A. Appendix / supplemental material

In this supplementary material, we first clarify the notations used in this paper and then revisit the proposed CompGS in Algorithms 1. The training details of CompGS will also be provided. Besides, we provide more numerical and visual evaluations to further validate the effectiveness of our model. We have provided **a demo video** in the attachment to display more visual comparisons between CompGS and other methods. We will make code public.

A.1. Notations

We compile a comprehensive list of all the notations utilized in this paper, as shown in Table 3.

Table 3. Notations.

Notation	Description
L	Total number of entities
V	Complex prompt (e.g., 'an owl perches on a branch near a pinecone')
I	Composed image generated by the 2D diffusion model
v_l	Entity-level prompt for entity l, $(l \in L)$
I_l	Segmented image containing entity l, $(l \in L)$
m_l	Rough triangle mesh of the 3D entity l, $(l \in L)$
θ_l	3D Gaussians for the entity l, $(l \in L)$
θ	Composed 3D Gaussians l
N	Number of points indexed from each mesh
μ_i^l	Center positions of each vertex of mesh m_l in $\mathbb{R}^3$
c_i^l	Texture colors queried from each vertex of mesh m_l in $\mathbb{R}^3$
bbox_l	3D bounding box for entity l, used for optimization
bbox_{std}	Standardized volumetric space for scaling
μ	Center positions of each vertex in the original 3D space
$\hat{\mu}$	Transformed center positions of entity Gaussian after scaling
β	Shift parameters for the center positions of the bounding box
λ	Scale parameters for standardizing the volumetric space
x	Rendered image from 3D Gaussians
$g(\cdot)$	Gaussian Splatting rendering function
β	the shift parameters for volume-adaptive optimization
λ	the scale parameters for volume-adaptive optimization
$\text{Mean}(\cdot)$	the operator computing the center coordinates of the given bounding box
$\hat{\theta}$	New Gaussians initialized from the edited 2D image

A.2. Algorithm

We provide pseudocode in Algorithm 1. Two core designs, including 3D Gaussian initialization with 2D compositionality and dynamic SDS optimization, are detailed.

A.3. Additional Training Details

CompGS is implemented in ThreeStudio [24]. We use DALL·E 3 [3], LangSAM [46] and TripoSR [67] to implement the text-to-image, text-guided segmentation, and image-to-mesh, respectively. For entity-level optimization, we adopt MVDream [58] as the 3D diffusion prior; while for composition-level optimization, we employ *stabilityai/stablediffusion-2-1-base* [57] as the 2D diffusion prior. We set all the diffusion guidance as 50. For all Gaussian parameters, we linearly decreased the learning rate for position μ from 10^{-3} to 10^{-5}, for scale from 10^{-2} to 10^{-3}, and for color c from 10^{-2} to 10^{-3}, respectively. Besides, we fixed the learning rate for opacity a to be 0.05, and for rotation to be 0.001. Additionally, we use a consistent batch size of 4 for both training and test, and a rendered resolution fixed at 1024×1024. Camera settings during training are set with distances ranging from 0.8 to 1.0 relative units, a field of view between 15 and 60 degrees, and elevation ranging up to 30 degrees. Additionally, there are no perturbations applied to camera position, center, or orientation, maintaining a controlled imaging environment. For test, we set the resolution of rendered image as 1024×1024 with specific camera distance and field of view for validation set to 3.5 units and 40 degrees, respectively. For each prompt, we train the model on an NVIDIA A100 GPU (40G) for 10,000 iterations, which takes approximately 70 minutes. We observed that training the model for 5,000 iterations already produces high-quality content with minimal loss of texture details. This indicates that the training duration can be shortened to around **30 minutes**. However, to achieve high-quality 3D textures, we use 10,000 iterations for training in this paper, unless otherwise specified.

