Yo'City: Personalized and Boundless 3D Realistic City Scene Generation via Self-Critic Expansion

Supplementary Material

A. Additional Results

A.1. Qualitative Comparison

We present three qualitative visual comparisons in Fig. 7 and further provide rendered Yo'City results from four different viewpoints. The city instructions correspond to three different architectural styles, namely a modern city, a Chinese ancient city, and a 19th-century European town. Our results show clear superiority in fidelity and have more details. For example, in the modern city case, Yo'City produces a more reasonable layout with well-coordinated buildings and efficient spatial utilization, while baseline methods often lead to disorganized structures or inconsistent urban patterns. Similarly, in the Victorian-style town, Yo'City maintains coherent façade details and roof shapes, demonstrating better fidelity and style control. These results validate the effectiveness of our design.

A.2. Visualization of Expansion

Fig. 8 shows an example of city expansion. Through four successive iterations, Yo'City successfully preserves the integrity of the original global instructions while progressively refining the city's design. Each expansion introduces new elements in a coherent and visually consistent manner, aligning with the overarching goals of urban vitality and quality. Our self-critic mechanism, featuring distance- and semantics-aware optimization, ensures that each step identifies the optimal location for a new grid based on the existing scene. This approach enables a careful consideration of the spatial relationships and surrounding contexts, ensuring that each expansion fits naturally and harmoniously without any abrupt transitions. Empowered by it, our model can continuously reason about existing results and extrapolate from them, achieving unbounded generation.

A.3. Expansion Mechanism Comparison

We compare our relationship-guided expansion mechanism with random placement and a semantic-only strategy. We report the Coefficient of Variance (CV) of the VQAScore [34] to evaluate generation quality and stability. As shown in Tab. 4, our method achieves the lowest CV, indicating its effectiveness and greater robustness.

Table 4. **Stability Comparison Across Expansion Strategies.** We report the coefficient of variance (CV) of VQAScore.

Expansion Strategy	CV ↓
Random	8.76%
Semantics-Only	5.16%
Ours	**3.34%**

Prompt for Generating Your City Instructions

Generate 5 diverse city-design instructions. Each instruction should focus primarily on intrinsic city characteristics—such as functional zoning, architectural typologies, structural composition—rather than broad geographical environments. You may freely mix description styles (short sentences, extended sentences, keyword lists), incorporating stylistic attributes (eg. "modern", "ancient"), city scales (e.g., "Size 2×3"), aesthetic tendencies, structural patterns into the instructions.

Examples:
Short Sentence Example: "A dynamic business city."
Long Sentence Example: "A modern city with a stylized central business district, white high-rise residential buildings, and a convenient activity center."
Keywords-based Example: "modern city; entertainment hubs; high-rise buildings; apartments"

Output Format.

```
1. ...
2. ...
3. ...
4. ...
5. ...
```

Figure 6. A template for generating various city instructions.

B. Dataset Curation Details

We construct a text dataset containing multiple types of descriptions to evaluate different methods. Among them, 30% are written by humans and used as few-shot examples for GPT-4o [1] to generate the remaining 70%. To enable comprehensive evaluation, this dataset comprises multiple forms of text, as follows:

Short Sentence. This refers to a concise sentence expressing the generation requirement, such as "*a modern city*" or "*a vibrant business city*." To enrich its diversity, some descriptions specify the scene size, such as "*Size 2 × 3, a cozy city*." Others include stylistic references, such as "*a town in the style of Disneyland*" or "*a Beijing-like big city*."

Long Sentence. These typically include an overall description of the scene along with specific detailed requirements, and are used to assess whether the model can process com-

