Improving Physical Object State Representation in Text-to-Image Generative Systems

Anonymous CVPR submission

Paper ID 61

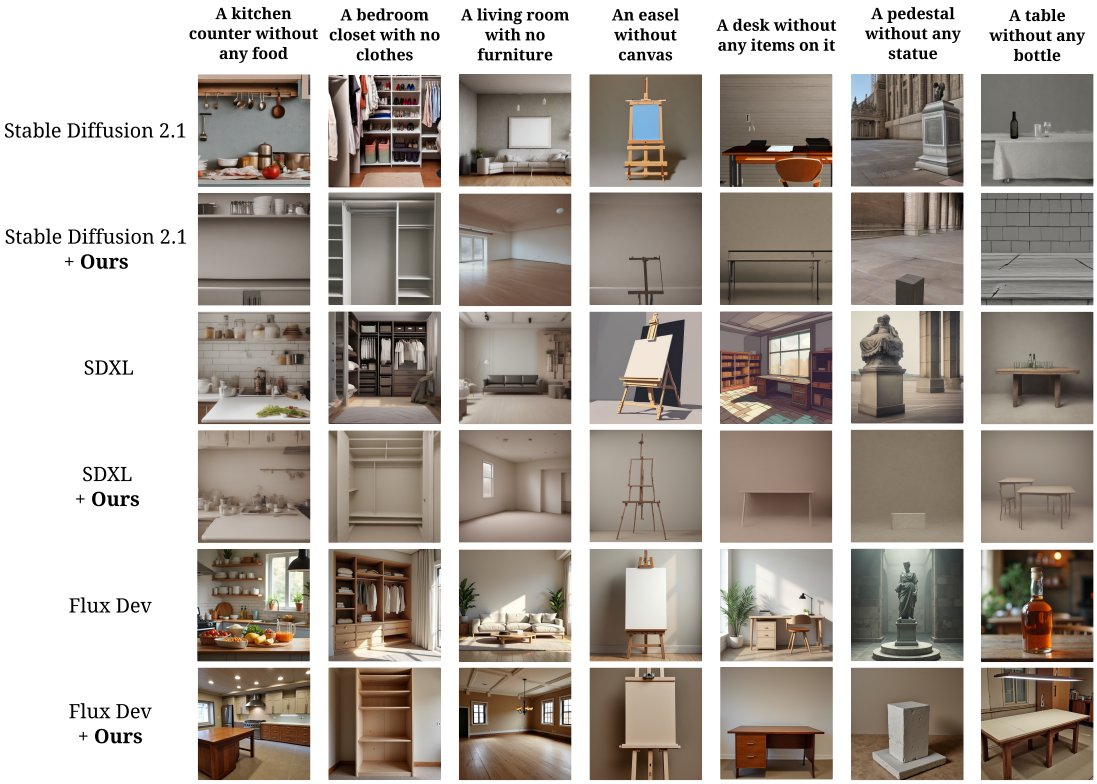


Figure 1. **Current text-to-image models struggle to depict common objects in varied physical states**, inaccurately include unintended objects or fail to depict the requested empty or absence state (e.g., prompting for "A kitchen counter without any food" still results in a kitchen count full of food). Our method addresses these issues and yields accurate object state representation.

Abstract

Current text-to-image generative models struggle to accurately represent object states (e.g., "a table without a bottle," "an empty tumbler"). In this work, we first design a fully-automatic pipeline to generate high-quality synthetic data that accurately captures objects in varied states. Next, we fine-tune several open-source text-to-image models on this synthetic data. We evaluate the performance of the fine-tuned models by quantifying the alignment of the generated images to their prompts using GPT, and achieve an average absolute improvement of ***8+%*** *across four models on the public GenAI-Bench dataset. We also curate a collection of 100 prompts with a specific focus on common objects in various physical states. We demonstrate a significant improvement of an average of* ***27+%*** *over the baseline on this dataset. We will release all evaluation prompts and code soon.*

1. Introduction

Recent advances in text-to-image generation [2, 4, 6, 7, 10, 18, 19, 22, 25, 26, 28] have significantly improved the visual quality and correctness of the generated images and unlocked incredible potential for human creative expression. Despite these significant improvements, as illustrated in Fig. 2, existing generative and visual language models still face challenges in accurately capturing spatial relationships [23] and simple real-world physical states such as presence, absence, empty, full [17], and so on. Despite being trained on billions of images, recent studies [30] and examples in Fig. 2 suggest that the generative systems are latching onto the intended and unintended co-occurring contexts represented in the training data and lack a fundamental understanding of object states.

Let us consider the example of a kitchen shelf. A simple web search indicates that a significant portion of images of a kitchen shelf are typically shelves filled with a variety of objects. We posit that this may inadvertently induce *contextual bias* into both generative and vision language models trained on such data, leading to text-to-image systems mostly generating a kitchen shelf in a "full" or "occupied" state. This scenario is further exacerbated given that captions generated on training data usually capture objects *present* in the image and not the ones *absent* in it. Recent studies [1, 24] have shown evidence that CLIP [21], the defacto text encoder in most generative image models, struggles to understand negation. How then should we impart the concept of absence of an object (e.g., a table without a vase) or negation of a physical state (e.g., an empty bottle) to the text to image generative system?

We tackle this issue in our work. Our key idea is to supply a text to image model with more evidence of a variety of objects in diverse physical states during training so that the model implicitly learns what absence or negation of an object should visually look like. We do this by first designing an automated pipeline to generate high-quality synthetic data that explicitly captures daily objects in varied, naturally feasible object states. The data generation pipeline (Fig. 3) comprises a step involving generating template-like prompts describing common objects in different physical states. Next, we use an off-the-shelf text to image model to generate synthetic images and subsequently filter out those not representative of objects in absent or empty states using a vision language model. We then finetune publicly available text to image models on this diverse generated data.

