# Lay-Your-Scene: Natural Scene Layout Generation with Diffusion Transformers

Supplementary Material

## 1. Scaling Factor

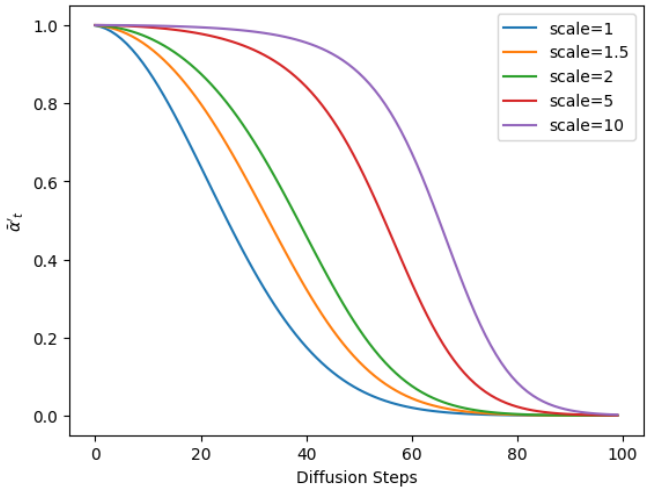



Figure 1. **Effect of scaling factor on denoising Process**. We plot the noise schedule $\bar{\alpha}'_t$ for diffusion process with 100 steps for different scaling factors $s$. We observe that $s > 1$ results in a more gradual destruction of information.

**Theorem 1.** *Given the forward process scaled by a factor $s$ and normalized input distribution*

$$X_t = s\sqrt{\alpha_t}X_0 + \sqrt{1-\alpha_t}\epsilon_t, \quad \epsilon_t \sim \mathcal{N}(0,1), \quad \mathbb{E}[X_0] = 0, \quad Var(X_0) = 1. \tag{1}$$

*the normalized process $\tilde{X}_t$ is given by*

$$\tilde{X}_t = \sqrt{\tilde{\alpha}_t}X_0 + \sqrt{1-\tilde{\alpha}_t}\epsilon_t, \tag{2}$$

*where*

$$\tilde{\alpha}_t = \frac{\sqrt{\alpha_t}s}{\sqrt{(s^2-1)\alpha_t + 1}} \tag{3}$$

*and has the property that $E[\tilde{X}_t] = 0$ and $Var(\tilde{X}_t) = 1$.*

*Proof.* We start with the expression for $X_t$:

$$X_t = s\sqrt{\alpha_t}X_0 + \sqrt{1-\alpha_t}\epsilon_t \tag{4}$$

**Step 1: Expectation of $X_t$**
Taking the expectation of both sides:

$$\mathbb{E}[X_t] = \mathbb{E}\left[s\sqrt{\alpha_t}X_0 + \sqrt{1-\alpha_t}\epsilon_t\right] \tag{5}$$

Since $\mathbb{E}[X_0] = 0$ and $\mathbb{E}[\epsilon_t] = 0$, it follows that:

$$\mathbb{E}[X_t] = s\sqrt{\alpha_t} \cdot \mathbb{E}[X_0] + \sqrt{1-\alpha_t} \cdot \mathbb{E}[\epsilon_t] = 0. \tag{6}$$

