Supplementary Material for LayoutGPT: Compositional Visual Planning and Generation with Large Language Models

Anonymous Author(s)
Affiliation
Address
email

A Implementation Details

In this section, we provide a detailed description of our prompt construction and instantiate instructions examples.

Task instructions As is shown in Table 1, the specific task instructions start with verbalized descriptions of the task and are followed by the formal definition of the CSS style. As for the indoor scene synthesis, we additionally provide a list of available furniture and the normalized frequency distribution for fair comparisons with the supervised method. Yet we discover that the provided frequency distribution has little effect on the generation results, based on the trivial change in the KL divergence. In some cases, it is important to make LLMs sample from a defined distribution instead of learning the distribution from in-context exemplars, which we leave for future work.

Table 1: The prepending instructions provided to GPT-3.5/4 during our LayoutGPT's 2D and 3D layout planning process. The instructions listed here are for the setting with CSS structure and with normalization.

Task	Instruction for GPT-3.5/4
2D Layout Planning	Instruction: Given a sentence prompt that will be used to generate an image, plan the layout of the image. The generated layout should follow the CSS style, where each line starts with the object description and is followed by its absolute position. Formally, each line should be like "object {width: ?px; height: ?px; left: ?px; top: ?px; }". The image is 64px wide and 64px high. Therefore, all properties of the positions should not exceed 64px, including the addition of left and width and the addition of top and height.
3D Layout Planning	Instruction: Synthesize the 3D layout of an indoor scene from the bottom-up view. The generated 3D layout should follow the CSS style, where each line starts with the furniture category and is followed by the 3D size, orientation, and absolute position. Formally, each line should follow the template: FURNITURE {length: ?px: width: ?px; height: ?px; left: ?px; top: ?px; depth: ?px; orientation: ?degrees;} All values are in pixels but the orientation angle is in degrees. Available furniture: armchair, bookshelf, cabinet, ceiling_lamp, chair, children_cabinet, coffee_table, desk, double_bed, dressing_chair, dressing_table, floor_lamp, kids_bed, nightstand, pendant_lamp, shelf, single_bed, sofa, stool, table, tv_stand, wardrobe Overall furniture frequencies: (armchair: 0.0045; bookshelf: 0.0076; cabinet: 0.0221; ceiling_lamp: 0.062; chair: 0.024; children_cabinet: 0.0075; coffee_table: 0.0013; desk: 0.0172; double_bed: 0.1682; dressing_chair: 0.0063; dressing_table: 0.0213; floor_lamp: 0.0093; kids_bed: 0.0079; nightstand: 0.2648; pendant_lamp: 0.1258; shelf: 0.0086; single_bed: 0.0211; sofa: 0.0018; stool: 0.012; table: 0.0201; tv_stand: 0.0308; wardrobe: 0.1557)

Submitted to 37th Conference on Neural Information Processing Systems (NeurIPS 2023). Do not distribute.

Base LLMs We use four variants of GPT models, (1) Codex [2] (`code-davinci-002`), an LLM that is fine-tuned with large-scale code datasets and can translate natural language into functioning code snippets; (2) GPT-3.5 [8] (`text-davinci-003`), which is trained to generate text or code from human instructions; (3) GPT-3.5-chat (`gpt-3.5-turbo`) and (4) GPT-4 [7] (`gpt-4`), which are both optimized for conversational tasks. For the last two models, we first feed the in-context exemplars as multiple turns of dialogues between the user and the model to fit into the API design. However, we generally observe that GPT-3.5-chat and GPT-4 are not as strong as GPT-3.5 in learning from the in-context demonstrations, especially when the dialogue format follows a certain structure instead of free-form descriptions.

Hyperparameters For all LLMs, we fix the sampling temperature to 0.7 and apply no penalty to the next token prediction. For image layouts evaluation in main paper Table 2, we fix the number of exemplars to 16 for numerical reasoning, and 8 for spatial reasoning, based on the best results of a preliminary experiment. However, we do not observe significant gaps in evaluation results when using different amounts of exemplars (see Sec. B.4). For each prompt, we generate five different layouts/images using baselines or LayoutGPT and thus result in 3810 images for numerical reasoning and 1415 images for spatial reasoning in all reported evaluation results. As for indoor scene synthesis, we fix the number of exemplars to 8 for bedrooms and 4 for living rooms to reach the maximum allowed input tokens. We set the maximum output token as 512 for bedrooms and 1024 for living rooms as bedrooms have $\sim$5 objects per room while living rooms have $\sim$11 objects per room. We generate one layout for each rectangular floor plan for evaluation.

B LayoutGPT for 2D Layout Planning

B.1 NSR-1K Benchmark Construction

We rely on the MSCOCO annotations to create NSR-1K with ground-truth layout annotations. Note that each image in COCO is paired with a set of captions and a set of bounding box annotations.