Our experiments indicate that finetuning on this curated synthetic dataset enriches a model's understanding of objects and their various physical states. Furthermore, we probe if finetuning leads to a more holistic understanding of objects in diverse physical states or mere memorization. To this end, we test the model on prompts comprising objects not part of the finetuning data. Across the four models we evaluate, we observe an average improvement of at least **5.6%** on novel, unseen objects, indicating that finetuning led to a better, more generalizable latent space. Qualitative comparisons (Fig. 1) further illustrate these improvements. Beyond enhancing the overall capability of generating objects in diverse states, we also examine the impact of finetuning on visual quality. Our analysis shows that both the CLIP score and the FID remain at similar levels after finetuning, with the FID showing no significant change, confirming that our synthetic data does not introduce undesirable visual artifacts. We also show that finetuning on high quality synthetic data does not degenerate performance on other prompts not related to object states. We summarize our key contributions below:

- We propose a **fully-automatic synthetic data generation pipeline** to systematically create high-quality training data that explicitly targets objects in empty, negation, and absent states (Sec.3).
- We fine-tune four open-source text-to-image generative models on this synthetic data and evaluate the generations on the publicly available GenAI-Bench dataset [11]. We show that the finetuned models yield an improvement of averaging **8+%** across the four models we experimented with, as measured by quantifying the alignment of the generated images to their prompts (Sec.4).
- We introduce a novel prompt collection, **Object State Bench**, specifically designed to evaluate models in complex physical states. All finetuned models yield an overall improvement of an average of **27+%** over their untuned baselines on this benchmark. This strong result underscores the substantial potential of our synthetic data pipeline in addressing the critical limitation of capturing object states in text-to-image generation. We will make **Object State Bench** publicly available very soon.

2. Related Work

Iterative Correction and Guidance Techniques: Recent studies have addressed shortcomings in text-to-image synthesis by incorporating iterative correction mechanisms. For example, Wu et al. [27] propose a Self-correcting LLM-controlled Diffusion framework that leverages large language models to iteratively refine generated images through latent space operations such as addition, deletion, and repositioning. Similarly, Liu et al. [14] introduces a particle filtering framework that uses external guidance, such as object detectors and real images, to mitigate errors such as missing objects and image distortions. Although these methods improve image fidelity, they typically require additional inference-time computations and complex feedback loops.

Synthetic Data Generation: Complementary to iterative correction methods, synthetic data generation has emerged as a promising strategy to overcome limitations in training datasets. Recent work [15] demonstrates that synthetic data

Figure 2. **Several state-of-the-art closed-source text-to-image generation models struggle to generate objects in absent or negation state**. This illustrates the limitations of current advanced generative systems in accurately representing object in simple and common physical states.

produced via diffusion models can enhance model robustness and generalization. Our method builds on this idea by adding a fully automatic pipeline that creates high-quality synthetic image-prompt pairs. These pairs clearly show different object states that aren't always shown in real-world data. Distinct from methods relying on iterative correction or resampling during inference, our pipeline directly enhances the training process. We fine-tune open-source text-to-image models on our synthetically generated data, thereby improving the semantic alignment between generated images and their textual descriptions. This data-centric framework not only simplifies the overall generation pipeline but also offers an extensible solution to address critical semantic limitations in text-to-image synthesis.

3. Approach

To address the gap in accurately generating objects in common physical states using text-to-image models, we propose a fully automatic synthetic data generation pipeline. As illustrated in Fig. 3, we first identify a diverse set of real-world objects and compose prompts referring to those objects in different physical states. Next, we generate images corresponding to those prompts using an off-the-shelf text to image generative model. Following this, we leverage Large Language Models (LLMs) to refine prompts and introduce diversity in their syntax. We also use a Large Vision-Language Model (LVLMs) for visual verification of the object state representation, to reduce noise in the generated data. We describe each step in detail next.

Noun identification, prompt, and image generation: First, we use a large language model and curate a wide

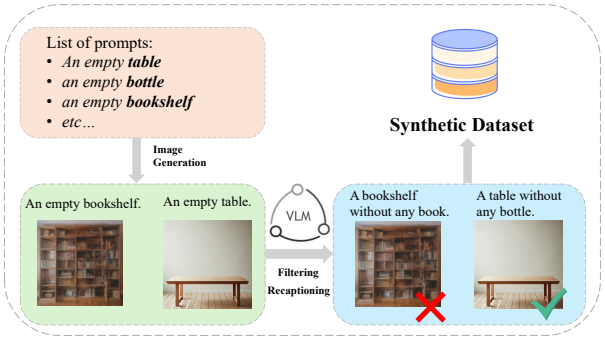


Figure 3. **Overview of the proposed synthetic data generation pipeline:** We generate prompts describing common objects in different physical states. We next create images from the prompts, evaluate for the correct representation of the object state using GPT4o-mini [9]. We rephrase prompts to introduce diversity in the sentence structures, length, and objects specified.

range of real-world objects such as containers, tables, shelves, rooms, and drawers, that can be depicted in empty, full, and absent states. We next use a vision language model to compose simple prompts around these objects that explicitly describe empty states (e.g., "an empty table"). We note that in our work, we focus only on curating prompts and images that represent only empty object states as they are more commonly underrepresented in most datasets.

Image Synthesis and Filtering: Next, we utilize an off-the-shelf text-to-image generation model to produce multiple candidate images per prompt using random seeds to ensure sufficient content diversity. Given that existing models struggle to generate images that represent object states cor-

Prompt for filtering generated images not representing absence or empty state of an object:

"You are an assistant that evaluates whether an image correctly represents the 'empty state' of an object as described in the caption. Specifically, check if the main object appears empty or unoccupied and confirm that the described absent object is not present in the image. Does the image accurately reflect both conditions? Return 'Yes' or 'No'."

Figure 4. **System prompt used** on the generated images for filtering out images not aligning with the provided prompt.

rectly, we pass each generation through a vision language model, and use visual-question-answering prompts to filter out generations that do not capture the object states correctly

Recaptioning: Finally, we use a large language model to introduce more diversity into the template-like initial prompt syntax. For example, the initial prompt "an empty table" could be refined to "a table without any bottle or book," increasing the complexity and clarity of the generated prompts. Our full pipeline is depicted in Fig. 3.

We finetune several open-sourced models on this synthetically generated image data, which we describe next.

4. Experiments

This section briefs about the datasets and evaluation metrics, baseline models, followed by fine-tuning. We also study the effect of different design choices we make in our overall approach.

4.1. Implementation Details

Synthetic data generation: The synthetic data pipeline has multiple modules involved to ensure high quality training dataset and uses GPT-4o-mini [9] in every step. Specifically, we use GPT-4o–mini to generate about 3000 different common objects and template-style prompts capturing these objects in empty or absent states. Next, we employ few-shot prompting technique [3] which has evidence to show better performance aligning with the prompted task in large language models. For generating the synthetic images, we use Stable Diffusion 1.5 [22] for 30 inference steps with a CFG scale of 5.0. We choose a slightly lower CFG value than the default to ensure more diversity in the synthetic training data while adhering to the actual input prompt. We again use GPT 4o-mini [9] to filter out images which incorrectly capture object states as mentioned in 3 and to rephrase the template-like prompts 9. This process resulted in 7600 synthetic image-text pairs.

