial reasoning, reaching 95.96% and 93.29%, respectively. Compared with fine tuned LLaMA 3.1 8B, PYP improves numerical accuracy by +25.12% and spatial accuracy by +6.65%. Moreover, PYP also achieves 98.92% recall, indicating strong coverage of the ground truth object categories and count. Its comparatively lower precision suggests that some layouts include objects from categories not present in the GT, which is partly consistent with PYP's design, which may introduce anchor, support, or background objects in the layout beyond the benchmark annotations to support better downstream image generation. Overall, PYP more reliably preserves the required object counts and spatial relations. RQ2 further shows that these stronger layouts improve prompt adherence in downstream L2I generation.

4.2 RQ2: Can PYP-generated layout improve prompt adherence in downstream image generation?

Dataset. We evaluate on the T2I-CompBench validation split, comprising 300 prompts spanning six skill categories: color, shape, texture, spatial, non-spatial, and complex compositional relationships.
Baselines. We compare PYP-generated layouts against LaYouSyn[21] layouts as a representative baseline for structured scene planning. We used the two tools to first generate layouts, and then both the layout sets are rendered into images using five L2I backbones: GLIGEN[16], MIGC [25], Siam-Layout Flux[24], and Laytrol [11].
Evaluation Metrics. We adopt the evaluation protocol defined by T2I-CompBench to assess prompt-image alignment. This involves four category-specific metrics: BLIP-VQA for attribute binding (color, shape, and texture), UniDet for spatial relationships, CLIP for non-spatial relationships, and an integrated metric for complex compositions.

RQ2 Results

Quantitative Results. As shown in Table 4, PYP consistently improves layout-guided image generation over LaYouSyn [21] across all four evaluated L2I backbones. The largest gains appear in spatial reasoning, where UniDet improves by more than 20 points on several backbones, including $22.05 \rightarrow 42.27$ on GLIGEN and $24.26 \rightarrow 46.22$ on SiamLayout-FLUX. Attribute-binding scores also increase across color, shape, and texture, with particularly large gains for MIGC. These results indicate that PYP's layout design provides stronger spatial and attribute grounding for downstream

image synthesis. More broadly, stronger layouts as an intermediate representation lead to more reliable and faithful image generation.

Table 3: Comparison of layout-guided image generation between PYP and LaYouSyn across four L2I backbones at 512×512 resolution. Higher scores indicate better prompt-image alignment.

L2I Model	Layout Method	Attribute Binding			Object Relationship		Complex
		Color BLIP-VQA	Shape BLIP-VQA	Texture BLIP-VQA	Spatial UniDet	Non-Spatial CLIPScore	Comp. 3-in-1
Gligen	LaYouSyn	31.50	37.49	45.95	22.05	30.38	27.78
	PYP (Ours)	**35.30** (+3.80)	**38.06** (+0.57)	**47.95** (+2.00)	**42.27** (+20.22)	**30.82** (+0.44)	**28.94** (+1.16)
MIGC	LaYouSyn	21.09	40.60	45.54	24.58	28.21	24.23
	PYP (Ours)	**67.45** (+46.36)	**57.90** (+17.30)	**57.70** (+12.16)	**41.79** (+17.21)	**28.66** (+0.45)	**32.00** (+7.77)
SiamLayout	LaYouSyn	71.48	51.80	66.60	24.26	31.22	36.59
	PYP (Ours)	**77.09** (+5.61)	**54.76** (+2.96)	**66.62** (+0.02)	**46.22** (+21.96)	**31.35** (+0.13)	**36.84** (+0.25)
Laytrol	LaYouSyn	71.18	49.46	62.87	25.72	**30.31**	35.97
	PYP (Ours)	**76.22** (+5.04)	**52.70** (+3.24)	**66.05** (+3.18)	**41.21** (+15.49)	**30.31** (+0.00)	**36.74** (+0.77)

Note: **Green** indicates improvement over LaYouSyn; red indicates lower performance.

Qualitative Results. Figure 4 showcases some examples and compares PYP with LaYouSyn [21] using the same downstream CreatiLayout generator. In the first example, LaYouSyn produces overlapping boxes for the cow and suitcase (a). This weakens separation between objects leading to poorer attribute binding, the cow and the suitcase both become brown. Moreover, in the turtle-man example, LaYouSyn gives the man a small, distant box which weakens the intended left-of relation (b). Similarly, in the complex example, LaYouSyn places the parrot inside the cage region rather than preserving the required separation, weakening the *next to* relation (c). PYP maintains clearer object separation across these cases. As a result, the final images follow the prompts more closely, consistent with the gains in Table 3. Applying to a generated image, PYP is able to further report the localized markers and reveal failure details that coarse scores cannot capture, motivating the fine-grained analysis in RQ3.

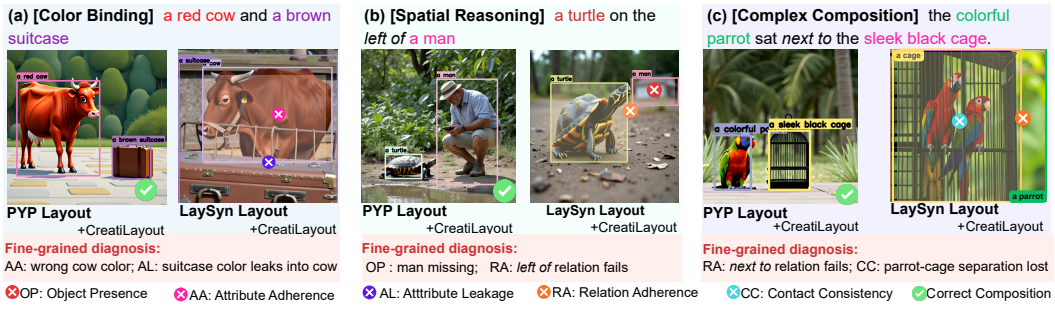

Figure 4: Qualitative comparison of PYP and LaYouSyn [21] using the same generation backbone. PYP-generated layouts lead to more prompt-faithful images.