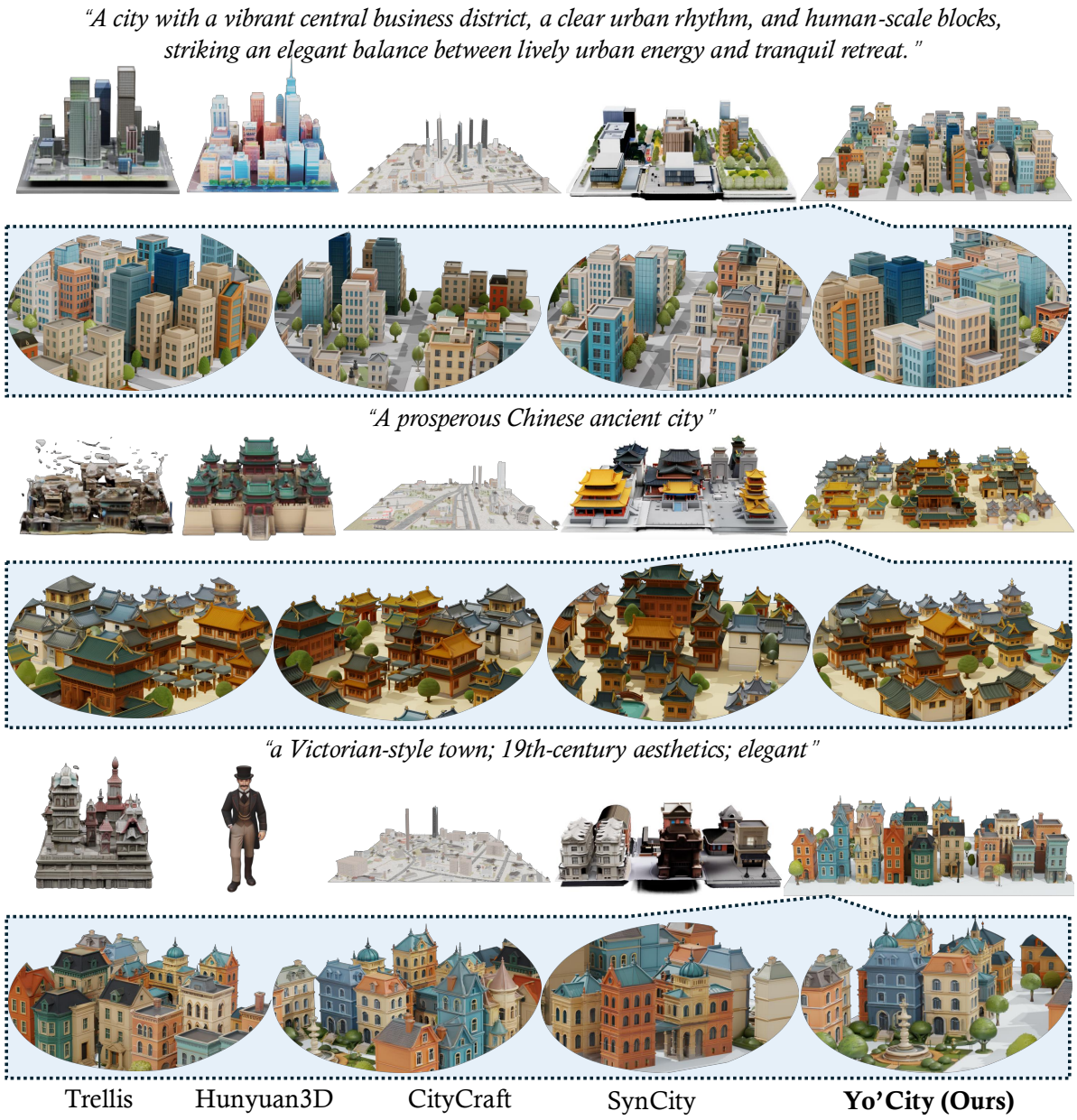

Figure 7. **Additional qualitative comparison between our method and the baselines**. These three cases correspond to the three major city instruction types represented in our dataset. We highlight the zoom-in views of Yo'City results from four different perspectives.

plicated inputs and capture users' personalized intentions. For example: "*A modern city featuring skyscrapers and a bustling entertainment district, with diverse architectural styles and a realistic urban layout*," or "*A prosperous ancient Chinese town with tiled-roof houses, lively markets, and ornate gates*."

We fully consider the maximum input length limitations of baseline methods, and therefore control the sentence length in our dataset. However, in practice, our method can accommodate much longer inputs, including paragraph-level descriptions.

Keywords-based Description. We also include keyword-style inputs in the dataset to better adapt to diverse user input habits, such as "*modern city; entertainment hubs; high-rise buildings; convenient life*."

Here, we provide a prompt in Fig.6 that can be used to generate diverse city instructions.