Figure 5. **Qualitative comparison of object state improvement for Stable Diffusion-1.5:** *(top)* row shows the Stable Diffusion-1.5 baseline model, while the *(bottom)* row displays fine-tuned with our synthetic data pipeline, yielding more precise object state representation.

Figure 6. **Qualitative comparison of object state improvement for Stable Diffusion-2.1:** *(top)* row shows the Stable Diffusion-2.1 baseline model, while the *(bottom)* row displays fine-tuned with our synthetic data pipeline, yielding more precise object state representation.

Evaluation benchmarks: There exist very few public datasets that specifically focus on evaluating models on prompts capturing objects in varied physical states. The recently introduced GenAI-Bench [11], and the subset of 347 prompts belonging to the "negation" category is the closest to the scenario we study in this work. A sample prompt from this set is: "the girl with glasses is drawing, and the girl without glasses is singing." Here the prompt challenges the model to generate both presence and absence of the

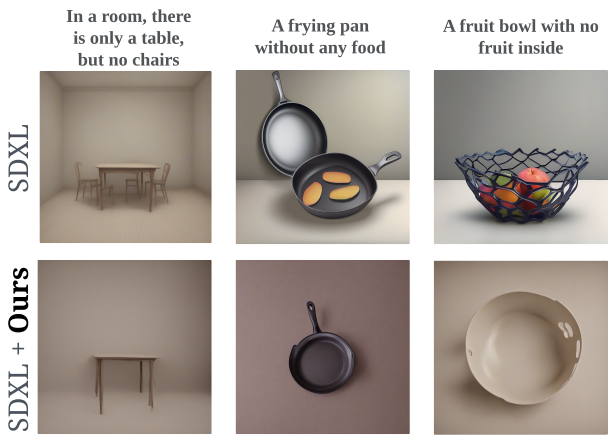


Figure 7. **Qualitative comparison of object state improvement for SDXL:** *(top)* row shows the SDXL baseline model, while the *(bottom)* row displays fine-tuned with our synthetic data pipeline, yielding more precise object state representation.

Figure 8. **Qualitative comparison of object state improvement for Flux Dev:** *(top)* row shows the Flux Dev baseline model, while the *(bottom)* row displays fine-tuned with our synthetic data pipeline, yielding more precise object state representation.

Prompt for recaptioning into passive voive prompts:

"The original prompt for the image is: '{original_prompt}'. Please refine the prompt by specifying an absent object if it is not already mentioned, but avoid redundant descriptions of emptiness. Ensure the refined prompt naturally integrates the missing object without repeating words like 'empty' or 'vacant'. For example: 'An empty table.' → 'A table without any bottles on it.', 'A deserted park.' → 'A park without any people.' If the original prompt is already sufficiently detailed, return it as is."

Figure 9. **System Recaptioning Prompt**: This figure shows the system prompt that transforms template-like prompts into passive voice. The examples instruct the model to enhance the prompt by adding a missing object and avoiding redundant emptiness descriptors.

Prompt for evaluating generated images on representing absence or empty state of an object:

"You are an assistant that evaluates whether an image correctly represents the 'empty state' of an object as described in the caption. The caption is: {original_prompt}. Specifically, check if the main object appears empty or unoccupied and confirm that the described absent object is not present in the image. Does the image accurately reflect both conditions? Return 'yes' or 'no'."

Figure 10. **System Prompt for Evaluation:** This figure presents the prompt used to assess whether a generated image accurately represents the absence or empty state of an object as described in the caption.

same object (glasses) but on different people. However, the negation category also has prompts that test absence of attributes (instead of objects), e.g., "a person with short hair is crying while a person with long hair is not." Thus, we manually go through these prompts and retain only those that are more aligned to our task, resulting in 214 prompts. We call this subset as **GenAI-Object-State** dataset.

Additionally, we manually curate a set of 100 prompts, titled **Object-state-Bench**. This benchmark consists of two parts: one-half of the prompts are generated using the synthetic prompt generation pipeline and the other half is curated by human annotators tasked with describing common objects around them in empty or absent states. This design incorporates both machine-generated and human-authored descriptions, ensuring diversity and realistic linguistic variability for a robust evaluation of model performance.

Evaluation metrics: We quantify our model performance using the Visual Question Answering score (VQA-score) introduced by Lin et al. [13]. The metric utilizes a fine-tuned version of the Google's FLAN-T5-XXL model [5] with contrastive language-image pre-training [21]. We use the default prompt given by the authors: "Does this figure show "prompt"? Please answer yes or no." Additionally, we also use OpenAI's GPT-4o-mini model [9] for evaluation, where we query with an evaluation prompt specified in Fig. 10 and the generated image. The model returns a yes or no based on whether the object state has been correctly depicted.

Method	GenAI-Object-State		Object State Bench	
	GPT (↑)	VQA (↑)	GPT (↑)	VQA (↑)
Stable Diffusion 1.5 [22]	16%	45%	49%	56%
Stable Diffusion 2.1 [22]	16%	47%	44%	53%
SDXL [16, 20]	15%	47%	39%	52%
Flux DEV [10]	11%	39%	27%	41%
Stable Diffusion 1.5 + Ours	21%	46%	**69%**	**57%**
Stable Diffusion 2.1 + Ours	**23%**	49%	68%	66%
SDXL + Ours	**23%**	**52%**	66%	67%
Flux Dev + Ours	22%	50%	65%	64%

Table 1. **Baseline vs finetuned results on different models**: GPT (↑) stands for GPT [9] correct rate and VQA (↑) stands for VQA-Score [13] (higher is better). The cyan colored rows represent the performance of fine-tuned models' with SD 1.5 [22] as synthetic data generator. The highlighted numbers indicate the highest performance for the corresponding dataset and metric.

4.2. Finetuning setup

Implementation Details: We finetune Stable diffusion 1.5, 2.1 [22], SDXL [16, 20], and Flux.1 Dev [10] on the proposed framework. For the Stable Diffusion family of models, we use a guidance scale of 7.5, which is their default value and for Flux DEV [10], we use a guidance scale of 3.5. Stable Diffusion 1.5 generates 512×512 dimensional output image while all other models generate 768×768 resolution. We infer Flux DEV version [10] with 50 inference timesteps as recommended in their documentation. For all other models, the number of inference time steps is 30.

We also ensure that the same seed of 1303 (chosen arbitrarily) is used across all the prompts of the dataset for a given run for all the models during baseline testing. We finetune all models using Low Rank Adapters (LoRA) [8]. The hyperparameters of LoRA are detailed in Appendix.

