Synthetic Curriculum Reinforces Compositional Text-to-Image Generation

Supplementary Material

A. Algorithm

We provide an overall algorithm of our CompGen in Algorithm 1.

Algorithm 1 CompGen

Require: Pre-trained T2I model with parameters θ, pre-trained MLLM p_{reward}, curriculum of difficulty ranges $\{[\text{Diff}_{\text{min}}, \text{Diff}_{\text{max}}]_k\}_{k=1}^{K}$.
Ensure: Fine-tuned T2I model with parameters θ'.

for each training step **do**
  Phase 1: Curriculum-based Data Synthesis
  Schedule a difficulty range $[\text{Diff}_{\text{min}}, \text{Diff}_{\text{max}}]$ from the curriculum according to Appendix C.
  Generate a scene graph $\mathcal{G}$ (as formulated in Definition 1) via adaptive MCMC, such that $\text{Diff}_{\text{min}} \leq \text{Diff}(\mathcal{G}) \leq \text{Diff}_{\text{max}}$ according to Algorithm 2.
  Phase 2: C-GRPO Optimization for Compositional Generation
  Derive natural language input prompt T from $\mathcal{G}$ using a constrained LLM.
  Generate a group of G images $\{I^{(i)}\}_{i=1}^{G}$ using T as input from policy $p_{\theta_{\text{old}}}(\cdot|T)$.
  Generate a set of binary question-answer pairs (e.g., Q_{object}, Q_{count}, $Q_{\text{attribute}}$, Q_{relation} with "Yes" answers) from $\mathcal{G}$ according to Section 4.2.
  Compute per-question rewards $\widehat{r}_j^{(i)}(t)$ for each image $I^{(i)}$ at step t.
  Compute overall image reward and normalized advantages $A_i(t)$ at step t according to Eq. (4).
  Optimize C-GRPO objective $\mathcal{J}_{\text{C-GRPO}}(\theta)$ with clipped importance sampling to update θ using Eq. (5).
end for
return Fine-tuned T2I model with parameters θ.

We also provide an illustrated example of key concepts, including difficulty-aware graphs, scene graph-based binary questions, and text-to-image generation in Figure 6.

B. Scene Graph Asset Generation

To instantiate the sampled scene graphs, we require corresponding data assets. These assets can be categorized into three types: objects, attributes, and relations.

B.1. Object Generation

To generate a diverse set of object assets that are well-representable by T2I models, we first selected 10 independent and broad object categories [1]: "Natural Landscapes", "City Infrastructure and Street Elements", "People", "Animals", "Plants", "Food and Beverages", "Sports and Fitness", "Technology Equipment and Industry", "Everyday Objects", and "Transportation". Based on these categories, we employed the DeepSeek-V3 [15] to generate 50 distinct objects for each, resulting in a total of 500 objects. To avoid duplicate names, we ensured uniqueness through case-normalized matching and assigned incrementing identifiers for traceability. The entire process involved multi-stage validation to ensure structural and semantic consistency, including schema checks on field types, required keys, and cardinality. Invalid outputs were regenerated, while valid ones were persisted atomically to maintain data integrity.

Algorithm 2 Scene Graph Generation via Adaptive MCMC

Require: Difficulty bounds $[\text{Diff}_{\text{min}}, \text{Diff}_{\text{max}}]$, max iterations T, an annealing schedule for temperature τ
Ensure: A generated scene graph $\mathcal{G}_T$

Initialize graph $\mathcal{G}_0$ as a simple prior graph
for $t \leftarrow 1$ to T **do**
  Randomly select a transformation operation $t_{\text{op}} \in \{t_{\text{add}}, t_{\text{delete}}\}$
  Propose a candidate graph $\mathcal{G}' \leftarrow t_{\text{op}}(\mathcal{G}_{t-1})$
  Compute Energy$(\mathcal{G}_{t-1})$ and Energy$(\mathcal{G}')$ via Eq. (2)
  Decrease temperature τ according to the annealing schedule
  Compute acceptance probability $\text{Acc}(\mathcal{G}'|\mathcal{G}_{t-1})$ via Eq. (3)
  if $\text{Uniform}(0,1) < \text{Acc}(\mathcal{G}'|\mathcal{G}_{t-1})$ **then**
    $\mathcal{G}_t \leftarrow \mathcal{G}'$ ▷ Accept proposal
  else
    $\mathcal{G}_t \leftarrow \mathcal{G}_{t-1}$ ▷ Reject proposal
  end if
end for
return $\mathcal{G}_T$

B.2. Attribute Generation

Attribute binding is a crucial component of a model's composability capability. To generate attribute assets that correspond to entities, we adopt a hierarchical generation process. Specifically, we first use the DeepSeek-V3 [15] to generate attribute concepts corresponding to the object, such as "color", "material", and "shape" for a chair. Next, we employ the DeepSeek-V3 [15] to generate valid values for each attribute concept. For example, for the material attribute of a chair, valid values such as "wood", "metal", and "plastic" are generated. Compatibility checks are applied at both stages

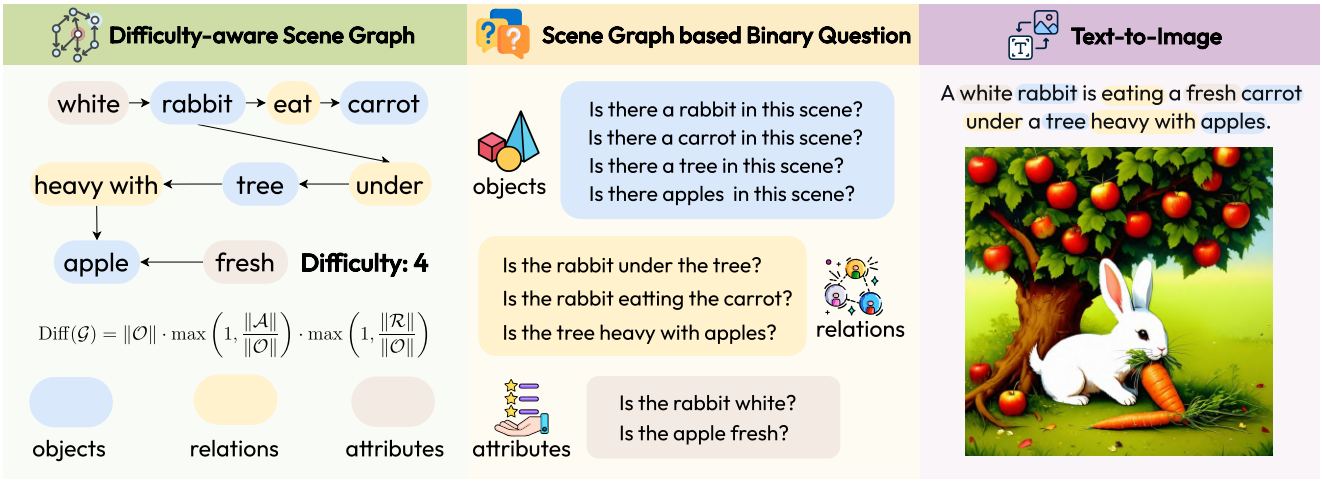

Figure 6. An illustrated example of scene graph complexity (Left), sample binary questions generated from the scene graph (with "Yes" answers) (Middle), and corresponding input text and output image pairs (Right).

of the process. For each object, we generate 5 attribute concepts, and for each concept, we generate 5 attribute values.