Numerical Reasoning We primarily focus on the competence of T2I models to count accurately, i.e., generate the correct number of objects as indicated in the input text prompt. The prompts for this evaluation encompass object counts ranging from 1 to 5. To design the template-based T2I prompts, we initially sample possible object combinations within an image based on the bounding box annotations. We only use the bounding box annotation of an image when there are at most two types of objects within the image. As a result, the template-based prompts consist of three distinct types: (1) *Single Category*, wherein the prompt references only one category of objects in varying numbers; (2) *Two Categories*, wherein the prompt references two categories of distinct objects in varying numbers; and (3) *Comparison*, wherein the prompt references two categories of distinct objects but specifies the number of only one type of object, while the number of the other type is indicated indirectly through comparison terms including "fewer than", "equal number of", and "more than". As for natural prompts, we select COCO captions containing one of the numerical keywords from "one" to "five" and filter out those with bounding box categories that are not mentioned to avoid hallucination.

Spatial Reasoning We challenge LLMs with prompts that describe the positional relations of two or more objects. Our spatial reasoning prompts consist of template-based prompts and natural prompts from COCO. To construct template-based prompts, we first extract images with only two ground-truth bounding boxes that belong to two different categories. Following the definitions from PaintSkill [3], we ensure the spatial relation of the two boxes belong to `(left, right, above, below)`. Specifically, given two objects A, B, their bounding box centers $(x_A, y_A), (x_B, y_B)$ and the Euclidean distance d between two centers, we define their spatial relation $\text{Rel}(A, B)$ as:

$$\text{Rel}(A,B) = \begin{cases} B \text{ above } A & \text{if } \frac{y_B - y_A}{d} \geqslant sin(\pi/4) \\ B \text{ below } A & \text{if } \frac{y_B - y_A}{d} \leqslant sin(-\pi/4) \\ B \text{ on the left of } A & \text{if } \frac{x_B - x_A}{d} < cos(3\pi/4) \\ B \text{ on the right of } A & \text{if } \frac{x_B - x_A}{d} > cos(\pi/4) \end{cases} \quad (1)$$

The definition basically dissects a circle centered at A equally into four sectors that each represent a spatial relation. While the definition may not stand for all camera viewpoints, it allows us to mainly focus on the **front view** of the scene. Then, we utilize the category labels and the pre-defined relations to form a prompt, as is shown in main paper Table 1. As for the natural COCO prompts, we select prompts that contain one of the key phrases (`the left/right of, on top of, under/below`) and ensure that the bounding box annotations align with our definition.

B.2 Evaluation Metrics

We denote the set of n object categories in the ground truth annotation as $\mathcal{C}_{GT} = c_1, c_2, \ldots, c_n$, where $x_{c_1}, x_{c_1}, \ldots, x_{c_n}$ represent the number of objects for each category. Additionally, we denote the set of m object categories mentioned in GPT-3.5/4's layout prediction as $\mathcal{C}_{pred} = c'_1, c'_2, \ldots, c'_m$, where $x'_{c'_1}, x'_{c'_2}, \ldots, x'_{c'_m}$ represent the number of objects for each category accordingly. If a category c_i is not mentioned in $\mathcal{C}_{pred}$, then x'_{c_i} is assigned a value of 0, and vice versa.

Categories	c_i	cat	bed	pillow
Ground Truth	x_{c_i}	2	1	2
Prediction	x'_{c_i}	1	0	3

$$precicion = \frac{\sum min(x_{c_k}, x'_{c_k})}{\sum x'_{c'_k}} = \frac{1+0+2}{1+0+3} = 75\,\%$$

$$recall = \frac{\sum min(x_{c_k}, x'_{c_k})}{\sum x_{c'_k}} = \frac{1+0+2}{2+1+2} = 60\,\%$$

Figure 1: An closeup example of how we compute the layout automatic evaluation metrics for numerical reasoning.

The numerical reasoning ability of GPT-3.5/4 on layout planning is assessed using the following metrics: (1) *precision*: calculated as $\frac{\sum_{k=1}^{n} \min(x_{c_k}, x'_{c_k})}{\sum_{k=1}^{m} x'_{c'_k}}$, is an indication of the percentage of predicted objects that exist in the groundtruth; (2) *recall*: calculated as $\frac{\sum_{k=1}^{n} \min(x_{c_k}, x'_{c_k})}{\sum_{k=1}^{n} x_{c_k}}$, indicates the percentage of ground-truth objects that are covered in the prediction; (3) *accuracy*: In the "comparison" subtask, an accuracy score of 1 is achieved when the predicted relation, whether it is an inequality or equality, between the two objects is accurately determined. For all other numerical subtasks, accuracy equals to 1 if the predicted categories and object numbers precisely match the ground truth. In other cases, the accuracy is 0. Fig. 1 shows an example of how we compute the *precision* and *recall*. The *accuracy* for this single example is 0 since the predicted object distribution does not match the ground truth in every category.

For spatial reasoning, we evaluate spatial accuracy based on the LLM-generated layouts and GLIP-based layouts. We adopt [4] finetuned on COCO to detect involved objects from the generated images and obtain the bounding boxes. For both types of layouts, we categorize the spatial relation based on the above definition and compute the percentage of predicted layouts with the correct spatial relation. For all evaluation benchmarks, we measure the CLIP similarity, which is the cosine similarity between the generated image feature and the corresponding prompt feature.