4.3. Overall performance improvements

In Table 1, we report the GPT evaluation score and VQA scores of models before and after finetuning. As shown in Table 1, our approach yields significant gains in semantic alignment, as measured by GPT evaluation scores. For instance, fine-tuning the Stable Diffusion 1.5 model with our synthetic dataset boosts its GPT score on the GenAI-object-state dataset from 15.6% to 21.0% (an improvement of 5.4 percentage points), and improves its performance on Object State Bench from 49% to 69%, a gain of 20 percentage points. Similarly, Stable Diffusion 2.1 improves from 16.0% to 23.0% (+7.0 percentage points) on GenAI-object-state and from 44% to 68% (+24 percentage points) on Object State Bench, while SDXL advances from 15.0% to 23.0% (+8.0 percentage points) on GenAI-object-state and from 39% to 66% (+27 percentage points) on Object State Bench respectively. Also, the Flux.1 DEV model shows an improvement of 11 percentage points from 11% to 22% on the GenAI-Object-State and a 38 percentage points improvement from 27% to 65% on the Object-State-Bench. These consistent improvements across all models demonstrates generalizability of the proposed data synthesis and finetuning strategy.

Furthermore, qualitative comparisons (Fig. 5, 6, 7, and 8) visually illustrate the enhanced semantic alignment in the generated images after fine-tuning on samples from both the GenAI-object-state and Object State Bench datasets.

While our fine-tuning strategy generally improves the visual representation of object states, qualitative analysis reveals several different failure modes, as illustrated in Fig. 11. In Fig. 11a, the tuned model improves significantly by approaching the correct empty state. However, the representation remains imperfect, indicating that the model converges toward the intended state without fully capturing all semantic details. In contrast, Figure 11b shows instances where the tuning process overemphasizes emptiness, resulting in an overrepresentation of the empty state, which, causes the object itself to be imperfectly represented. These observations highlight the delicate balance required in finetuning: while reinforcing the concept of emptiness is beneficial, overemphasis can degrade the precise representation of the object.

4.4. Effect of the synthetic data generator

Table 2 reports the performance of three tuned generative models when fine-tuned with synthetic data generated by different models. We evaluate performance on the GenAI-Object-State and Object State Bench using the GPT score. For each tuned generative model, the choice of synthetic data generator affects the results. For instance, when tuning Stable Diffusion 1.5 [22], using SDXL [16, 20] as the synthetic generator boosts the GenAI-Object-State score to 24% compared to 21% when using either Stable Diffusion 1.5 or 2.1, although the best Object State Bench performance (69%) is achieved when using SD 1.5 or 2.1. Similarly, when tuning SDXL, synthetic data generated by Stable Diffusion 1.5 leads to a GenAI score of 23% and an Object State Bench score of 66%, slightly outperforming data generated by the other models. In the case of Stable Diffusion 2.1, using synthetic data from Stable Diffusion 1.5 or SDXL results in comparable performance (23% on GenAI and 68% on Object State Bench), with data from Stable Diffusion 2.1 trailing slightly on the Object State Bench (65%).

These findings indicate that our synthetic data generation pipeline is robust and effective across different underlying generative models, as the overall performance differences remain relatively small. The results also reveal a trade-off between the evaluation benchmark, an improvement in the GenAI-Object-State Bench may sometimes coincide with a slight drop in the performance of the Object State Bench, suggesting that different synthetic data generators may emphasize different aspects of object state representation. Nevertheless, regardless of the specific synthetic

Tuned Generative Model	Synthetic Data Generator	GenAI-Object State	Object State Bench
Stable Diffusion 1.5[22]	Stable Diffusion 1.5[22]	21%	**69%**
	SDXL[16, 20]	**24%**	67%
	Stable Diffusion 2.1[22]	21%	**69%**
SDXL[16, 20]	Stable Diffusion 1.5[22]	23%	66%
	SDXL[16, 20]	20%	65%
	Stable Diffusion 2.1[22]	22%	67%
Stable Diffusion 2.1[22]	Stable Diffusion 1.5[22]	23%	68%
	SDXL[16, 20]	23%	68%
	Stable Diffusion 2.1[22]	22%	65%

Table 2. **Performance Comparison:** This table compares the performance of different synthetic data generators when used to finetune generative models on two benchmarks: GenAI-Object-State and Object State Bench. Results are reported as GPT evaluation scores (higher is better), with bold values indicating the best performance in each configuration.

data generator used, all tuned models show substantial improvements over baseline performance. This result confirms that our pipeline is robust and, consistently enhances model performance across various model architectures.

4.5. Performance on unseen object states

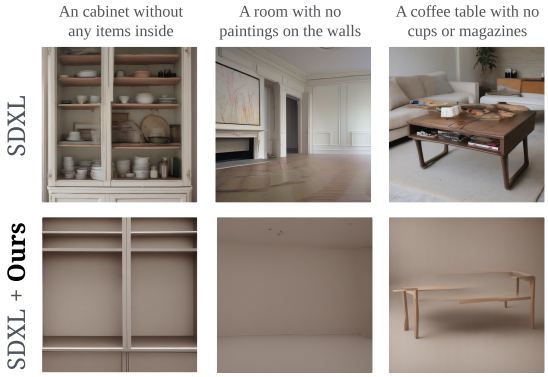
Our overarching goal is to teach the model the concept of emptiness or absence of an object. To this end, we study whether the model's understanding of object states learned during training generalizes to novel, unseen objects. We first identify 100 novel objects that are not part of the 3000 objects used for training. Furthermore, we manually inspect the list of objects and filter out if the new object is similar to the training objects (see Appendix A for the complete list of objects). Using our data generation pipeline described in Sec. 3, we generated images for these objects.

The results, reported in Table 3 (Column 2), show an improvement in the GPT score from 13.0 to 20.0 (+7 percentage points). Such significant improvements on unseen objects indicates that finetuning on synthetic data of objects in absent and negation states is leading to a more comprehensive understanding of physical states even on novel, unseen objects.

Model	Objects in non-empty states		Unseen objects	
	GPT (↑)	VQA (↑)	GPT (↑)	VQA (↑)
Stable Diffusion 1.5 [22]	**40%**	68%	13%	44%
Stable Diffusion 1.5 [22] + Ours	39%	**69%**	**20%**	**50%**

Table 3. **Evaluating Generalizability:** This table compares the performance of the baseline Stable Diffusion 1.5 with our finetuned variant on two evaluation sets: objects in non-empty states and unseen objects. GPT and VQA scores (higher is better) are reported for each category, with bold values indicating the best performance.