B.3. Relation Generation

For relations, our framework adopts a selective generation mechanism that produces semantic relations only for sampled object pairs, avoiding the computational cost of exhaustive pairwise enumeration. Relations are generated exclusively for edge nodes specified in the pre-computed scene graph topology, which ensures structural alignment while enabling semantic enrichment. Each relation is dynamically conditioned on the categorical and attribute-level properties of both participating objects through carefully designed LLM prompts that enforce contextual appropriateness.To maintain uniqueness while supporting diverse interactions across scene configurations, the framework employs a deterministic pair-identification scheme that prevents duplicate object-pair relations. This results in linear computational complexity relative to the number of objects, making the system scalable for complex scenes, while inherently preserving semantic coherence by jointly enforcing structural and attribute-based constraints during generation. Invalid outputs trigger automatic regeneration, whereas validated relations are atomically persisted to guarantee reliability throughout the pipeline.

C. Curriculum Scheduling Strategies

- **Random Scheduling.** As a straightforward method to prevent forgetting, the random scheduling strategy allows for sampling from all data with equal probability throughout the entire training process. This can be viewed as a trivial form of curriculum learning, or more generally, as the default behavior of most policy optimization algorithms when task difficulty is not considered. For M tasks, the sampling probability $p_{\text{random}}(t, j)$ for task j at any training iteration t is set to:

$$p_{\text{random}}(t, j) = \frac{1}{M}$$

This means that at each training step, all tasks have an equal chance of being selected. While this method effectively prevents the model from forgetting previously learned tasks, it might introduce more difficult tasks too early, leading to reward sparsity issues and potentially hindering the curriculum learning strategy from achieving optimal results.

- **Easy-to-Hard Scheduling.** Easy-to-Hard curriculum learning strategies adopt a phased approach, dividing the training process into a series of incrementally difficult stages. In each training phase, the model focuses solely on tasks of a specific difficulty level. We can define the sampling indicator function for task j at training step t as $p_{\text{E2H}}(t, j)$. For a total of M tasks ($j = 1, \ldots, M$), this function takes a value of 1 if the current training iteration t falls within the predefined stage $[\tau_j, \tau_{j+1}]$ for task j; otherwise, it is 0. Here, τ_j denotes the starting training step for the j-th stage, with $\tau_1 = 0$ and $\tau_{M+1} = N_T$ being the total number of training steps. This implies that at a given iteration t, only tasks corresponding to the current stage are sampled for training.

$$p_{\text{E2H}}(t, j) = \begin{cases} 1, & \text{if } \tau_j \le t < \tau_{j+1} \\ 0, & \text{otherwise} \end{cases}$$

Therefore, at training step t, the sampling distribution will be $[p_{\text{E2H}}(t, 1), \ldots, p_{\text{E2H}}(t, M)]$.

- **Gaussian Scheduling.** Gaussian scheduling models task sampling as a mixture of Gaussian distributions to provide flexible and fine-grained control over the training process.

Each task j ($j = 1, \ldots, M$) is assumed to follow a one-dimensional Gaussian distribution with the same variance σ^2 but different means

$$\mu_j = j - 1.$$

A latent curriculum position x_t moves from easier to harder tasks as training step t increases:

$$x_t = \left(\frac{t}{N_T}\right)^{\beta} (M - 1),$$

where N_T is the total number of training steps, $\beta > 0$ controls the moving speed of x_t, and M is the number of tasks.
The unnormalized sampling score for task j at step t is

$$s_{\text{Gaussian}}(t, j) = \exp\left(-\frac{(x_t - \mu_j)^2}{2\sigma^2}\right),$$

where σ determines the sampling concentration.
The normalized sampling probability is

$$p_{\text{Gaussian}}(t, j) = \frac{s_{\text{Gaussian}}(t, j)}{\sum_{m=1}^{M} s_{\text{Gaussian}}(t, m)}.$$

Smaller σ produces sharper, stage-like transitions, while larger σ smooths the task shifts. A lower β slows the move toward harder tasks, allowing more training on easy tasks in the early phase.

D. Datasets and Metrics

We summarize the benchmarks and crossponding metrics used to evaluate the compositional capabilities of text-to-image models below:

- **GenEval** [24] is a comprehensive benchmark comprising 553 meticulously designed and highly structured prompts that assess model performance across six key evaluation dimensions: single object, dual objects, color, count, spatial positioning, and attribute binding. The GenEval [24] framework automatically evaluates text-to-image alignment by using object detection to verify object presence, count, and position, and color classification models to verify attributes, calculating binary scores for single objects, dual objects, colors, counts, spatial positioning, and attribute binding, then averaging these scores for an overall compositional quality assessment, we use this overall score as the final score.
- **T2I-CompBench** [28] is a large-scale benchmark consisting of 6,000 carefully curated compositional text prompts designed to evaluate text-to-image models across varying levels of semantic and structural complexity. The prompts are organized into three primary evaluation domains—attribute binding, object relationships, and complex compositions—and further subdivided into six fine-grained categories: color binding, shape binding, texture binding, spatial relationships, non-spatial relationships, and complex compositions. For attribute binding (color, shape, texture), it employs Disentangled BLIP-VQA [33], which breaks prompts into individual object-attribute questions and calculates the product of the "yes" probabilities. Spatial relationships are verified through UniDet-based object detection using geometric rules such as relative positions and IoU thresholds. Complex compositions combine these with CLIPScore [25] in a 3-in-1 metric that averages the specialized evaluations. We select the complex compositions score as the final score.
- **TIFA** [27] is a comprehensive evaluation framework designed to assess the generation quality of text-to-image models through a combination of 4,000 diverse, human-curated text prompts and 25,000 automatically generated questions utilizing a Visual Question Answering (VQA) model. The benchmark spans 12 distinct question categories, including existence verification, object count, color identification, and spatial reasoning, each targeting different aspects of visual understanding and semantic alignment. TIFA's [27] overall score is the percentage of questions that visual models answer correctly when evaluating how well a generated image matches its text description. We use this overall score as the final score.
- **DPG-Bench** [26] is a benchmark that consists of 1,065 densely annotated prompts, each with an average token length of 83.91. These prompts are carefully crafted to describe complex visual scenarios that involve multiple objects, attributes, and modifiers, providing a rich and detailed source for evaluating text-to-image models. The prompts are designed to test a model's ability to handle intricate compositional structures, where objects are not only depicted in isolation but also in relation to one another, with various contextual modifiers such as color, size, orientation, and spatial relationships. DPG-Bench [26] evaluates models across Global, Entity, Attribute, Relation, and Other categories by using an MLLM judge to score the correctness of answers to scene-graph–based questions for each generated image, and the overall score is simply the average of all prompt-level scores. We use this overall score as the final score.
- **DSG** [12] is an evaluation framework for text-to-image (T2I) generation, designed to address the limitations of the Question Generation and Answering approach. It generates a set of questions from a given prompt, which are then answered by a VQA model. The core components of DSG [12] include questions about entities, their attributes, relationships, and global scene features. Each question is atomic, unique, and semantically complete. These ques-

tions are organized into a Directed Acyclic Graph (DAG), where the dependencies between questions are explicitly defined. For example, if a parent question (e.g., whether an object is present) receives a negative answer, its dependent questions (e.g., the object's color) are skipped to prevent inconsistent or irrelevant responses. The overall score is computed by averaging the accuracy of the VQA model's answers. Each question is validated based on whether its truth value can be clearly and reliably determined, and only such valid questions are included in the evaluation. The system also ensures that no redundant or illogical questions are generated. We use this overall score as the final score.

E. Baseline Models

The baseline models we compare against in our experiments are summarized as follows:

- **Stable-Diffusion-1.4** [54] is one of the earliest widely-adopted open-source text-to-image latent diffusion models. It was trained on a diverse and large-scale dataset of image-text pairs, which enabled it to generate high-quality images from natural language descriptions. By employing a latent diffusion process, Stable-Diffusion-1.4 [54] efficiently navigates high-dimensional image spaces, striking a balance between computational efficiency and image fidelity. This model not only demonstrated the potential of diffusion-based generative models but also established a strong foundation for subsequent versions and related models in the field of generative AI. Its open-source nature has fostered further research and innovation, leading to improvements in model scalability, image resolution, and the diversity of generated outputs.
- **Stable-Diffusion-1.5** [54] improves upon its predecessor, version 1.4, through refined data curation and additional training iterations, resulting in enhanced performance and greater image generation accuracy. The improved dataset, coupled with extended training cycles, allows the model to better capture intricate details and nuances in both the visual and textual domains. Stable-Diffusion-1.5 [54] demonstrated superior robustness in handling more complex prompts, making it a preferred choice for researchers and developers working on advanced generative tasks. Its ability to generate more coherent and contextually appropriate images from intricate or ambiguous text inputs has made it a standard baseline in the field of text-to-image generation, providing a reliable foundation for further model advancements and applications in diverse creative and technical domains.
- **Stable-Diffusion-2.1** [54] enhances high-resolution image synthesis and strengthens semantic alignment between text and image representations. Building upon the architectural and methodological foundations of earlier versions, it introduces refined training pipelines and improved noise scheduling strategies to achieve higher visual fidelity. The model leverages a more carefully curated and filtered dataset, reducing spurious artifacts and improving structural and compositional consistency in generated outputs. As a result, Stable-Diffusion-2.1 [54] produces images with sharper details, more accurate object boundaries, and improved correspondence to complex textual descriptions. These advancements make it particularly effective for tasks requiring fine-grained control over visual semantics and have established it as a benchmark for evaluating modern text-to-image diffusion models.
- **Playground-V2** [32] is optimized for creative and vivid image generation, focusing on enhancing aesthetic expressiveness and imaginative scene synthesis. Compared to earlier diffusion-based models, it introduces tuning strategies and architectural refinements that prioritize stylistic diversity, color richness, and artistic coherence. The model demonstrates strong capabilities in generating visually striking compositions that blend realism with creativity, making it particularly suitable for tasks involving conceptual design, digital art, and visual storytelling. By balancing fidelity and artistic abstraction, Playground-V2 [32] exemplifies how diffusion models can be adapted for open-ended, human-centric creative applications beyond conventional photorealistic synthesis.
- **Stable-Diffusion-XL** [49] incorporates a substantially larger model capacity and an improved decoder architecture, leading to significant gains in both visual realism and semantic precision. By expanding the number of parameters and enhancing the expressiveness of its latent representations, the model achieves more detailed, coherent, and photorealistic high-resolution outputs. The redesigned decoder contributes to sharper textures, smoother gradients, and better preservation of global scene structure. In addition, Stable-Diffusion-XL [49] demonstrates improved generalization across diverse visual domains, maintaining consistency in complex compositions and fine-grained visual elements. These advancements position it as a state-of-the-art framework for large-scale text-to-image synthesis and a critical reference point for subsequent research in high-fidelity generative modeling.
- **Lumina-Next** [72] represents a new generation of efficient text-to-image models, designed to deliver high-quality image synthesis with significantly reduced inference latency. By optimizing both the model architecture and inference pipelines, Lumina-Next [72] strikes a balance between computational efficiency and visual fidelity, enabling rapid generation of high-resolution images from textual prompts. This improvement in speed, without compromising on quality, makes the model particularly well-suited for interactive applications, where real-time performance is essential. Whether used in creative tools, virtual environments, or user-facing systems, Lumina-Next provides a robust

Table 4. Investigation on reward function using Stable-Diffusion-1.5 as backbone. Our adopted model is highlighted in yellow. The best performance is marked in **red**.