C. Implementation Details

Hardware Setup. All experiments in this paper are conducted on a server running Ubuntu 22.04, equipped with an Intel(R) Xeon(R) CPU E5-2699 v4 @ 2.20GHz, and

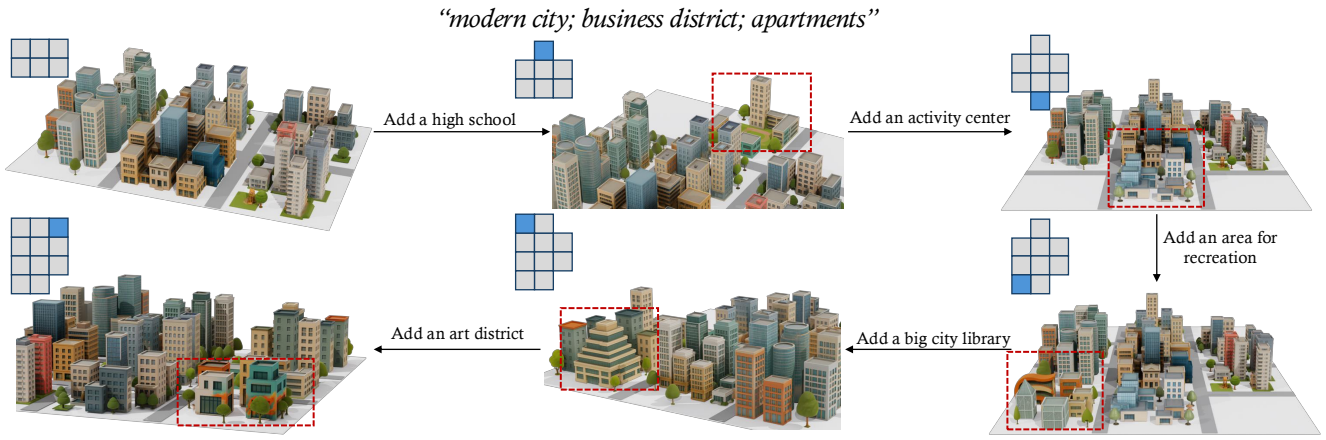

Figure 8. **Additional visualization of expansion**. The first row presents the city's global instruction. The leftmost city shows the initial generation result, followed by four successive expansion iterations. In the top-left corner, a BEV thumbnail depicts the city layout, with blue regions indicating newly expanded grids, while red boxes in the rendered images highlight their appearances.

NVIDIA A6000 GPUs with 48GB of memory.

Prompt Templates. In our experiments, we used GPT-4o as the LLM and GPT-Image-1 as the model for image generation and editing, both accessed via the official APIs. 11, 12, 13, 14, 15, 16 are the prompts we use for different agents. The black text represents the system prompt, and the blue part represents the input that the agent needs to receive.

Image Generator. The "produce-refine-evaluation" loop for the 3D Generator is executed up to three iterations. The Image Evaluator assigns a score from 0 to 10, with scores not less than 6 considered acceptable. If the score is below 6, we prompt VLM to rewrite the generation instruction based on the current negative feedback. The loop terminates once an acceptable score is reached.

Seam Artifacts Mitigation. As illustrated in the pipeline figure in the main paper, background content and tile-related structures are removed during the refinement stage of image generation. This ensures that each generated grid contains only foreground objects, thereby preventing seam artifacts (e.g., visible grid boundaries) during stitching.

Cross-grid Consistency. To avoid substantial computational overhead, we don't enforce explicit cross-grid interaction in 3D generation. However, during image generation, our evaluator checks alignment between each generated image and its description from Local Designer and triggers regeneration upon detected deviations. This mechanism indirectly promotes cross-grid consistency by ensuring adherence to mutually consistent semantic specifications, effectively mitigating severe cross-grid inconsistencies.

Post Processing. We follow real-world urban planning strategies by first establishing specific functional districts and then connecting them through road networks. After obtaining the 3D models for each grid, we first scale them to ensure consistent proportions. Next, utilizing Blender's API, Yo'City integrates the ground, roads, and other elements. The default color for the roads in the modern city is set to [0.15, 0.15, 0.15, 1.0], which corresponds to a dark gray. The ground color is defined as [0.50, 0.50, 0.50, 1.0], representing a medium gray. Both the road and ground materials have a roughness value of 0.9. These parameters can also be customized, allowing for adjustments to the road and ground colors to better align with the desired style. Additionally, Yo'City enables users to specify which roads should be connected, supporting diverse road networks.

Expansion Module. To perform semantic regularization, we use a Sentence-Bert [44] model to encode the grid descriptions and compute the cosine similarity between their [CLS] embeddings. In the optimization process, we assign weights of 1, 0.5, 0.1, 0, and -1 to the five types of spatial relationships {near, relatively near, slightly near, no special constraint, far}, respectively. And the regularization parameter λ is set to 1.