Algorithm 1 CompGS: 3D Gaussian Initialization and Dynamic SDS Optimization $V, \{v_l\}(l \in L)$: Input prompt and entity-level prompts.
$\{m_l\}(l \in L)$: Entity-level meshes.
$\theta, \{\theta_l\}(l \in L)$: Composition-level Gaussian parameters and entity-level Gaussian parameters.
bbox_{std}: Standardized volumetric space.
L: The number of entities.
N: The number of Gaussian parameters.
T2I: Text-to-Image models.
TGS: Text-guided segmentation models.
I2M: Image-to-Mesh models.
$\mathrm{Zoom}^{\uparrow}, \mathrm{Zoom}^{\downarrow}$: Zoom-in and Zoom-back operators in Eq. 4.
η: Learning rate.
T: Total training iterations.

```
Stage 1: Initializing 3D Gaussians with 2D Compositionality.
$I = \mathrm{T2I}(V)$    ▷ Generate well-composed Image from the given prompt
$\{v_l\} = \mathrm{LLM}(V)$    ▷ Obtain entity-level prompts via LLM
$\{m_l\} = \mathrm{I2M}(\mathrm{TGS}(\{v_l\}, I))$    ▷ Obtain entity-level meshes
$\mu_i(i \in N), c_i(i \in N) \leftarrow m_l(l \in L)$    ▷ Positions and colors of the 3D Gaussians.
$D \leftarrow \mu_i(i \in N)$    ▷ Distance between the nearest two positions.
$\Sigma_i(i \in N), \alpha_i(i \in N) \leftarrow D, 0.1$    ▷ Covariance and opacity of the 3D Gaussians.
$\mathrm{bbox}_l(l \in L) \leftarrow \mu_i(i \in N)$    ▷ Boundary of bounding box

Stage 2: Dynamic SDS Optimization.
for $t = 1$ to $T$ do
    $l \leftarrow \mathrm{randint}(1, L)$    ▷ Randomly select an integer $l$ from the range 1 to $L$
    if $i = 0$ then
        $\nabla_\theta \mathcal{L}^{2d}_{\mathrm{SDS}}(\phi, \mathrm{x} = g(\theta)) \triangleq \mathbb{E}_{t,\epsilon}\left[w(t)\left(\hat{\epsilon}_\phi(\mathbf{z}_t, V, t) - \epsilon\right)\frac{\partial \mathrm{x}}{\partial \theta}\right]$
            ▷ Obtain the gradients via SDS loss with 2D priors
        $\nabla_\theta \mathcal{L}^{3d}_{\mathrm{SDS}}(\phi, \mathrm{x} = g(\theta)) \triangleq \mathbb{E}_{t,\epsilon}\left[w(t)\left(\hat{\epsilon}_\phi(\mathbf{z}_t, v, t) - \epsilon\right)\frac{\partial \mathrm{x}}{\partial \theta}\right]$
            ▷ Obtain the gradients via SDS loss with 3D priors
        $\theta \leftarrow \theta - \eta(\nabla_\theta \mathcal{L}^{2d}_{\mathrm{SDS}} + \nabla_\theta \mathcal{L}^{3d}_{\mathrm{SDS}})$
            ▷ Update the compositional Gaussian parameters via back-propagation
    else
        $\hat{\theta}_l \leftarrow \mathrm{Zoom}^{\uparrow}(\theta_l, \mathrm{bbox}_l, \mathrm{bbox}_{std})$
            ▷ Dynamically zoom-in Gaussian parameters from $\mathrm{bbox}_l$ to a standardized space $\mathrm{bbox}_{std}$
        $\nabla_{\hat{\theta}_l} \mathcal{L}^{3d}_{\mathrm{SDS}}(\phi, \mathrm{x} = g(\hat{\theta}_l)) \triangleq \mathbb{E}_{t,\epsilon}\left[w(t)\left(\hat{\epsilon}_\phi(\mathbf{z}_t, v_l, t) - \epsilon\right)\frac{\partial \mathrm{x}}{\partial \hat{\theta}_l}\right]$
            ▷ Obtain the gradients via SDS loss with 3D priors
        $\hat{\theta}_l \leftarrow \hat{\theta}_l - \eta \nabla_{\hat{\theta}_l} \mathcal{L}^{3d}_{\mathrm{SDS}}$
            ▷ Update the compositional Gaussian parameters via back-propagation
        $\theta_l \leftarrow \mathrm{Zoom}^{\downarrow}(\hat{\theta}_l, \mathrm{bbox}_l, \mathrm{bbox}_{std})$
            ▷ Dynamically zoom-back Gaussian parameters from the standardized space $\mathrm{bbox}_{std}$ to $\mathrm{bbox}_l$

end for
```

A.4. Running Time Comparisons

We show runtime comparisons with other models in Tab. 4. Results in Tab. 4 indicate that the runtime is generally relative to the number of objects involved. Compared to open-source compositional 3D methods such as Set-the-Scene [15], Progressive3D [14], and GraphDraemer [23], our proposed CompGS is more efficient in training. For instance, given the prompt *"a parrot talks beside a perch and two bowls,"* Progressive3D takes approximately 250 minutes for 3D generation, while Set-the-Scene requires around 110 minutes. Since many compositional scene generation methods are not open-sourced, we also present the training steps listed in the papers for straightforward comparisons.

'A dripping paintbrush stands poised above a half-finished canvas'

'An intricately-carved wooden chess set'

'An old brass key sits next to an intricate, dust-covered lock'

'A scientist is examining a specimen under a microscope'

'A fisherman is throwing the fishing rod in the sea'

'A chessboard is set up, the king and queen standing in opposition'

'A brown horse in a green pasture'

DreamFusion Magic3D LatentNeRF Fantasia3D SJC ProlificDreamer VP3D CompGS (ours)

Figure 7. **Qualitative comparisons between CompGS and other text-to-3D models on T^{3}Bench (multiple objects track).** CompGS is better at generating highly-composed, high-quality 3D contents that strictly align with the given texts. Watch the animations by **clicking** them (Not all PDF readers support playing animations. Best viewed in Acrobat/Foxit Reader).