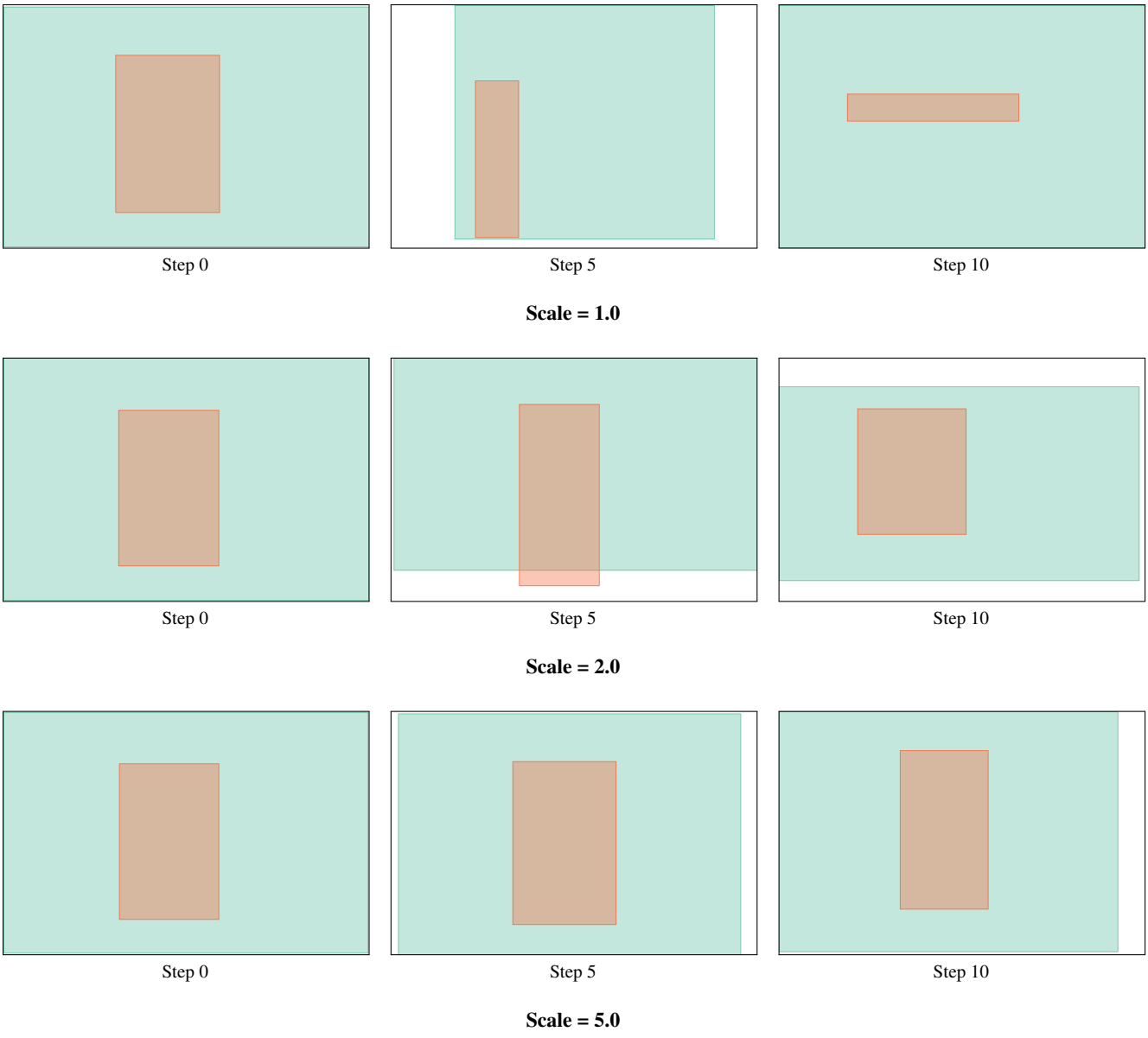



Figure 2. **Visualizing denoising process scale 1.0, 2.0, and 5.0**. The denoising process for higher scaling factor results in a more gradual destruction of information for the bounding box coordinates for layout with prompt *Snowboarder cuts his way down a ski slope*.

Thus,

$$\mathbb{E}[X_t] = 0. \tag{7}$$

**Step 2: Variance of $X_t$**
Next, we compute the variance of $X_t$:

$$\text{Var}(X_t) = s^2\alpha_t \text{Var}(X_0) + (1 - \alpha_t)\text{Var}(\epsilon_t). \tag{8}$$

Since $\text{Var}(X_0) = 1$ and $\text{Var}(\epsilon_t) = 1$, we have:

$$\text{Var}(X_t) = s^2\alpha_t + (1 - \alpha_t) = \alpha_t(s^2 - 1) + 1 \tag{9}$$