B.3 GPT-3.5/4 Prompting

In main paper Sec. 4.4, we investigate the impact of three components in the structured prompts: (1) *Instruction*, which examines whether detailed instructions explaining the task setup and the format of the supporting examples are included in the prompt. (2) *Structure*, which evaluates the impact of different formatting settings on the presentation of the bounding box aspects of height, width, top, and left. The "w/ CSS" setting formats the aspects in CSS, while the "w/o CSS" setting presents the four aspects in a sequence separated by a comma. (3) *Normalization*, which investigates the effects of rescaling the bounding box aspects to a specified canvas size and presenting them as integers in pixels in the "w/ Norm." setting, while the "w/o Norm." setting presents the aspects as relative scales to the canvas size in floats that range from (0, 1).

Table 1 shows the detailed prepending instructions LayoutGPT provided to GPT-3.5/4 models during 2D layout planning. Table 2 compares the formats of supporting examples with ablated structures and normalization settings.

Table 2: Closeup of various in-context example formats with ablated CSS structure and normalization for 2D layout planning.

CSS Structure	Normalization	In-context Example Format Demo
		Prompt: a teddy bear to the right of a book Layout: teddy bear: 0.50, 0.71, 0.50, 0.15 book: 0.50, 0.61, 0.00, 0.26
✓		Prompt: a teddy bear to the right of a book Layout: teddy bear {width: 0.50; height: 0.71; left: 0.50; top: 0.15; } book {width: 0.50; height: 0.61; left: 0.00; top: 0.26; }
	✓	Prompt: a teddy bear to the right of a book Layout: teddy bear: 32, 45, 31, 9 book: 31, 38, 0, 16
✓	✓	Prompt: a teddy bear to the right of a book Layout: teddy bear {width: 32px; height: 45px; left: 31px; top: 9px; } book {width: 31px; height: 38px; left: 0px; top: 16px; }

Table 3: The automatic metric scores of LayoutGPT (GPT-3.5) with different in-context sample selection approaches. All values are in percentage (%).

#	Exemplar Selection	# In-Context Exemplars	Numerical Reasoning				Spatial Reasoning	
			Precision↑	Recall↑	Layout Accuracy↑	GLIP Accuracy↑	Layout Accuracy↑	GLIP Accuracy↑
1	Fixed Random	16	64.83	92.71	87.66	47.10	80.14	47.07
2	Retrieval	4	88.93	95.02	76.17	50.20	85.30	51.66
3		8	93.32	95.63	82.68	50.58	82.54	52.86
4		16	94.81	96.49	86.33	51.25	82.40	51.09

B.4 Additional Experiments

Random In-Context Exemplars Empirically, selecting in-context exemplars can be critical for the overall performance of LLMs. Apart from our retrieval-augmented method in main paper Sec. 3, we also experiment with a **fixed random** set of in-context exemplars. Specifically, we randomly sample k examples from the training (support) set D to form a fixed set of in-context demonstrations for all test conditions $\mathcal{C}_j$. Therefore, the fixed random setting results in in-context exemplars that are unrelated to the test condition $\mathcal{C}_j$. The minor gap between lines 1&5 in Table 3 verifies that LayoutGPT is not directly copying from the in-context exemplars in most cases. Fig. 2 further justifies the argument with layout visualization of the most similar in-context exemplars and the LayoutGPT outputs.

Number of In-Context Exemplars We take a closer look at the effects of the number of in-context exemplars in the prompt as shown in Table 3. For counting, we observe that the number of exemplars is positively correlated with the counting accuracy. We conjecture that LLMs learn to make more accurate predictions for challenging prompts (e.g., comparison) by learning from more few-shot exemplars. As the layout accuracy also accounts for results where CSS parsing fails, we observe that the LLMs generate more consistent CSS-style code by learning from more examples. However, we cannot observe a similar trend in spatial reasoning prompts. We conjecture that LLMs only require as few as four demonstrations to learn the differences between the four types of spatial relations. The small optimal number of in-context exemplars implies that LLMs already have 2D spatial knowledge and can map textual descriptions to corresponding coordinate values. Yet it is important to find a proper representation to elicit such knowledge from LLMs as implied in main paper Sec. 4.4.

Performance on Numerical Subtasks Table 4 presents the performance of layout generation in various numerical reasoning subtasks. Regarding template-based prompts, the LayoutGPT demonstrates superior performance in the "Single Category" numerical reasoning task, exhibiting precision, recall, and accuracy values around 86%. However, when it comes to the "Two Category" numerical reasoning task, while precision and recall experience minimal changes, the accuracy drops to 66%.

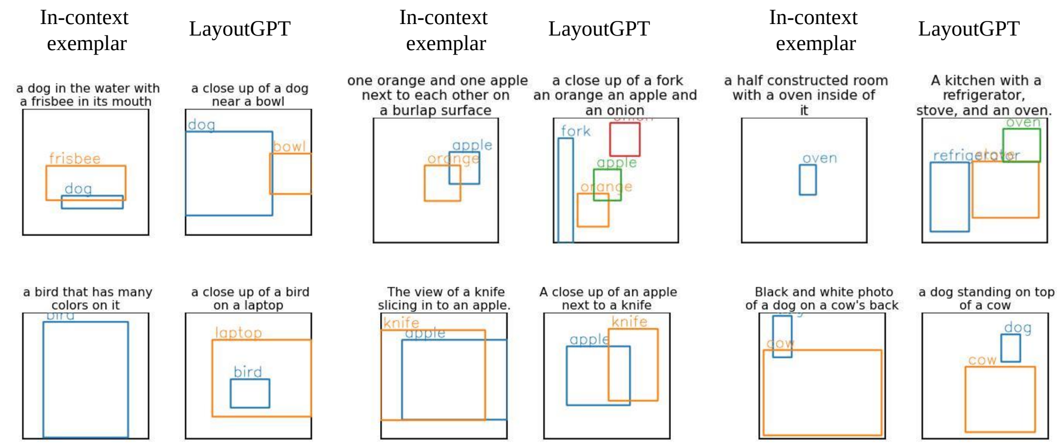

Figure 2: Comparison between the most similar in-context exemplar and the generation results of LayoutGPT.