'A castle-shaped sandcastle'

'A cherry red vintage lipstick tube'

'A fluffy, orange cat'

'A fuzzy pink flamingo lawn ornament'

'A hot air balloon in a clear sky'

'A paint-splattered easel'

'A rustic wrought-iron candle holder'

DreamFusion Magic3D LatentNeRF Fantasia3D SJC ProlificDreamer VP3D COMPGS (ours)

Figure 8. **Qualitative comparisons between COMPGS and other text-to-3D models on T^{3}Bench (single object track).** COMPGS is better at generating high-quality 3D assets that strictly align with the given texts. Watch the animations by **clicking** them (Not all PDF readers support playing animations. Best viewed in Acrobat/Foxit Reader).

Table 4. Runtime comparisons on both compositional generation and single object generation show the efficiency of CompGS.

	Compositional Generation				Single Object Generation			
Method	Open-source	3D Representations	Training Steps	Training Time (minutes)	Method	Open-source	3D Representations	Training Steps	Training Time (minutes)
Progressive3D [14]	✓	NeRF	40,000	220	DreamFusion [52]	✓	NeRF	-	360
Set-the-scene [15]	✓	NeRF	15,000	110	Magic3D [41]	✓	NeRF	-	340
CompNeRF [21]	×	NeRF	13,000	-	Fantastic3D [10]	✓	NeRF	-	380
SceneWiz3D [79]	×	NeRF	20,000	420	ProlificDreamer [71]	✓	NeRF	-	520
GraphDreamer [23]	✓	NeRF	20,000	420	GaussianDreamer [77]	✓	3D Gaussians	-	14
CompGS-10k (Ours)	-	3D Gaussians	10,000	70	CompGS-10k (Ours)	-	3D Gaussians	10,000	70
CompGS-5k (Ours)	-	3D Gaussians	5,000	30	CompGS-5k (Ours)	-	3D Gaussians	5,000	30

Table 5. **Quantitative comparisons with baselines on T^{3}Bench [26] (all three tracks)**. CompGS is compared with feed-forward models, optimization-based models, and models specifically designed for compositional generation.

Method	*Single Object*			*Single Object with Surroundings*			*Multiple Objects*		
	Quality	Alignment	Average	Quality	Alignment	Average	Quality	Alignment	Average
LRM [29]	29.4	38.2	33.8	20.3	35.1	27.7	15.2	25.5	20.4
TripoSR [67]	34.3	38.9	36.6	21.8	37.2	29.5	16.7	28.6	22.7
DreamFusion [52]	24.9	24.0	24.4	19.3	29.8	24.6	17.3	14.8	16.1
SJC [68]	26.3	23.0	24.7	17.3	22.3	19.8	17.7	5.8	11.7
LatentNeRF [47]	34.2	32.0	33.1	23.7	37.5	30.6	21.7	19.5	20.6
Fantasia3D [10]	29.2	23.5	26.4	21.9	32.0	27.0	22.7	14.3	18.5
ProlificDreamer [71]	51.1	47.8	49.4	42.5	47.0	44.8	45.7	25.8	35.8
Magic3D [41]	38.7	35.3	37.0	29.8	41.0	35.4	26.6	24.8	25.7
Set-the-Scene [15]	32.9	31.9	32.4	30.2	45.8	35.5	20.8	29.9	25.4
VP3D [11]	54.8	52.2	53.5	45.4	50.8	48.1	49.1	31.5	40.3
CompGS	55.1	52.5	53.8	43.2	46.8	45.0	54.2	37.9	46.1

A.5. Extended Experiments on Qualitative Comparisons

Qualitative Model Comparisons on Multi-objects Generation Fig. 7 showcases additional 3D assets produced by CompGS. The prompts are selected from T^{3}Bench (multiple objects track). Compared to previous methods, CompGS not only generates multiple objects but also produces more plausible interactions while maintaining 3D consistency among the objects. For example, in the first row, previous methods such as DreamFusion, Magic3D, LatentNeRF, Fantasia3D, SJC, and ProlificDreamer all fail to generate the canvas described in the given prompt. Although both VP3D and CompGS can generate the two entities (paintbrush and canvas), VP3D fails to maintain 3D consistency, as the back view of the canvas is not visually plausible. In this case, CompGS successfully captures both the key entities described in the prompt and generates reasonable spatial relationships and interactions between the two objects. This phenomenon can also be observed in other cases, such as the key and lock in the third row, and the fisherman in the sea in the fifth row, and so on. Besides the issue of 3D consistency, we found that CompGS performs better in texture alignment. For example, in the second-to-last row, other methods failed to display the combination of chessboard, king, and queen. Specifically, VP3D did not recognize the king and queen as chess pieces. In contrast, CompGS generates these entity details more accurately. Overall, the comparisons in both visual quality and textural alignment with previous methods demonstrate the effectiveness of the proposed CompGS.