4.3 RQ3: Can the PYP-VQA framework localize and quantify fine-grained failures in standard and layout-based T2I models?

Dataset: We constructed PYP-Bench, comprising 1,800 prompts from T2I-CompBench, 1,800 corresponding PYP layouts, and 12,503 atomic diagnostic questions. These questions target seven failure types (Table 1).
Baselines: We evaluate two categories of generative models on PYP-Bench. For *Text-to-Image (T2I)* baselines, we include three SOTA open-source models: Stable Diffusion 3.5 Turbo (SD3.5) [1], Flux.1 [12], and Flux2 [13]. For *Layout-to-Image (L2I)* baselines, we include four SOTA layout-conditioned generation methods: Gligen [16], MIGC [25], SiamLayout [24], and Laytrol [11]. We also conducted a case study using three top-ranked models from the T2I leaderboard [2], including GPT Image2.5 [19], Nano Banana2 [3], and MAI2.6 [18]. For each model, we used 10 challenging PYP-Bench prompts for each of the 6 skills and generated one image per prompt, resulting in 60 images per model.
Evaluation Metrics: We report failure counts using both the standard T2I-CompBench metrics and our proposed *PYP-VQA* metrics. T2I-CompBench evaluates attribute binding for color, shape, and texture using *BLIP-VQA* (Table 5, top panel), spatial relations using *UniDet*, non-spatial relations using *CLIP*, and complex compositions using the *3-in-1* score (Table 5, bottom panel). For *PYP-VQA*, Qwen3.6 evaluates each diagnostic question and assigns a question-level score, which we aggregate into the final *PYP-VQA* score (see

Appendix A.2.2 for details). We classify images with *PYP-VQA*score < 0.5 as *Fail* and *Pass* otherwise. We also measure agreement between *PYPScore* and human judgement from the T2I-CompBench human evaluation set using *Kendall's* τ and *Spearman's* ρ.

RQ3 Results

Previous Metrics vs. PYP-VQA. To assess the reliability of PYP-VQA, we measure its agreement with human judgments using Kendall's τ and Spearman's ρ. Table 4 compares PYP-VQA with existing T2I-CompBench metrics across compositional skills. While specialized metrics such as BLIP-VQA, UniDet, and 3-in-1 show strong agreement for the specific skills they evaluate, their coverage is limited (we marked "-" s in the table). Moreover, mGPT-CoT provides broader coverages across all categories, but its correlation with human judgments is considerably weaker. PYP-VQA bridges this gap by covering all compositional skills while maintaining strong agreement with human judgments across different VLM backbones. Beyond this broader and more reliable evaluation, PYP-VQA also localizes the source of individual failures. For example, an attribute-color failure reported by existing approaches can be traced to a *missing object*, *incorrect color assignment*, *attribute swapping*, or *unintended attribute sharing* in PYP. Thus, PYP-VQA identifies not only *which* compositional skill fails, but also *how* it fails, providing finer-grained diagnostic feedback than existing aggregate metrics.

Table 4: Evaluating the correlation of automated metrics with human judgement on T2ICompBench

Metrics	Att-Color		Att-Shape		Att-texture		Spatial rel		Non-spatial		Complex	
	$\tau\uparrow$	$\rho\uparrow$	$\tau\uparrow$	$\rho\uparrow$	$\tau\uparrow$	$\rho\uparrow$	$\tau\uparrow$	$\rho\uparrow$	$\tau\uparrow$	$\rho\uparrow$	$\tau\uparrow$	$\rho\uparrow$
CLIP	0.1938	0.2773	0.0555	0.0821	0.2890	0.4008	0.2741	0.3548	0.2470	0.3161	0.0650	0.0847
BLIP-VQA	**0.6297**	0.7958	0.2707	0.3795	0.5177	0.6995	–	–	–	–	–	–
UniDet	–	–	–	–	–	–	**0.4756**	0.5136	–	–	–	–
3-in-1	–	–	–	–	–	–	–	–	–	–	0.2831	0.3853
mGPT-CoT	0.3156	0.4151	0.1300	0.1805	0.3453	0.4664	0.1096	0.1239	0.1947	0.2137	0.1251	0.1463
PYP-Gemma4	0.5830	0.7240	**0.4155**	0.5324	0.5278	0.6741	0.4420	0.5468	0.3412	0.4142	0.4742	0.6116
PYP-Qwen3.5	0.6262	**0.8016**	0.3910	0.5390	0.5865	0.7720	0.4324	0.5553	**0.4036**	**0.5056**	0.5538	0.7044
PYP-Qwen3.6	0.6227	0.7962	0.4039	**0.5555**	**0.6056**	**0.7911**	0.4387	**0.5657**	0.3967	0.4992	**0.5612**	**0.7128**

Case study with SOTA models: Table 4.3 shows that recent T2I models still exhibit compositional failures across multiple skills. More importantly, Figure 5 shows that the same attribute-binding prompt can fail in different ways. For *"a red horse and a brown orange,"* GPT Image 2.5 preserves the red horse but misses the intended color of the orange [AA]. MAI Image 2.6 instead assigns the orange's intended brown attribute to the horse [AA, AS], while Nano Banana2 renders both objects with the brown attribute [AA, AS, AL]. A coarse color-binding score would mark all three as failures without revealing how they differ. PYP-VQA exposes these differences by identifying what failed and where, providing a more interpretable diagnosis than a single skill-level score (see Appendix A.2.3).

