CITYEXPANDER: AGENT-GUIDED SYNCHRONIZED DIFFUSION FOR LARGE-SCALE CITY GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Generating large-scale urban semantic layouts from text requires preserving both city-wide spatial coherence and fine-scale structural detail while satisfying spatial planning constraints. Directly generating high-resolution semantic image over large urban areas is computationally expensive, while lowering the resolution can erase narrow roads and merge nearby buildings. Sliding-window generation limits per-window computation but introduces sequential dependencies and may produce seams or repetitive patterns. Meanwhile, directly conditioning a generative model on text provides only coarse control over the locations, proportions, and spatial relations of land-use categories. We present CityExpander, a text-guided diffusion framework that addresses both challenges. First, We introduce a *compiler-grounded condition plan agent* that translates natural-language descriptions into executable zone-ratio conditions. To balance computational cost and structural fidelity, we use a global DiT for coarse city structure and a local DiT for fine-scale refinement. We propose *semantic consensus globally synchronized denoising* (SC-GSD) to achieve semantic consistent layout generation with simultaneous denoising, avoiding the sequential dependency of sliding-window sampling. Extensive experiments demonstrate that CityExpander effectively generates high-quality text-guided city layouts at both single-tile and large spatial scales. Further evaluations validate the effectiveness of SC-GSD in promoting semantic consistency and reducing time cost during large-scale synthesis.

1 INTRODUCTION

Generating high-quality 2D semantic layouts is fundamental to creating 3D city scenes for simulation, gaming, and digital twins (Zhang et al., 2024; Shang et al., 2024). Recent generative models, particularly diffusion models, have shown promise in synthesizing diverse and plausible city layouts (Deng et al., 2024; Qin et al., 2024; Huang et al., 2026). However, two fundamental challenges persist in real-world applications: scalability to larger areas and fine-grained controllability by text. Scaling to larger urban areas creates a trade-off between computational cost and structural fidelity.Directly generating large-scale city layouts at high resolution is computationally expensive, while operating only at a coarse resolution tends to sacrifice fine-grained geometric details. Existing methods often extend layout to large scales through sliding-window generation (Lee et al., 2025; Deng et al., 2025a) or autoregressive expansion (Xie et al., 2024). Sequential window generation becomes increasingly costly as the canvas grows and can accumulate visible seams, scan-order bias, and repetitive patterns. For controllability, current approaches often couple high-level text descriptions directly with low-level pixel generation (Xi et al., 2025; Deng et al., 2024; Huang et al., 2026). The resulting modality gap limits precise control over the locations, proportions, and spatial relations of land-use categories.

To address these challenges, we introduce CityExpander, a text-guided diffusion framework for large-scale 2D semantic city layouts. Our core insight is to decouple language understanding from pixel synthesis through an intermediate *zoning-ratio representation*. Furthermore, we propose a multi-block joint denoising strategy that simultaneously denoises multiple spatial blocks, eliminating the sequential dependency inherent in sliding-window sampling while preserving the structural consistency of the semantic map.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

As shown in Figure 1, we introduce a condition plan agent with a deterministic spatial compiler to instantiate the zone ratio representation form free-form descriptions. The planner translates textual intent into a coarse-to-fine organization of functional zones with explicit land-use compositions, while the compiler converts this plan into a geometrically valid and numerical consistent ratio field for diffusion conditioning through zone ratio ControlNet. To enable efficient large-scale synthesis, we develop semantic-consensus globally synchronized denoising (SC-GSD), which combines shared-canvas multi-window sampling (Bar-Tal et al., 2023; Lee et al., 2023) with explicit semantic consistent supervise. Instead of committing windows sequentially, SC-GSD jointly updates overlapping windows and aligns their semantic predictions in shared regions with an overlap-restricted Jensen–Shannon objective, reducing order bias and boundary artifacts without extra training. The shared-canvas formulation removes the sequential dependency of sliding-window inference, supports parallel window processing, and promotes agreement across window boundaries. To reconcile global structural consistency with local fidelity, we apply a multi-scale pipeline: a global DiT trained on 1 km tiles establishes kilometer-scale organization at coarse resolution, while a local DiT trained on 0.25 km patches refines fine-scale geometry under global guidance.

Extensive experiments demonstrate that CityExpander outperforms existing methods in single-tile layout generation. It also achieves better text-conditioned generation, highlighting the effectiveness of the intermediate spatial zone-ratio representation. Ablation studies further demonstrate the contributions of SC-GSD to semantic consistency and multi-scale diffusion to coherent urban structure and fine-grained detail. Finally, we convert generated layouts into procedural 3D city scenes, demonstrating their utility for downstream city modeling.

Our contributions are threefold:

- We introduce a **compiler-grounded hierarchical planning** framework that translates natural-language descriptions into intermediate zone-ratio conditions, bridging high-level language intent and low-level spatial control.
- We propose **Semantic-Consensus Globally Synchronized Denoising** (SC-GSD) for large-scale urban semantic generation. Beyond synchronizing overlapping windows through shared latent states and averaged noise predictions, SC-GSD explicitly aligns their predicted clean semantic distributions and preserves the compiled zone-ratio field.
- We design a **global–local city generation strategy** that separates kilometer-scale spatial structure from high-resolution fine-detail synthesis, enabling coherent generation for city at large scales.

2 Related Work

Diffusion models have become the standard for image generation. Early diffusion probabilistic models (Sohl-Dickstein et al., 2015; Ho et al., 2020; Nichol & Dhariwal, 2021) progressively removed Gaussian noise in pixel space. Latent diffusion (Rombach et al., 2022) shifted this process to a compressed latent space, enabling high-resolution synthesis. Diffusion transformers (DiTs) (Peebles & Xie, 2023) replaced the U-Net backbone with transformer architectures for improved quality. To enhance controllability, ControlNet (Zhang et al., 2023) and T2I-Adapter (Mou et al., 2024) introduced trainable conditioning modules that inject spatial or semantic guidance while keeping the base model frozen. Ramesh *et al.* (Ramesh et al., 2022) leveraged CLIP embeddings for text-image alignment. For large-scale coherence, Blended Diffusion (Avrahami et al., 2022) performed mask-guided local editing. MultiDiffusion (Bar-Tal et al., 2023) synchronizes multiple overlapping windows by fusing noise predictions, whereas SyncDiffusion (Lee et al., 2023) additionally introduces perceptual-similarity guidance. Our SC-GSD incorporates explicit semantic supervision into synchronized denoising to improve the coherence of generated urban semantic maps.

City layout generation spans both 2D semantic maps and 3D scenes. Early methods like UrbanGAN (Quan, 2022) and Infinicity (Lin et al., 2023) used generative adversarial networks (GANs) to generate layouts. GlobalMapper (He & Aliaga, 2023) employed a variational autoencoder (VAE) for layout completion, while CityDreamer (Xie et al., 2024) combined VAE encoding with autoregressive expansion. Recently, diffusion models have become mainstream due to their superior quality (Dhariwal & Nichol, 2021). COHO (He & Aliaga, 2024) employed graph-based masked autoencoders for hierarchical layout generation, and CityGen (Deng et al., 2025a) scaled 2D layouts

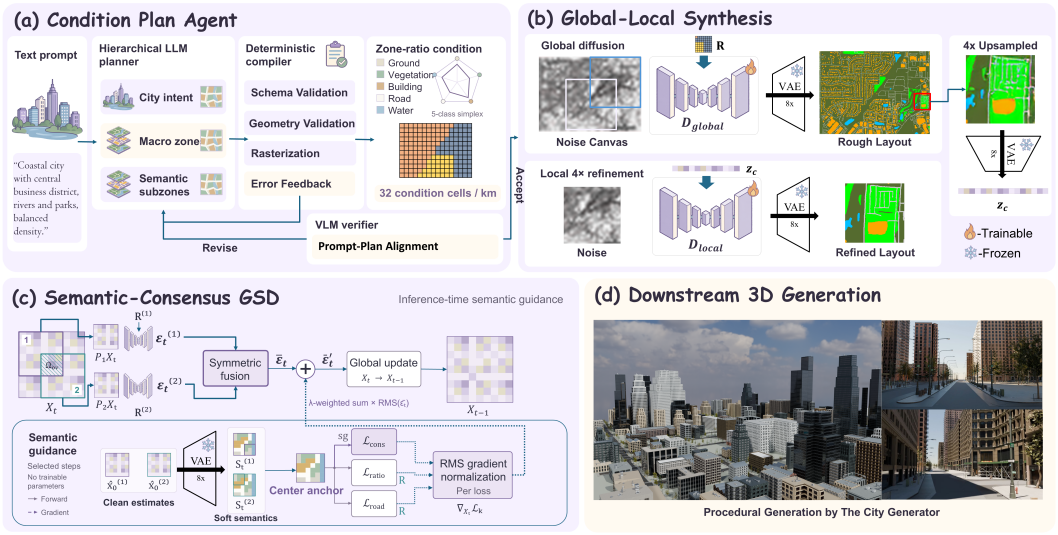

Figure 1: **Overview of CityExpander.** The Condition Plan Agent (a) compiles text into spatial zone-ratio conditions. Global–local diffusion (b) generates large-scale layouts, while SC-GSD (c) synchronizes overlapping windows through noise fusion and semantic constraints. Downstream 3D generation pipelines further transform the generated layouts (d) into diverse urban scenes.

via sliding windows. For 3D city generation, BlockFusion (Wu et al., 2024) and NuiScene (Lee et al., 2025) adopted tiling strategies, GaussianCity (Xie et al., 2025) used Gaussian splatting for real-time rendering, and CityX (Zhang et al., 2024) and UrbanWorld (Shang et al., 2024) leveraged LLM-based procedural generation. Yo'City (Lu et al., 2026) and MajustuCity (Huang et al., 2026) propose agentic method for 3D city generation. For controllability, CityCraft (Deng et al., 2024) supported both ratio-based and text-based conditioning, TerraCraft (Xi et al., 2025) applied LoRA fine-tuning, and Text2City (Qin et al., 2024) encoded text into parametric representations. However, these methods either struggle to scale beyond limited areas or lack fine-grained spatial control. CityExpander addresses both through synchronized large-area synthesis and compiler-backed zoning-ratio conditioning.