Model	Reward Model	GenEval	DPG	TIFA	T2I-CompBench	DSG	Avg.
Stable-Diffusion-1.5	–	42.08%	62.24%	78.67%	29.94%	61.57%	54.90%
Stable-Diffusion-1.5 w/ VQAScore [36]	LLaVA-v1.6-13B	44.02%	73.41%	80.19%	36.36%	73.23%	61.44%
CompGen (Stable-Diffusion-1.5)	CLIP-FlanT5-XXL	45.11%	71.26%	81.42%	31.60%	73.74%	60.63%
	InstructBLIP	42.04%	64.00%	76.29%	**39.21%**	64.58%	57.22%
	LLaVA-v1.5-13B	49.23%	74.54%	84.84%	37.43%	75.97%	64.40%
	LLaVA-v1.6-13B	**53.88%**	**78.67%**	**85.71%**	37.68%	**77.16%**	**66.62%**

solution for scenarios requiring swift, high-quality visual content generation, while maintaining a responsive and seamless user experience. Its efficiency and scalability position it as a key technology for next-generation interactive AI systems.

- **LlamaGen** [59] explores a novel language-model-driven paradigm for image generation, bridging the gap between large-scale linguistic understanding and visual synthesis. By integrating powerful language priors from large language models with a dedicated visual decoder, it effectively aligns semantic comprehension with image realization. This design enables the model to interpret nuanced textual descriptions and translate them into coherent and contextually rich visual outputs. LlamaGen [59] demonstrates that linguistic reasoning can significantly enhance visual generation quality, leading to improved semantic consistency, compositional accuracy, and prompt adherence. The approach underscores the emerging synergy between language and vision models, highlighting the potential of unified multimodal architectures for scalable, interpretable, and semantically grounded generative systems. visual decoder, it highlights the synergy between language and vision.
- **Show-o** [68] focuses on complex multi-object scene construction, advancing the capability of diffusion models to generate spatially coherent and semantically aligned compositions. It is designed to handle intricate visual relationships between multiple entities, ensuring that object placement, scale, and interactions conform to natural scene logic. Through enhanced conditioning mechanisms and improved scene representation learning, Show-o [68] achieves a higher degree of spatial organization and contextual consistency in generated outputs. The model excels at synthesizing scenes that maintain both local detail fidelity and global structural balance, making it especially valuable for applications in visual reasoning, synthetic dataset generation, and compositional image synthesis. Its emphasis on semantic layout understanding marks an important step toward controllable and interpretable multi-entity generative modeling.
- **SimpleAR** [63] is a lightweight autoregressive (AR) framework for high-quality text-to-image synthesis, focusing on simplicity and effectiveness. It uses a standard next-token prediction approach, where images are tokenized and modeled with text through a unified transformer. With a minimal model of just 0.5B parameters, SimpleAR generates 1024×1024 images with strong fidelity and structure. Its three-stage training pipeline—pretraining, supervised fine-tuning, and reinforcement learning (GRPO [56]) enhances prompt alignment, reasoning, and compositionality. Optimizations like KV-cache acceleration and speculative decoding reduce inference latency, allowing 1024×1024 images to be generated in about 14 seconds. SimpleAR demonstrates that high-quality visual generation can be achieved with a simple, efficient, and scalable AR model.
- **Emu3** [60] is a versatile multi-modal generation system capable of producing not only high-quality images but also videos and other cross-modal outputs. It is architected to enable task generalization across diverse input modalities, including text, audio, and visual signals, thereby supporting a unified generative framework. Through shared latent representations and multi-stage decoding strategies, Emu3 effectively captures complex inter-modal relationships and temporal dependencies, allowing it to synthesize coherent and contextually aligned outputs across formats. This flexibility makes Emu3 particularly suitable for next-generation applications such as interactive media creation, embodied AI, and multi-sensory content generation. Its design reflects a broader trend toward integrated generative systems that seamlessly bridge modalities, pushing the boundary of what multimodal AI models can achieve.
- **minDALL-E** [29] is a lightweight variant within the DALL-E family, designed to balance model efficiency with generative capability. Despite its substantially smaller parameter scale, it successfully preserves the core mechanisms underpinning text-to-image synthesis, including semantic understanding and compositional reasoning. The model's compact architecture enables faster inference and lower computational overhead, making it more accessible for research and educational purposes. By emphasizing reproducibility and open experimentation, minDALL-E [29] provides a practical platform for studying large-scale generative modeling principles without the extensive resource requirements of its larger counterparts. Its streamlined de-

sign illustrates how architectural simplification can coexist with strong generative performance, promoting inclusivity and transparency in multimodal AI research.

F. Experimental Setup

Implementation Details of CompGen. To generate high-quality training data for compositional RL, we employ our CompGen framework to construct 10K samples, which are evenly distributed across difficulty levels ranging from 1 to 10 according to our difficulty metric in Definition 1. We provide more detailed explorations of the difficulty measure in Section 5.4 and data difficulty distribution in Appendix G. To demonstrate the effectiveness and generalizability of CompGen, we apply it to two prominent T2I architectures representing different generation paradigms: the diffusion-based Stable-Diffusion-1.5 [54] and the auto-regressive SimpleAR-SFT [59].

For RL training with C-GRPO, we train Stable-Diffusion-1.5 following Xue et al. [69]'s setting, using a 1e-5 learning rate, AdamW [41] optimizer, 1.0 gradient clip norm, a batch size of 32, and generating 12 images per prompt at 512×512 resolution with a 1e-4 clip range. For SimpleAR, we adhere to the settings in Wang et al. [63], employing a 1e-5 learning rate, AdamW [41] optimizer, a batch size of 28, and generating 4 images per prompt at 1024×1024 resolution. All training is conducted on 8 NVIDIA H800 GPUs.

G. Additional Ablation Studies

G.1. Impact of Reward Function.

To validate our compositional reward function, we conduct an ablation study comparing our fine-grained reward design against the VQAScore baseline [36]. While VQAScore uses a single binary question to assess prompt-image alignment, our method decomposes the evaluation into multiple aspects like object existence, attribute binding, relation understanding and numerical counting. As shown in Table 4, using the same LLaVA-v1.6-13B reward model [39], our CompGen framework achieves a 66.62% average score, surpassing VQAScore by a significant 5.2%. This demonstrates that our fine-grained reward signals more effectively guide the T2I model towards mastering complex compositional generation tasks.

We provide a visual comparison between our method, CompGen, and the VQAScore baseline in Figure 7.

G.2. Impact of Data Difficulty Distribution.