Table 4: The layout performance on each numerical reasoning subtask. Results reported on LayoutGPT (GPT-4).

Prompt Source	**Subtask**	**Precision**	**Recall**	**Accuracy**
Template	Single Category	85.96	85.96	85.96
	Two Categories	85.14	85.04	66.60
	Comparison	-	-	77.80
Natural Prompts from MSCOCO		72.08	87.1	82.79
-	Total	78.36	86.29	78.43

For the "Comparison" subtask, the accuracy hovers around 78%. These outcomes indicate that LayoutGPT encounters greater challenges when confronted with multi-class planning scenarios, whether the number of objects is explicitly provided or indirectly implied through comparative clauses.

For natural prompts extracted from MSCOCO, a noteworthy observation is the high recall accompanied by relatively lower precision. This discrepancy arises due to the ground truth bounding box annotations encompassing only 80 object classes, whereas the natural prompts may mention objects beyond the annotated classes. Consequently, our LayoutGPT may predict object layouts corresponding to classes not present in the ground truth, which, despite lowering precision, aligns with the desired behavior.

Failure cases Fig. 3 shows typical failure cases in numerical and spatial relations. As previously discussed, we observe in Table 4 that numerical prompts that involves two type of objects ("Two Categories" and "Comparison") are more challenging to LayoutGPT and the image generation model. In these subtasks, LayoutGPT tends to predict much smaller bounding boxes to fit all objects within the limited image space. The small boxes further challenge GLIGEN to fit the object within the limited region, as shown in Fig. 3 (right).

C LayoutGPT for 3D Scene Synthesis

Due to the limitation in datasets, the conditions are room type and room size instead of text descriptions. While ATISS [9] utilizes the floor plan image as the input condition, LLMs are not compatible with image inputs. Therefore, we convert the floor plan image into the specification of the room size. Therefore, the input conditions are similar to "*Room Type: Bedroom, Room Size: max length 256px, max width 256px*".

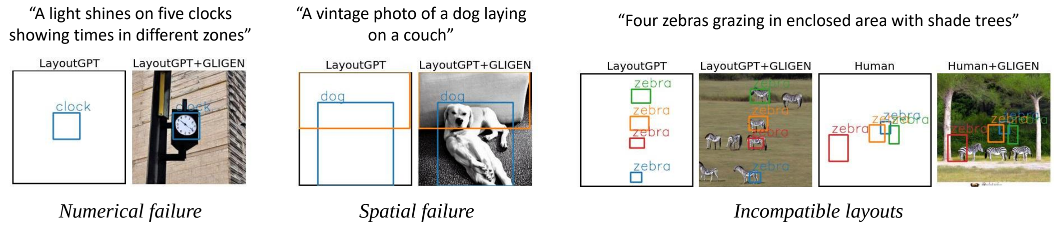

Figure 3: Typical failure cases of LayoutGPT and the generation results using GLIGEN.

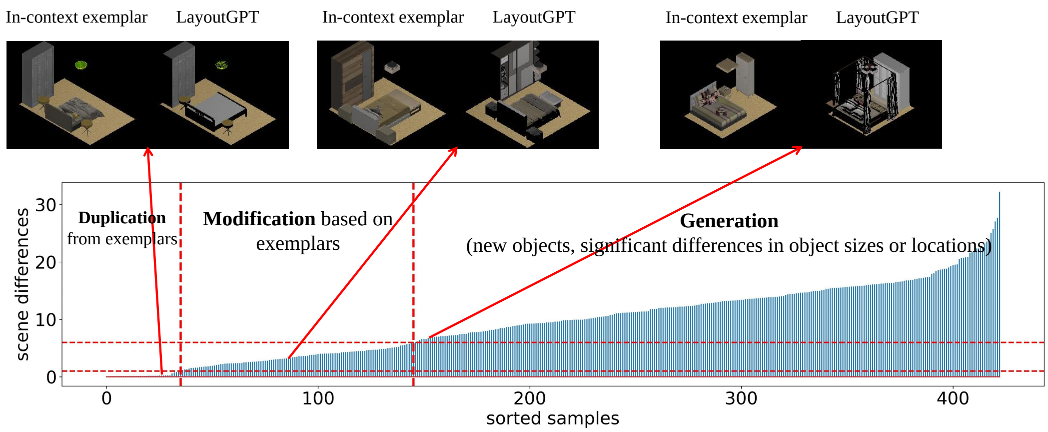

Figure 4: Sorted scene differences between LayoutGPT generated scenes and the most similar in-context exemplars of 423 testing bedroom samples. We partition the distribution into three segments representing different behaviors of LayoutGPT. Duplication: The generated scene is a duplication of the exemplar. Modification: LayoutGPT slightly modifies one exemplar as the generated layout. Generation: LayoutGPT generates novel scenes that are highly different from the exemplars.