(a) Tuned model shows improved object state, though not fully correct.



(b) Tuned model overrepresents the empty state, shifting the object depiction.

Figure 11. **Qualitative Tuning Effects:** Figure (a) shows cases where tuning improves the depiction of the object state (although with some imperfections), while Figure (b) illustrates instances where tuning overemphasizes emptiness, leading to a deviation from an accurate object representation.

4.6. Performance on non-empty object states

Given our finetuning data consists of objects in empty or absent states, we study the performance on prompts describing objects in full state (e.g., "a tumbler full of water"). We provide these prompts in the suppl. material. We report the results in Table 3 column 1. Our results indicate that there is minimal difference in the performance of the models even after fine-tuning. This experiment highlights that our approach does not lead to catastrophic forgetting of objects in full states even though this is not explicitly represented in the finetuning data.

4.7. Is synthetic data generation necessary?

To evaluate whether synthetic data is essential for improving text-to-image generation, we extract 12K training examples from both the COCO [12] dataset and a video dataset (VidOSC [29]) that contain captions related to object states.

Data Source	GPT (↑)	VQA (↑)
Stable Diffusion 1.5[22] (Baseline)	49%	56%
COCO[12]	46%	56%
VidOSC [29]	37%	46%
Stable Diffusion 1.5 + Synthetic Data (Ours)	69%	57%

Table 4. **Data Source Performance Comparison:** This table compares the performance of models fine-tuned on various data sources. Training on our synthetic dataset significantly boosts the GPT score (from 49% to 69%) and provides a modest improvement in VQA score over real-world datasets such as COCO [12] and VidOSC [29].

For the real-world data, we filter the captions to retain only those whose associated prompts and images suggest a potential empty state, even if not explicitly described, as such samples are relatively scarce, as we discussed in the above sections. We then refine these captions, following the same process as our synthetic pipeline. The key difference between the synthetic dataset and the real-world dataset is that, for each filtered prompt in the real-world datasets, we directly use the corresponding image. In contrast, because the synthetic pipeline lacks corresponding real images and is based on a limited set of original prompts, we generate each prompt with multiple random seeds and perform reception on each generation.

Table 4 compares the performance of models fine-tuned on these different data sources. Despite training all models on approximately 12K images for 400 steps, the synthetic dataset significantly outperforms the baseline. The modest gains from the real-world datasets suggest that, despite similar training set sizes, they contain relatively few high-quality samples that clearly represent object empty states. For a visual comparison of training samples across these datasets, please refer to the figure in the supplementary material. In contrast, our synthetic dataset, specifically designed to capture varied object states, provides a much stronger fine-tuning signal, leading to a significant improvement in image-prompt alignment.

Generative Model	FID (↓)	CLIP-Score (↑)
Stable Diffusion 1.5 [22]	24.32	0.31
Stable Diffusion 1.5 [22] + Ours	25.74	0.32

Table 5. **Effect of finetuning on synthetic data on visual quality:** Comparison of FID (lower is better) and CLIP-Score (higher is better) for Stable Diffusion 1.5 on COCO val2014 dataset, 10K randomly sampled subset.

4.8. Performance impact on prompts not related to object states

To verify whether the proposed framework impacts performance on prompts unrelated to object states, we leverage GenAI-Bench [11] and sample 50 random prompts which are specifically outside the "negation" set. We test the performance of both the base and finetuned Stable Diffusion 2.1 [22] models when the synthetic data is generated using Stable Diffusion-1.5. Upon evaluating the generated images on such randomly sampled prompts, we observe that the base model has a GPT score of 10% while our fine-tuned model has a performance of 16%(+6 percentage points). This demonstrates that our approach improves a generative model's understanding of object states without deteriorating performance on other non-object state related prompts.

4.9. Impact of recaptioning in the pipeline

One of the modules in the synthetic data pipeline is the recaptioning segment, where we convert the template like prompts to passive voice prompts. For example, a prompt such as "An empty table." can be recaptioned as "A table without any bottles on it" (see Fig. 9), shifting from a direct, template-like description to a more informal passive construction. We evaluate the importance of this step by finetuning Stable Diffusion 1.5 [22] and SDXL [16, 20] with and without recaptioning the training data. We report the results in Table. 6 and observe 3 and 2 percentage point improvement in GPT and VQA scores respectively on GenAI-Object-State. We also observe +9% GPT score and +3% VQA score points in Object-State-Bench. These results show that recaptioning to make prompts less-template like and more colloquial aligns them.

Model	Recaptioning	GenAI-Object-State		Object State Bench	
		GPT (↑)	VQA (↑)	GPT (↑)	VQA (↑)
Stable Diffusion 1.5 [22]	No	17%	44%	60%	54%
	Yes	21%	46%	69%	57%
Stable Diffusion 2.1 [16, 20]	No	20%	48%	64%	64%
	Yes	23%	49%	68%	66%
SDXL [16, 20]	No	20%	52%	63%	66%
	Yes	23%	52%	66%	67%

Table 6. **Effect of Recaptioning on Model Performance:** This table presents a comparison of models fine-tuned with and without recaptioning. The results show that incorporating recaptioning improves both GPT and VQA scores on the GenAI-Object-State and Object State Bench, demonstrating its effectiveness in enhancing semantic alignment.

5. Conclusion and Future Work

To improve the physical object state representation in existing text to image generative systems, we propose a fully automatic pipeline to generate high-quality synthetic data and use it to finetune any text-to-image models. We demonstrate that our approach improves the holistic understanding of objects in diverse physical states via two evaluation metrics GPT score and VQA score [13]. Future work entails exploring if such a data generation pipeline can be extended to other common failure models of image and video generative systems (compositional prompts, prompts involving generating text, generating objects of accurate counts, etc.) and exploring solutions with direct architectural tweaks to the model itself.