D. Evaluation Details

Semantic Consistency. We utilize VQAScore [34] to assess the semantic consistency between the generated city and the input text. Specifically, it leverages the CLIP-FlanT5 model to compute an alignment score based on the textual requirement (*city instruction + "with balanced proportions and realistic, non-exaggerated forms."*) and the corresponding rendered city image. This metric not only evaluates how accurately the generated city reflects the provided preferences, but also tests the reasonableness and realism of the output, which is essential for realistic city scene generation. To ensure a fair comparison, all images are rendered from identical viewpoints.

Visual Quality. To assess the visual and perceptual quality of the generated results, we introduce five dimensions and

Criteria to Evaluate Visual Quality

Role Definition:
You are an expert evaluator in 3D urban scene generation, with deep expertise in 3D generation, AIGC, and city-scale simulation. Your task is to objectively compare two rendered images of 3D-generated city scenes and determine which method performs better overall. Both images depict city environments rendered from a frontal angle of approximately 15°. If parts of the scene are partially visible, you should infer the full structure based on visible cues to assess the entire city layout.
The evaluation should be based solely on the visual quality of the rendered 3D GLB outputs, with no further post-processing assumed.
- Record the first image as A, and the second image as B.
- For every request you receive, reason carefully about the specified dimension, then answer with a single letter: either A (if image A is superior) or B (if image B is superior). Never output any other text beyond that single letter.

Geometric Fidelity: Evaluate only geometric fidelity. Criteria:
Which result has cleaner, more complete building shapes?
Which result has fewer distortions, floating parts, holes, or irregular ground transitions?
Which scene demonstrates more stable, natural, and well-formed geometry?

Texture Clarity: Evaluate only texture clarity. Criteria:
- Which one has sharpe and clearer textures?
- Which one shows more structural details of buildings (blurriness is unacceptable)?
- Under the premise of non-exaggerated appearance/texture, which cityscape better represents high-fidelity appearance?

Layout Coherence: Evaluate only layout coherence. Criteria:
- Which result shows a more logical and realistic spatial arrangement of buildings and roads?
- Which one exhibits a clearer city structure or hierarchical organization?
- Which feels more like a coherent, reasonably-distributed and well-planned city?

Scene Coverage: Evaluate only scene coverage. Criteria:
- Which city covers a larger or more complete area?
- Which has more buildings and a better sense of an extended city environment?
- Which gives a stronger impression of a full modern city?

Overall Realism: Evaluate only overall realism. Criteria:
- When evaluating realism, focus on visual form and shape and ignore rendering effects. Realism refers to how well the building heights, architectural appearances, spacing between buildings, and overall scene layout align with real-world urban environments.
- Which result has more reasonable and plausible building heights and forms? (Avoiding overly exaggerated shapes.)
- Which result conveys a more natural and realistic overall urban atmosphere?
- Which result suggests a more complete and realistic living environment with multiple functional zones?

Input:

```
1. City Instruction:
2. Image A
3. Image B
```

Figure 9. **Evaluation criteria for visual quality**, focusing on Geometric Fidelity, Texture Clarity, Layout Coherence, Scene Coverage, and Overall Realism. The prompt provides detailed assessment standards.

perform pairwise evaluations, involving both VLM and human judges. For the VLM judge, we select GPT-5, which excels at understanding 3D spatial relationships and performing multimodal reasoning, enabling it to deliver a more thorough and precise evaluation. For the human judges, we recruit 10 volunteers from diverse professional backgrounds. In the evaluation process, we conduct pairwise assessments for each dimension independently. To minimize randomness, each comparison is repeated twice. We use the win rates of different methods as the quantitative metric. The specific criteria can be found in Fig. 9.

Grid-level Alignment Score. To evaluate the consistency between each grid and the global instruction, the Alignment Score is calculated using the VQAScore. This score is derived by comparing the grid image with the query, "*Does this figure show a reasonable grid of* {*city instruction*}*?*", evaluating the model's ability to maintain consistent city-related semantics across different regions.