Qualitative Model Comparisons on Single-object Generation Though CompGS is specifically designed for compositional generation, it can naturally handle single-object generation as well. We present the qualitative comparisons between CompGS and previous works in Fig. 8. It is observed that CompGS performs better in maintaining multi-view consistency and generating fine-grained details of the object. For example, in the last row of Fig. 8, CompGS is capable of generating a 3D consistent candle holder, including detailed copper textures. In contrast, other methods either fail to produce the corresponding shape [10], only generate rough outlines without detailed textures [41, 47, 52, 68], or produce 3D patterns with discontinuities [11, 71].

Qualitative Model Comparisons with Scene-generation Methods We also compare CompGS with closed-source models [15, 81] that generate 3D scenes. Figures were selected from [81] and are presented in Fig. 9. The results indicate that CompGS excels in generating high-fidelity texture details and complex interactions. In the second row of Fig. 9, CompGS produces more detailed textures for table legs and rabbit fur. Regarding interaction generation, Set-the-Scene [15] fails to create complex spatial relationships, as shown with the dog and the Great Pyramid in the first row. Although GALA3D can generate reasonable spatial relationships, it fails to incorporate mutual interactions between objects. This is because it performs compositional generation by optimizing the layout of each object individually, neglecting other inter-interactions such as the rabbit's mouth on the cake and the dog's paw on the plate. In contrast, CompGS generates higher-fidelity textures (e.g., the table body, rabbit fur) and more realistic interactions among objects (e.g., the dog's paw hanging off the plate rather than just resting on top).

Qualitative Model Comparisons with Other Compositional Generation Methods In the main paper, we have compared CompGS with both open-sourced compositional 3D generation baselines (Set-the-scene and VP3D) in Table 1, and close-sourced baselines (GALA3D) in Figure 9. Results show that the 3D assets generated by CompGS are not only high-quality

'A puppy lying on the iron plate on the top of Great Pyramid'

'a rabbit is eating a birthday cake at the dining table'

'Panda in a wizard hat sitting on a Victorian-style wooden chair and looking at a Ficus in a pot'

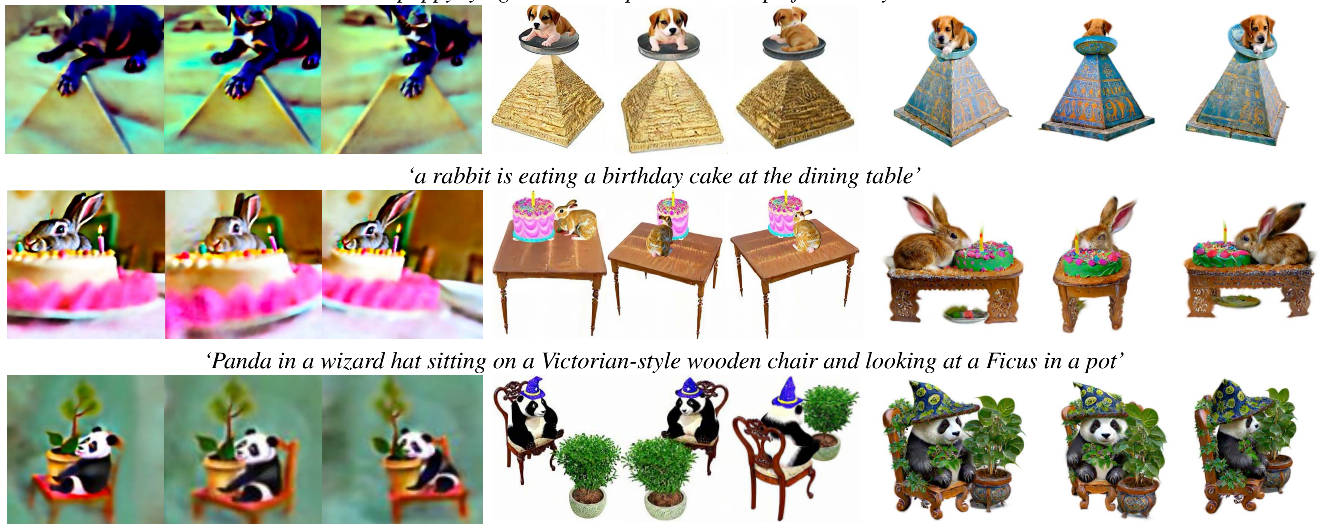

Figure 9. **Qualitative Comparisons Between COMPGS and 3D Scene Generation Methods.** We selected the figures from [81] for these comparisons due to the unavailability of the code. COMPGS performs better in generating object textures and complex interactions.

in appearance, but also align with the given prompts more strictly. We have included qualitative comparisons with both GraphDreamer [23] and DreamGaussian [65] in Fig. 10. Results show that COMPGS demonstrates superior performance on both generation quality and text-3d alignment.