References

[1] Kumail Alhamoud, Shaden Alshammari, Yonglong Tian, Guohao Li, Philip Torr, Yoon Kim, and Marzyeh Ghassemi. Vision-language models do not understand negation. *arXiv preprint arXiv:2501.09425*, 2025. 2

[2] Omri Avrahami, Dani Lischinski, and Ohad Fried. Blended diffusion for text-driven editing of natural images. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 18208–18218, 2022. 2

[3] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020. 4

[4] Xiaokang Chen, Zhiyu Wu, Xingchao Liu, Zizheng Pan, Wen Liu, Zhenda Xie, Xingkai Yu, and Chong Ruan. Janus-pro: Unified multimodal understanding and generation with data and model scaling. *arXiv preprint arXiv:2501.17811*, 2025. 2

[5] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53, 2024. 5

[6] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021. 2

[7] Anastasis Germanidis. Introducing gen-3 alpha: A new frontier for video generation, 2024. Accessed: 2025-03-27. 2

[8] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022. 6

[9] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024. 3, 4, 5, 6

[10] Black Forest Labs. Flux. https://github.com/black-forest-labs/flux, 2024. 2, 6, 11

[11] Baiqi Li, Zhiqiu Lin, Deepak Pathak, Jiayao Li, Yixin Fei, Kewen Wu, Tiffany Ling, Xide Xia, Pengchuan Zhang, Graham Neubig, et al. Genai-bench: Evaluating and improving compositional text-to-visual generation. *arXiv preprint arXiv:2406.13743*, 2024. 2, 4, 8

[12] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer vision–ECCV 2014: 13th European conference, zurich, Switzerland, September 6-12, 2014, proceedings, part v 13*, pages 740–755. Springer, 2014. 7, 8, 11, 13

[13] Zhiqiu Lin, Deepak Pathak, Baiqi Li, Jiayao Li, Xide Xia, Graham Neubig, Pengchuan Zhang, and Deva Ramanan. Evaluating text-to-visual generation with image-to-text generation. *arXiv preprint arXiv:2404.01291*, 2024. 5, 6, 8

[14] Yujian Liu, Yang Zhang, Tommi Jaakkola, and Shiyu Chang. Correcting diffusion generation through resampling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8713–8723, 2024. 2

[15] Eugenio Lomurno, Matteo D'Oria, and Matteo Matteucci. Stable diffusion dataset generation for downstream classification tasks. *arXiv preprint arXiv:2405.02698*, 2024. 2

[16] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Guided image synthesis and editing with stochastic differential equations. *arXiv preprint arXiv:2108.01073*, 2021. 6, 7, 8

[17] Kaleb Newman, Shijie Wang, Yuan Zang, David Heffren, and Chen Sun. Do pre-trained vision-language models encode object states? *arXiv preprint arXiv:2409.10488*, 2024. 2

[18] OpenAI. Dall-e 3: Opinionated, boring, 2023. 2

[19] OpenAI. Sora: Generating videos from text, 2024. Accessed: 2025-03-27. 2

[20] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952*, 2023. 6, 7, 8

[21] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021. 2, 5

[22] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10684–10695, 2022. 2, 4, 6, 7, 8, 11

[23] Ander Salaberria, Gorka Azkune, Oier Lopez de Lacalle, Aitor Soroa, Eneko Agirre, and Frank Keller. Improving explicit spatial relationships in text-to-image generation through an automatically derived dataset. *arXiv preprint arXiv:2403.00587*, 2024. 2

[24] Jaisidh Singh, Ishaan Shrivastava, Mayank Vatsa, Richa Singh, and Aparna Bharati. Learn" no" to say" yes" better: Improving vision-language models via negations. *arXiv preprint arXiv:2403.20312*, 2024. 2

[25] Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023. 2

[26] Xinlong Wang, Xiaosong Zhang, Zhengxiong Luo, Quan Sun, Yufeng Cui, Jinsheng Wang, Fan Zhang, Yueze Wang, Zhen Li, Qiying Yu, et al. Emu3: Next-token prediction is all you need. *arXiv preprint arXiv:2409.18869*, 2024. 2

[27] Tsung-Han Wu, Long Lian, Joseph E Gonzalez, Boyi Li, and Trevor Darrell. Self-correcting llm-controlled diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6327–6336, 2024. 2

[28] Shitao Xiao, Yueze Wang, Junjie Zhou, Huaying Yuan, Xingrun Xing, Ruiran Yan, Shuting Wang, Tiejun Huang, and Zheng Liu. Omnigen: Unified image generation. *arXiv preprint arXiv:2409.11340*, 2024. 2

[29] Zihui Xue, Kumar Ashutosh, and Kristen Grauman. Learning object state changes in videos: An open-world perspective. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18493–18503, 2024. 7, 8, 11, 13

[30] Juntu Zhao, Junyu Deng, Yixin Ye, Chongxuan Li, Zhijie Deng, and Dequan Wang. Lost in translation: Latent concept misalignment in text-to-image diffusion models. In *European Conference on Computer Vision*, pages 318–333. Springer, 2024. 2

Appendices

A. List of Unseen Objects

In this appendix, we provide the complete list of 100 novel objects used for evaluation. These objects were carefully selected and manually filtered to ensure they are distinct from the 3000 training objects. [The detailed list is provided here.]

- birdhouse
- meat locker
- medicine bottle
- print cartridge
- shipping container
- fuel can
- bird's nest
-
- fuel drum
- shopping cart
- squeeze bottle
- photobook
- paper towel holder
- lipstick tube
- sunglasses rack
- jewelry roll
- wristwatch case
- gadget case
- marionette theatre
- drill holder
- silicone mold
- speech bubble
- caviar tin
- crisper drawer
- resealable bag
- hawser reel
- luggage trunk
- coffee tin
- mason jar
- picnic hamper
- jewelry stand
- picture frame
- tackle pouch
- flower vase
- pool hall
- exam room
- motel room
- kitchen closet
- terracotta planter
- patio chair
- operating theater
- animal cage
- phone booth
- quilt bag
- dessert cup
- sitting room
- ring box
- sleeping bag
- flower vase with no water
- traveling case
- banana holder
- rooftop garden bed
- pressure cooker
- guitar rack
- sous vide container
- cane basket
- treehouse
- planter
- charcoal holder
- dog treat tin
- glaze bucket
- hair tie holder
- bead organizer
- scale pan
- hammock frame
- bulletin board
- incense tray
- book safe
- picture rail
- mantle clock case
- snow globe
- football bladder
- card holder
- protein container
- chess board
- animal trap
- trailer
- mannequin
- candy mold
- pet water fountain
- vendor cart
- film reel
- stamp album
- first-adi kit
- toboggan
- ice cream maker
- hand towel rail
- flower sac
- toothbrush holder
- control pannel
- scout cabin
- preserving jar
- guitar case
- sunhat box
- noodle bowl
- wristlet
- salt grinder
- cake pop stand
- wine testing glass
- gas lamp
- billiard table

Figure 12. List of 100 unseen objects used for evaluation. These objects, which are distinct from the 3000 training objects, are detailed here for reproducibility.

B. List of Full state prompts

In this section, we provide the full state prompts 13 used for ablation study on performance on non-empty object states. These prompts were inspected manually to ensure there was no empty state reference.