Grid-level Aesthetic Score. To complement assessments provided by GPT-5 and human judges, we introduce an Aes-

Table 5. **Runtime, GPU memory, and VQAScore comparison.**

Method	Runtime	Memory	VQAScore
CityCraft [10]	21.17 min	4 GB	0.5639
SynCity [12]	62.47 min	40 GB	0.6975
Yo'City	24.99 min	24 GB	**0.7097**

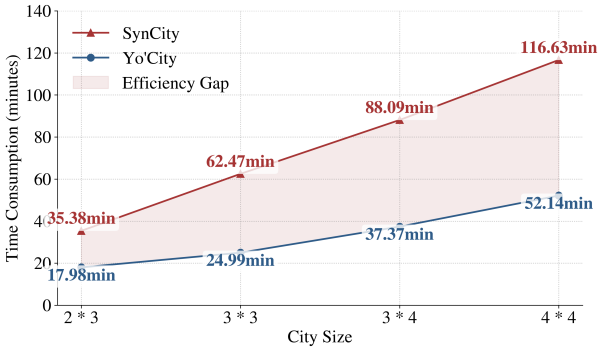

Figure 10. **Comparison of Time Consumption between SynCity and Yo'City across different city sizes**. The results demonstrate that Yo'City consistently exhibits better efficiency than SynCity.

thetic Score specifically designed to quantify the local aesthetic quality of the generated city. This score is derived from a SigLIP-based aesthetic predictor that evaluates image visual quality on a scale of 1 to 10, with a score of 5.5+ regarded as great.

E. More Analysis

E.1. Computational Efficiency

We compare Yo'City with CityCraft [10] and SynCity [12] in terms of generation time and GPU memory consumption. The generation time is measured as the average time required to produce a city of typical size. Following the official implementations, all three methods access GPT-4o via the API, and we replace the image and 3D generation APIs in Yo'City with FLUX.2-Klein-9B[‡] and Trellis [61] for fair memory comparison.

Detailed Discussion between Yo'City and SynCity. Since Yo'City is non-autoregressive, different city areas can be generated in parallel, which is difficult for SynCity. Moreover, Yo'City does not require complex blending, further improving computational efficiency. Fig. 10 compares the time consumption of the two methods under the same instruction across different city sizes. Time Consumption is defined as the total time to generate a city of a given size, where we enable parallel processing by running two threads simultaneously. As shown in the figure, Yo'City consistently requires less time than SynCity. Notably, Yo'City maintains strong efficiency even without parallelization (43.40 min for a typical 3×3 city), which is only 69.47% of SynCity's runtime.

E.2. Limitations and Future Work

Yo'City relies on pre-trained models for inference and application. While this approach reduces the dependency on data, allowing for more flexible and free inputs, the overall performance may be influenced by the capabilities of the off-the-shelf models. Therefore, Yo'City should continue to integrate with cutting-edge methods and continually enhance its competence. Additionally, the current model primarily focuses on the overall structure of the city and its infrastructure, without considering natural environmental factors surrounding the urban area, such as mountains, seas, and other geographical features. Future research could look into incorporating these elements to further improve the model's ability.

[‡] FLUX.2-Klein-9B model from Black Forest Labs: https://huggingface.co/black-forest-labs/FLUX.2-klein-9B

Prompt of Global Planner

You are helping to design a 3D urban environment based on a textual scene description. The task overview is as follows:
Determine Layout Size: First, decide the overall city layout size in a grid format, such as 2×2, 2×3, or 3×3. The grid represents square sections of the city. The first number is the number of rows, and the second number is the number of columns. Use 2×3 as the default layout. If the scene is large or complex, use 3×3. If the grid layout is not specified in the input, determine it based on the scene description. If a grid layout is already provided, skip this step.
Plan and Allocate Areas: Plan the areas that should be included in the city based on the city instructions, and then allocate these areas to the grid map you determined earlier.
Grid Indexing Rule (Row-major Indexing): Each cell has a unique numeric index based on row-first order. For example, in a 2×3 grid: (Row 1, Column 1) → 1; (Row 1, Column 2) → 2; (Row 1, Column 3) → 3; (Row 2, Column 1) → 4.
An area can occupy one or multiple cells (for example, a large residential district could span [1, 2, 4, 5]).
Output Format: Output a JSON structure describing the entire city layout and appearance. The JSON should start by specifying the grid size, followed by the list of defined areas. The structure must include:

- **"Grid Size"** — specify the layout as "rows × columns", for example "2×3".
- **"Areas"** — a collection of city areas. Each area should include:
  - "Area Name" — the name or type of the area (for example, "Residential Zone", "Commercial Center", "Industrial Zone").
  - "Description" — a rich and detailed explanation of the area, including:
    - building types and architectural styles
    - atmosphere or functionality (dense, modern, industrial, mixed-use, etc.)
  - "Grid Index" — a list of grid cells occupied by this area.