LLM-guided text to layout has gained traction due to the reasoning capabilities of LLMs (Brown et al., 2020; Wei et al., 2022; Wang et al., 2026; Guo et al., 2026), particularly for indoor scenes (Çelen et al., 2024; Aguina-Kang et al., 2024; Deng et al., 2025b; Sun et al., 2025). Early approaches such as LayoutGPT (Feng et al., 2024) and LLplace (Yang et al., 2024a) directly prompted LLMs to predict layout structures, but often produced physically or semantically inconsistent outputs for complex scenes. Recent methods narrow this gap through scene graphs (Yang et al., 2024b; Fu et al., 2024; Liu et al., 2025), procedural parameters (Liu et al., 2024; Zhou et al., 2025), domain-specific languages (Zhang et al., 2025), or image intermediates (Ling et al., 2025). Urban generation poses an additional difficulty: textual intent must be converted into localized controls whose complexity grows with physical extent. CityExpander combines an explicit hierarchical zoning-ratio plan with a deterministic compiler: the LLM reasons about semantic regions and quantitative composition, the compiler defines their executable spatial condition, and the diffusion model realizes local morphology.

3 Method

3.1 Condition Plan Agent

The *Condition Plan Agent* uses an LLM to determine the physical canvas extent E from the input text T and construct a hierarchical plan. Starting from a single region covering the full extent, the LLM uses the current plan at each round to decide whether to submit it or select a region for further subdivision. Each response includes the updated plan $\mathcal{P} = (\mathbf{b}, \mathcal{R})$, where $\mathbf{b}$ is the default zone-ratio

vector and $\mathcal{R} = \{r_k\}_{k=1}^{K}$ contains the planned regions. Each region is represented as

$$r_k = (s_k, g_k, \boldsymbol{\alpha}_k, p_k), \tag{1}$$

where s_k describes the region's intended function, g_k is an ordered list of polygon vertices encoded as row–column grid coordinates (e.g., `r6c6`), $\boldsymbol{\alpha}_k$ gives its zone ratios over background, vegetation, building, road, and water, and p_k identifies its parent region, preserving the hierarchy formed through subdivision. To discourage implausible zone ratios, we require explicit functional semantics for each region and provide text-to-ratio examples and composition anchors drawn from the dataset in the prompt.

The compiler rasterizes the plan on a square grid with $32E$ cells per side and clips each child geometry to its ancestors. Each grid cell receives the average land-use composition of all regions covering it, or the default composition $\mathbf{b}$ if none covers it. This produces the five-class ratio field $\mathbf{Q}$, independently of region order. After applying a 3×3 box filter and normalizing each cell's proportions to sum to one, the vegetation, building, road, and water channels form the diffusion condition $\mathbf{R}$. The compiler also audits geometry, hierarchy, coverage, overlap, feature size, and target-ratio deviation.

The compiler additionally renders $\mathbf{Q}$ as a five-class preview. A separate reviewer VLM jointly examines this preview, the original text T, and the compiler diagnostics to assess text–plan consistency. It either accepts the plan or proposes bounded structured edits to $\mathcal{P}$. Each revision is recompiled and retained only if it passes the deterministic audits; otherwise, the last valid plan is preserved. The review concerns the condition plan rather than the final layout. Detailed prompts and validation rules are provided in the supplementary material.

3.2 Multi-Scale Diffusion

Each $1\,\mathrm{km}$ layout is represented as an $S\times S$ image over five land-use classes and encoded by a pretrained VAE into $\mathbf{x}_0 \in \mathbb{R}^{C\times s\times s}$. Following latent diffusion (Rombach et al., 2022), we train each DiT to predict the noise added at timestep t:

$$\mathcal{L}_{\mathrm{diff}} = \mathbb{E}_{t,\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\|\boldsymbol{\epsilon} - \epsilon_\theta(\mathbf{x}_t, t; \mathbf{c})\|_2^2\right], \tag{2}$$

where $\mathbf{c}$ is the ratio field $\mathbf{R}$ for the global model and a coarse structural latent $\mathbf{z}_{\mathrm{c}}$ for the local model.

The global DiT is trained on $1\,\mathrm{km}$ tiles to model city-scale organization under zone-ratio control. The local DiT operates on $0.25\,\mathrm{km}$ patches with the same latent dimensions and learns high-resolution refinement conditioned on $\mathbf{z}_{\mathrm{c}}$, constructed by VAE reconstruction and random cropping during training.

For the global model, we inject $\mathbf{R}$ through ControlNet (Zhang et al., 2023) and adopt classifier-free guidance (CFG) (Ho & Salimans, 2022) to control condition strength. ControlNet produces conditional and unconditional residuals, which are interpolated before being injected into the DiT (Peebles & Xie, 2023):

$$\begin{aligned} \mathbf{h}_c &= \mathcal{F}_{\mathrm{CN}}(\mathbf{x}_t, t; \mathbf{R}), \quad \mathbf{h}_u = \mathcal{F}_{\mathrm{CN}}(\mathbf{x}_t, t; \mathbf{0}), \\ \mathbf{h}_{\mathrm{CFG}} &= \mathbf{h}_u + c(\mathbf{h}_c - \mathbf{h}_u), \qquad \boldsymbol{\epsilon}_\theta = \mathcal{F}_{\mathrm{DiT}}(\mathbf{x}_t, t; \mathbf{h}_{\mathrm{CFG}}), \end{aligned} \tag{3}$$

where c is the CFG scale controlling the trade-off between condition fidelity and generation diversity. During training, $\mathbf{R}$ is obtained by SLIC partitioning (Achanta et al., 2010) and regional averaging of ground-truth layouts; during inference, it is produced by the Condition Plan Agent.

At inference, the global model first samples a coarse layout conditioned on $\mathbf{R}$; SC-GSD expands this stage over overlapping $1\,\mathrm{km}$ windows for multi-kilometer extents. The decoded result is then upsampled $4\times$, re-encoded as $\mathbf{z}_{\mathrm{c}}$, and refined by the local DiT over overlapping $0.25\,\mathrm{km}$ windows.

3.3 Semantic-Consensus Globally Synchronized Denoising (SC-GSD)

SC-GSD is an inference-time sampler jointly denoises overlapping windows on a shared canvas (Bar-Tal et al., 2023; Lee et al., 2023) and uses semantic guidance to align their land-use predictions. Let $\mathbf{X}_t \in \mathbb{R}^{C\times H\times W}$ be the latent canvas and $\mathbf{P}_i$ extract its i-th overlapping $s\times s$ window. Each window uses its local compiled condition to predict $\boldsymbol{\epsilon}_t^{(i)} = \epsilon_\theta(\mathbf{P}_i\mathbf{X}_t, t; \mathbf{R}^{(i)})$. The predictions

are symmetrically fused as

$$\bar{\boldsymbol{\epsilon}}_t = \frac{\sum_{i=1}^{N} \mathbf{P}_i^\top \boldsymbol{\epsilon}_t^{(i)}}{\sum_{i=1}^{N} \mathbf{P}_i^\top \mathbf{1}}. \tag{4}$$

Averaging noise does not ensure that windows assign the same land-use label to a shared physical location. At selected steps, we decode each window's clean estimate $\hat{\mathbf{x}}_0^{(i)} = \frac{\mathbf{P}_i \mathbf{X}_t - \sqrt{1-\bar{\alpha}_t}\,\boldsymbol{\epsilon}_t^{(i)}}{\sqrt{\bar{\alpha}_t}}$ through the frozen VAE and apply a differentiable palette projection to obtain soft five-class distributions $\mathbf{S}_t^{(i)}$. These predictions form a shared semantic reference, the center anchor $\mathbf{A}_t(u) = \sum_{i:u\in\Omega_i} w_i(u)\mathbf{S}_t^{(i)}(u)$, where Ω_i is the window footprint in global pixel coordinates. The weights $w_i(u)$ sum to one over covering windows, giving central predictions more influence than boundary predictions. Each window is encouraged to match this reference in overlapping pixels Ω_{ov} through Eq. 5.

$$\mathcal{L}_{\text{cons}} = \frac{\sum_i \sum_{u\in\Omega_i\cap\Omega_{\text{ov}}} n(u)^{-1} D_{\text{JS}}\Big(\mathbf{S}_t^{(i)}(u), \text{sg}[\mathbf{A}_t(u)]\Big)}{\sum_i \sum_{u\in\Omega_i\cap\Omega_{\text{ov}}} n(u)^{-1}}, \tag{5}$$

where $n(u)$ counts covering windows and $\text{sg}[\cdot]$ stops gradients through the anchor. The ratio-preservation loss $\mathcal{L}_{\text{ratio}} = \text{SmoothL1}(\mathcal{D}_\sigma(\mathbf{A}_{t,\text{fg}}), \mathbf{R})$ aligns local land-use proportions with the compiled condition; $\mathcal{D}_\sigma$ pools and smooths the four foreground channels onto the condition grid. A target-gated boundary loss $\mathcal{L}_{\text{road}}$ uses $\mathbf{A}_{t,\text{road}}$ and $\mathbf{R}_{\text{road}}$ to encourage road crossings only where the condition supports them. Gradients through $\mathbf{A}_t$ remain enabled for both losses. We normalize each loss gradient by its RMS, then combine them to correct the fused prediction through Eq. 6. The corrected prediction drives one global sampler update.

$$\bar{\boldsymbol{\epsilon}}_t' = \bar{\boldsymbol{\epsilon}}_t + \text{RMS}(\bar{\boldsymbol{\epsilon}}_t) \sum_{k\in\{\text{cons,ratio,road}\}} \lambda_k \frac{\nabla_{\mathbf{X}_t}\mathcal{L}_k}{\text{RMS}(\nabla_{\mathbf{X}_t}\mathcal{L}_k) + \varepsilon}. \tag{6}$$

Intuitively, $\mathcal{L}_{\text{cons}}$ encourages windows covering the same location to predict the same land-use class; $\mathcal{L}_{\text{ratio}}$ aligns the local land-use proportions of the denoised canvas with the compiled zone-ratio field; and $\mathcal{L}_{\text{road}}$ promotes road continuity across window boundaries where supported by the compiled road condition.