We investigate how the training data difficulty distribution affects model performance in Table 5. All experiments are conducted on Stable Diffusion 1.5 [54], with each trained on 10k samples. We compare our uniform data difficulty distribution (uniformly covering difficulty levels 1-10) against two skewed distributions: "Skew-easy" (10k samples from levels 1-3) and "Skew-difficult" (10k samples from levels 8-10). The results are stark: both skewed distribution perform poorly. Training solely on easy data fails to prepare the model for complex prompts, while focusing only on hard data appears to cause catastrophic forgetting, crippling generalization. Our approach, which provides a balanced curriculum across all difficulty levels, substantially outperforms both, achieving an average score of 66.62%. This highlights that a curriculum spanning a comprehensive range of difficulties is crucial for enhancing compositional text-to-image generation.

Table 5. Investigation on different *data difficulty distributions* using Stable-Diffusion-1.5 with 10K training data. The data distribution we adopted is marked with yellow background. The best performance for each benchmark is marked in **red**. "Skew-easy" means training on easier instances, "Skew-difficult" means harder ones, and "Uniform" means training on a balanced mix of difficulties..

Difficulty Distribution	GenEval	DPG	TIFA	T2I-Bench	DSG	Avg.
Skew-easy	30.24%	40.20%	48.64%	29.52%	55.64%	40.85%
Skew-difficult	24.78%	50.88%	60.31%	25.68%	50.27%	42.38%
Ours (Uniform)	**53.88%**	**78.67%**	**85.71%**	**37.68%**	**77.16%**	**66.62%**

Table 6. Comparison of sampling efficiency and graph diversity using different sampling methods. The proposed method is marked with yellow background. The best performance is marked in **red**. **SR** denotes Success Rate, and **NTD** denotes Node Type Diversity.

Graph Sampling Method	Easy		Medium		Hard	
	SR	NTD	SR	NTD	SR	NTD
Random Rejection Sampling	62.50%	39	48.40%	32	35.60%	21
Greedy Sampling	90.40%	35	88.30%	46	84.10%	37
Ours (Adaptive MCMC Sampling)	**98.80%**	**42**	**95.20%**	**88**	**91.50%**	**130**

Table 7. Analysis of different initialization strategies for the prior graph. The default setting used in our approach is marked with yellow background. The best performance is marked in **red**. **SR** denotes Success Rate, and **NTD** denotes Node Type Diversity.

Initial Strategy of Prior Graph	Easy		Medium		Hard	
	SR	NTD	SR	NTD	SR	NTD
Empty Initial Graph	97.40%	37	94.10%	85	87.20%	114
Dense Initial Graph	98.00%	**46**	93.80%	84	89.90%	127
Ours	**98.80%**	42	**95.20%**	**88**	**91.50%**	**130**

H. Additional Experiments on Scene Graph Generation

H.1. Efficiency of Adaptive MCMC Graph Sampling

To demonstrate the necessity and effectiveness of our proposed Adaptive MCMC sampling approach in navigating the combinatorial graph space, we conduct a comparative study against standard sampling baselines.

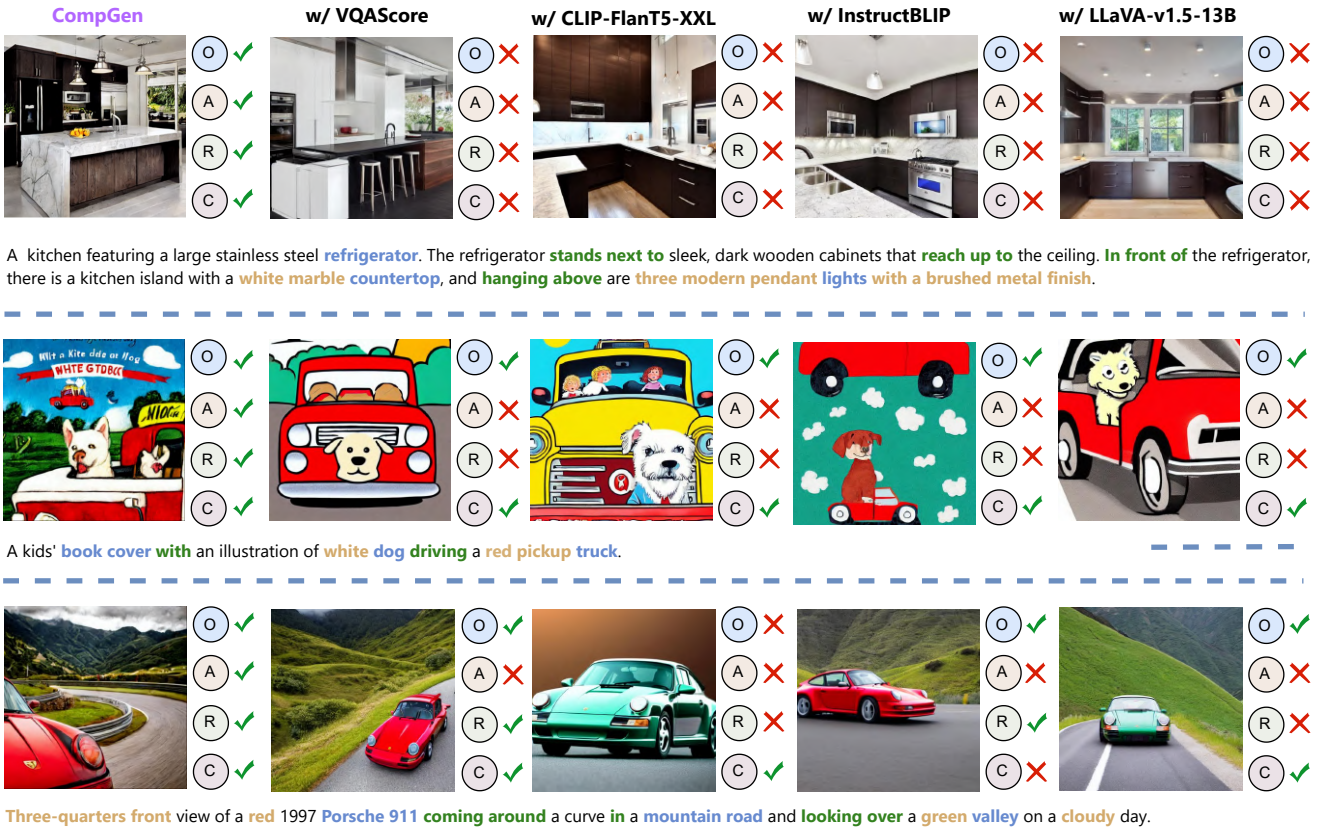

Figure 7. Qualitative comparison of compositional generation across CompGen and ablation models (w/ VQAScore, w/ CLIP-FlanT5-XXL, w/ InstructBLIP, w/ LLaVA-v1.5-13B). Within each prompt, we color the elements for which at least one model makes an error: the object in **blue**, the attribute in **brown**, the relationship in **green**, and the count in **purple**. Ⓞ, Ⓐ, Ⓡ, Ⓒ denote Object, Attribute, Relationship, and Count, respectively. A ✓ indicates correct generation, while a ✗ indicates an error.