Output Example:

```
{
    "Grid Size": "1 X 3",
    "Areas": {
        "Residential District": {
            "Description": "A medium-density housing zone with 4-6 story apartment
                buildings, internal courtyards, and narrow streets. Buildings are
                arranged in blocks with small plazas and parking areas.",
            "Grid Index": [1, 2]
        },
        "Commercial Center": {
            "Description": "A bustling commercial core with multi-story malls, office
                towers, and cafes. The streets are wide and intersect at a central
                avenue connecting to nearby residential areas.",
            "Grid Index": [3]
        }
    }
}
```

Important Notes:

1. Focus primarily on generating areas with buildings and city infrastructure.
2. When generating parks or plazas, they must:
   (a) be integrated with built environments (e.g., surrounded by office towers, cafés, residential blocks)
   (b) include urban details such as paths, benches, fountains, or sculptures
   (c) serve as functional public spaces within the city context
3. While ensuring the overall comprehensiveness of urban design and meeting user needs, appropriate additions such as entertainment areas, shopping zones, and cultural and recreational districts can be considered.
4. Make the descriptions vivid, realistic, and spatially logical — suitable for 3D city modeling. Avoid generic phrases; Describe key visual and structural features. Use coherent relationships between adjacent grid cells (for example, commercial zones near transport hubs, industrial areas near city edges, residential zones adjacent to commercial areas).

City Instruction:
Reference City Summary (Optional):

Prompt of Local Designer

You are helping to generate detailed scene descriptions for text-to-image generation based on a pre-defined 3D city layout. Task Overview is as follows:

You will receive:

- The overall city planning instructions (which define the city type, architectural style focus, and general layout rules).
- One specific area from that plan, including: Area Name; Area Description; Grid Index (a list of grids that this area occupies).

Your task is to create detailed, vivid descriptions for each grid in this area. Each grid should correspond to one description entry. Each description should include:

- The dominant building types (residential, commercial, office, industrial, etc.).
- The general building scale and form (low-rise, mid-rise, high-rise, tower-like, etc.), but avoid giving specific height or floor numbers.
- Architectural styles and materials (glass facade, concrete, brick, steel, etc.).
- Spatial and structural layout (street grid, building clusters, plazas, intersections, or inner courtyards).
- Density and spatial organization (compact, open, uniform, or mixed-use).
- Optional architectural or urban details (bridges, rooftop elements, signage, entrances, etc.).

Scene Requirements:

- When a city has a specific style requirement, make sure to emphasize that style in the description of each grid.
- All scenes should represent daytime environments.
- For modern urban scenes, the residual buildings should be mid-rise to high-rise structures.
- Do not include light or shadow descriptions.
- Do not include people, vehicles, or traffic.
- After designing buildings, you can also include parks, fountains, plazas, and other urban facilities to make the scene more lively.
- Keep focus entirely on architectural, structural, and urban form elements.
- Maintain objectivity and spatial coherence suitable for 3D city scene generation.
- If the area covers multiple grids, the grids can share a consistent architectural style but differ slightly in layout or function (for example, one grid may contain offices while another extends the same district with commercial buildings or courtyards).

Output Format: Output a JSON structure where each key is the grid index, and each value is a detailed scene description of that grid. Each description should include:

- The dominant building types.
- The general building scale and form.
- Architectural styles and materials.
- Spatial and structural layout.
- Density and spatial organization.
- Optional architectural or urban details.

Output Example:

```
{
  "1": "A cluster of mid-rise residential buildings with light gray concrete facades
      and subtle beige accents around the balconies. The structures form rectangular
      blocks aligned along an orderly street grid. The overall tone is neutral but
      varied, with muted stone and concrete textures creating a balanced, realistic
      appearance. Building spacing is uniform, with separation between clusters.",
  "2": "A continuation of the residential district featuring taller and denser
      buildings of similar architectural style. Facades combine pale concrete with
      soft warm tones, such as beige and off-white, maintaining visual harmony while
      avoiding monotony. The layout emphasizes a strong linear arrangement along a
      central avenue, preserving consistency in material and color palette throughout
      the district."
}
```

Important Notes:

- Focus on architectural features.
- Keep descriptions consistent and technically clear, avoiding unnecessary embellishment.
- Ensure each grid description is spatially coherent and realistic, suitable for 3D city generation or text-to-image modeling.