C.1 Exemplar Selection

Similar to Sec. B.4, we investigate the effect of using a random set of in-context exemplars for indoor scene synthesis. When we apply 8 random bedroom layouts from the training set as in-context exemplars, the out-of-bound rate increases from 43.26% in main paper Table 4 to 85.58%. The significant differences suggest that LayoutGPT heavily relies on rooms with similar floor plans to maintain objects within the boundary. Yet we verify that the generated layouts from LayoutGPT are not duplicates of the in-context exemplars in most cases.

We first define a training scene layout as a set of objects $S^t = \{\mathbf{o}_1^t, \dots, \mathbf{o}_m^t\}$, and a generated scene layout as $S^g = \{\mathbf{o}_1^g, \dots, \mathbf{o}_n^g\}$. Note that $\mathbf{o}_j$ consists of category $\mathbf{c}_j$, location $\mathbf{t}_j \in \mathbb{R}^3$, size $\mathbf{s}_j \in \mathbb{R}^3$, and orientation $\mathbf{r}_j \in \mathbb{R}$, i.e. $\mathbf{o_j} = (\mathbf{c}_j, \mathbf{t}_j, \mathbf{s}_j, \mathbf{r}_j)$ We define the scene difference $D(\cdot|\cdot)$ between S^t and S^t as

$$D(S^t|S^g) = \sum_{i=1}^{n} \min_{j, \mathbf{c}_j^t = \mathbf{c}_i^g} (\|\mathbf{t}_j^t - \mathbf{t}_i^g\|_1 + \|\mathbf{s}_j^t - \mathbf{s}_i^g\|_1). \tag{2}$$

We set $\mathbf{t}_j^t, \mathbf{s}_j^t$ to $\mathbf{0}$ if S^t does not have a single object that belongs to the same category as $\mathbf{c}_i^g$. For each testing sample of the bedroom, we compute the scene differences between the generated layout and all eight in-context exemplars and use the minimum value as the final scene difference. Note that all parameters used for computation are in “meters” instead of “pixels”.

We plot the scene differences of all 423 testing samples in Fig. 4. We empirically discover that a scene difference below 1.0 means S^g is highly similar to S^t, which we conclude as **duplication** from in-context exemplars. A scene difference below 6.0 shows moderate differences in object sizes or locations between two scenes, representing a **modification** based on S^t to generate S^g. Finally, a scene difference larger than 6.0 represents new objects or significant differences in object sizes or

Figure 5: Typical failure cases of LayoutGPT.

Figure 6: Plausible examples of LayoutGPT(GPT-4) planning keypoints distributions before conducting text-conditioned image generation.

locations between the exemplar and the generated layouts, i.e. true **generation**. Fig. 4 shows that 34/111/278 scenes belong to duplication/modification/generation. Among each category, 30/67/143 scenes have no out-of-bound furniture. Therefore, LayoutGPT is performing generation instead of duplicating in-context exemplars in most cases.

C.2 Failure Cases

While LayoutGPT achieves comparable results as ATISS, LayoutGPT cannot avoid typical failure cases as shown in Fig. 5, such as out-of-bound furniture and overlapped objects. Fig. 5 (right) shows an incorrect placement of nightstands on the same side of the bed while they are commonly placed on each side of the bed headboard. Future work could focus on more sophisticated in-context learning or fine-tuning methods to improve the LLMs' understanding of 3D concepts.

D LayoutGPT for 2D Keypoint Planning

In addition to its application in 2D and 3D layout planning, we investigate the feasibility of leveraging LayoutGPT for 2D keypoint planning to facilitate text-conditioned image generation. In this approach, we utilize LayoutGPT to predict keypoint distributions based on a given text prompt, and subsequently employ GLIGEN [5] for keypoint-to-image generation. The keypoint format used aligns with the specifications outlined in MSCOCO2017 [6], focusing on 17 keypoints that correspond to the human skeleton. Similar to our methodology for selecting supporting examples in the context of 2D layout planning (Section B), we retrieve the k-most similar examples from the training set of MSCOCO2017 and utilize these examples to provide keypoint distributions as input to GPT-3.5/4. Table 5 presents an illustrative example of the input format employed for keypoint planning with GPT-3.5.

Fig. 6 presents several illustrative examples that compare the images generated by conditioning on keypoints planned by our LayoutGPT with those generated by end-to-end models such as StableDiffusion-v2.1 [10] and Attend-and-Excite [1]. In this preliminary demonstration, we observe that LayoutGPT

Table 5: The prompting input provided to GPT-3.5 for LayoutGPT keypoint planning.

Instruction:
Given a sentence prompt that will be used to generate an image, plan skeleton keypoints layout of the mentioned objects. The skeleton keypoints include the following 17 nodes: nose, left_eye, right_eye, left_ear, right_ear, left_shoulder, right_shoulder, left_elbow, right_elbow, left_wrist, right_wrist, left_hip, right_hip, left_knee, right_knee, left_ankle, right_ankle. The generated keypoints layout should follow the CSS style, where each line starts with the keypoint node name and is followed by its absolute position.
Formally, each line should be like "node_name {left: ?px; top: ?px; }". Please follow this format strictly. Do not display in other variation of formats. Notice that some keypoint nodes may not be visible on the canvas. In such cases, simply put "node_name {left: 0px; top: 0px; }" for the invisible nodes. The image is 64px wide and 64px high. Therefore, all properties of the positions should not exceed 64px.