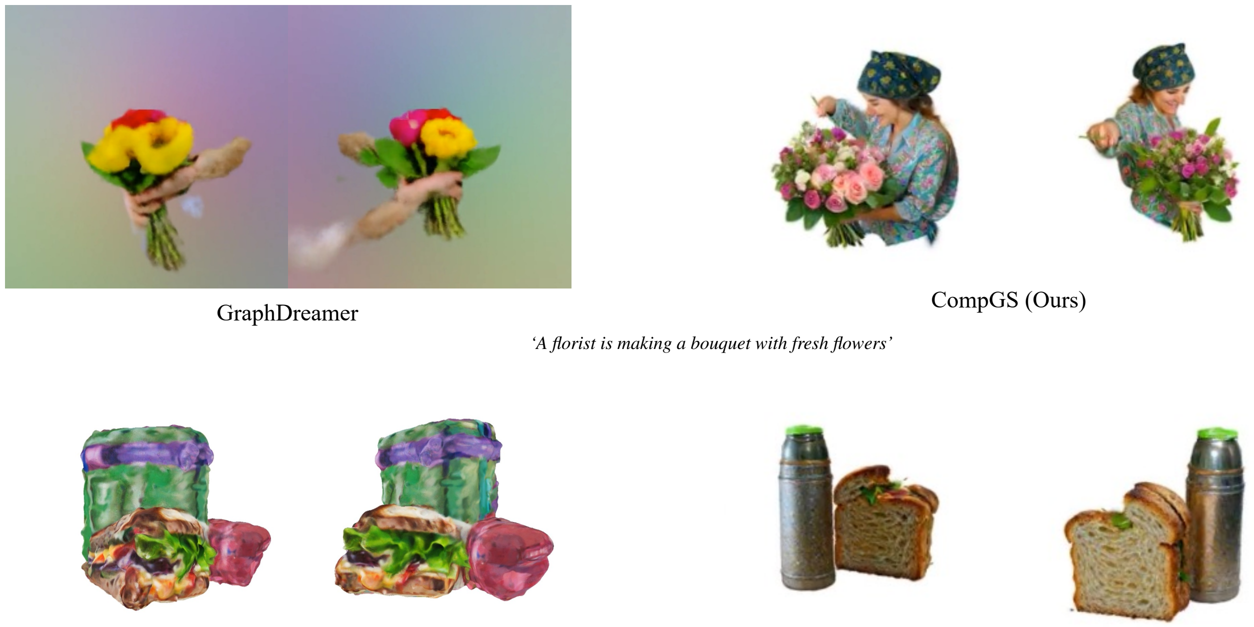
GraphDreamer
CompGS (Ours)
'A florist is making a bouquet with fresh flowers'
DreamGaussian
CompGS (Ours)
'A half-eaten sandwich sits next to a lukewarm thermos'

Figure 10. Extended comparisons with GraphDreamer [23] and DreamGaussian [65].

A.6. Quantitative Model Comparisons

Tab. 5 presents the complete quantitative comparisons on all three tracks of T^{3}Bench. The results indicate that CompGS achieved state-of-the-art performance in compositional generation and slightly outperformed competitors in the single object track. For instance, in the multiple object track, our model surpassed the second-best work [11] by 5.1 in quality and 6.4 in texture alignment. In the single object track, our model also slightly outperformed the second-best work [11] by 0.3 in both quality and alignment.

However, it is worth noting that our model did not achieve state-of-the-art performance in generating single objects with surroundings. This is attributed to the text-guided segmentation model we use, which does not effectively segment the background (e.g., ground, sky, etc.). We have explained this in Sec. 5 and leave it for future improvement. Despite a slight decline in our texture alignment metric in this track, our model still performed significantly better than other methods [10, 15, 29, 41, 47, 52, 67, 68, 71], except for VP3D.

A.7. Examples in User Study

We provide examples of images and scenes used in our user study. In particular, we present concatenated rendering videos and ask participants to rank the eight methods shown in the video based on the overall quality of the 3D objects and the alignment between the text and the 3D models. We average the rank number as its ranking score for comparisons in Tab. 1.