C. Comparison of Training Samples

In this section, we provide a visual and qualitative comparison of training samples extracted from three different data sources: our synthetic dataset, COCO [12], and VidOSC [29]. Our synthetic dataset is specifically generated to capture object empty states with clear and consistent imagery. In contrast, the COCO dataset often fails to clearly represent object absence: frequently, the described object is not the focal point, while the VidOSC dataset, derived from video frames, suffers from motion blur and inconsistent viewpoints. These limitations in real-world data help explain why models fine-tuned on our synthetic dataset perform significantly better in generating accurate object states.

D. Hyperparameters for LoRA in finetuning

We otuline the hyperparameters used for LoRA during finetuning of the family of Stable Diffusion models [22] and Flux.1 DEV [10] in table 7.

Table 7. Hyperparameters used for fine-tuning with LoRA.

Hyperparameter	**SD family [22]**	**Flux.1 DEV [10]**
LoRA Rank	4	16
Resolution	512	512
Center Crop	True	True
Random Flip	True	False
Mixed Precision	fp16	bf16
Allow TF32	True	False
Training Batch Size	32	8
Gradient Accumulation Steps	1	1
Gradient Checkpointing	True	True
Learning Rate	1e-04	1e-04
Max Gradient Norm	1	1

E. Additional Qualitative Examples of Object State Failures

In this appendix, we provide further qualitative results that illustrate additional failure cases of closed-source text-to-image generation models when handling object state prompts. These examples reinforce our observations and emphasize the need for improved model training and data recaptioning.

F. Impact of Tuning Steps on Accuracy

In this section, we analyze how varying the number of fine-tuning steps affects generation accuracy, as measured by

1. A full bottle of water is placed on the table.
2. The cup is filled to the brim with hot coffee.
3. The plate is loaded with a delicious meal.
4. The bowl is full of fresh fruit.
5. The glass is filled with orange juice.
6. The jar is packed with homemade jam.
7. The container is filled with rice.
8. The box is stuffed with chocolates.
9. The bag is filled with groceries.
10. The wallet is thick with cash.
11. The suitcase is packed with clothes for the trip.
12. The backpack is filled with school supplies.
13. The envelope is stuffed with important documents.
14. The fuel tank is completely full, ready for a long drive.
15. The trash can is overflowing with garbage.
16. The sink is full of dirty dishes.
17. The bathtub is filled with warm, soapy water.
18. The fridge is stocked with fresh food.
19. The freezer is packed with frozen meals.
20. The oven is full of baking cookies.
21. The pan is filled with sizzling vegetables.
22. The pot is bubbling with hot soup.
23. The dish rack is full of clean plates.
24. The storage box is packed with winter clothes.
25. The wardrobe is filled with dresses and suits.
26. The bookshelf is packed with novels and textbooks.
27. The laundry basket is full of dirty clothes.
28. The washing machine is loaded with clothes.
29. The dryer is tumbling a full load of laundry.
30. The pencil case is filled with pens and markers.
31. The toolbox is stocked with hammers and screwdrivers.
32. The drawer is stuffed with office supplies.
33. The file cabinet is filled with paperwork.
34. The purse is heavy with personal items.
35. The shopping cart is loaded with groceries.
36. The refrigerator drawer is filled with fresh vegetables.
37. The spice rack is stocked with herbs and spices.
38. The medicine cabinet is filled with bottles of pills.
39. The candy jar is brimming with sweets.
40. The flower vase is full of fresh roses.
41. The aquarium is teeming with colorful fish.
42. The tea kettle is filled with boiling water.
43. The thermos is full of hot coffee.
44. The lunchbox is packed with sandwiches and snacks.
45. The picnic basket is overflowing with food and drinks.
46. The trash bag is full and needs to be taken out.
47. The egg carton is completely full.
48. The gas cylinder is filled with propane.
49. The rain barrel is full after the storm.
50. The bathtub is overflowing with bubbles.
51. The hard drive is full of stored files.
52. The email inbox is filled with unread messages.
53. The car trunk is packed with luggage.
54. The bread basket is full of warm rolls.
55. The coffee pot is filled with fresh-brewed coffee.
56. The pet food bowl is full for dinner time.
57. The ice cube tray is full and ready to freeze.
58. The cup holder is filled with soda cans.
59. The suitcase pocket is stuffed with travel essentials.
60. The fishing net is full of fresh catch.
61. The raincoat pockets are filled with small items.
62. The coin purse is full of loose change.
63. The fruit basket is overflowing with apples and bananas.
64. The measuring cup is filled with flour.
65. The battery pack is fully charged.
66. The balloon is filled with helium.
67. The notepad is full of handwritten notes.
68. The chalkboard is covered with writing.
69. The gift bag is stuffed with presents.
70. The music playlist is full of favorite songs.
71. The wine cellar is stocked with vintage bottles.
72. The parking lot is completely full.
73. The stadium is packed with cheering fans.
74. The toy chest is overflowing with stuffed animals.
75. The makeup bag is full of beauty products.
76. The tool shed is stocked with gardening equipment.
77. The bakery display case is filled with fresh pastries.
78. The cookie jar is full of chocolate chip cookies.
79. The seed packet is full of flower seeds.
80. The pet carrier is filled with cozy blankets.
81. The luggage rack is stacked with heavy suitcases.
82. The fishing bucket is full of water and fish.
83. The scrapbook is filled with memories.
84. The classroom board is covered with notes.
85. The violin case is packed with accessories.
86. The music stand is filled with sheet music.
87. The bike basket is loaded with fresh groceries.
88. The file folder is stuffed with reports.
89. The bread bin is stocked with fresh loaves.
90. The lemonade pitcher is full and ready to serve.
91. The attic is packed with old furniture and boxes.
92. The beach bag is full of towels and sunscreen.
93. The hospital bed is occupied with a patient.
94. The rain boot is filled with water after the storm.
95. The marshmallow jar is overflowing with sweets.
96. The milk carton is completely full.
97. The Christmas stocking is filled with gifts.
98. The dog's food bowl is filled with kibble.
99. The holiday suitcase is packed with vacation clothes.
100. The bus is completely full of passengers.

Figure 13. List of 100 full state prompts used for ablation study.

Figure 14. **Training Sample Comparison from Different Data Sources:** The top row shows samples from our synthetic dataset, the middle row displays samples from COCO [12], and the bottom row presents samples from VidOSC [29]. Our synthetic dataset clearly captures object empty states with focused and consistent imagery. In contrast, COCO samples often fail to clearly depict the absence of an object, with the described object not being the focal point, while VidOSC samples suffer from motion blur and inconsistent viewpoints. These factors contribute to the superior performance of our synthetic dataset.

the GPT score over GenAI-Object-State and Object-State-Bench. Table 8 summarizes the results of each model on the Object-State-Bench and GenAI-Object-State benchmarks.