Method	Color					Shape					Texture				
	Prompt	OP	AA	AL	AS	Prompt	OP	AA	AL	AS	Prompt	OP	AA	AL	AS
GPT Image2.5	6	0	6	0	1	2	0	1	0	0	6	0	0	0	0
MAI 2.6	6	0	6	2	3	7	0	4	2	4	5	0	2	1	3
Nano Banana2	8	0	6	5	5	9	1	9	0	3	6	0	2	1	3

Method	Spatial					Non-spatial					Complex							
	Prompt	OP	RA	RS	CC	Prompt	OP	RA	RS	CC	Prompt	OP	AA	AL	AS	RA	RS	CC
GPT Image2.5	**1**	0	1	5	4	**2**	1	2	1	5	6	1	1	0	0	6	4	4
MAI 2.6	2	1	2	6	3	3	1	2	0	5	5	2	1	0	0	5	4	4
Nano Banana2	3	0	2	5	3	3	2	3	2	6	4	2	1	0	0	4	4	3



Figure 5: Qualitative failure localization across SOTA T2I models for the same attribute-binding prompt.

Overall Failure Analysis. Table 5 uses PYP-VQA to localize failures across three T2I and four L2I models. The fine-grained analysis reveals two key findings. (1) **Layout conditioning consistently improves spatial control, but not all spatial errors equally.** L2I models achieve the two lowest spatial failure counts. Within the Flux family, Laytrol and SiamLayout are layout-conditioned variants built upon Flux1. Laytrol, for example, reduces spatial failures from 1,894 $\rightarrow$ 1,151 relative to Flux1. PYP-VQA shows that this gain comes from Relation Adherence (RA), dropping from 1,325 $\rightarrow$ 374, and Contact Consistency (CC), from 979 $\rightarrow$ 479. Object Presence (OP) also improves, but Relative Size (RS) remains a bottleneck. This shows that layout guidance improves placement and relational control, but some geometric errors persist. (2) **These gains still depend on the base generator.** Despite the gains from layout control, Flux2 achieves the lowest failure count

in five of the six skill groups. This pattern is especially visible in attribute binding. GLIGEN and MIGC use older Stable Diffusion backbones and retain substantial Attribute Adherence (AA), Attribute Swap (AS), and other binding errors. Similarly, SiamLayout and Laytrol improve over Flux1, yet Flux2 remains stronger in several categories. Thus, layout guidance improves *where* objects appear, but the fidelity of *how* they are rendered still depends strongly on the base model. Finally, these results highlight the value of fine-grained failure localization. Aggregate failure counts show whether performance changes, but not *why*. PYP-VQA reveals which capabilities improve and which failures persist.

Table 5: Failure analysis on PYP-Bench across three groups. Colored section headers denote Attribute Binding, Object Relationship, and Complex Composition. Results are raw failure counts over 3,000 samples. Methods above the dashed line are T2I baselines and those below are L2I.

Attribute Binding *Color · Shape · Texture*

Method	Color						Shape						Texture					
	Total Failure		Localized Failure				Total Failure		Localized Failure				Total Failure		Localized Failure			
	BLIP-VQA	PYP-VQA	OP	AA	AL	AS	BLIP-VQA	PYP-VQA	OP	AA	AL	AS	BLIP-VQA	PYP-VQA	OP	AA	AL	AS
SD3.5 [1]	541	738	209	551	49	192	1319	1746	308	1485	144	370	796	994	329	728	39	116
Flux1 [12]	631	763	168	628	78	217	1571	1731	351	1512	252	439	1133	1247	406	1018	78	177
Flux2 [13]	182	**236**	55	159	23	62	981	**876**	73	621	80	223	416	**323**	84	173	17	41
Gligen [16]	1994	2202	368	2111	382	1255	1924	2094	616	1900	367	531	1526	1615	408	1374	119	320
MIGC [25]	754	869	45	736	122	239	830	1500	173	1286	85	230	926	928	143	736	16	110
Siam-Layout [24]	503	596	126	464	54	161	1260	1330	142	1076	156	327	829	658	124	481	36	73
Laytrol [11]	483	464	79	343	32	96	1313	1352	181	1100	130	274	877	771	126	567	40	91

Object Relationship *Spatial · Non-Spatial* **Complex Composition** *All seven failure categories*

Method	Spatial						Non-Spatial						Complex Composition							
	Total Failure		Localized Failure				Total Failure		Localized Failure				Total Failure		Localized Failure					
	UniDet	PYP-VQA	OP	RA	RS	CC	CLIP	PYP-VQA	OP	RA	RS	CC	BLIP-VQA	PYP-VQA	OP	AA	AL	AS	RA	RS	CC
SD3.5 [1]	2223	2092	274	1249	940	875	—	1246	598	602	458	513	—	1518	589	486	23	9	902	737	509
Flux1 [12]	1900	1894	357	1325	869	979	—	986	405	468	283	471	—	1307	472	343	47	51	759	629	474
Flux2 [13]	1528	1246	28	465	463	577	—	**665**	103	193	94	1450	—	**910**	205	144	27	4	419	346	322
Gligen [16]	1358	1393	153	496	854	590	—	935	196	582	328	374	—	1810	519	1016	144	239	983	775	482
MIGC [25]	—	**1096**	39	377	694	488	—	1187	212	830	368	355	—	1273	176	379	29	35	731	499	325
Siam-Layout [24]	1526	1249	92	334	761	459	—	731	174	302	149	338	—	1106	295	262	21	4	694	505	317
Laytrol [11]	1664	1151	54	374	731	479	—	881	257	473	254	340	—	1156	220	232	23	14	701	443	313

Notes: OP = Object Presence, AA = Attribute Adherence, AL = Attribute Leakage, AS = Attribute Swap, RA = Relation Adherence, RS = Relative Size, CC = Contact Consistency. — = unavailable. **Green** and blue indicate the best and second-best PYP-VQA failure counts.

5 Related Work

Text-to-Image and Layout-to-Image Models: Text-to-image (T2I) models have made substantial progress in high-fidelity image synthesis, yet still struggle with multi-object composition, attribute binding, and spatial relations. Layout-to-image (L2I) methods address these limitations by conditioning pretrained T2I models on explicit object layouts, typically bounding boxes [16; 25; 11; 24]. However, their performance depends heavily on input layout quality, motivating open-vocabulary text-to-layout generation.