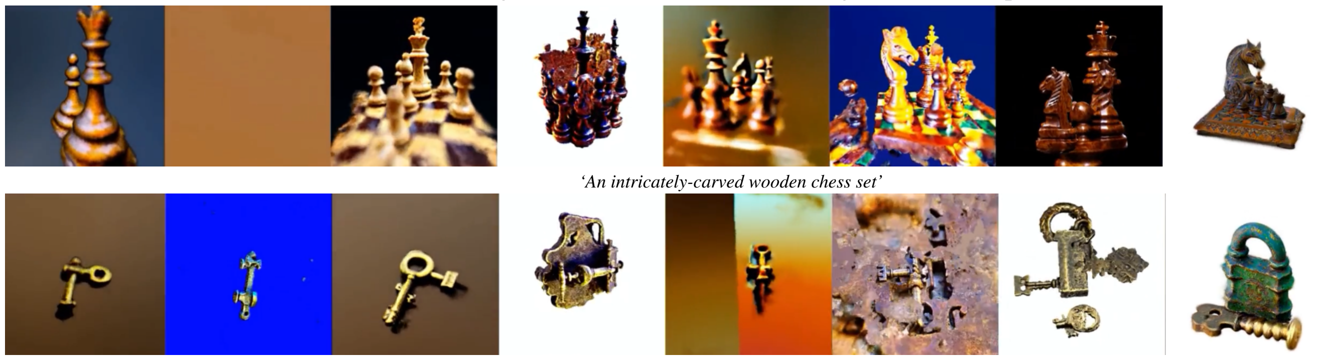
'An intricately-carved wooden chess set'
'An old brass key sits next to an intricate, dust-covered lock'

Figure 11. Examples used in our user study.

A.8. Robustness

We empirically found that CompGS demonstrates the ability to address certain deficits caused by off-the-shelf model priors (e.g., T2I and segmentation priors). Here are some illustrative examples: (1) If certain parts of the target objects are not correctly segmented, CompGS can complete the unsegmented part with correct 3D information. This is demonstrated in Fig. 12(left), where the swing has not been segmented but has been generated by CompGS correctly. This is facilitated through the Entity-level Optimization procedure proposed in the DO strategy. (2) If the T2I models fail to generate proper intra-object interactions, CompGS can correct the multi-object interactions. This is shown in Fig. 12(right), where the spatial relationships in the given image are incorrect and then corrected in the text-to-3D process. This is achieved by the Composition-level Optimization in the proposed DO strategy.

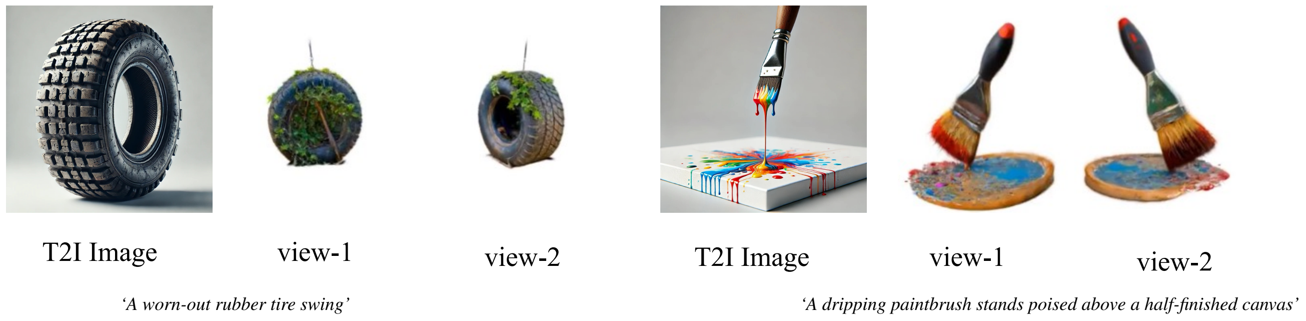
T2I Image
view-1
view-2
T2I Image
view-1
view-2
'A worn-out rubber tire swing'
'A dripping paintbrush stands poised above a half-finished canvas'

Figure 12. CompGS demonstrates the ability to address certain deficits caused by off-the-shelf priors.