Prompt: a man on a surfboard in a river near a couple of trees and branches
Keypoints:
person#1:
nose {left: 36px; top: 33px; }
left_eye {left: 36px; top: 33px; }
right_eye {left: 36px; top: 33px; }
left_ear {left: 37px; top: 33px; }
right_ear {left: 0px; top: 0px; }
left_shoulder {left: 38px; top: 34px; }
right_shoulder {left: 36px; top: 35px; }
left_elbow {left: 35px; top: 34px; }
right_elbow {left: 35px; top: 38px; }
left_wrist {left: 33px; top: 32px; }
right_wrist {left: 33px; top: 39px; }
left_hip {left: 39px; top: 39px; }
right_hip {left: 37px; top: 40px; }
left_knee {left: 38px; top: 44px; }
right_knee {left: 37px; top: 44px; }
left_ankle {left: 39px; top: 49px; }
right_ankle {left: 37px; top: 48px; }

[MORE SUPPORTING EXAMPLES]

Prompt: a man leaning on a surfboard in the water riding a wave
Keypoints:

exhibits promising potential in offering inherent control over specific movements or actions through keypoint planning.

Nevertheless, it is worth noting that keypoints planning presents considerably greater challenges compared to bounding box layout planning, attributable to several evident factors. Firstly, keypoints planning necessitates the prediction of the positions of 17 nodes, which is significantly more complex than the 2D layout planning involving four aspects or the 3D layout planning encompassing seven aspects. Secondly, the distribution of keypoints encompasses a much larger array of spatial relations due to the numerous possible body movements. In contrast, previous 2D layout planning tasks only involve four types of spatial relations. These inherent complexities render keypoint planning heavily reliant on in-context demonstrations. However, the limited availability of annotations pertaining to body movements in the MSCOCO dataset further exacerbates the challenges associated with reliable keypoint planning. Therefore, we leave the exploration of this potential direction to future research endeavors.

E Ethical Statement

In addition to the layouts predicted by GPT-3.5/4, we also incorporate human-planned layouts as a natural baseline for comparative analysis. To facilitate this, we provide annotators with an interface featuring a blank square space where they can draw bounding boxes. Alongside the input text prompt, we also present the noun words or phrases from the prompt to human annotators, instructing them to draw a bounding box for each corresponding element. We intentionally refrain from imposing additional constraints, enabling annotators to freely exercise their imagination and create layouts based on their understanding of reasonable object arrangements. To compensate annotators for their efforts, we offer a payment rate of \$0.2 US dollars per Human Intelligence Task (HIT). The average completion time of approximately 30 seconds per HIT, which corresponds to an average hourly payment rate of \$24.

F Limitations

The current work has several limitations that provide opportunities for future research. Firstly, while this work focuses on 2D and 3D bounding box layouts and makes a preliminary attempt at keypoints, there exist various other methods for providing additional spatial knowledge in image/scene generation, such as segmentation masks and depth maps. Future work could explore integrating LLMs with these alternative visual control mechanisms to broaden the scope of visual planning capabilities. Secondly, the current work primarily addresses visual generation tasks and lacks a unified framework for handling other visual tasks like classification or understanding. Extending the proposed framework to encompass a wider range of visual tasks would provide a more comprehensive and versatile solution. Thirdly, this work is a downstream application that attempts to distill knowledge from LLMs' extensive knowledge bases. Future research could explore more fundamental approaches that directly enhance the visual planning abilities of various visual generation models. By developing specialized models that are explicitly designed for visual planning, it may be possible to achieve more refined and dedicated visual generation outcomes. Overall, while the current work demonstrates the potential of using LLMs for visual planning, there are avenues for future research to address the aforementioned limitations and further advance the field of visual generation and planning.

G Broader Impact

The utilization of LLMs for conducting visual planning in compositional 2D or 3D generation has significant broader impacts. Firstly, LLMs alleviate the burden on human designers by simplifying the complex design process. This not only enhances productivity but also facilitates scalability, as LLMs can efficiently handle large-scale planning tasks. Secondly, LLMs exhibit remarkable capabilities in achieving fine-grained visual control. By conditioning on textual inputs, LLMs can easily generate precise and detailed instructions for the desired visual layout, allowing for precise composition and arrangement of elements. Moreover, LLMs bring a wealth of commonsense knowledge into the planning process. With access to vast amounts of information, LLMs can incorporate this knowledge to ensure more accurate and contextually coherent visual planning. This integration of commonsense knowledge enhances the fidelity of attribute annotations and contributes to more reliable and realistic visual generation outcomes.

It is worth noting that this work represents an initial foray into the realm of visual planning using LLMs, indicating the potential for further advancements and applications in this area. As research in this field progresses, we can anticipate the development of more sophisticated and specialized visual planning techniques, expanding the scope of LLMs' contribution to diverse domains, such as architecture, virtual reality, and computer-aided design.