Text-to-Layout Generation: LLM-based T2L planners [17; 4; 6; 20] directly predict object bounding boxes from text. This leaves object proportions and inter-object geometry implicit in coordinate generation. Training-based methods [9; 14; 21] instead learn spatial arrangements from prior layout distributions. In both cases, box placement occurs without first explicitly resolving the prompt's geometric requirements, so relations such as relative size or support must be inferred during placement. PYP instead converts each inter-object relation into an explicit pairwise geometric constraint that guides placement and enables direct layout validation.

Text-to-Image Evaluation: Recent benchmarks evaluate compositional skills such as attribute binding, object relations, counting, and spatial reasoning [10; 8; 15; 23]. However, most remain at the skill level and do not reveal the cause of failure. For example, an attribute-binding failure may arise from an incorrect attribute or leakage to another object, while a relation failure may stem from incorrect ordering, relative scale, or physical contact. This limits failure localization and targeted model improvement. PYP-VQA addresses this gap by decomposing each skill into explicit failure types for targeted diagnosis.

6 Conclusions

In this work, we address prompt-adherence limitations in text-to-image generation through a layout-guided agentic framework for both generation and evaluation. We find that existing LLM-based planners often leave object properties, spatial relations, and relative scale implicit during layout construction.Our results show that resolving object properties and pairwise geometry constraints before box generation improves numerical and spatial layout accuracy and leads to images that better follow the prompt. We further show that skill-level scores can conceal distinct failure modes. PYP-VQA localizes these errors across T2I and L2I models, provides interpretable failure diagnoses and reveals systematic weaknesses even in SOTA T2I models.

AI USE STATEMENT

In this work, we used generative AI tools as part of the proposed framework for scene decomposition, layout generation, and image evaluation. LLM and VLM models were used to operate the agents in PYP, generate structured scene representations and layouts, and answer diagnostic questions for evaluating generated images. We also used generative AI tools during research development and manuscript preparation. These tools assisted with debugging and revising code, as well as editing and polishing the writing for clarity, grammar, and style. All AI-assisted code and outputs used in our experiments were reviewed and tested by the authors, and all AI-assisted writing was reviewed and revised before inclusion in the manuscript. The authors take responsibility for the experimental design, analysis, claims, and final content of this work, including text, code, and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This paper introduces the PYP framework and PYP-Benchmark for guiding image generation and evaluating the compositional capabilities of text-to-image models. Our framework uses open-source LLM and VLM models with human oversight. All prompts and diagnostic question templates are designed and reviewed by human experts to ensure that they are logically consistent, neutral, and free from harmful or personally identifiable information. Our research focuses on improving prompt adherence and identifying and explaining current limitations of text-to-image models.

REPRODUCIBILITY STATEMENT

We support reproducibility by releasing the code used for the proposed framework and evaluation pipeline. The main paper and appendix document the prompts, agent components, scene representation, evaluation procedure, and experimental settings used in our experiments. Additional implementation details, model configurations, and evaluation settings needed to reproduce the reported results are provided in the appendix and supplementary materials. We also provide the code and supporting resources through an anonymous repository for review `https://figshare.com/s/f64f422eda57b012b983`

REFERENCES

[1] Stability AI. Stable diffusion 3.5 large turbo. `https://huggingface.co/stabilityai/stable-diffusion-3.5-large-turbo`, 2024.

[2] Artificial Analysis. Text to image leaderboard, 2026. URL `https://artificialanalysis.ai/image/leaderboard/text-to-image`. Accessed: 2026-09-17.

[3] Google DeepMind. Nano banana 2: Google's latest ai image generation model (gemini 3.1 flash image), 2026. URL `https://blog.google/innovation-and-ai/technology/ai/nano-banana-2/`.

[4] Weixi Feng, Wanrong Zhu, Tsu jui Fu, Varun Jampani, Arjun Akula, Xuehai He, Sugato Basu, Xin Eric Wang, and William Yang Wang. Layoutgpt: Compositional visual planning and generation with large language models, 2023. URL `https://arxiv.org/abs/2305.15393`.

[5] Yutong Feng, Biao Gong, Di Chen, Yujun Shen, Yu Liu, and Jingren Zhou. Ranni: Taming text-to-image diffusion for accurate instruction following, 2024. URL `https://arxiv.org/abs/2311.17002`.

[6] Hanan Gani, Shariq Farooq Bhat, Muzammal Naseer, Salman Khan, and Peter Wonka. LLM blueprint: Enabling text-to-image generation with complex and detailed prompts. In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=mNYF0IHbRy`.

[7] Hanan Gani, Shariq Farooq Bhat, Muzammal Naseer, Salman Khan, and Peter Wonka. Llm blueprint: Enabling text-to-image generation with complex and detailed prompts, 2024. URL `https://arxiv.org/abs/2310.10640`.

[8] Dhruba Ghosh, Hanna Hajishirzi, and Ludwig Schmidt. Geneval: An object-focused framework for evaluating text-to-image alignment, 2023. URL `https://arxiv.org/abs/2310.11513`.

[9] Kamal Gupta, Justin Lazarow, Alessandro Achille, Larry Davis, Vijay Mahadevan, and Abhinav Shrivastava. Layouttransformer: Layout generation and completion with self-attention, 2021. URL `https://arxiv.org/abs/2006.14615`.

[10] Kaiyi Huang, Kaiyue Sun, Enze Xie, Zhenguo Li, and Xihui Liu. T2i-compbench: A comprehensive benchmark for open-world compositional text-to-image generation. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 78723–78747. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/f8ad010cdd9143dbb0e9308c093aff24-Paper-Datasets_and_Benchmarks.pdf.

[11] Sida Huang, Siqi Huang, Ping Luo, and Hongyuan Zhang. Laytrol: Preserving pretrained knowledge in layout control for multimodal diffusion transformers, 2025. URL https://arxiv.org/abs/2511.07934.