For the Object-State-Bench, all three models show steady improvements as the number of tuning steps increases. For example, Stable Diffusion 1.5 improves from 67% at 200 steps to 70% at 800 steps, while SDXL improves from 65% to 70% within the same range. Stable Diffusion 2.1 maintains a consistent performance of 68%, regardless of tuning steps. These trends indicate that for benchmarks that require precise object state representation, additional tuning steps can consistently improve performance, though gains may stop at higher steps.

In contrast, the results from the GenAI-Object-State benchmark show less monotonic behavior. The performance of Stable Diffusion 1.5 improves slightly from 20% at 200 steps to 21% at 400 steps, but then decreases to 18% at 800 steps. Stable Diffusion 2.1 shows a significant improvement from 18% to 23% between 200-400 steps, followed by a small decrease to 22% at 800 steps. Meanwhile, SDXL achieves a more stable progress, gradually increasing from 23% at 200 steps to 24% at 800. These findings suggest that, while additional tuning can improve performance, there appears to be an optimal range of approximately 400 steps. Based on complementary evaluations of another 50-sample validation set generated by our pipeline, we discovered that 400 tuning steps provide the best balance of semantic accuracy and overall image quality. To ensure consistency, all subsequent experiments reported in this paper include 400 tuning steps.

Overall, our findings indicate that increasing the number of tuning steps improves object state representation, particularly on the Object-State-Bench, whereas the GenAI-Object-State results show a more nuanced relationship that necessitates model-specific tuning for optimal performance.

Model	Object-State-Bench			GenAI-Object-State		
	200 Steps	400 Steps	800 Steps	200 Steps	400 Steps	800 Steps
Stable Diffusion 1.5	67%	69%	70%	20%	21%	18%
Stable Diffusion 2.1	68%	68%	68%	18%	23%	22%
SDXL	65%	66%	70%	23%	23%	24%

Table 8. GPT evaluation scores (in percentage) for different tuning steps on GenAI-Object-State and Object-State-Bench.

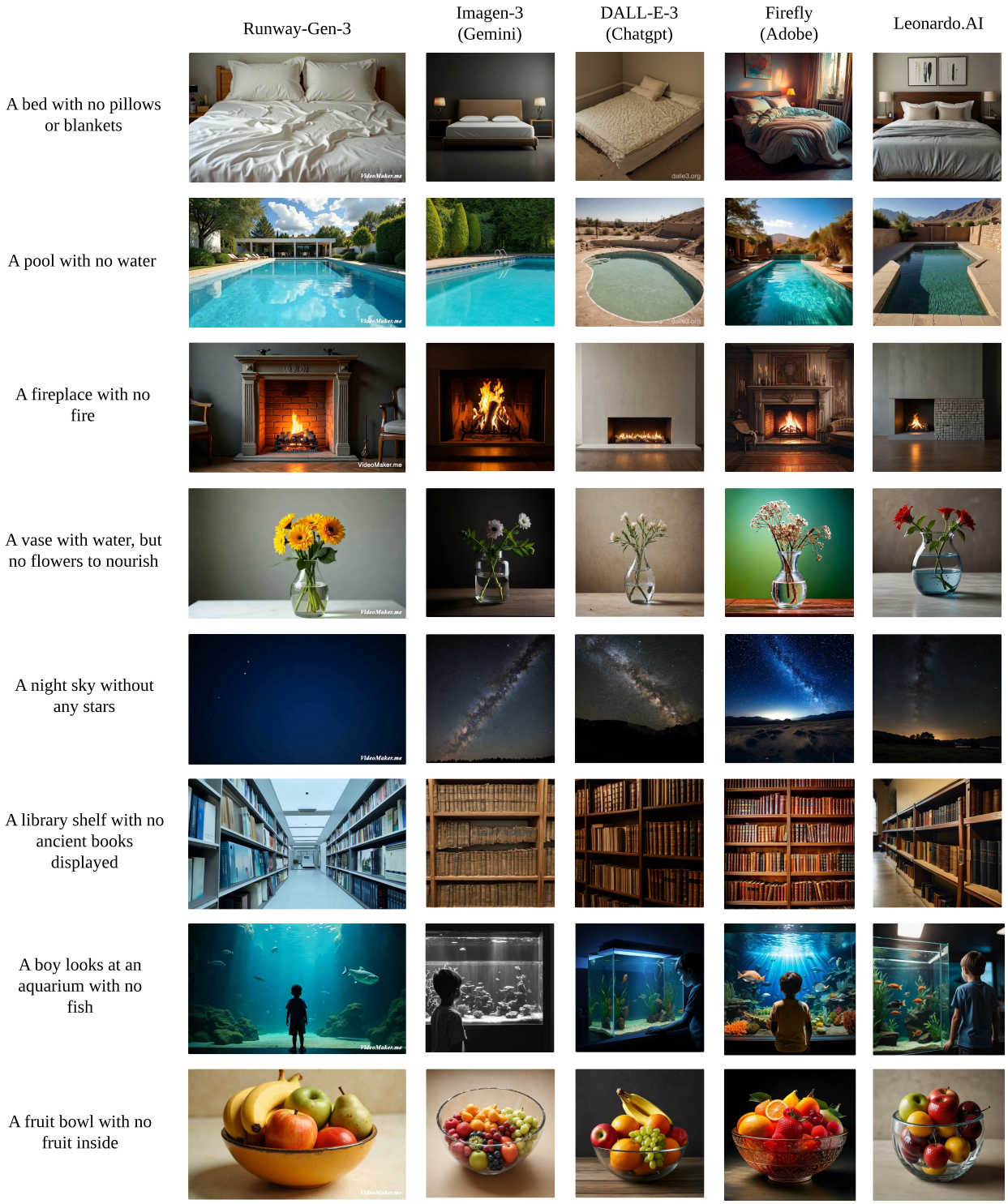


Figure 15. Additional qualitative examples of object state failures in advanced text-to-image generation models. For example, the figure shows a bed without pillows or blankets, a fireplace with no fire, a pool with no water, a vase filled with water but missing flowers, a night sky devoid of stars, a library shelf lacking ancient books, an aquarium without fish, and a fruit bowl with no fruit. These deficiencies are observed in outputs from models such as Runway-Gen-3, Imagen-3 (Gemini), DALL-E-3 (ChatGPT), Firefly (Adobe), and Leonardo.AI.