H Additional Qualitative Examples

We present additional visual showcases to demonstrate the capabilities of LayoutGPT in different contexts. Fig. 7 showcases examples related to 2D numerical reasoning, Fig. 8 illustrates examples of 2D spatial reasoning, and Fig. 9 displays examples of 3D scene synthesis. These showcases offer further insights into the effectiveness and versatility of our approach across various domains.

References

[1] Hila Chefer, Yuval Alaluf, Yael Vinker, Lior Wolf, and Daniel Cohen-Or. Attend-and-excite: Attention-based semantic guidance for text-to-image diffusion models. *arXiv preprint arXiv:2301.13826*, 2023. 7

[2] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde, Jared Kaplan, Harrison Edwards, Yura Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott

Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, David W. Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William H. Guss, Alex Nichol, Igor Babuschkin, S. Arun Balaji, Shantanu Jain, Andrew Carr, Jan Leike, Joshua Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew M. Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code. *ArXiv*, abs/2107.03374, 2021. 2

[3] Jaemin Cho, Abhay Zala, and Mohit Bansal. Dall-eval: Probing the reasoning skills and social biases of text-to-image generative transformers. *arXiv preprint arXiv:2202.04053*, 2022. 2

[4] Liunian Harold Li, Pengchuan Zhang, Haotian Zhang, Jianwei Yang, Chunyuan Li, Yiwu Zhong, Lijuan Wang, Lu Yuan, Lei Zhang, Jenq-Neng Hwang, et al. Grounded language-image pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10965–10975, 2022. 3

[5] Yuheng Li, Haotian Liu, Qingyang Wu, Fangzhou Mu, Jianwei Yang, Jianfeng Gao, Chunyuan Li, and Yong Jae Lee. Gligen: Open-set grounded text-to-image generation. *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 7

[6] 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

[7] OpenAI. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. 2

[8] Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke E. Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Francis Christiano, Jan Leike, and Ryan J. Lowe. Training language models to follow instructions with human feedback. *ArXiv*, abs/2203.02155, 2022. 2

[9] Despoina Paschalidou, Amlan Kar, Maria Shugrina, Karsten Kreis, Andreas Geiger, and Sanja Fidler. Atiss: Autoregressive transformers for indoor scene synthesis. *Advances in Neural Information Processing Systems*, 34:12013–12026, 2021. 5

[10] Robin Rombach, A. Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10674–10685, 2021. 7