4 Experiments and Results

4.1 Experiment Setup

Dataset and preprocessing. From OpenStreetMap (OSM) (OpenStreetMap contributors, 2017), we sample 650k images of size 1 km×1 km at 1024×1024 resolution: 500k for training and 150k for validation and testing. Tiles for global diffusion training are downsampled to S=256, while patches for local diffusion training are randomly cropped at S=256 from full-resolution images. The dataset encodes five land-use types (Section 3.1) using a fixed color palette. For the large-area sampler ablation (Section 4.3), we use 5k validation maps at 2.5 km×2.5 km. To prevent geographic leakage, training and test tiles are sampled from non-overlapping geographic regions with a minimum separation of 5 km. We further verify that no test tile overlaps with training tile's bounding box.

Baselines and setup. We compare against three recent methods: CityDreamer (Xie et al., 2024) autoregressively predicts layout patches using a VQ-VAE codebook; CityGen (Deng et al., 2025a) applies a U-Net-based diffusion model in one-hot pixel space; CityCraft (Deng et al., 2024) employs DiT-based latent diffusion.

Evaluation metrics. We employ multiple metric groups tailored to different aspects: a) *Image quality.* **FID** (Heusel et al., 2017) and **KID** (Bińkowski et al., 2018) measure distribution similarity in

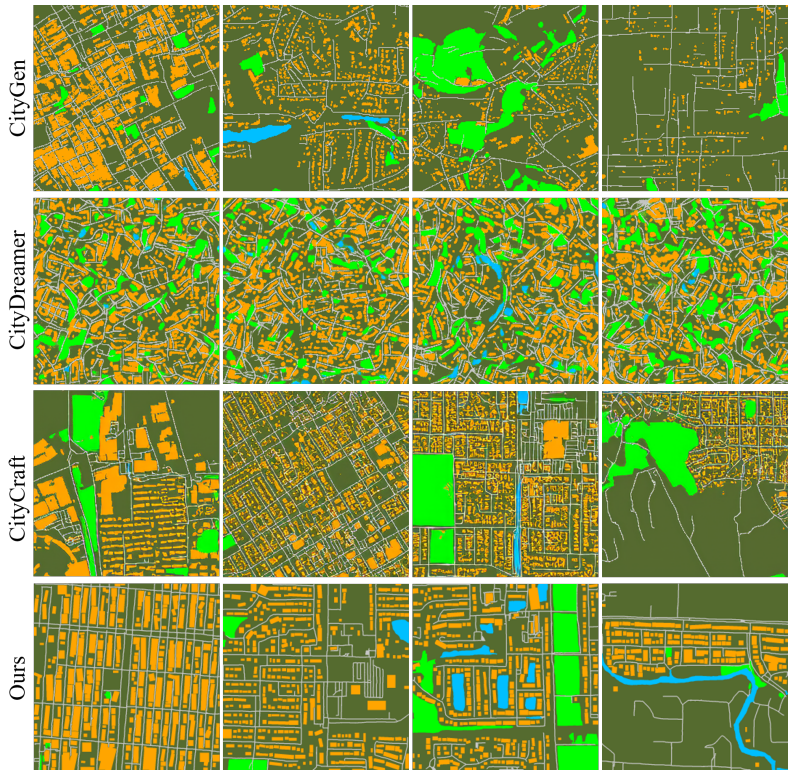

Figure 2: **Single-tile generation comparison.** Our method produces the most balanced spatial structure and coherent local details compared to baselines.

feature space. **CMMD** (Jayasumana et al., 2024) provides a more robust CLIP-based (Radford et al., 2021) assessment. b) *Image diversity.* **HI** (homogeneity index) computes average pairwise cosine similarity in Inception feature space (Shang et al., 2024). Lower values indicate higher diversity. c) *Structural quality.* Compared to RGB-image-based metrics, the statistics of semantic structures in a layout may provide more insights. For road networks, **Road L1** and **Road JS** measure L1 distance and Jensen-Shannon divergence (Lin, 2002) of endpoint degree distributions, while **Ang. EMD** computes Earth Mover's Distance (Rubner et al., 2000) for road orientations. **Bld. EMD** measures EMD for building area distributions. d) *Image-image similarity.* **CLIP-sim** computes average cosine similarity between CLIP embeddings of generated and ground-truth images. **Cat. L1** and **Cat. JS** measure category ratio differences via L1 distance and JS divergence. e) *Large-area consistency.* **Mismatch** is the fraction of adjacent labels that differ across seams. **Road F1** measures road-pixel agreement across seams with a 2-pixel matching tolerance. **Road LCC** is the fraction of road pixels in the largest 8-connected component. **Road (%)** reports the fraction of image pixels occupied by roads.

Implementation Details. We adopt a pretrained Stable Diffusion 3.5 (Esser et al., 2024) VAE with scaling factor r=8, yielding latent dimensions $\mathbb{R}^{16\times32\times32}$. The global DiT uses a standard DiT backbone paired with a ControlNet of matching depth, while the local DiT uses a smaller DiT-S variant for efficiency. We train the global DiT for 100 epochs and the local DiT for 300 epochs, both on 500k samples using 4 NVIDIA A100-SXM4-80GB GPUs. We use 250 denoising steps for global generation and 20 steps for local refinement. We apply SC-GSD on last 20 step at global stage, with overlap ratio $\rho = 0.5$. Qwen3.7-plus is used as the Condition Plan Agent with temperature 0.2. We set the ControlNet scale and conditional guidance coefficient to 0.6. $\lambda_{\text{cons}}, \lambda_{\text{ratio}}$ and λ_{road} are set to 0.2, 0.05, and 0.1, respectively.

Table 1: Single-tile generation across baselines and CityExpander variants on 10k test samples. Uncond supplies an all-zero ratio field to ControlNet. Full model uses zoning-ratio conditioning with c=0.6. Bold denotes the best.

Method	FID↓	KID↓	HI↓	CMMD↓	Road L1↓	Road JS↓	Ang. EMD↓	Bld. EMD↓
Baseline comparison								
CityCraft	40.21	0.033	0.658	44.52	0.173	0.008	0.661	15.63
CityDreamer	299.38	0.361	0.840	131.61	0.263	0.013	1.763	13.64
CityGen	117.63	0.122	0.683	56.67	0.171	0.006	0.980	36.58
CityExpander component variants								
Uncond.	39.37	0.036	0.668	22.06	0.106	0.002	0.267	8.10
w/o Global	165.72	0.188	0.778	50.53	0.168	0.005	0.693	**4.79**
w/o Local	46.62	0.034	0.698	43.27	0.201	0.010	1.464	23.64
Global+SR	45.50	0.032	0.690	42.34	0.153	0.009	1.738	21.52
Full	**32.60**	**0.024**	**0.654**	**20.50**	**0.081**	**0.001**	**0.151**	7.91

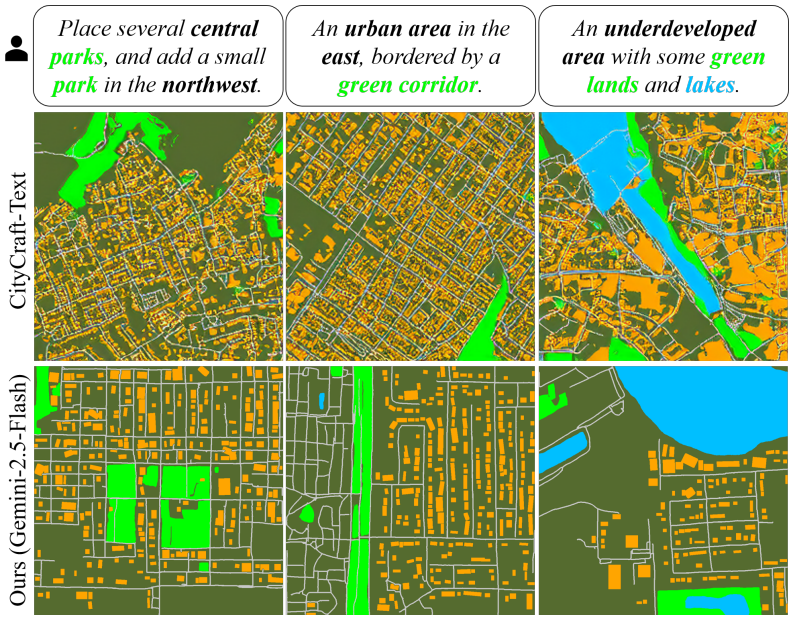

Figure 3: **Text-to-layout alignment comparison.** Compared with CityCraft's direct text embedding, the intermediate spatial-ratio representation improves localization of textual requirements in this single-scale study.

4.2 Comparative Studies

Single-tile generation quality. We first evaluate all methods on 10k test samples for single-tile generation. Table 1 consolidates the baseline comparison and component analysis. With an all-zero ratio field (Uncond.), CityExpander outperforms on most metrics, especially on those for structural quality, indicating closer agreement with real layouts in road-degree and orientation distributions. These results demonstrate not only better image generation but improved single-tile layout quality as well. The Full variant additionally uses test-derived zoning-ratio conditions, which achieves better generation. CityCraft produces reasonable overall layouts but suffers from local artifacts without dedicated local refinement. CityGen, operating directly in pixel space, struggles to capture complex spatial relationships and yields weaker metrics. CityDreamer, despite learning local patterns, generates repetitive structures and fails to produce coherent city-scale layouts. Qualitive results are illustrated in Figure 2.

Text-to-layout alignment. Next, we evaluate text-guided controllability against CityCraft on 1k human-validated text-layout pairs. To isolate the effect of the intermediate ratio representation from

Table 2: Text-to-layout alignment on 1k test pairs. Bold denotes the best.

Method	CLIP-sim↑	Cat. L1↓	Cat. JS↓
CityCraft-Text	0.895	0.315	0.039
Ours (Qwen)	0.908	0.255	0.027
Ours (Gemini)	**0.911**	**0.237**	**0.020**

Table 3: Stage-wise sampler comparison on 5k paired validation cases. Global outputs are evaluated at 640×640; local outputs at 2560×2560 with same global inputs. Road LCC and road coverage are measured over the whole image. Time measures stage sampling in seconds per case, excluding agent planning. Values are case means; bold denotes the best within each stage.

Sampler	Mismatch (%)↓	Road F1↑	LCC (%)↑	Road (%)	Time (s)↓
Global stage					
Sliding	**13.83**	0.7835	2.02	6.37	379.87
Vanilla GSD	15.82	0.7629	1.94	6.37	**95.29**
SC-GSD	14.99	**0.8643**	**2.27**	6.33	136.53
Local stage (same global inputs)					
Sliding	6.72	0.9512	41.63	7.78	113.81
Vanilla GSD	6.30	0.9553	45.38	7.79	**7.60**
SC-GSD	**5.07**	**0.9683**	**46.31**	7.77	104.69

scale-adaptive planning, Table 2 uses a fixed planning protocol and compares CityCraft's direct text conditioning with CityExpander using Qwen3.7-plus (Yang et al., 2025) or Gemini-2.5-Flash. For CityCraft, we use Gemini-2.5-Pro to annotate 50k training samples, while our method requires no additional diffusion-model training. Both CityExpander variants improve image-image similarity and spatial text alignment. Figure 3 illustrates that a spatial ratio field can localize abstract constraints (e.g., "underdeveloped") and directional requirements (e.g., "east") that a global text embedding may miss.

Condition agent robustness. We evaluate the Condition Plan Agent on 25 semantic prompt families, each instantiated at 1, 2, 3, and 4 km, for a total of 100 tasks. After bounded repair and case-level retry, all tasks at every scale produce compiler-valid raster conditions, yielding a 100% final success rate; 89 tasks succeed on the first attempt, nine on the second, and two on the third. Within the successful attempt, the average number of model calls increases from 5.92 at 1 km to 13.68 at 4 km, while the average plan size grows from 8.84 to 48.88 regions. This indicates that the agent maintains reliable execution across scales while allocating additional calls to the hierarchical refinement required by larger spatial extents.

4.3 Ablation Studies

Large-area expansion. We compare SC-GSD with Sliding and vanilla GSD, which averages overlapping noise predictions as in MultiDiffusion (Bar-Tal et al., 2023), on 5k validation maps spanning 2.5 km×2.5 km. Global samplers share reference-derived ratio conditions and noise; local samplers share the same SC-GSD global outputs, noise, and decoding settings. Table 3 isolates the effects at each stage. At the global stage, vanilla GSD produces higher seam mismatch and weaker road connectivity than Sliding, showing the limits of simple noise averaging. Adding explicit semantic constraints reduces mismatch by 5.3% relative to vanilla GSD and gives SC-GSD the highest Road F1 and LCC among all three methods, with similar road coverage. SC-GSD also uses 64.1% less sampling time than Sliding, although its seam mismatch remains higher. At the local stage, SC-GSD achieves the lowest seam mismatch and highest Road F1 and LCC. However, repeated VAE decoding and gradient computation add substantial overhead, leaving only an 8.0% time saving over Sliding. Vanilla GSD already produces competitive results at much lower cost (7.60 s versus 104.69 s for SC-GSD), therefore we use vanilla GSD for local stage in practice.

Multi-scale pipeline components. To assess the contribution of each component, we ablate the global stage and the local stage individually on 10k samples. We also test the effectiveness of our local model by replacing it with Real-ESRGAN (Wang et al., 2021), a commonly used super-

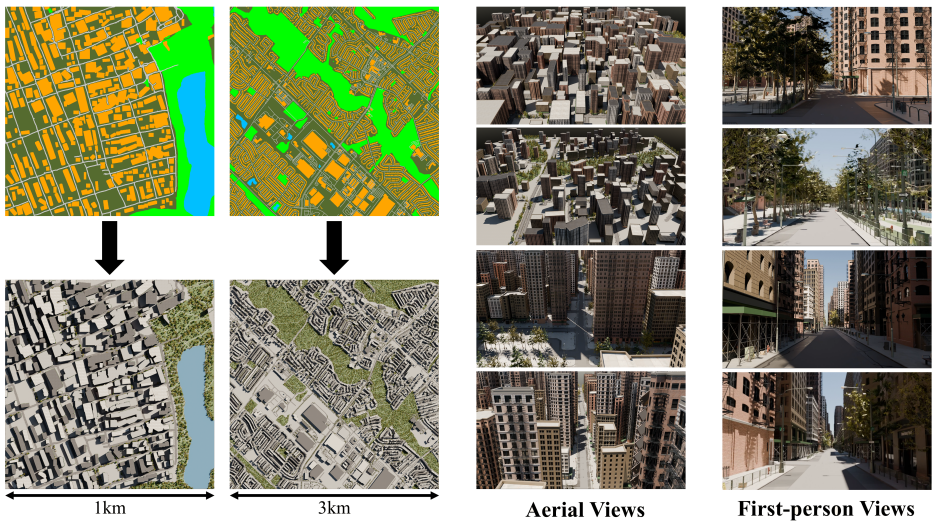

Figure 4: **3D city procedural generation** from our 2D layouts.

resolution (SR) model. The lower block of Table 1 consolidates these results. Removing local refinement (w/o Local) yields results similar to CityCraft, with noticeable local artifacts that degrade overall quality. Removing the global stage (w/o Global) and relying solely on unconditional local generation produces fragmented layouts. While this variant preserves fine building details well, it fails to establish kilometer-scale spatial coherence and align with long-range dependencies, mirroring the failure mode of CityDreamer. This suggests that repeatedly composing locally plausible patterns without global structural guidance can cause the generated layouts to deviate from the distribution of real cities. The SR variant (Global+SR) achieves only marginally better metrics than w/o Local and remains substantially inferior to the full pipeline, suggesting that our local DiT provides semantic and structural refinement beyond pixel-level enhancement. These results confirm that both global structure and local refinement are essential for high-quality layout generation.

4.4 Applications

To demonstrate practical applicability, we present 3D city scenes transformed from our generated layouts (Figure 4). Following diffusion-based generation, we apply post-processing to extract road networks and building/vegetation/water footprints. These elements are then fed into The City Generator (Dürr, 2025), a procedural generation add-on for Blender (Blender Online Community, 2025), which instantiates detailed 3D geometries and textures. The resulting 3D scenes preserve the spatial coherence and structural organization of the source 2D layouts, validating CityExpander's potential for large-scale city generation from text descriptions.

5 Conclusion and Future Work

We presented CityExpander, a text-guided framework for generating large-scale city layouts through spatial zoning-ratio conditions. Its Condition Plan Agent combines LLM-based proposal, deterministic compilation, and VLM-based review to translate language into executable, localized ratio fields. A global–local diffusion backbone then synthesizes kilometer-scale structure and fine-grained morphology, while SC-GSD synchronizes overlapping windows under spatially varying conditions and promotes semantic agreement across them, avoiding the scan-order dependency of sequential sliding-window inference.

Several limitations remain. First, our evaluation uses OSM-derived data with a fixed five-class taxonomy, and generalization to other geographic domains and land-use definitions remains to be established. Moreover, CityExpander constrains regional road occupancy but does not explicitly specify a road graph, so detailed topology is synthesized rather than guaranteed.

REFERENCES

Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, Sabine Süsstrunk, et al. Slic superpixels. Technical report, Technical report EPFL, 2010.

Rio Aguina-Kang, Maxim Gumin, Do Heon Han, Stewart Morris, Seung Jean Yoo, Aditya Ganeshan, R Kenny Jones, Qiuhong Anna Wei, Kailiang Fu, and Daniel Ritchie. Open-universe indoor scene generation using llm program synthesis and uncurated object databases. *arXiv preprint arXiv:2403.09675*, 2024.

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

Omer Bar-Tal, Lior Yariv, Yaron Lipman, and Tali Dekel. Multidiffusion: Fusing diffusion paths for controlled image generation. 2023.

Mikołaj Bińkowski, Danica J Sutherland, Michael Arbel, and Arthur Gretton. Demystifying mmd gans. *arXiv preprint arXiv:1801.01401*, 2018.

Blender Online Community. Blender - a 3d modelling and rendering package. `https://www.blender.org`, 2025. Version 4.4.

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

Ata Çelen, Guo Han, Konrad Schindler, Luc Van Gool, Iro Armeni, Anton Obukhov, and Xi Wang. I-design: Personalized llm interior designer. *arXiv preprint arXiv:2404.02838*, 2024.

Jie Deng, Wenhao Chai, Junsheng Huang, Zhonghan Zhao, Qixuan Huang, Mingyan Gao, Jianshu Guo, Shengyu Hao, Wenhao Hu, Jenq-Neng Hwang, et al. Citycraft: A real crafter for 3d city generation. *arXiv preprint arXiv:2406.04983*, 2024.

Jie Deng, Wenhao Chai, Jianshu Guo, Qixuan Huang, Junsheng Huang, Wenhao Hu, Shengyu Hao, Jenq-Neng Hwang, and Gaoang Wang. Citygen: Infinite and controllable city layout generation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 1995–2005, 2025a.

Wei Deng, Mengshi Qi, and Huadong Ma. Global-local tree search in vlms for 3d indoor scene generation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 8975–8984, 2025b.

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

Andreas Dürr. The city generator. `https://superhivemarket.com/products/the-city-generator`, 2025. Blender Add-on, version for Blender 4.3–4.5.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024.

Weixi Feng, Wanrong Zhu, Tsu-jui Fu, Varun Jampani, Arjun Akula, Xuehai He, Sugato Basu, Xin Eric Wang, and William Yang Wang. Layoutgpt: Compositional visual planning and generation with large language models. *Advances in Neural Information Processing Systems*, 36, 2024.

Rao Fu, Zehao Wen, Zichen Liu, and Srinath Sridhar. Anyhome: Open-vocabulary generation of structured and textured 3d homes. In *European Conference on Computer Vision*, pp. 52–70. Springer, 2024.

Chunchao Guo, Jinpeng Li, Yang Li, and Zilong Huang. Worldclaw: Agentic 3d open-world generation at scale. *arXiv preprint arXiv:2608.05248*, 2026.

Liu He and Daniel Aliaga. Globalmapper: Arbitrary-shaped urban layout generation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 454–464, 2023.

Liu He and Daniel Aliaga. Coho: Context-sensitive city-scale hierarchical urban layout generation. In *European Conference on Computer Vision*, pp. 1–18. Springer, 2024.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Zilong Huang, Jun He, Xiaobin Huang, Ziyi Xiong, Yang Luo, Junyan Ye, Weijia Li, Yiping Chen, and Ting Han. Majutsucity: Language-driven aesthetic-adaptive city generation with controllable 3d assets and layouts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 31608–31618, 2026.

Sadeep Jayasumana, Srikumar Ramalingam, Andreas Veit, Daniel Glasner, Ayan Chakrabarti, and Sanjiv Kumar. Rethinking fid: Towards a better evaluation metric for image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9307–9315, 2024.

Han-Hung Lee, Qinghong Han, and Angel X Chang. Nuiscene: Exploring efficient generation of unbounded outdoor scenes. *arXiv preprint arXiv:2503.16375*, 2025.

Yuseung Lee, Kunho Kim, Hyunjin Kim, and Minhyuk Sung. Syncdiffusion: Coherent montage via synchronized joint diffusions. *Advances in Neural Information Processing Systems*, 36:50648–50660, 2023.

Chieh Hubert Lin, Hsin-Ying Lee, Willi Menapace, Menglei Chai, Aliaksandr Siarohin, Ming-Hsuan Yang, and Sergey Tulyakov. Infinicity: Infinite-scale city synthesis. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 22808–22818, 2023.

Jianhua Lin. Divergence measures based on the shannon entropy. *IEEE Transactions on Information theory*, 37(1):145–151, 2002.

Lu Ling, Chen-Hsuan Lin, Tsung-Yi Lin, Yifan Ding, Yu Zeng, Yichen Sheng, Yunhao Ge, Ming-Yu Liu, Aniket Bera, and Zhaoshuo Li. Scenethesis: A language and vision agentic framework for 3d scene generation. *arXiv preprint arXiv:2505.02836*, 2025.

Jia-Hong Liu, Shao-Kui Zhang, Chuyue Zhang, and Song-Hai Zhang. Controllable procedural generation of landscapes. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pp. 6394–6403, 2024.

Yuheng Liu, Xinke Li, Yuning Zhang, Lu Qi, Xin Li, Wenping Wang, Chongshou Li, Xueting Li, and Ming-Hsuan Yang. Controllable 3d outdoor scene generation via scene graphs. *arXiv preprint arXiv:2503.07152*, 2025.

Keyang Lu, Sifan Zhou, Hongbin Xu, Gang Xu, Zhifei Yang, Yikai Wang, Zhen Xiao, Jieyi Long, and Ming Li. Yo'city: Personalized and boundless 3d realistic city scene generation via self-critic expansion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3219–3230, 2026.

Chong Mou, Xintao Wang, Liangbin Xie, Yanze Wu, Jian Zhang, Zhongang Qi, and Ying Shan. T2i-adapter: Learning adapters to dig out more controllable ability for text-to-image diffusion models. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pp. 4296–4304, 2024.

Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pp. 8162–8171. PMLR, 2021.

OpenStreetMap contributors. Planet dump retrieved from https://planet.osm.org . `https://www.openstreetmap.org`, 2017.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 4195–4205, 2023.

Yiming Qin, Nanxuan Zhao, Bin Sheng, and Rynson WH Lau. Text2city: One-stage text-driven urban layout regeneration. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pp. 4578–4586, 2024.

Steven Jige Quan. Urban-gan: An artificial intelligence-aided computation system for plural urban design. *Environment and Planning B: Urban Analytics and City Science*, 49(9):2500–2515, 2022.

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

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 1(2):3, 2022.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10684–10695, 2022.

Yossi Rubner, Carlo Tomasi, and Leonidas J Guibas. The earth mover's distance as a metric for image retrieval. *International journal of computer vision*, 40(2):99–121, 2000.

Yu Shang, Yuming Lin, Yu Zheng, Hangyu Fan, Jingtao Ding, Jie Feng, Jiansheng Chen, Li Tian, and Yong Li. Urbanworld: An urban world model for 3d city generation. *arXiv preprint arXiv:2407.11965*, 2024.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pp. 2256–2265. pmlr, 2015.

Fan-Yun Sun, Weiyu Liu, Siyi Gu, Dylan Lim, Goutam Bhat, Federico Tombari, Manling Li, Nick Haber, and Jiajun Wu. Layoutvlm: Differentiable optimization of 3d layout via vision-language models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 29469–29478, 2025.

Dilin Wang, Hyunyoung Jung, Tom Monnier, Kihyuk Sohn, Chuhang Zou, Xiaoyu Xiang, Yu-Ying Yeh, Di Liu, Zixuan Huang, Thu Nguyen-Phuoc, et al. Worldgen: From text to traversable and interactive 3d worlds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 27124–27135, 2026.

Xintao Wang, Liangbin Xie, Chao Dong, and Ying Shan. Real-esrgan: Training real-world blind super-resolution with pure synthetic data. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 1905–1914, 2021.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*, 2022.

Zhennan Wu, Yang Li, Han Yan, Taizhang Shang, Weixuan Sun, Senbo Wang, Ruikai Cui, Weizhe Liu, Hiroyuki Sato, Hongdong Li, et al. Blockfusion: Expandable 3d scene generation using latent tri-plane extrapolation. *ACM Transactions on Graphics (ToG)*, 43(4):1–17, 2024.

Zichen Xi, Zhihao Yao, Jiahui Huang, Zi-Qi Lu, Hongyu Yan, Tai-Jiang Mu, Zhigang Wang, and Qun-Ce Xu. Terracraft: City-scale generative procedural modeling with natural languages. *Graphical Models*, 141:101285, 2025.

Haozhe Xie, Zhaoxi Chen, Fangzhou Hong, and Ziwei Liu. Citydreamer: Compositional generative model of unbounded 3d cities. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9666–9675, 2024.

Haozhe Xie, Zhaoxi Chen, Fangzhou Hong, and Ziwei Liu. Generative gaussian splatting for unbounded 3d city generation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 6111–6120, 2025.

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.

Yixuan Yang, Junru Lu, Zixiang Zhao, Zhen Luo, James JQ Yu, Victor Sanchez, and Feng Zheng. Llplace: The 3d indoor scene layout generation and editing via large language model. *arXiv preprint arXiv:2406.03866*, 2024a.

Yue Yang, Fan-Yun Sun, Luca Weihs, Eli VanderBilt, Alvaro Herrasti, Winson Han, Jiajun Wu, Nick Haber, Ranjay Krishna, Lingjie Liu, et al. Holodeck: Language guided generation of 3d embodied ai environments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16227–16237, 2024b.

Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 3836–3847, 2023.

Shougao Zhang, Mengqi Zhou, Yuxi Wang, Chuanchen Luo, Rongyu Wang, Yiwei Li, Zhaoxiang Zhang, and Junran Peng. Cityx: Controllable procedural content generation for unbounded 3d cities. *arXiv preprint arXiv:2407.17572*, 2024.

Yunzhi Zhang, Zizhang Li, Matt Zhou, Shangzhe Wu, and Jiajun Wu. The scene language: Representing scenes with programs, words, and embeddings. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 24625–24634, 2025.

Mengqi Zhou, Yuxi Wang, Jun Hou, Shougao Zhang, Yiwei Li, Chuanchen Luo, Junran Peng, and Zhaoxiang Zhang. Scenex: Procedural controllable large-scale scene generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 10806–10814, 2025.

A LLM Planner Implementation

This section provides implementation details of the Condition Plan Agent introduced in Section 3.1. We use one planning prompt for both initial city planning and subsequent subdivision (Figure 5), with shared dataset guidance and examples. At each round, the LLM receives the current plan and decides whether to submit it or select a region for subdivision, returning the complete updated plan in either case. Figure 6 illustrates one refinement round. The deterministic compiler converts the submitted plan into a zone-ratio condition, and a separate reviewer VLM examines the compiled preview and proposes plan revisions (Figure 7).

Inference Configuration. We use Qwen3.7-plus with temperature 0.2. The initial planning round uses a thinking budget of at most 1000 tokens. For subsequent planning rounds and plan revision, we disable thinking for faster responses. The same planning prompt is reused across rounds, with subdivision depth limited to 2.

Dataset Guidance. We provide dataset statistics (archetypes and quantiles) and region-level text-to-ratio examples in the prompt to guide ratio assignments for different functional regions. The planner uses these examples together with the user requirements and explicit region semantics to assign each region a five-class land-use composition.

Iterative Hierarchical Planning. The LLM determines the physical canvas extent from the user text when it is not already specified and starts with one region covering the full canvas. Each round uses the latest plan and a schematic of its regions as spatial context. The LLM either submits the plan or selects a region and adds child regions to refine it. Every response contains the full

System Prompt: Iterative Hierarchical Planning

Task

You are an expert urban architect. Carefully plan a city with spatial regions and per-region class ratios according to the user requirements and city scale. Each region represents a functional urban zone (e.g., a residential district) with a specific land-use composition. Starting from one region covering the full canvas, inspect the current plan at each round and decide whether to submit it or select a region for further subdivision. Use the selected region's geometry, composition, and role in the global plan to guide its refinement.

Background Information

The city is represented as a semantic map with 5 land-use types: (0) background: undeveloped/open areas; (1) vegetation: parks/forests/green spaces; (2) building: residential/commercial/industrial structures; (3) road: streets/highways/transportation networks; (4) water: rivers/lakes/ponds.
Each region's "class_ratios" contains 5 nonnegative floats summing to 1, ordered as [bg, veg, bld, road, water]. "base_ratios" provides the composition for uncovered locations.

Input

{"user_intent": (the desired city layout), "city_bbox": ([length, width] in km, or null before inference), "current_plan": (the latest complete plan, or null on initialization), "max_depth": 2}
A schematic of the current plan is also supplied when a plan exists. Dataset anchors and text-to-ratio examples are supplied in the shared guidance below.

Output Format

Return a strict JSON object with a complete updated plan:
{"city_bbox": ([length, width] in km), "action": ("submit" or "subdivide"), "selected_region_id": (the region to subdivide, or null for submission),
"plan": {"base_ratios": ([bg, veg, bld, road, water]),
"regions": [
{"id": (unique region identifier), "semantic_label": (the region's function),
"geometry": {"type": "polygon", "polygon": (ordered vertex identifiers, e.g., "r6c6")},
"class_ratios": ([bg, veg, bld, road, water]), "parent": (parent identifier, or null for the root)},
{(region_2)}, ...]}}
The "plan" object represents $\mathcal{P} = (\mathbf{b}, \mathcal{R})$.

Figure 5: **Unified planning prompt template.** The same prompt initializes the city plan and guides subsequent refinement rounds. Every response specifies an action and returns the complete plan. Shared rules, anchors, and examples continue below.

plan $\mathcal{P} = (\mathbf{b}, \mathcal{R})$. Region records retain unique identifiers and parent references alongside their semantic labels, polygon vertices, and class ratios, following Eq. 1. The selected region's geometry, composition, and depth are obtained from these records and their parent links. The response also records the canvas extent, action, and selected region identifier; these describe the planning step rather than adding fields to $\mathcal{P}$.

Compilation and Plan Review. The compiler rasterizes the plan on a $G \times G$ lattice with $G = 32E$ and clips each child geometry to its ancestors. For each cell, it averages the land-use compositions of all covering regions and uses the default composition for uncovered cells, as described in Section 3.1. After a 3×3 box filter and simplex normalization, the four foreground channels form the diffusion condition. The compiler audits geometry, hierarchy, coverage, overlap, feature size, and deviation from any composition targets stated in the user requirements. A separate reviewer VLM jointly examines the original text, the five-class compiled preview, and these diagnostics to assess text–plan consistency. It accepts the plan or proposes bounded structured edits. Each revision is recompiled and retained only if it passes the deterministic audits; otherwise, the last valid plan is preserved. Review concerns the condition plan before diffusion sampling.

B Evaluation Metric Definitions

We provide detial definitions for the custom metrics used in the main paper and following parts.

Road L1 / Road JS / Ang. EMD / Bld. EMD. We extract road skeletons and building footprints from semantic layouts using connected component analysis. **Road L1** and **Road JS** measure L1 distance and Jensen-Shannon divergence of road endpoint degree distributions between generated and

System Prompt: Iterative Hierarchical Planning (continued)

Planning and Subdivision Rules

If "city_bbox" is null, infer the physical canvas extent from the user intent. If "current_plan" is null, initialize one root region covering the full canvas before deciding whether to submit or subdivide. Reuse the established canvas extent in later rounds.

For "subdivide", select a region below "max_depth" and return the full updated plan, including the selected region, its new children, and the other existing regions. Set each new child's "parent" to "selected_region_id". Preserve existing identifiers and use unique identifiers for new regions. For "submit", set "selected_region_id" to null and return the complete plan. Submit when no further subdivision is needed within the depth limit; do not return only a list of children or an empty list.

Use the parent's composition and geometry as guidance, together with the user intent and current plan. Keep children within their ancestors and preserve requested spatial relations. For example, if the parent's ratios are approximately [0.2, 0.7, 0.05, 0.03, 0.02], consider mainly large vegetation-dominant subregions. The root has depth 0; follow parent links to determine each region's depth.

Coordinates and Constraints

For the square canvas of side length E km, use $G = 32E$ cells per side. Encode polygon vertices as "r{row}c{column}", e.g., "r6c6". Vertex indices range from 0 to G, with "r0c0" at the northwest corner and "rGcG" at the southeast corner. Rows increase southward and columns eastward. List vertices in boundary order and use simple polygons within the canvas. Keep parent references valid and acyclic. The compiler averages the compositions of all regions covering each cell and uses "base_ratios" for uncovered cells.

Dataset Guidance

Typical region archetypes:
A0 (share={share0}%, veg∼{veg0}, bld∼{bld0}, ...),
A1 (share={share1}%, veg∼{veg1}, bld∼{bld1}, ...), ...
Quantiles (p10∼p50∼p90):
veg={veg_p10}∼{veg_p50}∼{veg_p90},
bld={bld_p10}∼{bld_p50}∼{bld_p90}, ...

Region-Level Text-to-Ratio Examples

- "dense CBD core" → [{bg}, {veg}, {bld}, {road}, {water}]
- "park with pond" → [{bg}, {veg}, {bld}, {road}, {water}]
- ...

Consider these examples when assigning ratios to each region. Do not simply follow them, but think thoroughly about the user requirements.

Figure 5: **Unified planning prompt template (continued).** These rules, dataset anchors, and text-to-ratio examples belong to the same prompt and are reused at every planning round.

real data. **Ang. EMD** computes the Earth Mover's Distance of road orientation angle distributions. **Bld. EMD** computes the EMD of building area distributions in m^2.

ACE (Average Category Error). Given a generated layout and a reference ratio vector $\boldsymbol{\alpha}^*$, we compute pixel-level category ratios $\hat{\boldsymbol{\alpha}}$ over the entire tile and report the L1 distance: $\text{ACE} = \frac{1}{5}\sum_{c=0}^{4}|\hat{\alpha}_c - \alpha_c^*|$.

zACE (Zone-level ACE). For each zone k in the zoning plan, we compute pixel-level ratios $\hat{\boldsymbol{\alpha}}_k$ within that zone's polygon and report the area-weighted L1 distance: $\text{zACE} = \sum_k \frac{A_k}{\sum_j A_j} \cdot \frac{1}{5}\sum_{c=0}^{4}|\hat{\alpha}_{k,c} - \alpha_{k,c}^*|$.

BC (Boundary Consistency). For adjacent tiles in a large-area canvas, we compute the mean squared error between pixels on either side of tile boundaries: $\text{BC} = \frac{1}{|\mathcal{B}|}\sum_{(i,j)\in\mathcal{B}}\|p_i - p_j\|^2$, where $\mathcal{B}$ is the set of cross-boundary pixel pairs and p denotes RGB values normalized to $[0, 1]$. Lower BC indicates fewer visible seams.

SC (Spatial Consistency). We extract Inception-v3 features for each tile in a multi-tile canvas and compute the average pairwise cosine similarity: $\text{SC} = \frac{2}{N(N-1)}\sum_{i<j}\cos(f_i, f_j)$. Higher SC indicates more coherent style across tiles.

Few-Shot Example: One Planning Round

Input

{“user_intent”: “Metropolitan city, divided by an east–west river along with a green belt immediately south of the river”, “city_bbox”: [3.0, 3.0], “max_depth”: 2,
“current_plan”: { “base_ratios”: [0.65, 0.02, 0.08, 0.25, 0.00],
“regions”: [{“id”: “city”, “semantic_label”: “urban area”,
“geometry”: {“type”: “polygon”, “polygon”: [“r0c0”, “r0c96”, “r96c96”, “r96c0”]},
“class_ratios”: [0.65, 0.02, 0.08, 0.25, 0.00], “parent”: null}]}}

The same dataset anchors, ratio examples, and current-plan schematic accompany this input.

Output

{“city_bbox”: [3.0, 3.0], “action”: “subdivide”, “selected_region_id”: “city”,
“plan”: {“base_ratios”: [0.65, 0.02, 0.08, 0.25, 0.00],
“regions”: [
{“id”: “city”, “semantic_label”: “urban area”,
“geometry”: {“type”: “polygon”, “polygon”: [“r0c0”, “r0c96”, “r96c96”, “r96c0”]},
“class_ratios”: [0.65, 0.02, 0.08, 0.25, 0.00], “parent”: null},
{“id”: “river”, “semantic_label”: “east–west river”,
“geometry”: {“type”: “polygon”, “polygon”: [“r27c0”, “r27c96”, “r30c96”, “r30c0”]},
“class_ratios”: [0.05, 0.05, 0.00, 0.00, 0.90], “parent”: “city”},
{“id”: “greenbelt”, “semantic_label”: “southern green belt”,
“geometry”: {“type”: “polygon”, “polygon”: [“r30c0”, “r30c96”, “r32c96”, “r32c0”]},
“class_ratios”: [0.00, 1.00, 0.00, 0.00, 0.00], “parent”: “city”}]}}

In a later round, the LLM can return “action”: “submit” with “selected_region_id”: null and the full plan if no further subdivision is needed.

Figure 6: **A few-shot example for iterative planning.** The LLM selects the city region, adds the river and green belt as its children, and returns the full updated plan. Polygon vertices are expressed on the 96×96 grid for the 3 km canvas.

Table 4: Intermediate representation comparison. Despite CityCraft-Ratio achieving slightly lower Cat. L1 through direct ratio conditioning, our grid representation achieves better perceptual quality and substantially better text-to-layout alignment.

Method	CLIP-sim↑	Cat. L1↓	CMMD↓
CityCraft-Ratio	0.903	**0.229**	38.01
Ours (Gemini)	**0.911**	0.237	**26.32**

C Extra Experiments

C.1 Comparison with CityCraft's Ratio Control

The main paper's Table 1 demonstrates that our method has superior generation quality over CityCraft's unconditional baseline, and Table 2 shows improved text-layout alignment over CityCraft's direct text conditioning. Since CityCraft also supports conditioning by ratio, here we present a comparison study against it under the text-to-layout task.

C.1.1 Experimental Setup

We use a similar setup as described in **Text-to-layout alignment** in main paper's Section 4.2. To fairly assess CityCraft's global ratio control and our ratio grid control, we compute the area-weighted average from our zoning plan as the global ratio vector for CityCraft's conditioning. We evaluate on 1k text-layout pairs using image-image similarity metrics (**CLIP-sim** and **Cat. L1**), a representative quality metric **CMMD**.

C.1.2 Results and Analysis

Table 4 reveals that CityCraft-Ratio achieves marginally lower category error, which is expected since it directly learns the conditioning from ratios to layouts. Our method outperforms across other

System Prompt: Plan Revision

Task
You are an expert urban architect reviewing a city condition plan. Examine the original user requirements, the five-class preview produced by the compiler, and its diagnostics. Assess whether the functional regions, spatial relations, and land-use compositions are consistent with the user intent. Accept the plan if no revision is needed; otherwise, propose structured edits to the plan.

Background Information
(Same five land-use types, ratio ordering, and plan fields as the unified planning prompt.) The preview represents the compiled condition plan before diffusion sampling.

Input
{"user_intent": (the original user text), "city_bbox": ([length, width] in km),
"current_plan": (base ratios and all region records),
"compiler_diagnostics": (geometry, hierarchy, coverage, overlap, feature size, and deviation from any user-specified composition targets)}
(The compiler's five-class condition preview is also provided as an image.)

Output Format
A strict JSON object containing the review decision and proposed edits:
{"decision": ("accept" or "revise"),
"edits": [{"region_id": (affected region), "updates": (the region fields to update and their new values)}, ...]}
Return an empty edit list for acceptance. For a revision, use the same semantic-label, polygon-geometry, class-ratio, and parent fields as the planning prompt. Use row–column vertex identifiers and preserve region identifiers.

Revision Rules
Review the preview and compiler diagnostics jointly with the original text. Adjust region geometry, composition, or hierarchy where needed to address a mismatch. Preserve portions of the plan that already satisfy the requirements. The proposed edits are applied to the structured plan and recompiled. A revision is retained only if it passes the deterministic audits; otherwise, the last valid plan is preserved. Keep revisions within the configured review budget.

Coordinates and Constraints & Guidance & Examples
(Same coordinate convention, composition constraints, dataset guidance, and region-level examples as the unified planning prompt.)

Figure 7: **Plan revision prompt template.** A separate reviewer VLM uses the text, compiled preview, and compiler diagnostics to accept the plan or propose structured edits for recompilation.

Table 5: Conditioning scale c ablation. We select c=0.6 as the default, which achieves strong ratio alignment (ACE/zACE) while maintaining high generation quality (FID/KID/CMMD).

c	FID↓	KID↓	CMMD↓	ACE↓	zACE↓
0.2	32.38	0.025	20.61	0.053	0.092
0.4	**32.15**	0.025	20.52	0.050	0.090
0.6	32.60	**0.024**	**20.50**	**0.046**	0.087
0.8	34.65	0.027	20.78	0.047	**0.086**
1.0	36.12	0.029	20.99	**0.046**	**0.086**

metrics with advantages in quality and text-to-layout alignment, suggesting that the ratio grid is much more informative for spatial conditioning compared to a single ratio vector.

C.2 Ablations for CFG Scale

We evaluate the CFG scale $c \in \{0.2, 0.4, 0.6, 0.8, 1.0\}$ for ratio conditioning on 10k samples (Table 5). While the differences are not significant, higher c values generally improve ratio alignment (lower ACE/zACE) but degrade perceptual quality (higher FID/CMMD). We select $c = 0.6$ as a moderate default for an optimal balance between controllability and naturalness.

Table 6: Overlap ratio ρ ablation on 2.5 km×2.5 km large-area generation. We select $\rho = 0.5$ as the default balance of quality and efficiency.

Strategy	FID↓	KID↓	CMMD↓	BC↓	SC↑
$\rho = 0.25$	38.41	0.032	50.88	0.054	0.752
$\rho = 0.5$	**35.90**	**0.030**	**46.04**	0.052	0.758
$\rho = 0.75$	36.38	**0.030**	47.13	**0.051**	**0.760**

C.3 Ablations for Overlap Ratio

Ablation for overlap ratio selection is shown in Table. 6, $\rho = 0.25$ achieves 52% inference time relative to $\rho = 0.5$ but sacrifices quality and consistency. Conversely, $\rho = 0.75$ requires 337% time with only marginal consistency gains. We therefore select $\rho = 0.5$ as the default.

D Dataset Construction Details

This section provides additional details on constructing our training dataset from OpenStreetMap (OSM) data.

D.1 Data Acquisition

Source. To ensure geographic diversity, we extract vector data covering diverse urban regions from multiple continents, including hundreds of major cities worldwide. These data are stored in .osm.pbf format.

Sampling Pipeline. For each OSM region file, we apply the following pipeline in Python for extraction:

1. Extract vector geometries (buildings, roads, water bodies, green areas) using the library *osmium*.
2. Project coordinates to a local UTM coordinate system (e.g., EPSG:326XX) for metric accuracy.
3. Generate candidate tiles uniformly on a regular grid with a target scale (e.g., 1 km).
4. Apply weighted sampling based on building density, where higher building counts yield higher sampling probabilities. This significantly increases the coverage of urban content and filters most of the undeveloped areas.
5. Apply a strict filtering strategy to the selected tiles and reserve the valid ones.
6. Rasterize the valid tiles into the target resolution (e.g., 1024 px) using the library *rasterio*. For potential overlapping elements, we use the following priority from high to low: building → road → water → vegetation.
7. If necessary, perform postprocessing such as downsampling and cropping.

Filtering Protocols. Filtering protocols could significantly affect the data quality. We apply two filters:

1. *Content coverage:* Divide each tile into a 16×16 grid of sub-blocks. Retain only tiles where 16×14 sub-blocks (≥87.5%) contain non-background content.
2. *Green area ratio:* Discard tiles where vegetation and water exceed 50% to avoid overly sparse or undeveloped layouts.

D.2 Dataset Statistics

Here, we report region archetypes as the representative dataset statistics for reference (Table 7). We extract typical functional region archetypes from the 500k training data using K-means clustering, where the K-means++ initialization determines $k = 6$ as the optimal via the elbow method. These archetypes are used to guide the LLM planner toward realistic ratio assignments.

Table 7: Region archetype centers with interpretations. Each archetype represents a typical functional zone pattern in real cities. Background ratio is implicit: $bg = 1 - veg - bld - road - water$.

ID	Share	Veg	Bld	Road	Water	Description
A0	25.1%	0.016	0.112	0.071	0.002	Sparse/underdeveloped
A1	23.7%	0.062	0.226	0.112	0.004	Typical residential
A2	23.0%	0.101	0.081	0.050	0.017	Mixed-use/suburban
A3	19.2%	0.811	0.014	0.027	0.006	Parks/green spaces
A4	7.3%	0.009	0.565	0.063	0.001	Commercial/industrial
A5	1.7%	0.032	0.007	0.012	0.792	Water bodies

E Training Details

E.1 CityExpander Training Configuration

Global DiT + ControlNet. The global DiT uses a standard DiT backbone (hidden=768, depth=12, 12 heads). The ControlNet mirrors the DiT encoder with an additional 4-channel input projection for the ratio grid. Both are trained jointly using AdamW optimizer (LR=1×10^{-4}, weight decay=0.01, batch=128) for 100 epochs on 4 NVIDIA A100-SXM4-80GB GPUs. We use 1000 diffusion steps during training with linear noise schedule and 250 steps at inference with DDPM sampling. The VAE (Stable Diffusion 3.5, frozen) compresses 256×256 RGB images to 16×32×32 latents. Classifier-free guidance dropout rate is 10%.

Local DiT-S. The local DiT uses the DiT-S architecture (hidden=384, depth=12, 6 heads, ∼33M parameters). It is trained on 0.25 km patches (256×256 crops from 1024×1024 full-resolution images) using AdamW (LR=1×10^{-4}, batch=256) for 300 epochs on 4 A100 GPUs. The structural condition $\mathbf{z}_c$ is constructed by encoding ground-truth images through the frozen VAE, decoding, and re-encoding to simulate the global stage's reconstruction artifacts. We use 1000 training diffusion steps but only 20 inference steps with DDIM sampling.

E.2 Baseline Reproduction Protocol

For fair comparison, all baselines are retrained on our dataset (500k samples, 1 km tiles at 256×256 resolution). We provide full training configurations below.

CityGen. CityGen's official codebase and weights are not publicly available. We reimplement it based on the paper description: a U-Net backbone (base channels=128, ∼19M parameters) operating in one-hot pixel space with 1000 diffusion steps. Training uses Adam optimizer (LR=2×10^{-5}, batch=24) for 200 epochs on 4 GPUs.

CityCraft. We use CityCraft's official codebase with its DiT-B architecture (hidden=768, depth=12, ∼130M parameters). Training uses AdamW (LR=1×10^{-4}, batch=128) for 300 epochs. For the text-conditioned variant (CityCraft-Text), we annotate 50k training samples using Gemini-2.5-Pro and fine-tune an additional text projection layer.

CityDreamer. We use CityDreamer's official pipeline: VQ-GAN encoder (∼75M) + absorbing diffusion sampler (∼77M) + GanCraft renderer (∼90M). The VQ-GAN is trained for 100 epochs, and the sampler uses 256 diffusion steps at inference. We evaluate only the 2D layout component (VQ-GAN + sampler) for consistency with our semantic map outputs.

F Additional Qualitative Results

This section presents extended qualitative examples to complement the main paper.

Single-Tile Generation Gallery. Figure 9 shows random results of our method on single-tile generation, with their top-down visualizations of corresponding 3D cities. In most cases, our method produces layouts with clear structures and details.

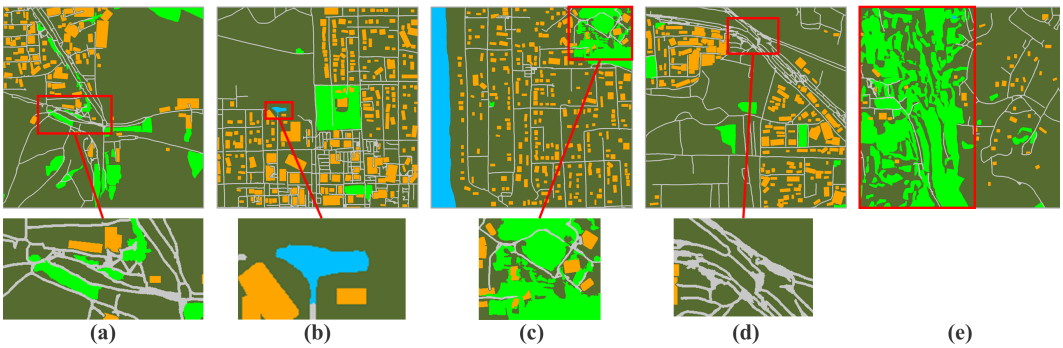

Figure 8: **Failure cases.** (a)&(d): Messy or disconnected road structure. (b): Illogical semantic connections. (c)&(e): Fragmented and bizarre vegetation layouts.

Text-to-Layout Examples. Figure 10 presents additional text-guided generation results, with several layouts generated from each prompt. These examples show our method's capability of generating diverse layouts while maintaining adherence to the text requirements, demonstrating the success of the zoning-ratio representation to bridge the LLM planning and the diffusion process.

Failure Cases. Figure 8 presents representative failure cases of our method. Common issues include messy or disconnected roads, fragmented vegetation, and other semantic errors. These failures stem from two main sources: (1) the zoning-ratio representation cannot enforce topological constraints such as road connectivity, and (2) the diffusion model occasionally generates implausible local patterns, particularly in mixed-use areas. Addressing these limitations through topology-aware conditioning or post-processing refinement is an important direction for future work.

G LLM USAGE ANNOUNCEMENT

We used large language models to assist with the writing and language polishing of this manuscript.

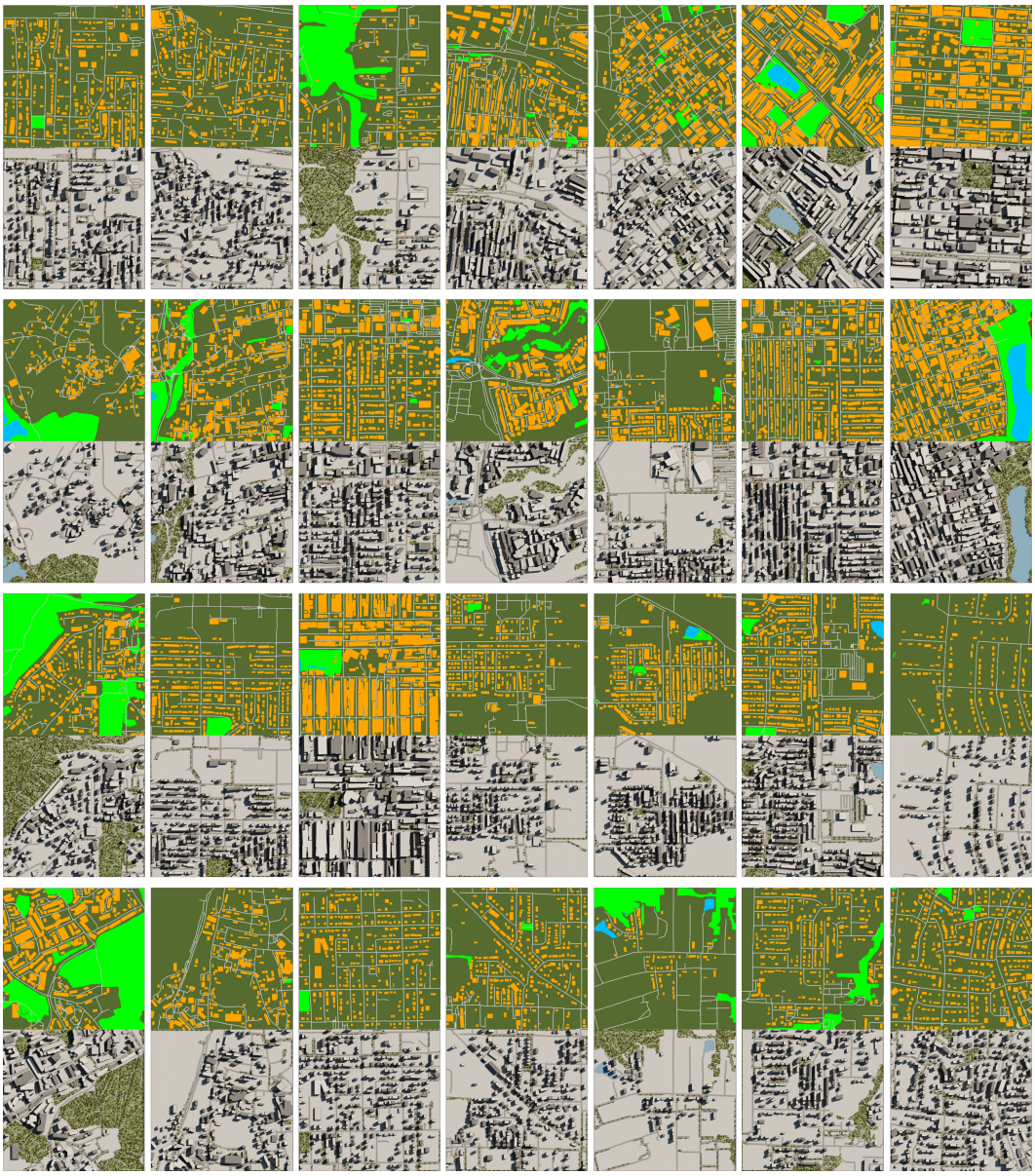

Figure 9: **Single-tile generation gallery.**



Figure 10: **Text-to-layout examples.**