A.9. Failure Cases

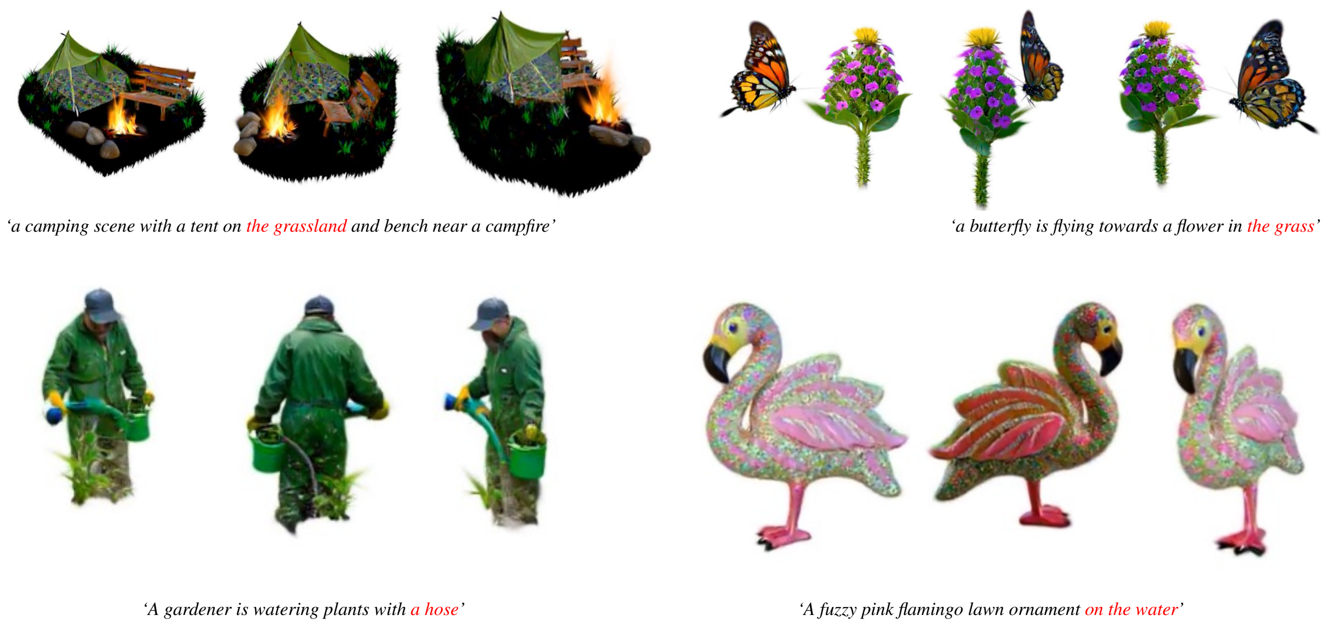
‘a camping scene with a tent on the grassland and bench near a campfire’
‘a butterfly is flying towards a flower in the grass’
‘A gardener is watering plants with a hose’
‘A fuzzy pink flamingo lawn ornament on the water’

Figure 13. **Failure Cases of CompGS in background generation** When text-guided segmentation mode fails to segment the backgrounds, CompGS may generate background with poor visual quality or fails to generate background.

As discussed in Sec. 5, CompGS exhibits limitations in generating backgrounds, such as ground and sky. This is likely due to the current text-guided segmentation model’s inability to effectively segment these abstract concepts. When the background is not well-segmented, we lose the corresponding 2D compositionality needed for initializing 3D Gaussians. This leads to two failure cases: (1) the absence of background in the compositional 3D scenes, as seen with the missing grass in the second column of Fig. 13, or (2) background generation of poor visual quality, such as the vague and unclear depiction of grass in the first column of Fig. 13. It’s crucial to note that such limitations, whilst exist, are not the focus of this work. These shortcomings can be overcome by enhancing the capabilities of off-the-shelf models, effectively mitigating the manifested issues.

A.10. Progressive 3D Adding Examples

CompGS offers a user-friendly approach to progressively conduct 3D editing for compositional 3D generation. More visual examples are presented in Fig. 14. For instance, given a compositional prompt such as *‘A puppy lying on the iron plate on the top of the Great Pyramid, with a pharaoh nearby’*, we divide the generation process into four stages. Initially, we generate *‘the Great Pyramid’* on the left, then progressively add *‘the plate’*, *‘the puppy’*, and *‘the pharaoh’* to complete the 3D scene. Notably, both the interactions and texture details can be well-produced during the editing pipeline of CompGS.

A.11. LLM Prompts

The used prompt for LLM to decomposed the given sentence for 3D generation are provided in Tab. 6

Please analyze the following sentence and break it down into distinct entities by identifying meaningful, standalone components. Each entity should capture a uniqu
entity; instead, isolate each component to preserve its individual meaning. For instance, in a sentence like “An owl perches on a branch near a pinecone,” you woul

1. 1. “an owl”
2. 2. “a branch”
3. 3. “a pinecone”

Follow this approach to segment the sentence accurately, ensuring that each extracted entity reflects a clear and self-contained element of the original context. Plea

Table 6. LLM Prompt for decomposing

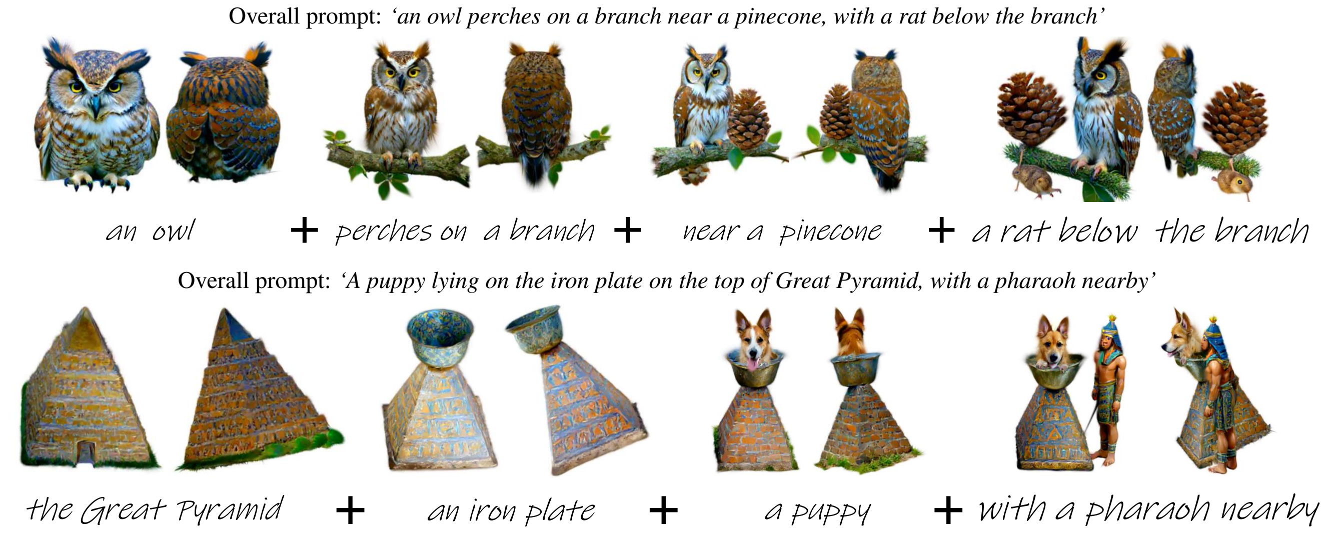
Overall prompt: *'an owl perches on a branch near a pinecone, with a rat below the branch'*
an owl
+ perches on a branch
+ near a pinecone
+ a rat below the branch
Overall prompt: *'A puppy lying on the iron plate on the top of Great Pyramid, with a pharaoh nearby'*
the Great Pyramid
+ an iron plate
+ a puppy
+ with a pharaoh nearby

Figure 14. **More examples of progressive adding 3D objects.** COMPGS provides a user-friendly way to progressively edit on 3D scenes for compositional generation.

A.12. Impacts and Limitations

As a text-to-3D model, we do not foresee obvious undesirable ethical/social impacts. For limitations, we notice three and leave them for future explorations.

- **Background.** One limitation is that when the given prompt includes backgrounds (e.g., ground, sky), COMPGS may fail to generate these elements adequately. This is due to the current text-guided segmentation model's inability to segment such abstract concepts.
- **Complex scaling.** The other is the number scaling. Regarding the limitation on the number of objects supported, we empirically found that the effectiveness of CompGS depends on the capabilities of the text-to-image model being used. If the text-to-image model can accurately generate multiple objects based on a given text input, CompGS can indeed extend to the corresponding number of objects. In other words, the limitation of CompGS in generating multiple objects primarily stems from the compositional 2D generation ability of text-to-image models. Although this is an important research topic and has been widely explored (e.g., [30]), it is not the focus of this work. Besides, it is also important to note that methods [10, 41] based on SDS for text-to-compositional 3D generation often face this limitation.
- **Potential semantic leakage.** We observe potential semantic leakage, which typically arises from significant overlap between the two entities. A possible solution could involve replacing the 3D bounding box prior with more fine-grained spatial priors, such as 3D masks.

A.13. Future Work

We leave several explorations for future works due to limited resources:

- **Adopting VSD** Theoretically, VSD [4] can indeed enhance 3D multi-view consistency and improve visual appearance quality. However, in our implementation, we empirically found that VSD consumes significantly more GPU memory. For example, training CompGS with SDS on the A100 (40G) GPU typically takes around 30-36 GB of GPU memory, while training CompGS with VSD causes out-of-memory (OOM) errors as the number of 3D Gaussians increases.
- **Additional optimization on rotation.** In Equation (4), we focused our optimization on translation and scaling parameters only. An alternative approach could be to include additional rotation parameters in the optimization. However, we found empirically that the prior knowledge embedded in current text-to-image models already offers strong compositional capabilities. As a result, incorporating rotations did not yield substantial improvements and, in some cases, added unnecessary complexity to the optimization process. Therefore, the inherent compositionality provided by the models appears sufficient for achieving robust alignment without requiring additional transformations. Nevertheless, we leave optimizing rotations for future explorations on improving the performance on corner cases.