**Step 3: Normalization of $X_t$**

We define the normalized process $\tilde{X}_t$ as:

$$\tilde{X}_t = \frac{X_t - \mathbb{E}[X_t]}{\sqrt{\text{Var}(X_t)}} = \frac{s\sqrt{\alpha_t}X_0 + \sqrt{1-\alpha_t}\epsilon_t}{\sqrt{\alpha_t(s^2-1)+1}} \tag{10}$$

This can be simplified to:

$$\tilde{X}_t = \sqrt{\tilde{\alpha}_t}X_0 + \sqrt{1-\tilde{\alpha}_t}\epsilon_t \tag{11}$$

where

$$\tilde{\alpha}_t = \frac{\sqrt{\alpha_t}s}{\sqrt{(s^2-1)\alpha_t + 1}} \tag{12}$$

This completes the proof. □

## 2. Description Set Generation

We use a pre-trained LLM to generate a set of object descriptions $\mathcal{D}$ from a given prompt. The LLM is instructed to follow a structured process to extract noun phrases from the prompt that can be depicted in the scene and to output these objects along with their counts in JSON format. The prompt we use is as follows:

```
You are a creative scene designer who predicts a scene from a natural language prompt. A scene is a
     JSON object containing a list of noun phrases with their counts {"phrase1": count1, "phrase2":
     count2, ...}. The noun phrases contain **ONLY** common nouns. You strictly follow the below
    process for predicting plausible scenes:

Step 1: Extract noun phrases from the prompt. For example, "happy people", "car engine", "brown dog
    ", "parking lot", etc.
Step 2: Limit noun phrases to common nouns and convert the noun phrase to its singular form. For
    example, "happy people" to "person", "tall women" to "woman", "group of old people" to "person
    ", "children" to "child", "brown dog" to "dog", "parking lot" remains "parking lot", etc.
Step 3: Predict the count of each noun phrase and ensure consistency with the count of other
    objects in the scene. If a particular object does not have any explicit count mentioned in the
    prompt, use your creativity to assign a count to make the overall scene plausible but not too
    cluttered. For example, if the prompt is "a group of young kids playing with their dogs," the
    count of "kid" can be 3, and the count of "dog" should be the same as the count of "kid".
Step 4: Output the final scene as a JSON object, only including physical objects and phrases
    without referring to actions or activities.

Complete example:

Prompt: Three white sheep and few women walking down a town road.
Steps:
Step 1: noun phrases: white sheep, women, town road
Step 2: noun phrase in singular form: sheep, woman, town road
Step 3: Since the count of women is not mentioned, we will assign a count of 2 to make the scene
    plausible. The count of "sheep" is 3 and the count of "town road" is 1.
Step 4: {"sheep": 3, "woman": 2, "town road": 1}
Plausible scene: {"sheep": 3, "woman": 2, "town road": 1}

Other examples with skipped step-by-step process:

Prompt: A desk and office chair in the cubicle
Plausible scene: {"office desk": 1, "office chair": 1, "cubicle": 1}

Prompt: A pizza is in a box on a corner desk table.
Plausible scene: {"pizza": 1, "box": 1, "desk table": 1}

Note: Print **ONLY** the final scene as a JSON object.
```

## 3. Aspect Ratio

Our model can generate layouts at different aspect ratios by conditioning on the aspect ratio of the image. We visualize the layouts generated at different aspect ratios in Fig. 3 for a given caption. We observe that our model can generate layouts consistent with the image's aspect ratio.

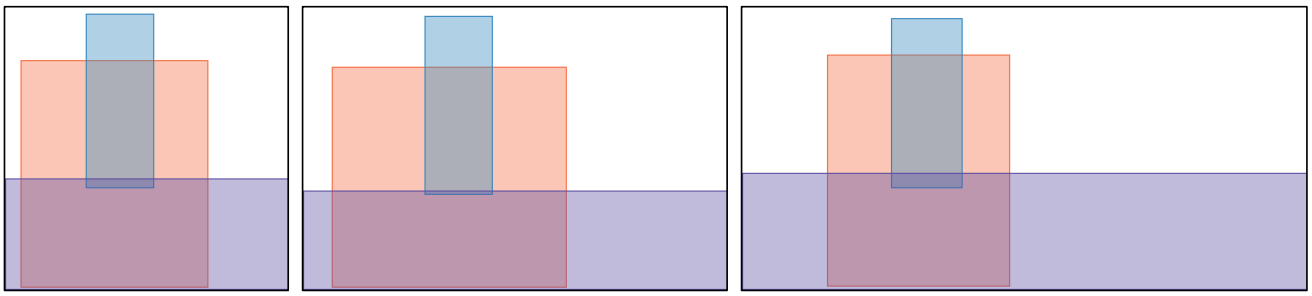
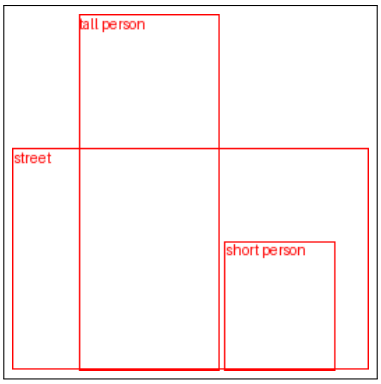
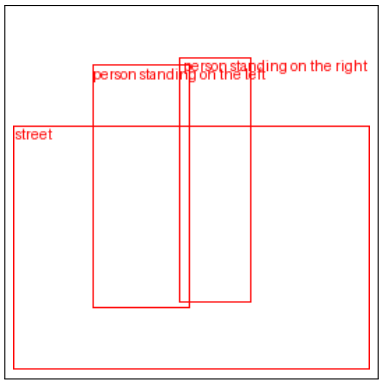



Figure 3. **Layout generation with varying aspect ratios**. Layouts generated at different aspect ratios for prompt: *A man riding a horse on the street.* The model adjusts the position and aspect ratio corresponding to the man and the horse to produce natural-looking layouts.




**Prompt**: Two people on the street
**Description set**: tall person, short person, street




**Prompt**: Two people on the street
**Description set**: person standing on the left, person standing on the right, street

Figure 4. **Visualizing generalization of our model on fine-grained description set**

## 4. Description Set Generalizability

The COCO-GR dataset provides labels that are restricted to object names (e.g., "person") and does not include detailed descriptions paired with these object names (e.g., "tall person"). However, training with the description set in an open-domain manner allows our model to generalize to fine-grained object descriptions. The results are visualized in Fig. 4.

## 5. Additional Baselines

### 5.1. Chain-of-Thought Baseline

We modified LayoutGPT to incorporate a Chain-of-Thought (CoT) process by generating object descriptions before layout prediction. Although this CoT variant improves over the vanilla LayoutGPT baseline in L-FID and achieves similar performance on numerical and spatial reasoning benchmarks, it still falls short of our proposed method (see Tab. 1).

| Method | L-FID (↓) | Numerical Resoning | | | | Spatial Resoning | |
|---|---|---|---|---|---|---|---|
| | | Prec (↑) | Recall (↑) | Acc (↑) | GLIP (↑) | Acc (↑) | GLIP (↑) |
| LayoutGPT (4o-mini) | 6.72 | 73.82 | 86.84 | 77.51 | 57.96 | 92.01 | 60.49 |
| LayoutGPT w/ CoT (4o-mini) | 5.91 | 74.96 | 86.49 | 76.61 | <u>55.95</u> | 92.01 | <u>60.57</u> |
| LayouSyn (GRIT pretraining) | **3.31** | **77.62** | **99.23** | **95.14** | 56.20 | **92.58** | 58.94 |

Table 1. Spatial and Counting Evaluation on the NSR-1K Benchmark with LayoutGPT Chain-of-Thought.