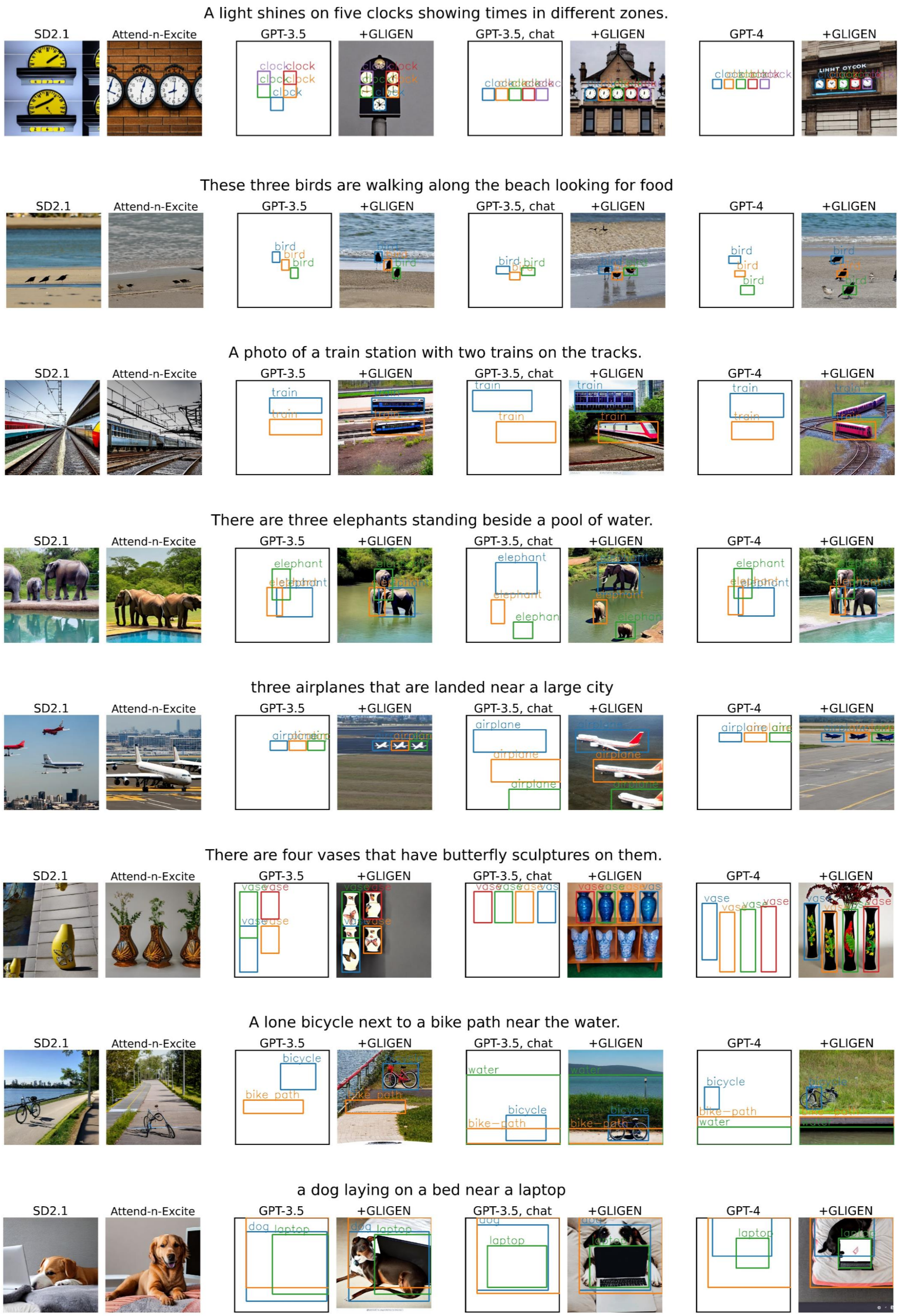

Figure 7: Qualitative examples of variants of LayoutGPT on numerical reasoning prompts.

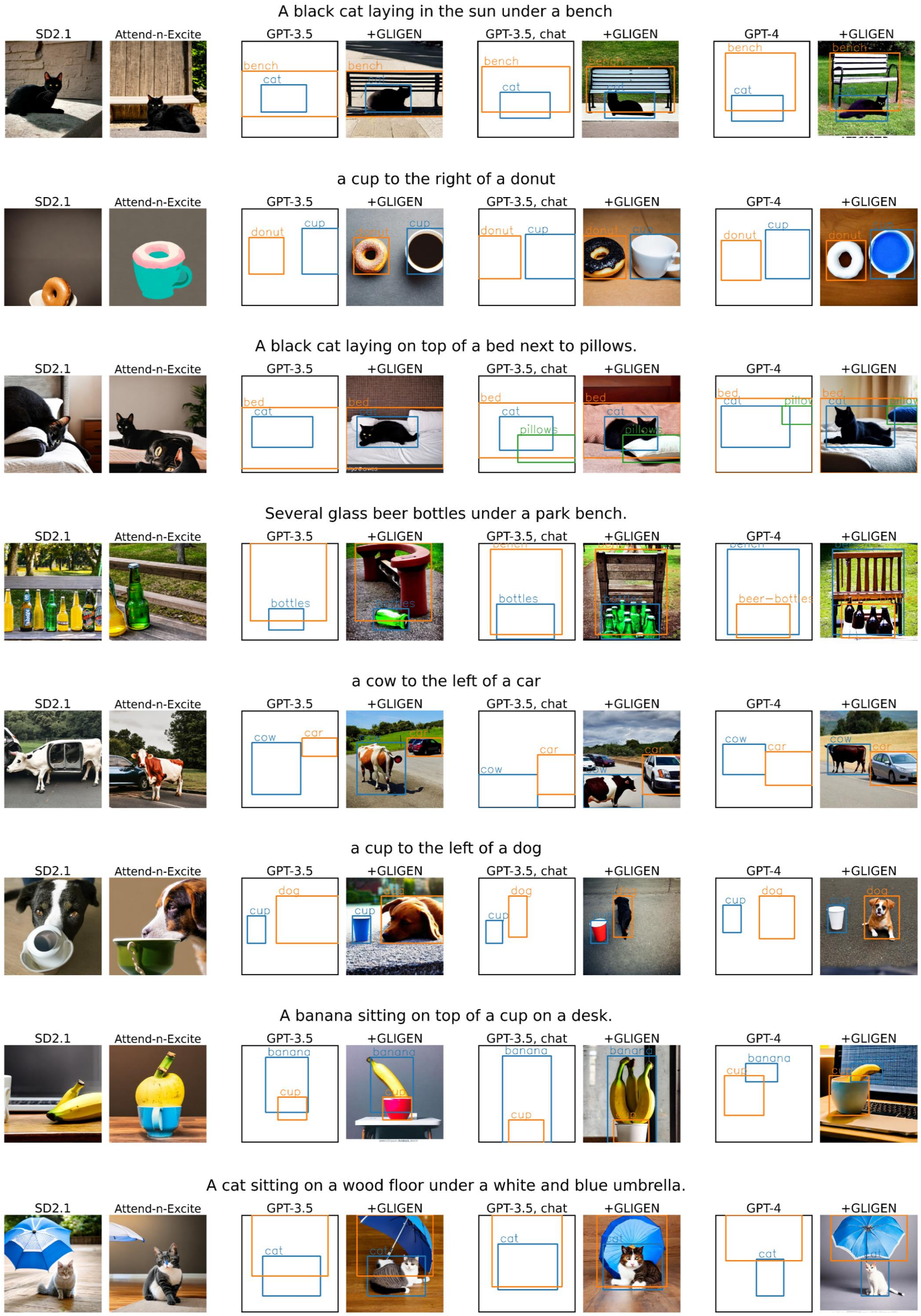

Figure 8: Qualitative examples of variants of LayoutGPT on spatial reasoning prompts.

Figure 9: Additional qualitative examples of variants of LayoutGPT in bedroom scene synthesis.