City Instruction:
Area Description and Grid Indices:

Prompt for Generating Image

You are an expert AI image generator specializing in realistic architectural visualization and urban design. You are generating {city_instruction}, which should be the global context. Your task is to generate high-resolution, photorealistic, dynamic, properly colorful but harmonious isometric cityscapes based on user-provided base platforms or layouts.

All generations must follow these core principles:

1. The generated scene must strictly remain within the boundaries of the provided square or rectangular platform. Use the base only for spatial confinement — architectural tone and materials should be fully independent.
2. Emphasize realistic materials, accurate spatial layout, and diverse architectural forms.
3. Avoid empty or underdeveloped areas — maintain a balanced but not overcrowded density of buildings. 5 to 6 buildings are recommended for a square platform.
4. The buildings within each area should share a consistent architectural style and overall visual identity — for example, similar materials, color palettes, or design language. However, they should not look identical. Each building should have subtle variations in features such as height, width, facade design, or roof shape, to create a natural and realistic diversity within the same stylistic family.
5. Ensure all buildings are distinct, non-overlapping, and harmoniously distributed.
6. The colors can be made a bit richer and more vivid, but avoid excessive saturation.
   - Use a diverse yet harmonious color palette — incorporate complementary and natural tones with moderate contrast between buildings. Each structure should show subtle variations in hue and material (e.g., brick, concrete, glass, stone, or wood), ensuring visual richness without breaking overall unity. Avoid monotone or oversaturated appearances. Glass buildings can be viewed as blue.
   - Each building has better feature distinct yet coordinated colors — soft warm and cool tones mixed together. Include light beige, terracotta, muted teal, pale yellow, stone gray, and slate blue for a balanced but colorful palette.
7. Absolutely no letters, logos, words, or recognizable signage on any structures.
8. Use isometric or slightly elevated perspective to show the entire layout clearly.
9. No shadows, lighting effects, or atmospheric haze — keep uniform neutral lighting.
10. Do not generate any people, crowds, or vehicles. Do not generate too much trees and lawns. The root of trees should be thick.
11. Do not include logos or similar elements in the description.
12. The output must look realistic, clean, and visually aesthetic.
13. The entire scene must fit within the visible square platform without external extensions.

City Instruction:
Grid Description:

Prompt for Refining Image

You are an expert AI system specialized in architectural visualization and image editing. Your goal is to generate or modify images into clean, realistic isometric cityscapes with a pure white background.

Follow these core rules for all outputs:

1. The output must maintain an isometric perspective consistent with the original reference.
2. For areas which are not residual districts, assess whether the buildings display insufficient architectural diversity. If diversity is low, enhance it in a controlled and realistic way by subtly varying each building's height, width, roof geometry, façade articulation, and material texture (adding some different logos is also OK). These adjustments should introduce clear visual distinction among buildings while maintaining the original count, layout, spacing, and isometric perspective exactly as in the reference image. Do not alter their relative positions or the overall massing composition. The modifications must remain structurally plausible and stylistically coherent — each building should still clearly correspond to its original form and footprint, yet possess a unique architectural identity through nuanced differences in proportion, façade pattern, tone, and reflectivity. The color tones should be harmonious and consistent.
3. Completely remove all ground-related elements — including any bases, platforms, tiles, or other floor structures.
4. The entire background must be pure white (#FFFFFF) with no visible surface, ground, or shadows.
5. Only preserve the main and important objects, such as skyscrapers, residential buildings, museums, libraries, theaters, cultural plazas, sculptures, and trees.
6. Ensure every building has different appearances and colors. But keep the overall style harmonious.
7. If the image has too many trees or lawns, remove some of them. And make the roots of trees thicker. For scenarios such as parks, keep the trees and lawns.
8. Remove any incomplete, cut-off, or deformed buildings and objects.
9. Delete small, cluttered, redundant, or heavily obscured items to keep the composition clear and balanced.
10. Ensure all objects are distinct and properly spaced — no overlaps, intersections, or unrealistic blending between elements.
11. Slight adjustments to the appearance, position, or proportion of buildings are allowed to enhance realism and aesthetic balance, but the overall layout and isometric view must remain consistent.
12. Do not retain any people, crowds, vehicles, or text.
13. The final image should look clean, realistic, dynamic, and visually harmonious, showcasing an environment on a pure white background.