### 5.2. Grounding Evaluation with GroundingDINO

We report counting and spatial accuracy with Grounding-DINO [4] in Tab. 2. Our method shows an advantage over the GLIP-based evaluation, particularly in spatial alignment.

| Metric | GT | LayoutGPT (4o-mini) | LayouSyn | LayouSyn (GRIT pretraining) |
|---|---|---|---|---|
| Counting Score | 49.16 | 56.10 | 56.97 | **56.69** |
| Spatial Score | 86.22 | 86.50 | 82.83 | **88.69** |

Table 2. Spatial and counting accuracy with Grounding-DINO.

### 5.3. Controlled L-FID Evaluation with LLMs

We control the LLM used in training dataset construction and object extraction from prompt at inference to eliminate bias from mismatched LLMs in evaluation. For this, we additionally construct COCOGR-QWEN using Qwen2.5 [5], and refer to the original LLaMA3.1 [1]-based dataset as COCOGR-LLAMA. More generally, datasets built with LLM $X$ are denoted COCOGR-$X$. We train LayouSyn on COCOGR-$X$ and use LLM $X$ for object extraction at inference. For LayoutGPT, we use in-context examples with LLM $X$ for layout generation. All models are evaluated on the COCOGR-$X$ validation split. As shown in Tab. 3, our model consistently outperforms LayoutGPT with both LLaMA3.1 and Qwen2.5, further demonstrating its effectiveness.

| LLM | LayoutGPT | LayouSyn (Ours) |
|---|---|---|
| LLaMA-3.1-8B | 10.81 | **3.07** |
| Qwen2.5-8B | 7.29 | **3.89** |

Table 3. **Controlled L-FID Evaluation with LLMs**: Comparing LayoutGPT and LayouSyn using the same LLM in dataset construction, inference and evaluation stage on L-FID (↓).

### 5.4. Cross-Dataset Generalization

To assess true out-of-distribution generalization, we evaluate on COCOGR-QWEN ensuring that neither Qwen nor COCO are used for training our model. LayouSyn is trained on GRIT and uses LLaMA-3.1 for object extraction at inference. For baselines, we include off-the-shelf Ranni [2] and LLMBlueprint-GPT-4o [3] (without in-context learning). Results in Tab. 4 show our two-stage method outperforms these baselines.

| | Ranni | LLM Blueprint (GPT-4o-mini) | LayouSyn(Ours) |
|---|---|---|---|
| **L-FID** (↓) | 17.44 | 33.87 | **5.35** |

Table 4. **Cross-dataset generalization with LayouSyn**: LayouSyn is trained on GRIT, uses LLaMA-3.1 for object extraction, and is evaluated on COCOGR-QWEN.

| Object Extraction Model | Precision (↑) | Recall (↑) | Accuracy (↑) |
|---|---|---|---|
| GPT-3.5 | 75.73 | 99.47 | 95.01 |
| GPT-4o-mini | 76.33 | 99.43 | 94.23 |
| LLaMA-3.1-8B | 77.62 | 99.23 | 95.14 |

Table 5. Object extraction performance across LLMs.

### 5.5. Evaluating Object-Extraction Capability of LLMs

We assess the ability of LLMs to extract objects from prompts in the NSR-1K dataset, measuring performance with precision, recall, and accuracy. Precision is the percentage of predicted objects in the ground-truth objects set, Recall is the percentage of ground-truth objects in the predicted object set, and Accuracy is defined as 1 if the ground-truth object set and predicted object set overlap exactly and 0 otherwise. Tab. 5 confirms that lightweight LLMs can effectively extract object descriptions.

## References


[1] Abhimanyu Dubey et al. The llama 3 herd of models, 2024.

[2] Yutong Feng, Biao Gong, Di Chen, Yujun Shen, Yu Liu, and Jingren Zhou. Ranni: Taming text-to-image diffusion for accurate instruction following. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4744–4753, 2024.

[3] Hanan Gani, Shariq Farooq Bhat, Muzammal Naseer, Salman Khan, and Peter Wonka. Llm blueprint: Enabling text-to-image generation with complex and detailed prompts. In *The Twelfth International Conference on Learning Representations*, 2024.

[4] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. *arXiv preprint arXiv:2303.05499*, 2023.

[5] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.