Baselines. We employ two baseline strategies for comparison: (1) *Random Rejection Sampling* [4]: To serve as a brute-force lower bound, this method randomly generates a graph from scratch at each step and accepts it only if it strictly satisfies the difficulty constraints. (2) *Greedy Sampling* [6]: To simulate a local optimization approach, this method proposes modifications to the current graph but only accepts transformations that reduce the energy function (i.e., move closer to the target difficulty), unconditionally rejecting any moves that temporarily increase energy.

Evaluation Protocol. To systematically evaluate performance across varying complexities, we define three difficulty intervals based on the score $\text{Diff}(\mathcal{G})$: **Easy** ($1 < \text{Diff} \leq 4$), **Medium** ($4 < \text{Diff} \leq 7$), and **Hard** ($7 < \text{Diff} \leq 10$). For each method and difficulty level, we perform 1,000 independent sampling trials with a maximum budget of $T = 100$ iterations per trial. To quantify the quality of the sampling process, we report two metrics: **Success Rate (SR)**, indicating the percentage of trials that yield a valid graph within the iteration budget; and **Node Type Diversity (NTD)** [20]. Given that a scene graph comprises object, attribute, and relation nodes, we characterize the graph's structural signature using the tuple $\mathbf{s} = (N_{\text{object}}, N_{\text{attribute}}, N_{\text{relation}})$. NTD quantifies the total number of unique valid signatures $\mathbf{s}$ discovered by the method, serving as a metric for the diversity of the generated graphs.

Results Analysis. As presented in Table 6, *Random Rejection Sampling* exhibits a sharp performance degradation as complexity increases, dropping to a low Success Rate (35.60%) in the Hard setting. This failure highlights the intractability of blindly searching the high-dimensional graph space. To improve search efficiency, *Greedy Sampling* achieves respectable success rates in simpler tasks but suffers from mode collapse, evidenced by its significantly lower NTD compared to our method (e.g., 37 vs. 130 in Hard mode). This indicates a tendency to get trapped in local minima, repeatedly reproducing limited structural combinations. In contrast, to effectively balance exploration and

exploitation, our Adaptive MCMC method leverages the Metropolis-Hastings criterion to escape local optima. Consequently, it achieves a Success Rate exceeding 91% across all levels while maintaining superior semantic diversity (reaching 130 NTD in Hard mode).

H.2. Performance of Varying Initial Prior Graph

To verify the robustness of our method, we investigate whether the convergence of the Markov chain is sensitive to the initialization of the prior graph $\mathcal{G}_0$.

Baselines. We compare three distinct initialization strategies: (1) *Empty Initial Graph*: To test generation from scratch, we initialize with a null graph $\mathcal{G}_0 = \emptyset$. (2) *Dense Initial Graph*: To test the ability to refine chaotic structures, we initialize with a randomly generated dense graph containing a high number of nodes (4-7) and edges. (3) *Ours*: To provide a neutral starting point, we initialize with a minimally complex graph containing a small set of random object nodes (1-3).

Evaluation Protocol. We adhere to the same experimental setup described in Sec. H.1. Specifically, for each initialization strategy, we conduct 1,000 independent trials across the three defined difficulty intervals (Easy, Medium, Hard) with a fixed iteration budget of $T = 100$. We evaluate performance using Success Rate (SR) to assess convergence reliability and Node Type Diversity (NTD) to analyze the impact of the starting state on the diversity of the final graph.

Results Analysis. We analyze the impact of initialization in Table 7. First, the overall performance variance across strategies is marginal; for instance, in the Hard setting, the Success Rate remains consistently high for all strategies, confirming the robustness of our Adaptive MCMC sampler to initial states. Second, regarding the trade-off between efficiency and diversity, the *Dense Initial Graph* yields a slightly higher diversity in the Easy setting (NTD 46 vs. 42). This suggests that starting with high entropy can help explore more combinations in simple tasks, but this complexity results in a lower Success Rate compared to our method (89.90% vs. 91.50%) in Hard tasks. Ultimately, our default initialization strikes the best balance, achieving the highest Success Rate while maintaining competitive or superior graph diversity across all difficulty levels.

I. Instruction Prompt

I.1. Scene Graph Asset Generation

As introduced in Appendix B, scene graph asset generation consists of three components: object generation, attribute generation, and relation generation. We provide the corresponding prompt templates in the following prompt boxes.

Prompt Template for Object Generation

Task: Generate exactly 50 unique objects for the category '`{category}`' that are simple, common, and clearly visualizable for use in text-to-image generation.

Output Format:
The response must be a JSON dictionary with this exact structure:

```
{
    "category_name": "{category}",
    "objects": [
        {"id": 1, "name": "object1"},
        {"id": 2, "name": "object2"},
        ...
    ]
}
```

Requirements:

- Provide exactly 50 objects
- All object names must be unique (case-insensitive comparison)
- Object names must be common, easily visualizable examples of the category
- Use lowercase names for consistency
- Ensure each object can be clearly visualized in a text-to-image setting
- Avoid abstract or overly complex objects
- Return ONLY the JSON dictionary

Category: `{category}`

Prompt Template for Attribute Generation

Task: Generate a JSON response with exactly 5 VISUALIZABLE attribute concepts and 5 VISUALIZABLE values per concept for the object '`{object_name}`'.

Response Structure:

```
{
    "object_name": "{object_name}",
    "attributes": {
        "concept1": ["value1", "value2", ...],
        "concept2": ["value1", "value2", ...],
        ...
    }
}
```

Requirements:

- Attribute concepts must be:
  - Instantly visually recognizable and concrete
  - Directly related to physical appearance or form
  - Simple, fundamental, not abstract
- Values must be:
  - Single words when possible
  - Clearly distinct from each other
  - Common and visually obvious
- DO NOT include abstract adjectives like "relaxed", "alert", "matte"
- Strictly avoid underscores (_); use spaces instead

Good Example (Simple and highly visual):

`{"object_name": "painting", "attributes": {"color": ["red", "blue", "green", "yellow", "black"], "style": ["abstract", "realistic", ...], "size": ["small", "medium", "large", ...]}}`

Bad Example (Too complex):
`{"color_palette": ["monochromatic", ...], "brushwork_style": ["smooth/blended", ...], ...}`

Object to analyze: `{object_name}`

Prompt Template for Relation Generation

Task: Generate one visual relation between two objects in a text-to-image scene.
Given objects:
- Subject: `{subject_name}`
- Object: `{object_name}`

Requirements:
- Generate a relation where `{subject_name}` acts on `{object_name}`
- The relation must represent natural interaction between subject and object

Relation Categories (prioritize spatial):
Spatial Relations (PREFERRED):
- Position: on, under, above, below, beside, next to, near, far from, in front of, behind
- Containment: inside, within, outside, around, surrounding, enclosing
- Contact: touching, against, leaning on, resting on, attached to, connected to
- Orientation: facing, pointing to, directed toward, aligned with
- Relative: left of, right of, between, among, across from

Action Relations:
- Physical: holding, carrying, pushing, pulling, lifting, dropping
- Interaction: using, operating, playing with, examining, touching
- Movement: approaching, moving toward, following, chasing

Functional Relations:
- Purpose: for, used by, designed for, intended for
- Ownership: belongs to, owned by, part of
- State: connected to, linked to, associated with

Examples:
- person + chair → "sitting on" or "standing beside"
- book + table → "on" or "lying on"
- car + road → "on" or "driving on"
- bird + tree → "perched on" or "flying near"
- cup + saucer → "on" or "resting on"
- dog + house → "inside" or "in front of"

Output Format:
Return ONLY the base-form relation word/phrase (no subject/object names, no full sentences).
Examples: "on", "holding", "next to", "inside", "behind"

I.2. Input Text Generation

We also provide the prompt template to generate the input text in the following prompt box.

Prompt Template for Input Text Generation

Input Format:
Given the following scene description components in JSON format:
`{scene_graph}`

Task: Generate a single, coherent description that EXACTLY represents all given elements.

STRICT REQUIREMENTS:

- ONLY mention relationships EXPLICITLY defined in the "relations" array
- DO NOT create new relationships between objects
- DO NOT use prepositions like "in", "on", "with", "near" unless they are EXACT relation words in the "relations" array
- NEVER group objects together unless they have an explicit relationship
- List all objects separately if they don't have relationships with other objects
- Include EVERY object with its EXACT attributes as specified
- For objects with the same name, use their attributes or explicit relationships to distinguish them
- DO NOT use artificial identifiers like "#1" or "#2"
- MUST use the ORIGINAL wording for ALL elements
- MUST NOT add any information not present in the scene graph
- List every object instance explicitly. For identical objects, state their count (e.g., 'two identical apples')

EXAMPLES:
Scenario 1: Scene graph: "desert with shrubs", "eagle", "rocket" (NO relations)
BAD: "In a desert with shrubs, there is an eagle and a rocket"
GOOD: "A desert with shrubs, an eagle, and a rocket"

Scenario 2: Scene graph: "truck", "pizza with olive oil sauce", "truck with a flat cabin type", "motorcycle with a boxy shape", "orange juice", "meadow", "monitor" (NO relations)
BAD: "There is a truck, a pizza with olive oil sauce, a truck with a flat cabin type..."
GOOD: "There are two trucks, one with a flat cabin type, a pizza with olive oil sauce, a motorcycle with a boxy shape, orange juice, a meadow, and a monitor."

Scenario 3: Scene graph: "dog" (white), "dog" (yellow), "hamburger", relation: "dog (white) eating hamburger"
GOOD: "A white dog is eating a hamburger, and a yellow dog is present."

More Examples:
Bad: "A fire hydrant, a slim router, an adult, a waterfall, a cow, a manhole cover, a waterfall with a veil shape, and another manhole cover."
Good: "A fire hydrant, a slim router, an adult, two distinct waterfalls (one standard and one veil-shaped), a cow, and two standard manhole covers."

Output Format:
Output ONLY this JSON format with NO additional text:
`{{"prompt": "Generated description here"}}`

J. Additional Visualization Results

In Section 5.2, our visualized examples in Figure 4 demonstrate CompGen's superior compositional capabilities over strong baselines (SDXL, Lumina-Next). Here, we provide additional visualized results in Figures 8 and 9.

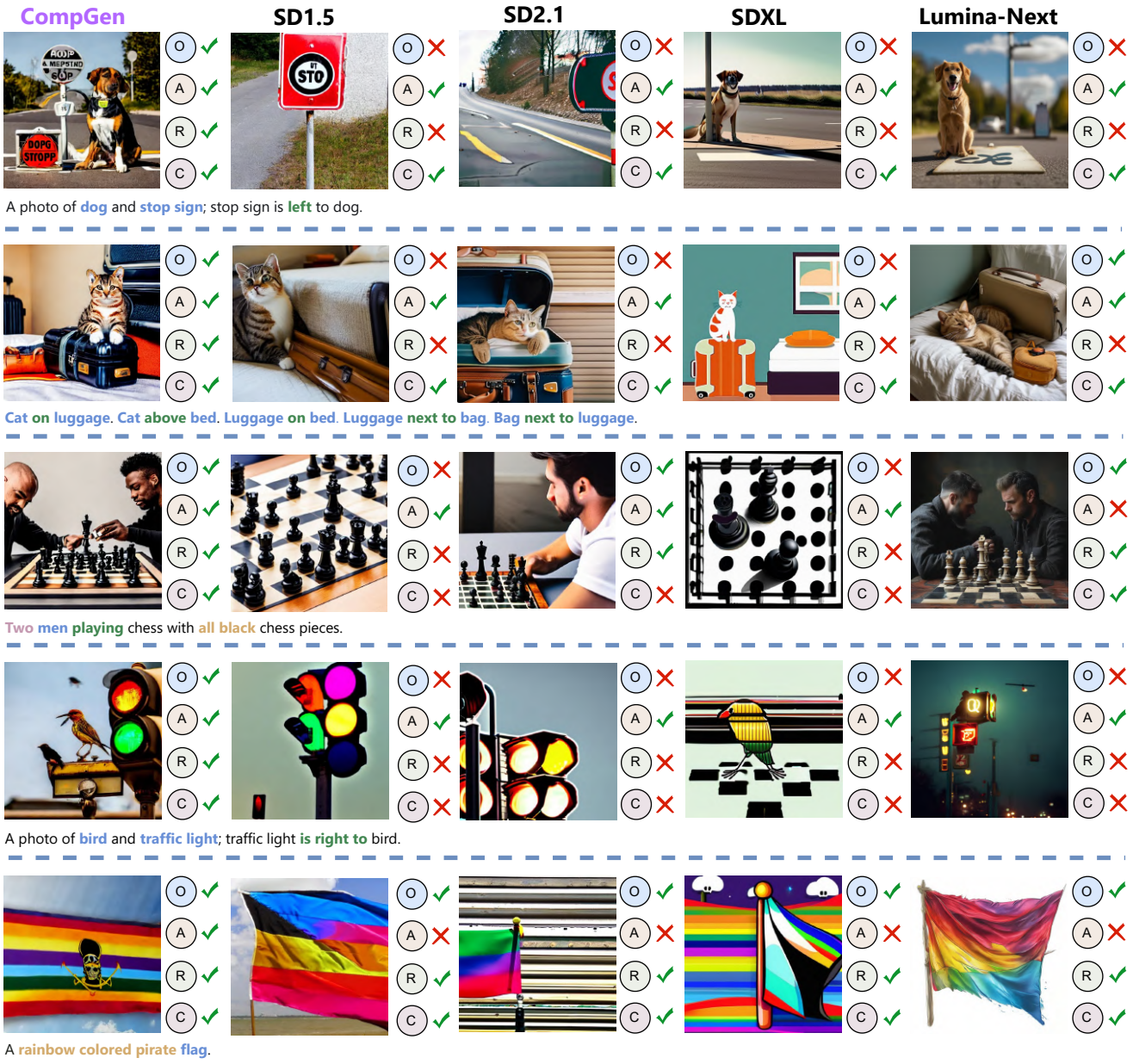

Figure 8. Extended qualitative comparison of our CompGen with other strong text-to-image models (SD1.5, SD2.1, SDXL, and Lumina-Next). Within each prompt, we color the elements for which at least one model makes an error: the object in **blue**, the attribute in **brown**, the relationship in **green**, and the count in **purple**. Ⓞ, Ⓐ, Ⓡ, Ⓒ denote Object, Attribute, Relationship, and Count, respectively. A ✓ indicates correct generation, while a ✗ indicates an error.

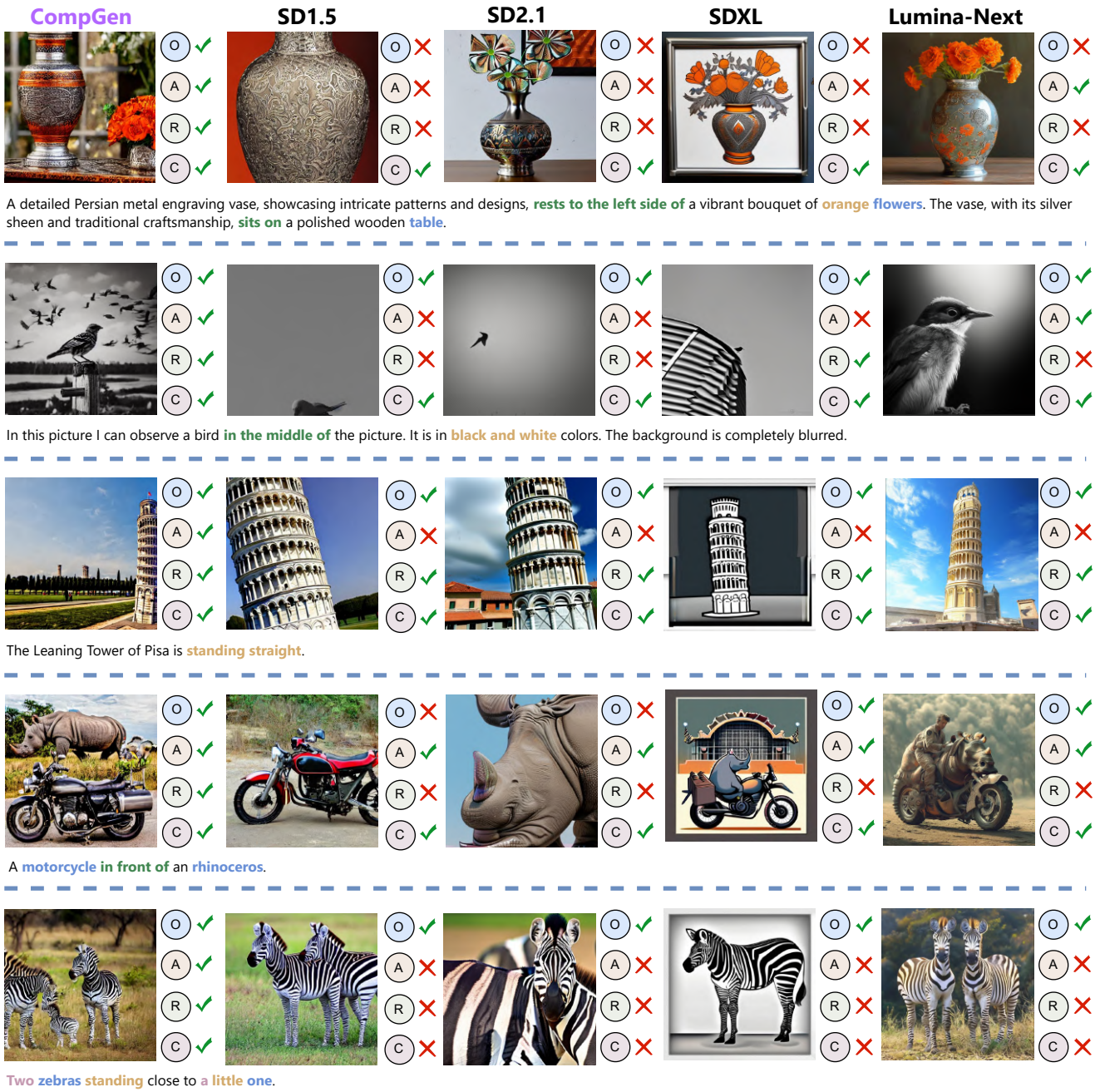

Figure 9. Extended qualitative comparison of our CompGen with other strong text-to-image models (SD1.5, SD2.1, SDXL, and Lumina-Next). Within each prompt, we color the elements for which at least one model makes an error: the object in **blue**, the attribute in **brown**, the relationship in **green**, and the count in **purple**. Ⓞ, Ⓐ, Ⓡ, Ⓒ denote Object, Attribute, Relationship, and Count, respectively. A ✓ indicates correct generation, while a ✗ indicates an error.