[12] Black Forest Labs. Flux.1 [dev]. https://huggingface.co/black-forest-labs/FLUX.1-dev, 2024.

[13] Black Forest Labs. Flux.2: Frontier visual intelligence. https://huggingface.co/black-forest-labs/FLUX.2-dev, 2025.

[14] Lauren Levine and Amir Zeldes. Llmbridge: An llm pipeline for end-to-end referential bridging resolution in english, 2026. URL https://arxiv.org/abs/2605.29048.

[15] Baiqi Li, Zhiqiu Lin, Deepak Pathak, Jiayao Li, Yixin Fei, Kewen Wu, Tiffany Ling, Xide Xia, Pengchuan Zhang, Graham Neubig, and Deva Ramanan. Genai-bench: Evaluating and improving compositional text-to-visual generation. *arXiv preprint arXiv:2406.13743*, 2024.

[16] Yuheng Li, Haotian Liu, Qingyang Wu, Fangzhou Mu, Jianwei Yang, Jianfeng Gao, Chunyuan Li, and Yong Jae Lee. Gligen: Open-set grounded text-to-image generation, 2023. URL https://arxiv.org/abs/2301.07093.

[17] Long Lian, Boyi Li, Adam Yala, and Trevor Darrell. Llm-grounded diffusion: Enhancing prompt understanding of text-to-image diffusion models with large language models. *arXiv preprint arXiv:2305.13655*, 2023.

[18] Microsoft AI. Mai-image-2.6: Pushing the quality-cost frontier, September 2026. URL https://microsoft.ai/news/pushing-the-quality-cost-frontier-with-mai-image-2-6/.

[19] OpenAI. Introducing GPT-Image-2.5 Flare and Sunburst. OpenAI API Documentation and Business Announcement, September 2026. URL https://openai.com/index/introducing-chatgpt-images-2-5/.

[20] Leigang Qu, Shengqiong Wu, Hao Fei, Liqiang Nie, and Tat-Seng Chua. Layoutllm-t2i: Eliciting layout guidance from llm for text-to-image generation, 2023. URL https://arxiv.org/abs/2308.05095.

[21] Divyansh Srivastava, Xiang Zhang, He Wen, Chenru Wen, and Zhuowen Tu. Lay-your-scene: Natural scene layout generation with diffusion transformers, 2025. URL https://arxiv.org/abs/2505.04718.

[22] Llama Team. The llama 3 herd of models. *CoRR*, abs/2407.21783, 2024. doi: 10.48550/ARXIV.2407.21783. URL https://doi.org/10.48550/arXiv.2407.21783.

[23] Zengbin Wang, Xuecai Hu, Yong Wang, Feng Xiong, Man Zhang, and Xiangxiang Chu. Everything in its place: Benchmarking spatial intelligence of text-to-image models, 2026. URL https://arxiv.org/abs/2601.20354.

[24] Hui Zhang, Dexiang Hong, Yitong Wang, Jie Shao, Xinglong Wu, Zuxuan Wu, and Yu-Gang Jiang. Creatilayout: Siamese multimodal diffusion transformer for creative layout-to-image generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 18487–18497, 2025.

[25] Dewei Zhou, You Li, Fan Ma, Xiaoting Zhang, and Yi Yang. Migc: Multi-instance generation controller for text-to-image synthesis, 2024. URL https://arxiv.org/abs/2402.05408.

A APPENDIX

The appendix provides additional details on the PYP framework and PYP-Bench. Section A.1 describes the framework components and provides additional examples of layout-guided image generation, while Section A.2 presents benchmark statistics, diagnostic question design, evaluation details, and example.

A.1 PYP Framework

A.1.1 Prompt Parser
A.1.2 Relation Extractor
A.1.3 Layout Generator

A.2 PYP Benchmark

A.2.1 Benchmark Statistics and Analysis
A.2.2 Diagnostic Question Design
A.2.3 Evaluation Protocol
A.2.4 Failure Localization with PYP-VQA

A.1 PYP-FRAMEWORK

In our PYP framework, we used three different agents. We provide the details and prompt template used with Gemma4 27B during prompt decomposition and layout generation. The details on each each agent is given below

A.1.1 PROMPT PARSER

Given a prompt T, the parser organizes the caption using the following components:

- **Things (t_i):** Countable, bounded objects represented as individual nodes. For example, in "a red chair and two vases," the chair and each vase become separate Thing nodes.
- **Stuff (s_i):** Unbounded regions or surfaces that act as support, container, or spatial medium for Things. For example, in "a vase on the table," the table may serve as Stuff when used as the supporting region.
- **Groups (g_i):** Preserve plural or quantity-based references by linking their corresponding Thing IDs. For example, "two vases" may form $g_1 = \{t_2, t_3\}$.
- **Viewpoint:** Records the camera perspective stated or implied by the caption. For example, "top-down view" is preserved as the scene viewpoint; otherwise the predefined default viewpoint is used.
- **Negative Prompt (NP):** Records explicitly excluded objects or attributes. For example, "a table without flowers" adds *flowers* to NP and excludes it from the object nodes.

Prompt Parser Output.

Prompt Parser (Agent 1)
Input Prompt: <*input caption*>
Nodes (N): • **Thing:** `id`: t_i, `category`: `<object name>`, `label`: `<caption phrase>`, `description`: `<visual description>`, `attribute`: `<color/texture/shape>` • **Stuff:** `id`: s_i, `name`: `<region name>`, `label`: `<caption phrase>`, `description`: `<visual description>`, `attribute`: `<color/texture/shape>`
Groups (G): `id`: g_i, `members`: [$t_i, \ldots, t_j$], `name`: `<plural/quantity phrase>`
Viewpoint: `<scene viewpoint>`
Negative Prompt (NP): "`<excluded object/attribute>, ...`"

Table 6: Generic output structure of the Prompt Parser(Agent 1). The parser extracts viewpoint, Thing and Stuff nodes, plural or quantity-based groups, and explicit visual exclusions from the input caption.

A.1.2 RELATION EXTRACTOR

Given the caption T, nodes N, groups G, and viewpoint, the Relation Extractor constructs pairwise relation edges between objects. If a relation involves a group, it decomposes the group into pairwise relations for each member. Each extracted edge contains a directed subject–relation–object triplet together with its grounded geometric constraint.

- **Relation:** Represented as `subject`, `relation`, and `object`. The relation preserves the predicate and direction from the caption. For example, `subject`: A, `relation`: *left of*, `object`: B.
- **Dominant axis:** Specifies the primary spatial dimension: *horizontal*, *vertical*, *depth*, *containment*, or *none*. For example, A left of B uses the horizontal axis, while A inside B uses containment.
- **Ordering and grounded relation:** Specifies where A lies relative to B along the dominant axis, using the grounded spatial interpretation of the predicate. For example, A above B requires vertical top–bottom ordering.
- **Relation-specific rule:** Defines how the relation should be geometrically realized: separation for horizontal/vertical relations, support for *on*, enclosure for *inside*, localized connection for attachment, partial occlusion for depth, and closeness for proximity.
- **Alignment:** Controls placement along the non-dominant axis using one of *centered*, *edge-aligned*, *offset*, or *loosely aligned*. For example, A above B may still be horizontally centered with B.
- **Contact mode:** Specifies box interaction using exactly one of *gap*, *touch*, *support*, or *overlap*. For example, A above B typically uses *gap*, while A on B uses *support*.
- **Relative size and shape:** Uses exactly one size comparison: *smaller*, *larger*, or *comparable*, optionally with one shape comparison such as *taller than*, *wider than*, or *more compact than*.

Output Format.

Relation Extractor (Agent 2)

Input: Caption T, Nodes N, Groups G, Viewpoint

Relations (R):

• `subject`: t_i/s_i, `relation`: `<predicate>`, `object`: t_j/s_j

• `geometric_constraint: <dominant axis, ordering, alignment, contact mode, relative size>`

Geometric Constraint Format:

```
Dominant axis is <axis>. <subject> appears <grounded predicate>
<object> with <ordering>. <relation-specific rule>. <alignment>.
Contact mode is <contact mode>. <relative comparison>.
```

Table 7: Generic output structure of the Relation Extractor. The agent preserves the directed subject–predicate–object relation and grounds each pairwise edge with a geometric constraint describing spatial ordering, alignment, contact, and relative size.

A.1.3 LAYOUT GENERATOR

Given the caption, viewpoint, nodes, and relation edges, the Layout Generator assigns a bounding box $[x, y, w, h]$ to every node on a 512×512 canvas. The layout is constructed using the following components:

- **Bounding box:** Assigns each object a valid box $[x, y, w, h]$ within the canvas. Boxes preserve object visibility and use the available canvas effectively.
- **Size and proportion:** Determines object width and height from object semantics, metadata, canonical proportions, and the relative size specified in the geometric constraints. For example, if A is smaller than B, its box is scaled accordingly while preserving its aspect ratio.
- **Constraint realization:** Uses the grounded `geometric_constraint` as the authoritative placement instruction, preserving its dominant axis, ordering, alignment, contact, overlap, or enclosure. For example, if A is left of B, the horizontal ordering must remain clear.
- **Contact enforcement:** Realizes the specified contact mode: *gap* keeps A and B separated, *touch* joins their boundaries, *support* places A on or against B, and *overlap* allows localized intersection for attachment or occlusion.

- **Structural hierarchy:** Maintains physically plausible support, containment, and attachment structure. Each object uses one clear parent unless multiple parents are explicitly required.
- **Layout refinement:** Expands or shifts boxes when space allows while preserving proportions, relative sizes, and geometric constraints. A final global shift and scale reduce unnecessary canvas margins.
- **Relation validation:** Rechecks all relations after refinement to ensure that dominant axis, ordering, contact, and containment remain satisfied. Only minimal local adjustments are applied when correction is needed.
- **Viewpoint and depth:** Interprets the layout under a 2D photographic viewpoint. Depth cues and occlusion are used only when explicitly required; otherwise objects retain neutral 2D ordering.

Layout Output. Each input node is mapped to exactly one bounding box:

```
{ "nodes": [ {"id": "t1", "bbox": [x, y, width, height]},
{"id": "t2", "bbox": [x, y, width, height]} ] }
```

A.2 PYP-BENCHMARK

A.2.1 BENCHMARK STATISTICS AND ANALYSIS

This section summarizes the composition of PYP-Bench. The benchmark contains 1,800 prompts and 12,503 PYP-generated diagnostic questions spanning seven question categories and six compositional skills. Table 8 shows the distribution of these questions across skills and categories.

Table 8: Distribution of diagnostic questions in PYP-Bench across compositional skills.

	Attribute Binding				Object Relation			
Skill	**OP**	**AA**	**AS**	**AL**	**RA**	**RS**	**CC**	**Total**
Color	637	628	577	299	–	–	–	2,141
Shape	624	606	600	305	–	–	–	2,135
Texture	711	708	692	416	–	–	–	2,527
Spatial	600	–	–	–	300	300	233	1,433
Non-spatial	704	–	–	–	423	427	191	1,745
Complex	673	573	165	102	394	394	221	2,522
Total	**3,949**	**2,515**	**2,034**	**1,122**	**1,117**	**1,121**	**645**	**12,503**

OP: Object Presence; *AA*: Attribute Adherence; *AS*: Attribute Swap; *AL*: Attribute Leakage; *RA*: Relation Adherence; *RS*: Relative Size; *CC*: Contact Consistency. A dash indicates that the diagnostic category is not applicable for the corresponding skill.

A.2.2 QUESTION DESIGN

Before question generation, the object metadata extracted from each prompt is cleaned and manually reviewed. A human reviewer verifies the extracted properties against the original prompt and corrects missing or incorrect entries, ensuring that all explicitly specified object properties are retained. Since PYP already decomposes each prompt into individual objects, attributes, and relations, PYP-VQA uses failure-specific question templates from table 1 populated with these extracted elements. This produces consistent yet prompt-specific diagnostic questions targeting concrete failure modes in the generated image. Rather than assigning a single prompt-level judgment, the questions isolate failures at the object, attribute, and relation levels as follows:

- **Object Presence.** We use separate question templates for single and multiple objects to verify whether the required instances are generated. This allows PYP-VQA to localize missing objects or incomplete object sets as Object Presence (OP) failures.
- **Attribute Binding.** We distinguish different ways an object-attribute binding can fail. Attribute Adherence (AA) checks whether the target object exhibits its intended attribute. Attribute Leakage (AL) checks whether an attribute intended for another object incorrectly appears on the target object. Attribute Swap (AS) checks whether distinct attributes are assigned to the wrong objects. Together, these questions distinguish a missing attribute from leakage and cross-object misbinding.

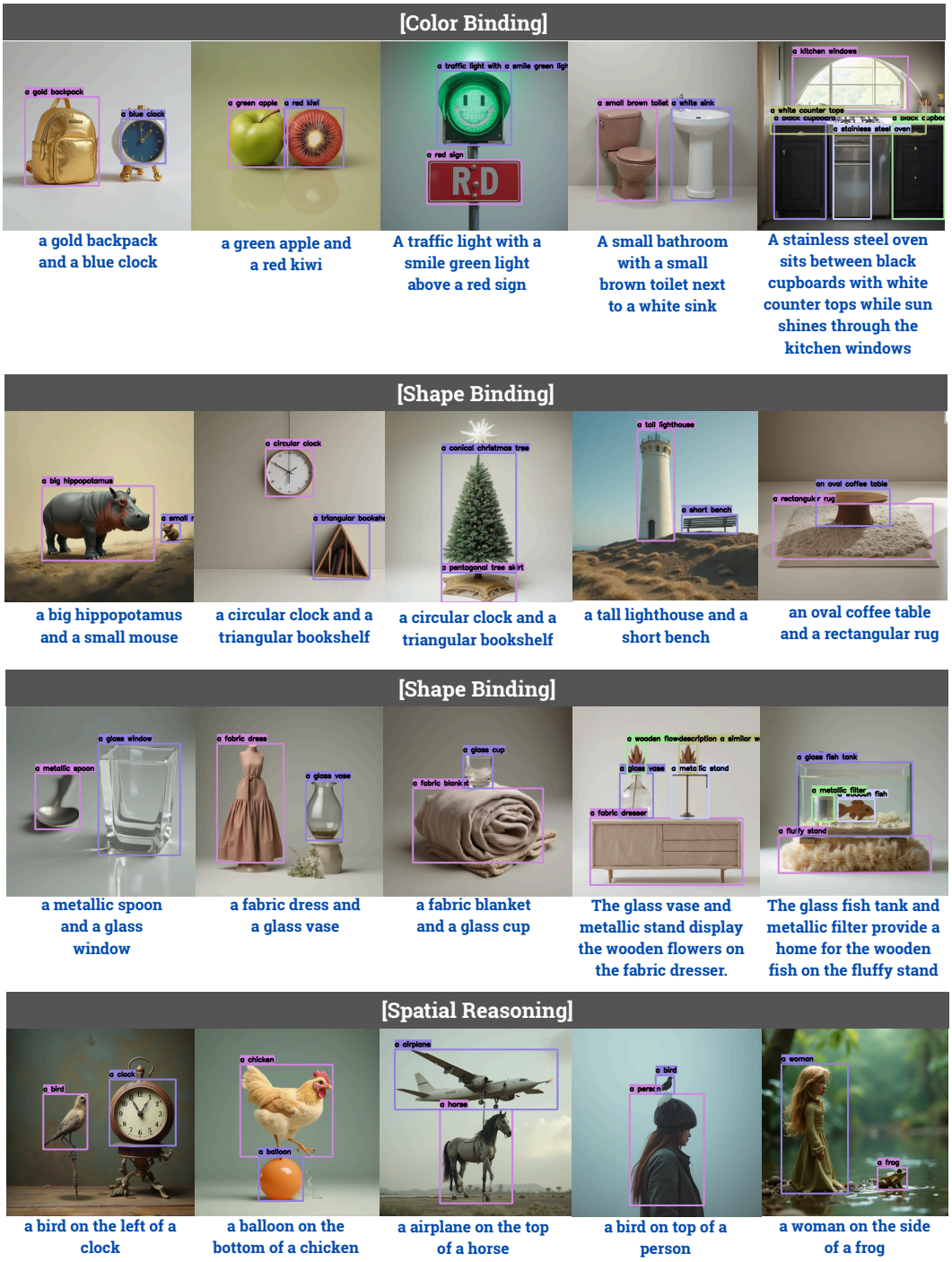

Figure 6: Additional examples of PYP's layout-guided image generation on T2I-CompBench. Images are generated from the predicted layouts using Laytrol.

- **Object Relations.** We separately diagnose different failures in pairwise object geometry. Relation Adherence (RA) checks whether the intended spatial relation between two objects is preserved, while Relative Size (RS) checks whether their required size relationship is maintained. Contact Consistency (CC) further detects incorrect gaps, overlaps, or contacts relative to the intended interaction. These questions localize which aspect of the specified object relation fails in the generated image.

EVALUATION

Image Generation. For each prompt, we generate 10 images per image-generation model using different random seeds. This captures generation variability rather than evaluating each model from a single sample.

VLM Evaluation. Each generated image is evaluated with its corresponding PYP-VQA diagnostic questions using Qwen3.6-27B as the VLM evaluator. We use a fixed seed and fixed decoding configuration for all VLM evaluations to reduce run-to-run variation. For each image–question pair, the evaluator receives the image, the diagnostic question, and the following instruction:

```
You are a visual judge. Inspect the image and answer visual questions
↪  using only visible evidence.

For each question:
1. Examine the relevant image region carefully.
2. Base your decision on direct visible proof, clarity, and presence of
↪  features.
3. If evidence is missing, blurred, or occluded, answer "no".

Return ONLY a valid JSON object where the keys are the EXACT IDs provided
↪  in the input.
No markdown, no reasoning, and no text outside the JSON.

Required Output Format:
{
  "ID_NUMBER": "yes/no"
}

Rules:
- Use only the provided IDs as keys.
- Output ONLY the JSON object.
- Include exactly one result for every supplied question.
- Do not repeat the questions.
- Do not omit any QID.
- Do not provide any explanation.
```

Question-Level Scoring. Each diagnostic question has an expected binary answer(GT) determined during question generation. Rather than using only the decoded Yes/No response, we use the probability assigned to the expected answer as the question-level score. For a question q with expected answer y_q, the score is

$$s_q = \begin{cases} p_q(\text{Yes}), & y_q = \text{Yes}, \\ 1 - p_q(\text{Yes}), & y_q = \text{No}. \end{cases}$$

Failure-Level and Image-Level Scoring. Question-level scores are first aggregated within each failure category F_k using the geometric mean:

$$F_k = \left(\prod_{q \in F_k} s_q \right)^{\frac{1}{|F_k|}}, \tag{1}$$

where $|F_k|$ denotes the number of diagnostic questions in category F_k. We use the resulting category scores directly as OP, AA, AL, AS, RA, RS, and CC.

For **attribute-binding** prompts,

$$\text{PYP}_{\text{attribute}} = \text{OP} \cdot \text{AA} \cdot \text{AL} \cdot \text{AS}.$$

For relational reasoning, we first define object score

$$R = \text{RA} \left[1 - (1 - \text{RA}) \left(1 - \frac{\text{RS} + \text{CC}}{2} \right) \right]$$

For **object-relation** prompts,

$$\text{PYP}_{\text{obj_relation}} = \text{OP} \cdot R.$$

For **complex** prompts,

$$\mathrm{PYP}_{\mathrm{complex}} = \mathrm{OP} \cdot \mathrm{AA} \cdot \mathrm{AL} \cdot \mathrm{AS} \cdot R.$$

An image is classified as *Fail* when its corresponding *PYPScore* falls below 0.5, and as *Pass* otherwise. We retain the individual category-level scores alongside the image-level score, allowing PYP-VQA to localize the specific failure modes contributing to each prediction. The resulting category-level and overall failure statistics are reported in Table 5.

A.2.3 Failure Localization with PYP-VQA

Failure Case: Attribute Binding

Prompt: *a red cow and a brown suitcase*



Scene Structure

Nodes:

ID	Object	Description	Attributes	BBox
t1	red cow	a red cow with a smooth coat and sturdy build	color: red	[60, 180, 180, 150]
t2	brown suitcase	a brown suitcase with a rectangular shape	color: brown	[260, 220, 100, 80]

Relation (R):

Subject	Predicate	Object	Constraint
t1	and	t2	*Dominant axis:* horizontal; *Ordering:* t1 next to t2, left-to-right (next-to rule observed); *Alignment:* centered; *Contact mode:* gap.

Negative Prompt (NP): None

PYP-VQA Diagnosis

Type	Diagnostic Question	Expected	Answer	Score
OP	Focusing on the cow: is it clearly visible?	yes	yes	0.9579
OP	Focusing on the suitcase: is it clearly visible?	yes	yes	0.9900
AA	Focusing on the cow: is that cow red?	yes	no	0.2678
AA	Focusing on the suitcase: is that suitcase brown?	yes	yes	0.9900
AL	Focusing on the cow and suitcase: do they have the same color?	no	yes	0.2967
AS	Focusing on the cow: is the color brown?	no	yes	0.0154
AS	Focusing on the suitcase: is the color red?	no	no	0.9900

OP: Object Presence; *AA*: Attribute Adherence; *AL*: Attribute Leakage; *AS*: Attribute Swap. *Answer*: the VQA model's answer; *Score*: question-level score; scores below 0.5 are shown in red.

Figure 7: Figure 7 shows a failure case from an image generated by Nano Banana 2. PYP-VQA localizes the error through its diagnostic questions, identifying which prompt constraints are satisfied and which fail.

Failure Case: Object Relation

Prompt: *a plate on the bottom of a fish*



Scene Structure

Nodes:

ID	Object	Description	Attributes	BBox
t1	plate	a round flat plate with a smooth surface, appearing still and intact	–	[180, 250, 152, 50]
t2	fish	the lower part of a fish with a streamlined shape, appearing smooth and motionless	–	[100, 100, 312, 200]

Relation (R):

Subject	Predicate	Object	Constraint
t1	bottom of	t2	*Dominant axis:* vertical; *Ordering:* t1 on t2, top-to-bottom; *Alignment:* vertically aligned; *Contact mode:* support.

Negative Prompt (NP): None

PYP-VQA Diagnosis

Type	Diagnostic Question	Expected	Answer	Question-level Score
OP	Focusing on the fish: is it clearly visible?	yes	yes	0.9389
OP	Focusing on the plate: is it clearly visible?	yes	yes	0.9900
RA	Focusing on the a plate and the a fish: Is the a plate bottom of the a fish in the image?	yes	yes	0.5204
RS	a plate bottom of a fish: Is their relative size consistent with the scene and camera perspective?	yes	yes	0.9216
CC	Focus on the a plate and the a fish: Does the image show the a plate physically supported by or resting on the a fish?	yes	no	0.2253

OP: Object Presence; *RA*: Relation Adherence; *RS*: Relative Size; *CC*: Contact Consistency. *Answer*: the VQA model's answer; *Question-level Score*: scores below 0.5 are shown in red.

Figure 8: Fine-grained failure diagnosis for the prompt *"a plate on the bottom of a fish."* The image is generated by Nano Banana 2 and analyzed by PYP-VQA, which constructs the scene representation and localizes failures through diagnostic questions.