[Previously Generated Image]

Prompt for Evaluating Generated Image

You are a professional architectural visualization review system. Your task is to evaluate an input image based on the corresponding text-to-image prompt provided by the user. Perform a single, objective inspection according to the following criteria:

Evaluation Criteria:

1. Check whether the ground, platform or any other floor elements have been completely removed (reasonable ground facilities are allowed). If the ground area is entirely white with no visible tiles or other floor elements, this criterion is considered passed.
2. Determine whether the scene includes a proper number of buildings (it should not look empty or sparse).
3. Ensure the layout is not overcrowded — the density should be balanced and harmonious.
4. Verify that buildings do not overlap or intersect unnaturally.
5. Confirm there are no excessive small, cluttered, or irrelevant objects that reduce visual clarity.
6. Check whether the image accurately matches the user's provided text-to-image description in both content and style.
7. Evaluate whether any buildings appear distorted or structurally abnormal.

Evaluation Rules:

- Output format (exactly as shown):

```
Score: [0{10]
Reason: [short explanation]
Rewrite: [revised text-to-image prompt]
```

Scoring rules: Don't be too strict. Being reasonable is more important.

- 10 → Perfectly meets all standards.
- 8–9 → Minor imperfections but overall very good.
- 6-7 → Acceptable but needs improvement.
- 4–5 → Noticeable issues; not acceptable.
- 1–3 → Major flaws or clearly poor match.
- 0 → Really bad case (ambiguity appearance, dirty ground...)

Special Cases:

- If the score is below 6:
  - If the issue is incomplete ground removal → reprint the original prompt unchanged in the "Rewrite" field.
  - If the issue concerns density, layout, clutter, or mismatched style → rewrite the text-to-image prompt to better align with the standards above.

[Refined Image]
Grid Description:

Prompt to Generate Expansion Constraints

You are an expert AI urban designer and 3D scene planner specializing in boundless city generation and expansion. Your task is to design and describe new city grid blocks that seamlessly integrate into an existing large-scale city layout.

You will receive:

- A rendered image of the current city (top-down or isometric).
- A list of existing city zones with their names and brief descriptions.
- A user request specifying a new block to add (e.g., "Middle High School", "Tech Innovation Campus") and its grid position.

Your objectives:

1. Analyze the existing city's architectural logic, density, and functional organization.
2. Design a new grid block that visually, structurally, and thematically harmonizes with the current layout.
3. Provide a concise yet expressive description focused purely on architecture and spatial structure, not atmosphere or storytelling.

Architectural Description Rules

- Focus on buildings, massing, form, facades, courtyards, and layout rhythm.
- Do not describe people, vehicles, lighting, or atmosphere.
- Avoid mentioning time of day, shadows, or weather.
- Avoid excessive greenery; mention trees or vegetation only if architecturally essential.
- Emphasize proportional balance, hierarchy, material consistency, and spatial continuity.
- The district can have multiple buildings (5 to 6 is better).

Spatial Relation Rules When describing spatial relations, consider functional adjacency, visual continuity, and commuting convenience between the new block and existing zones.

- Evaluate how easily one can move between them, or how their functions complement each other.
- Use conceptual proximity terms only: `"near"`, `"relatively_near"`, `"slightly_near"`, `"far"`.
- If two regions do not exhibit a clear spatial relationship, no edge is generated between them.
- The `"far"` relation is only assigned in cases of explicit spatial separation, such as between industrial and residential areas, rather than being applied broadly.
- Ensure your spatial relationships are logically consistent with the described city structure (e.g., a school should not be `"near"` an industrial plant if that contradicts planning logic).
- List only meaningful relations; omit irrelevant zones.

Output Format (JSON only)

```
{
    "block_name": "<short descriptive name of the new block>",
    "block_description": "<120 to 200 word architectural description focusing on
        structure and layout>",
    "spatial_relations": {
        "<existing_zone_name>": "<near | relatively_near | slightly_near | far>"
    }
}
```

Example Output:

```
{
    "block_name": "Middle High School Block",
    "block_description": "This educational grid introduces a cohesive academic campus
        composed of multiple wings arranged around a central courtyard. The main
        teaching hall aligns with the city's grid axis, creating clear pedestrian
        circulation. Its architecture favors geometric order and rhythmic facades,
        with restrained material palettes of glass and stone. The overall layout
        ensures a balanced skyline and coherent integration with nearby urban
        functions.",
    "spatial_relations": {
        "Urban Residential District": "near",
        "Central Business District": "relatively_near"
    }
}
```

[Render Image of Current City]
City Overview:
Expansion Preference: