SceneFrom3D: Geometry-Conditioned Outdoor 3D Scene Generation via View Scheduling with Object-Level Control

GEONUNG KIM, POSTECH, Republic of Korea
JEONGEUN PARK, POSTECH, Republic of Korea
NURI RYU, POSTECH, Republic of Korea
DI LIU, Meta Reality Labs, United States of America
SUNGHYUN CHO, POSTECH, Republic of Korea

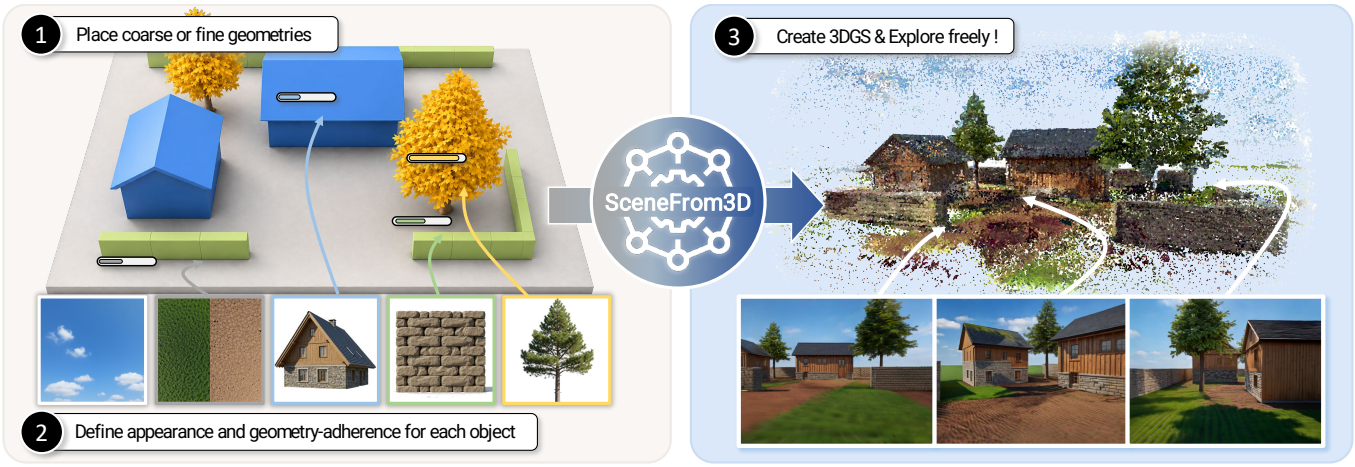


Fig. 1. Overview of SceneFrom3D. Given coarse or fine object geometries with per-object appearance and geometry-adherence conditions, SceneFrom3D generates a 3DGS scene that can be rendered and explored from arbitrary viewpoints.

Geometry-conditioned 3D scene generation enables the creation of 3D environments from user-provided geometry, offering direct control over scene structure and object layout. To generate such 3D scenes, current methods commonly adopt a three-stage design that first defines a view schedule, then synthesizes multi-view observations along the scheduled views, and finally reconstructs a 3D representation from the generated images. However, defining the view schedule becomes a major bottleneck for outdoor scenes, where large, unstructured, and unbounded geometry makes it difficult to obtain views that provide sufficient coverage while supporting stable generation. To address this bottleneck, we present SceneFrom3D, a framework that automatically schedules views from outdoor input geometries. SceneFrom3D constructs a directed generation graph whose nodes represent anchor views and whose edges represent interpolation trajectories, defining which views to synthesize, which view pairs to interpolate, and in which order generation should proceed. Beyond automatic view scheduling, SceneFrom3D further improves controllability through object-level conditioning, assigning each object an identity image for appearance guidance and a geometry-adherence parameter for region-wise control over the input geometry. Experiments demonstrate that SceneFrom3D achieves state-of-the-art geometry-conditioned outdoor 3D scene generation, producing high-quality scenes with controllable object appearance and geometry adherence.

Authors' Contact Information: Geonung Kim, POSTECH, Republic of Korea, k2woong92@postech.ac.kr; Jeongeun Park, POSTECH, Republic of Korea, koyy001@postech.ac.kr; Nuri Ryu, POSTECH, Republic of Korea, ryunuri@postech.ac.kr; Di Liu, Meta Reality Labs, United States of America, lsn33096@gmail.com; Sunghyun Cho, POSTECH, Republic of Korea, s.cho@postech.ac.kr.

1 Introduction

3D scene generation aims to synthesize a spatially consistent 3D representation of a scene, typically represented by Neural Radiance Fields (NeRF) [Mildenhall et al. 2021] or 3D Gaussian Splatting (3DGS) [Kerbl et al. 2023]. These representations allow rendering from arbitrary viewpoints while maintaining view consistency, enabling users to navigate and explore the generated scene freely. This capability makes them well-suited for immersive and interactive experiences, leading to broad applications in virtual reality, gaming, simulation, robotics, and autonomous driving.

Recent research in 3D scene generation has explored various input modalities, including text prompts [Höllein et al. 2023], images [Shriram et al. 2024], scene graphs [Liu et al. 2025], and geometric conditions [Schult et al. 2024]. Among these, geometric conditions provide explicit spatial priors for scene composition in the form of meshes or semantic layouts. Such conditions significantly reduce the ambiguity of the generated scene, leading to more reliable results while providing users with practical control over the generation process.

Since paired data between geometry and high-quality 3D scenes is scarce and difficult to acquire, recent state-of-the-art methods commonly decompose geometry-conditioned 3D scene generation into a three-stage pipeline [Fang et al. 2025; Kim et al. 2025; Schneider and Dai 2026; Yang et al. 2024]. Specifically, given geometric conditions as input, the first stage defines camera viewpoints or trajectories that cover the underlying scene. The second stage employs

diffusion models to synthesize geometry-conditioned multi-view observations. The final stage uses the synthesized observations to optimize a 3D representation, such as NeRF or 3DGS. By separating multi-view generation from 3D reconstruction, this pipeline can exploit strong image or video diffusion priors while avoiding the need to train a direct mapping from geometry to complete 3D scenes.

The success of this three-stage pipeline critically depends on the first stage of camera viewpoint definition. The selected viewpoints must provide sufficient coverage of the input geometry so that the generated observations support complete 3D reconstruction. At the same time, they must be suitable for diffusion-based generation, with an appropriate number of views and well-distributed viewing directions. Existing methods obtain such viewpoints under restrictive assumptions. Indoor methods exploit bounded and structured room layouts, where rule-based camera placement can be effective [Fang et al. 2025; Schneider and Dai 2026; Yang et al. 2024]. Other methods assume that camera trajectories are provided as input [Kim et al. 2025]. These assumptions are difficult to satisfy for arbitrary outdoor geometry, which often has unstructured layout, open boundaries, and no canonical camera range. Consequently, existing pipelines lack a general mechanism for automatically producing view schedules for arbitrary outdoor geometry.

In this paper, we propose SceneFrom3D, a geometry-conditioned framework for outdoor 3D scene generation that automatically produces the view schedule required by this three-stage pipeline. Given arbitrary input geometry, SceneFrom3D constructs a directed generation graph whose nodes represent anchor views and whose edges represent interpolation trajectories between anchor-view pairs. Our view scheduling algorithm first estimates a compact set of anchor views that covers the input geometry, then connects suitable view pairs for interpolation, and finally orients the graph to define generation dependencies. The graph determines which views to synthesize, which view pairs to interpolate, and in which order multi-view synthesis should proceed, enabling scalable generation in large and unstructured outdoor scenes without predefined camera paths.

In addition to view scheduling, SceneFrom3D also improves the controllability of generated scenes. Previous geometry-conditioned methods mainly use input geometry to control structural or semantic layout, but provide limited control over the appearance of individual objects and how closely each object follows its input geometry. SceneFrom3D addresses this limitation through object level conditioning. Each object is assigned an identity image to guide its appearance, while the strength of geometric conditioning is adjusted for each region to control geometry adherence. This design enables both flexible control over global scene structure and precise object-level appearance with adherence to the input geometry. Together, automatic view scheduling and object-level conditioning allow SceneFrom3D to generate outdoor 3DGS scenes from input geometries with object-level controllability, as shown in Fig. 1.

Our contributions are summarized as follows:

- We propose SceneFrom3D, the first geometry-conditioned framework for outdoor 3D scene generation that does not require explicit camera trajectories as input.
- We introduce an automatic view scheduling algorithm that selects anchor views, interpolation trajectories, and generation order from arbitrary outdoor scene geometry.
- We introduce an object-level conditioning mechanism that adds control over object specific appearance and region wise adherence to input geometry.
- Extensive experiments demonstrate that SceneFrom3D generates high quality 3D scenes in complex and unstructured outdoor environments.

2 Related Work

Geometry-guided 3D generation. Early geometry-guided 3D generation methods primarily focused on single objects using Score Distillation Sampling (SDS) [Chen et al. 2024, 2023; Dong et al. 2024; Metzer et al. 2023; Poole et al. 2022] or multi-view image generation under forward-facing view assumptions [Ryu et al. 2023; Shi et al. 2023; Wang et al. 2024b]. Subsequent works extended these approaches to scene-level generation conditioned on untextured meshes [Schneider and Dai 2026; Tang et al. 2023] or semantic layouts [Fang et al. 2025; Schult et al. 2024; Wang et al. 2025, 2024a; Yang et al. 2024]. However, many of these methods target indoor scenes, where bounded and structured layouts make heuristic camera placement effective. A few recent methods handle outdoor scenes [Kim et al. 2025; Lu et al. 2024; Yang et al. 2024], but they assume user-specified or externally provided camera trajectories. Designing such camera trajectories is difficult for large and unstructured scenes, since inefficient placement increases generation cost while incomplete coverage can leave important regions unobserved. In contrast, SceneFrom3D automatically schedules anchor views and interpolation trajectories from the input geometry, enabling scalable geometry-conditioned generation for outdoor scenes.

View scheduling. View scheduling has also been widely studied in robotics and vision, most notably as next-best-view (NBV) planning, which selects the most informative next viewpoint for tasks such as active object recognition [Dickinson et al. 1997; Roy et al. 2000], inspection [Tarbox and Gottschlich 1995; Trucco et al. 2002], and 3D reconstruction [Tarbox and Gottschlich 1995; Trucco et al. 2002]. Aerial path planning further extends this idea by optimizing camera trajectories and viewpoints for large-scale reconstruction using drones [Liu et al. 2022; Roberts et al. 2017; Smith et al. 2018; Tang et al. 2025; Zhang et al. 2021; Zhou et al. 2020]. Although these methods also determine viewpoints and trajectories from geometric information, their objectives and constraints differ from ours. They target reconstruction under physical acquisition constraints, where a camera must follow a single continuous path. In our setting, no physical camera traverses the scene, and generation can branch from any synthesized anchor view. A single global path is therefore inefficient, so SceneFrom3D formulates view scheduling as graph construction and plans multiple local interpolation trajectories tailored to geometry-conditioned outdoor 3D scene generation.

Object-level conditioning. Controllable generation has increasingly moved from global conditioning toward spatially localized guidance. In 2D image generation, early text-conditioned diffusion

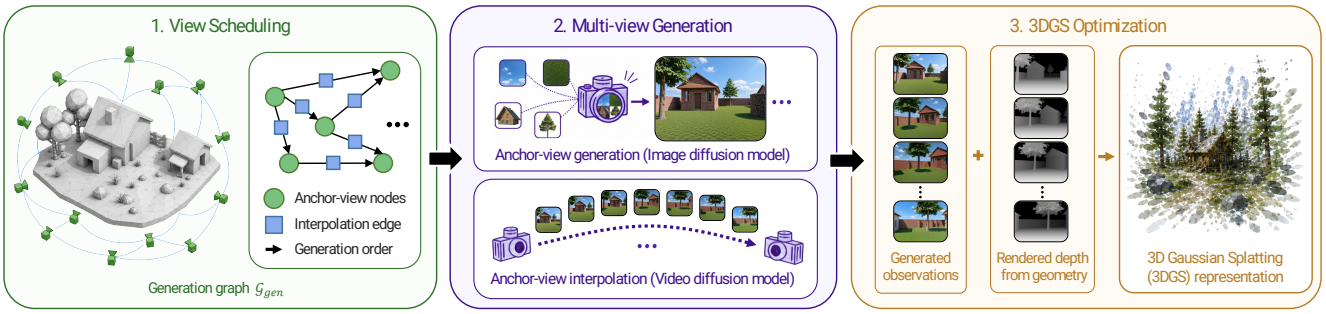


Fig. 2. Framework overview. SceneFrom3D first performs view scheduling from the input mesh to construct a directed generation graph. It then generates multi-view images by using identity images and geometry-adherence parameters for anchor-view generation and video diffusion for anchor-view interpolation. Finally, the generated posed observations and rendered depth maps from the input mesh are jointly used to optimize a 3DGS representation.

models [Rombach et al. 2022] were extended with structural controls such as depth, edges, and segmentation maps [Zhang et al. 2023], as well as region-level conditioning for localized editing and composition [Yang et al. 2023]. A similar trend appears in 3D scene generation, where text-driven methods [Höllein et al. 2023; Shriram et al. 2024] have been extended with scene-level structural priors such as layouts, semantic maps, and coarse geometry [Kim et al. 2025; Schult et al. 2024]. However, existing 3D methods mainly use such conditions to guide the global scene structure, leaving limited control over the appearance of individual objects and the degree to which each object should follow the input geometry. SceneFrom3D addresses this limitation through object-level conditioning, where each object is associated with an identity image for appearance guidance and a geometry-adherence parameter for region-wise control over the input geometry.

3 Methods

SceneFrom3D aims to generate an outdoor 3D scene from object-level geometry prompts. Let $O \equiv \{1, \ldots, N\}$ denote the set of object indices in the scene. The input is a set of attributed object meshes,

$$\mathcal{M} \equiv \{\mathbf{m}_o\}_{o \in O}, \qquad \mathbf{m}_o \equiv (M_o, I_o, \alpha_o), \tag{1}$$

where M_o denotes the input geometry of object o, I_o denotes an identity image specifying its target appearance, and $\alpha_o \in [0, 1]$ denotes a geometry-adherence parameter. Given $\mathcal{M}$, SceneFrom3D synthesizes a 3DGS representation of the output scene.

3.1 Overview

Fig. 2 illustrates the overall framework of SceneFrom3D. First, SceneFrom3D performs automatic view scheduling over the input geometry. The scheduler constructs a directed generation graph, where nodes represent anchor views and directed edges represent interpolation trajectories with generation dependencies. This graph determines the anchor views, interpolation pairs, and generation order.

Second, SceneFrom3D generates multi-view observations according to the generation graph. We follow the generation strategy of VideoFrom3D [Kim et al. 2025], which combines image-based anchor-view synthesis with video-based view interpolation for geometry-conditioned scene generation. Unlike VideoFrom3D, which relies on given camera trajectories, SceneFrom3D uses the generation graph to organize the same generation process. During anchor-view synthesis, rendered geometry, object identity images, and region-wise conditioning strengths guide appearance synthesis and adherence to the input geometry.

Finally, SceneFrom3D optimizes a 3DGS representation from the generated multi-view observations and mesh-rendered depth maps. The generated images provide posed appearance supervision, while the depth maps encourage alignment with the input geometry. We describe the three stages in detail in the following subsections.

3.2 View scheduling

The view-scheduling stage constructs a directed generation graph

$$\mathcal{G}_{\text{gen}} \equiv (\mathcal{V}, \mathcal{E}_{\rightarrow}), \tag{2}$$

where $\mathcal{V} \equiv \{v_i\}_{i=1}^{N_{\text{v}}}$ is the set of anchor-view nodes, v_i denotes the i-th anchor view, N_{v} is the number of anchor views, and $\mathcal{E}_{\rightarrow}$ is the set of directed interpolation edges without self-loops. Each edge $(v_i, v_k) \in \mathcal{E}_{\rightarrow}$ represents an interpolation trajectory from v_i to v_k and specifies that v_i is generated before v_k. View scheduling proceeds through node construction, edge construction, and direction construction, which we describe in order below.

3.2.1 Node construction. The objective of node construction is to find a compact anchor-view set $\mathcal{V}$ that broadly covers the input geometry with minimal cameras. This compactness is important because intermediate regions are later densely completed through video interpolation, so excessive anchor views provide limited benefit while increasing computation. In addition, as discussed in VideoFrom3D [Kim et al. 2025], placing anchor views too closely may worsen multi-view inconsistency, since stochastic diffusion sampling can produce conflicting details in shared observed regions.

To construct such a set, we define a visibility measure over sampled surface points of the input meshes to estimate how well candidate cameras observe the input geometry. Using this measure, we initialize anchor views through visibility-guided densification and then optimize their poses to improve coverage while preserving generation-friendly viewpoints. We next describe the visibility measure, node initialization, and pose optimization.

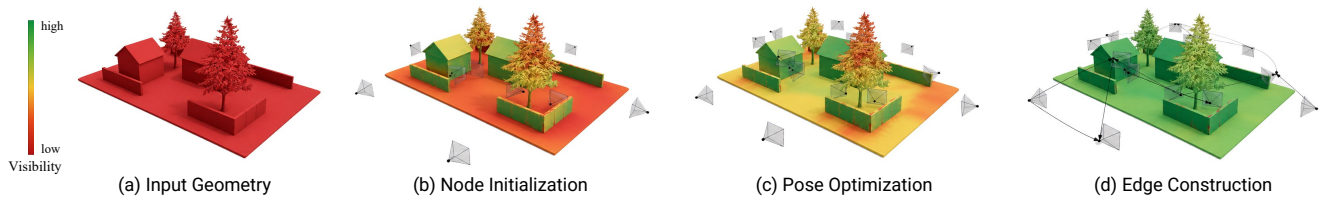


Fig. 3. Visualization of the view scheduling process. (a) The input geometry is color-coded by surface-point visibility. (b) Visibility-guided node initialization selects a compact set of anchor views that broadly cover the input geometry. (c) Pose optimization refines the anchor-view poses to improve both surface coverage and generation suitability. (d) Edge construction links suitable anchor-view pairs with interpolation trajectories.

Visibility measure. To evaluate how well the anchor views cover the input geometry, we uniformly sample surface points from the input meshes as $\mathcal{P} \equiv \{(\mathbf{p}_n, \mathbf{n}_n)\}_{n=1}^{N_\mathrm{p}}$, where $\mathbf{p}_n$ and $\mathbf{n}_n$ denote the 3D position and outward normal of the n-th sample, respectively, and N_p denotes the number of surface samples. We evaluate coverage by relating these samples to each anchor-view camera. Each anchor view $v_i \in \mathcal{V}$ is associated with a camera center $\mathbf{q}_i \in \mathbb{R}^3$ and a world-to-camera rotation $\mathbf{R}_i \in \mathrm{SO}(3)$, which together define the world-to-camera transform for view v_i.

For an anchor view v_i and a surface sample $\mathbf{p}_n$, we express the sample in the coordinate system of camera v_i as

$$(x_{i,n}^c, y_{i,n}^c, z_{i,n}^c)^\top \equiv \mathbf{R}_i(\mathbf{p}_n - \mathbf{q}_i), \tag{3}$$

where the superscript c denotes camera coordinates and the positive z^c axis points forward. We also define the camera-to-sample distance and ray direction as

$$r_{i,n} \equiv \|\mathbf{p}_n - \mathbf{q}_i\|_2, \qquad \boldsymbol{\omega}_{i,n} \equiv \frac{\mathbf{p}_n - \mathbf{q}_i}{r_{i,n}}. \tag{4}$$

Using these geometric quantities, we define the soft visibility score between anchor view v_i and surface sample $\mathbf{p}_n$ as

$$V_{i,n} = V_{i,n}^{\mathrm{fov}} \cdot V_{i,n}^{\mathrm{dist}} \cdot V_{i,n}^{\mathrm{front}} \cdot M_{i,n}^{\mathrm{occ}}. \tag{5}$$

This score measures whether a surface sample is useful for view scheduling by combining four criteria. The sample should lie inside the camera field of view, be observed from an appropriate distance, face the camera, and remain unoccluded by the input geometry.

The field-of-view term encourages samples to lie inside the camera frustum. Since all anchor views share fixed horizontal and vertical field-of-view angles, denoted by FOV_x and FOV_y, we first compute the angular offsets of each sample in camera coordinates as $\gamma_{i,n}^x \equiv \mathrm{atan2}(x_{i,n}^c, z_{i,n}^c)$ and $\gamma_{i,n}^y \equiv \mathrm{atan2}(y_{i,n}^c, z_{i,n}^c)$. We then define the field-of-view term as

$$V_{i,n}^{\mathrm{fov}} = \mathrm{sigmoid}\left(\beta\left(1 - \frac{2|\gamma_{i,n}^x|}{\mathrm{FOV}_x}\right)\right) \cdot \mathrm{sigmoid}\left(\beta\left(1 - \frac{2|\gamma_{i,n}^y|}{\mathrm{FOV}_y}\right)\right), \tag{6}$$

where β controls the sharpness of the frustum boundary.

The distance term favors a preferred viewing distance d_0,

$$V_{i,n}^{\mathrm{dist}} = \exp\left(-\frac{(r_{i,n} - d_0)^2}{\sigma_{\mathrm{dist}}^2}\right), \tag{7}$$

where σ_{dist} controls the tolerance around d_0.

The front-facing term favors surface samples whose normals face the camera,

$$V_{i,n}^{\mathrm{front}} = (1 - \lambda_{\mathrm{front}}) + \lambda_{\mathrm{front}} \max\big(0, -\langle \boldsymbol{\omega}_{i,n}, \mathbf{n}_n \rangle\big), \tag{8}$$

where λ_{front} controls the strength of the front-facing preference, and $\langle \cdot, \cdot \rangle$ denotes the inner product.

Finally, $M_{i,n}^{\mathrm{occ}} \in \{0, 1\}$ is a binary visibility mask that equals 1 when $z_{i,n}^c > 0$ and the segment from $\mathbf{q}_i$ to $\mathbf{p}_n$ is not occluded by the input geometry, and equals 0 otherwise.

Given the anchor-view set $\mathcal{V}$, we aggregate visibility for each surface sample by a soft union,

$$\bar{V}_n = 1 - \prod_{i=1}^{N_\mathrm{v}} (1 - V_{i,n}). \tag{9}$$

The aggregated score $\bar{V}_n$ becomes high when at least one anchor view observes the surface sample with high visibility.

Node initialization. This stage determines the number of anchor views and their initial camera poses, as shown in Fig. 3(b). Starting from an empty anchor-view set, we repeatedly sample a surface point $n^\star$ whose aggregated visibility is below the threshold, $\bar{V}_{n^\star} < \delta_{\mathrm{vis}}$, and add a new anchor view for it. The camera center is placed along its normal at the preferred distance d_0, i.e., $\mathbf{q}_{\mathrm{new}} = \mathbf{p}_{n^\star} + d_0 \mathbf{n}_{n^\star}$, with yaw and pitch set to look back toward $\mathbf{p}_{n^\star}$. We repeat this process until all surface samples satisfy the visibility threshold.

Since this process targets under-covered surface regions, it can add cameras with negligible visibility contribution or strong overlap with existing views. To obtain a compact initialization, we apply a refinement loop that removes low-contribution cameras, merges camera pairs with high shared visibility, and re-densifies remaining under-covered regions. We repeat this loop until the camera set sufficiently covers the input geometry without redundant views. Additional details are provided in the supplementary material.

Camera pose optimization. After initialization, we refine the anchor-view poses while fixing the view count. Specifically, we optimize the camera centers $\{\mathbf{q}_i\}_{i=1}^{N_\mathrm{v}}$ and rotations $\{\mathbf{R}_i\}_{i=1}^{N_\mathrm{v}}$ to improve coverage while maintaining generation-friendly viewpoints. We minimize

$$\mathcal{L}_{\mathrm{node}} = \mathcal{L}_{\mathrm{cov}} + \lambda_{\mathrm{rep}} \mathcal{L}_{\mathrm{rep}} + \lambda_{\mathrm{tilt}} \mathcal{L}_{\mathrm{tilt}}, \tag{10}$$

where λ_{rep} and λ_{tilt} are weighting coefficients.

The coverage loss encourages the anchor views to cover all surface samples by maximizing aggregated visibility,

$$\mathcal{L}_{\text{cov}} = -\frac{1}{N_{\text{p}}} \sum_{n=1}^{N_{\text{p}}} \bar{V}_n. \tag{11}$$

To prevent cameras from moving into the input geometry, we use a repulsion loss,

$$\mathcal{L}_{\text{rep}} = \sum_{i=1}^{N_{\text{v}}} \left[\max\left(0, d_{\text{safe}} - d(\mathbf{q}_i, \{M_o\}_{o\in O})\right)\right]^2, \tag{12}$$

where $d(\mathbf{q}_i, \{M_o\}_{o\in O})$ denotes the shortest distance from the camera center $\mathbf{q}_i$ to the input geometry, and d_{safe} is a safety margin.

We also regularize camera tilt to prevent cameras from being biased toward densely sampled ground regions. Let $\mathbf{f}_i$ denote the forward direction of camera v_i in world coordinates. Given the global up direction $\mathbf{g}$, we define a tilt loss,

$$\mathcal{L}_{\text{tilt}} = \frac{1}{N_{\text{v}}} \sum_{i=1}^{N_{\text{v}}} \langle \mathbf{f}_i, \mathbf{g} \rangle^2. \tag{13}$$

Together, the coverage, repulsion, and tilt terms guide the optimization toward anchor-view poses that comprehensively cover the input geometry, avoid entering the input geometry, and maintain viewing directions suitable for generation, as shown in Fig. 3(c).

3.2.2 Edge construction. After constructing the anchor-view nodes, we connect suitable anchor-view pairs so that video interpolation can provide denser multi-view supervision for 3DGS optimization. Since interpolation becomes unreliable when two anchor views are too far apart or observe largely different regions, we connect only pairs with non-negligible shared visibility and collision-free motion. Specifically, for each pair of distinct anchor views v_i and v_k, we compute the shared visibility score $S_{i,k} = \sum_{n=1}^{N_{\text{p}}} \min(V_{i,n}, V_{k,n})$. We connect them if $S_{i,k} > \delta_{\text{shared}}$ and the line segment between their camera centers is collision-free. For each connected pair of anchor views v_i and v_k, we store an interpolation trajectory $\Gamma_{i,k}$ that defines the camera path used to generate intermediate views. Details of the trajectory construction are provided in the supplementary material.

While the visibility-based criterion provides reliable interpolation edges, it can produce disjoint subgraphs or leaf nodes, potentially missing useful trajectories between nearby anchor views. To complement this, we add a small number of distance-based edges by connecting disjoint subgraphs through shortest paths and linking leaf nodes whose camera distance is below δ_{leaf}. This effectively recovers nearby connections that are beneficial but not captured by the visibility-based criterion.

3.2.3 Direction construction. The final step orients the connections. Specifically, we use the node indices as the generation order and orient each connection from the lower-index view to the higher-index view, yielding the directed edge set $\mathcal{E}_{\rightarrow}$ and the final generation graph $\mathcal{G}_{\text{gen}}$. Since all edges follow this order, the graph is acyclic, and topological sorting gives a valid anchor-view generation order. For each directed edge $(v_i, v_k) \in \mathcal{E}_{\rightarrow}$, view v_i is generated before v_k and can serve as a parent conditioning view to improve multi-view consistency during anchor-view synthesis.

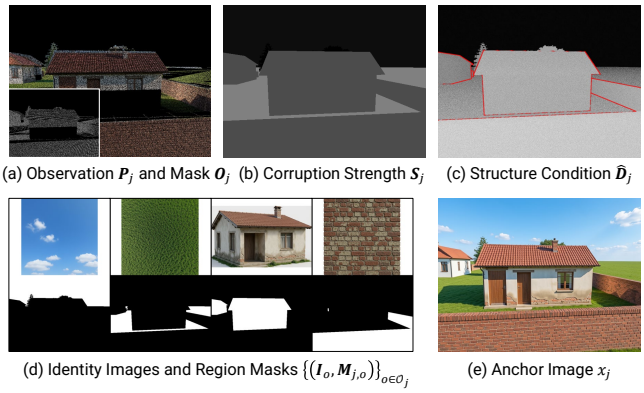


Fig. 4. Example inputs and output for anchor-view generation.

3.3 Multi-view generation

Given the generation graph $\mathcal{G}_{\text{gen}}$, the multi-view generation stage synthesizes posed observations for 3DGS training. The stage consists of anchor-view generation on graph nodes and view interpolation along graph edges. We describe these two steps in order.

3.3.1 Anchor-view generation. Following the topological order of $\mathcal{G}_{\text{gen}}$, we generate an anchor image for each node $v_j \in \mathcal{V}$. Let $O_j \subseteq O$ denote the set of objects visible from v_j. For each anchor view, we construct a view-specific conditioning input

$$C_j \equiv \Big(\mathbf{P}_j,\ \mathbf{O}_j,\ \hat{\mathbf{D}}_j,\ \mathbf{S}_j,\ \{(I_o, \mathbf{M}_{j,o})\}_{o\in O_j},\ \tau_j\Big), \tag{14}$$

where $\mathbf{P}_j$ is the partial observation, $\mathbf{O}_j$ is the observation mask, $\hat{\mathbf{D}}_j$ is the structure condition, $\mathbf{S}_j$ is the corruption strength map, $(I_o, \mathbf{M}_{j,o})$ pairs the identity image of object o with its visible-region mask, and τ_j is the text conditioning prompt. We feed C_j to the anchor-view generator to synthesize the anchor image $\mathbf{x}_j$, as visualized in Fig. 4.

We first construct the partial observation $\mathbf{P}_j$ and observation mask $\mathbf{O}_j$ from the generated parent views of v_j (Fig. 4(a)). For each parent view connected to v_j by an incoming edge, we warp its generated image into the target view using correspondences derived from input geometry and camera poses, then merge valid warped pixels to obtain $\mathbf{P}_j$. The mask $\mathbf{O}_j$ is defined by $\mathbf{O}_j(u) = 1$ if pixel u receives a valid warp from any parent view, and $\mathbf{O}_j(u) = 0$ otherwise.

We then construct the geometry-based conditions. Specifically, we render the input meshes at the target camera pose to obtain a clean depth map $\mathbf{D}_j$ and object masks $\{\mathbf{M}_{j,o}\}_{o\in O_j}$. From these masks and the geometry-adherence parameters $\{\alpha_o\}_{o\in O_j}$, we construct the corruption strength map $\mathbf{S}_j$ (Fig. 4(b)), setting the strength for object o according to $1 - \alpha_o$ so that larger α_o enforces stronger geometric adherence. We sample a Gaussian noise map $\boldsymbol{\epsilon}_j$ and form the corrupted depth map as $\tilde{\mathbf{D}}_j = (1 - \mathbf{S}_j) \odot \mathbf{D}_j + \mathbf{S}_j \odot \boldsymbol{\epsilon}_j$, where higher corruption weakens exact geometric guidance. Finally, we overlay boundary cues extracted from the object masks onto $\tilde{\mathbf{D}}_j$ to obtain the structure condition $\hat{\mathbf{D}}_j$ (Fig. 4(c)), helping align generated object boundaries with the input geometry and reduce warping errors in later partial-observation construction.

The identity-region conditions provide object-specific appearance guidance (Fig. 4(d)). For each visible object $o \in O_j$, the pair $(I_o, \mathbf{M}_{j,o})$

Fig. 5. Qualitative results across various scenarios. For each scenario, the figure shows multiple identity images, a 3D model, and rendered images from the camera viewpoints marked in the 3D model. Colored outlines indicate corresponding elements between the identity images and the 3D model. The labels A–D denote the camera viewpoints used to generate the rendered images.

specifies that the identity image I_o should guide the appearance of the image region indicated by $\mathbf{M}_{j,o}$. The text prompt τ_j summarizes the visible object semantics and global scene description based on a predefined rule. Given all conditioning inputs, the anchor-view generator produces the anchor image $\mathbf{x}_j$ (Fig. 4(e)).

In implementation, we convert the paired conditioning inputs $(\mathbf{P}_j, \mathbf{O}_j)$, $(\hat{\mathbf{D}}_j, \mathbf{S}_j)$, and $\{(I_o, \mathbf{M}_{j,o})\}_{o\in\mathcal{O}_j}$ into conditioning images through height-wise concatenation. The resulting conditioning images are then combined through image-token concatenation and fed to the generator. We construct a synthetic training dataset and use it to fine-tune the pretrained FLUX.2-klein-9B[1] diffusion model for this conditioning format. Additional details on dataset construction, training, and inference are provided in the supplementary material.

3.3.2 Anchor-view interpolation. After generating the anchor images, we densify the observation set along the directed edges of $\mathcal{G}_{\text{gen}}$. For each edge $(v_i, v_j) \in \mathcal{E}_{\rightarrow}$ with its associated trajectory $\Gamma_{i,j}$, we provide the endpoint anchor images $\mathbf{x}_i$ and $\mathbf{x}_j$, along with depth conditions rendered along $\Gamma_{i,j}$, to an off-the-shelf video diffusion model, VACE[2]. The model synthesizes an interpolation sequence between the two anchors, and the resulting frames provide dense posed observations for the subsequent 3DGS optimization.

3.4 3DGS optimization

Finally, we optimize a 3DGS scene from the generated posed observations. For each generated view $m \in \mathcal{J}$, let $\mathbf{I}_m$ be its RGB image and $\mathbf{D}_m^{\text{mesh}}$ be the metric depth rendered from the input meshes at the same pose. We optimize the 3DGS by minimizing

$$\mathcal{L}_{\text{3DGS}} = \frac{1}{|\mathcal{J}|}\sum_{m\in\mathcal{J}}\Big[\lambda_{\text{r}}\mathcal{L}_1(\tilde{\mathbf{I}}_m, \mathbf{I}_m) + \lambda_{\text{s}}\mathcal{L}_{\text{DSSIM}}(\tilde{\mathbf{I}}_m, \mathbf{I}_m) + \lambda_{\text{p}}\mathcal{L}_{\text{LPIPS}}(\tilde{\mathbf{I}}_m, \mathbf{I}_m) + \lambda_{\text{d}}\|\tilde{\mathbf{D}}_m - \mathbf{D}_m^{\text{mesh}}\|_1\Big], \tag{15}$$

where $\tilde{\mathbf{I}}_m$ and $\tilde{\mathbf{D}}_m$ are the RGB and depth rendered from the current 3DGS, and λ_{r}, λ_{s}, λ_{p}, and λ_{d} are loss weights. The RGB reconstruction and DSSIM losses encourage the optimized 3DGS to match the generated observations. The LPIPS term helps reduce perceptual inconsistencies inherited from the generative model [Gao et al. 2024], while the depth term encourages the reconstructed geometry to remain aligned with the input meshes [Schneider and Dai 2026].

4 Experiments

In this section, we evaluate SceneFrom3D through comprehensive experiments. Implementation details, including hyperparameters and thresholds, as well as additional comparisons, ablations, and analyses, are provided in the supplemental document.

4.1 Scene Generation Results

Fig. 5 shows scene generation results using SceneFrom3D across diverse geometry layouts, where each scene is guided by a varying set of identity images ranging from four to seven. The generated scenes faithfully follow the input 3D layouts, preserving the arrangement and shapes of scene elements. At the same time, SceneFrom3D successfully transfers appearance cues from the identity images to the corresponding elements, including sky, ground, trees, buildings, and walls. Moreover, the rendered images from cameras A–D further show that the generated 3DGS representations can be consistently rendered from arbitrary viewpoints that are not included in the generated observations, while maintaining high visual quality.

4.2 Baseline Comparisons

Our framework consists of three stages. To evaluate its effectiveness, we derive representative baselines by replacing individual stages with existing methods where applicable, enabling comparison across different generation paradigms. First, we compare to UrbanArchitect [Lu et al. 2024], which follows an SDS-based optimization [Poole et al. 2022] paradigm for scene generation. Since UrbanArchitect requires camera poses, we provide the camera poses obtained from our view scheduling algorithm. Second, we compare to YoNoSplat [Ye et al. 2025], which follows a feedforward reconstruction paradigm. For this baseline, we provide our generated multi-view images with their corresponding camera poses, from which YoNoSplat reconstructs a 3DGS representation. In addition, to evaluate the effectiveness of our view scheduling algorithm, we compare it with a drone path-planning method [Zhang et al. 2021]. This method is conceptually related to our task, as it also selects reconstruction-friendly viewpoints and computes trajectories from an input proxy mesh of the scene.

To measure visual quality, we use CLIP Aesthetic [LAION-AI 2023], and MUSIQ [Ke et al. 2021] on test frames. For each scene, we construct a continuous camera trajectory by using the anchor-view nodes as control points of a Bézier curve, resulting in approximately 400–700 test frames. These test frames do not overlap with the views used for optimization. For structural fidelity, we measure scale-invariant PSNR on depth maps (PSNR-D) [Kim et al. 2025]. We also report Chamfer Distance and F-score with a distance threshold of 0.1. To compute these metrics, we uniformly sample 100,000 points from the input mesh and compare them with the centers of the optimized Gaussian points. For UrbanArchitect, which is based on a NeRF representation, we first extract a density field, reconstruct a mesh using marching cubes, and then uniformly sample points from the reconstructed mesh.

For the test dataset, we construct 9 types of outdoor 3D scene layouts, consisting of either manually created coarse geometry or free assets. Each scene contains a varying number of objects, ranging from 9 to 16. From these scene models, we generate 10 different scenes for each baseline and use them for comparison.

Fig. 6 presents qualitative comparisons with the baseline methods. Since Zhang et al.'s method targets aerial drone path planning, the resulting views tend to be closer to top-down views, as shown in Fig. 9. When depth maps are rendered from coarse scene geometry under these viewpoints, object shapes and camera poses can become ambiguous, which often leads to view generation failures. Also, relying on such views degrades rendering quality at typical eye-level viewpoints. UrbanArchitect suffers from blurry appearances and missing textures, as it relies only on SDS-based optimization with 2D image priors for 3D scene generation. YoNoSplat is trained on real captured images and therefore suffers from a domain gap when applied to generated views. Moreover, its applicability is limited to

[1] https://huggingface.co/black-forest-labs/FLUX.2-klein-9B
[2] https://huggingface.co/Wan-AI/Wan2.1-VACE-14B

Fig. 6. Qualitative comparison with baseline methods, including Zhang et al.'s method [2021], UrbanArchitect [Lu et al. 2024], and YoNoSplat [Ye et al. 2025].

Fig. 7. Qualitative ablation comparison of our full model and its variants, including configurations without tilt loss $\mathcal{L}_{tilt}$, repulsion loss $\mathcal{L}_{rep}$, and anchor-view interpolation, respectively.

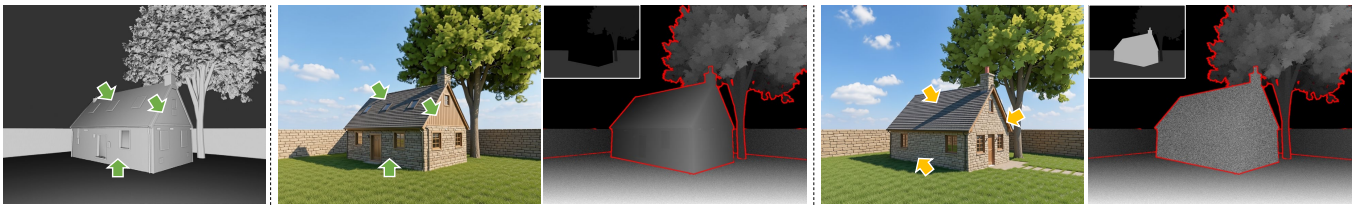
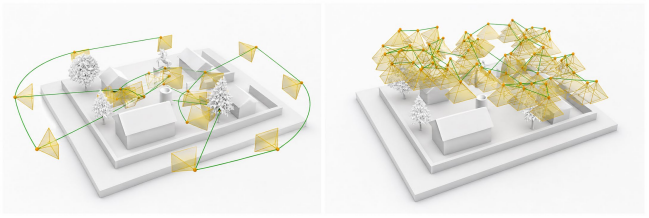


Fig. 8. Effect of geometry-adherence control. Given the same fine-detailed input geometry, larger α makes the generated object adhere more closely to the input structure map, while smaller α relaxes the geometry constraint.

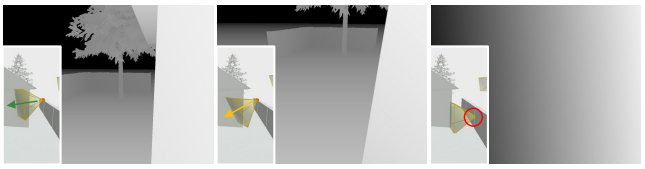
Table 1. Quantitative comparison with various baselines, including UrbanArchitect [Lu et al. 2024], YoNoSplat [Ye et al. 2025], and Zhang et al.'s method [2021], as well as ablation variants removing $\mathcal{L}_{\text{tilt}}$, $\mathcal{L}_{\text{rep}}$, or video interpolation. Bold and underline indicate the best and second-best scores, respectively.

Configuration			Visual Quality		Structural Fidelity		
View Scheduling	Multi-view Generation	3D Optimization	CLIP Aesthetic↑	MUSIQ↑	PSNR-D↑	Chamfer Distance↓	F-score↑
Our Module	UrbanArchitect		4.360	41.905	14.521	264.265	0.00001
Our Module	Our Module	YoNoSplat	3.413	39.096	15.796	46.664	0.00282
Zhang et al.	Our Module	Our Module	3.412	33.782	14.114	39.173	0.00990
W/o $\mathcal{L}_{\text{tilt}}$	Our Module	Our Module	5.066	45.454	20.647	<u>20.100</u>	0.01258
W/o $\mathcal{L}_{\text{rep}}$	Our Module	Our Module	<u>5.987</u>	49.588	<u>21.531</u>	21.061	<u>0.01398</u>
Our Module	W/o interpolation	Our Module	5.143	**59.653**	19.082	23.989	0.00831
Our Module	Our Module	Our Module	**6.194**	<u>54.474</u>	**21.974**	**19.255**	**0.01399**



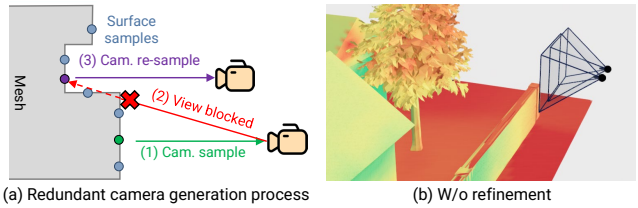
(a) Ours (b) Zhang et al.

Fig. 9. View scheduling comparison with Zhang et al.'s method [2021].



(a) W/ $\mathcal{L}_{tilt}$ & $\mathcal{L}_{rep}$ (b) W/o tilt loss $\mathcal{L}_{tilt}$ (c) W/o repulsion loss $\mathcal{L}_{rep}$

Fig. 10. Comparison of optimized camera poses and corresponding depth maps with and without the tilt and repulsion losses.



(a) Redundant camera generation process (b) W/o refinement

Fig. 11. Effect of the refinement loop in node initialization. (a) Illustration of how redundant cameras are generated when occluded views are not refined. (b) Node initialization result without the refinement loop.

fewer than 100 input views, making it less effective for reconstructing large outdoor scenes from hundreds of generated observations. Table 1 demonstrates the effectiveness of our method, showing that it outperforms the baselines across all metrics.

4.3 Ablation & Analysis

Ablation on tilt loss $\mathcal{L}_{\text{tilt}}$. Fig. 10(b) shows an example of camera pose optimization without the tilt loss. As indicated by the yellow arrow, the optimized camera direction is tilted toward the ground. This is because the ground mesh often accounts for a large fraction of the sampled surface points, which biases the optimized camera poses toward the ground plane. Consequently, the optimized views may fail to cover the full visible scene content, leading to incomplete appearances, as shown in Fig. 7(c).

Ablation on repulsion loss $\mathcal{L}_{\text{rep}}$. Fig. 10(c) shows an optimization example without the repulsion loss. As indicated by the red circle, the optimized cameras can move inside the input mesh. This results in invalid depth maps and collisions between camera paths and the input geometry, producing unreliable views and preventing these cameras from being connected to other nodes in the view graph. Consequently, some scene regions remain uncovered, leading to degraded reconstruction quality, as shown in Fig. 7(d).

Ablation on video interpolation. Fig. 7(e) shows the reconstruction result without video interpolation. In this setting, the available views are insufficient for stable 3DGS training, resulting in degraded rendering quality. Although this variant achieves the best scores on one visual quality metrics in Table 1, these scores are likely affected by high-frequency artifacts favored by the metrics.

Analysis on refinement loop. Fig. 11(a) illustrates how redundant cameras can arise during node initialization without the refinement loop. Specifically, an initialized camera is expected to cover nearby surface samples, but irregular input geometry can occlude even adjacent samples. The initialization process then treats these occluded samples as uncovered and generates additional cameras near the existing one. Since these cameras are spatially close, they often observe largely overlapping regions, leading to redundant cameras and poor scene coverage (Fig. 11(b)). Our refinement loop mitigates this issue by removing low-contribution cameras, merging cameras with similar visibility, and adding new cameras from under-covered surface samples that have not yet initialized cameras.

Analysis on geometry-adherence control. Fig. 8 shows how the geometry-adherence parameter α controls the degree of geometric guidance during anchor-view generation. Given detailed input geometry, setting α close to one makes the generated object closely follow the input structure map. As indicated by the green arrows in Fig. 8(b), fine geometric details such as the roof dormer and the positions of the door and windows are accurately reflected in the generated result. In contrast, decreasing α relaxes the geometric constraint by adding noise to the structure map. As indicated by the yellow arrows in Fig. 8(c), the result follows the overall shape of the house while allowing more flexible appearance generation and weaker adherence to local geometric details. This demonstrates that α provides controllable guidance between faithful geometry following and loose shape-level generation.

Fig. 12. Limitation of shadow consistency. The yellow arrows highlight shadows whose directions vary across views, leading to inconsistent appearance.

5 Conclusion

In this paper, we introduced SceneFrom3D, a geometry-conditioned framework for outdoor 3D scene generation without requiring explicit camera trajectories. By combining automatic view scheduling with an anchor-view-and-interpolation generation pipeline, our method enables high-quality 3DGS generation in large and unstructured outdoor scenes. As a first step toward object-level controllable 3D scene generation, our framework also provides control over object appearance and geometry adherence. Extensive experiments demonstrate its effectiveness across diverse scenarios.

Limitations. Anchor-view generation may fail when a single view contains more than eight distinct object identities, due to the limited number of input images that the pretrained model can effectively incorporate. In addition, the lack of an explicit prior for global illumination can occasionally result in variations in shadow direction and size across anchor views, leading to shadow inconsistencies in the 3DGS output, as shown in Fig. 12.

References

Hansheng Chen, Ruoxi Shi, Yulin Liu, Bokui Shen, Jiayuan Gu, Gordon Wetzstein, Hao Su, and Leonidas Guibas. 2024. Generic 3d diffusion adapter using controlled multi-view editing. *arXiv preprint arXiv:2403.12032* (2024).

Rui Chen, Yongwei Chen, Ningxin Jiao, and Kui Jia. 2023. Fantasia3d: Disentangling geometry and appearance for high-quality text-to-3d content creation. In *Proceedings of the IEEE/CVF international conference on computer vision*. 22246–22256.

Robert L Cook. 1986. Stochastic sampling in computer graphics. *ACM Transactions on Graphics (TOG)* 5, 1 (1986), 51–72.

Sven J Dickinson, Henrik I Christensen, John K Tsotsos, and Göran Olofsson. 1997. Active object recognition integrating attention and viewpoint control. *Computer vision and image understanding* 67, 3 (1997), 239–260.

Wenqi Dong, Bangbang Yang, Lin Ma, Xiao Liu, Liyuan Cui, Hujun Bao, Yuewen Ma, and Zhaopeng Cui. 2024. Coin3d: Controllable and interactive 3d assets generation with proxy-guided conditioning. In *ACM SIGGRAPH 2024 Conference Papers*. 1–10.

Chuan Fang, Heng Li, Yixun Liang, Jia Zheng, Yongsen Mao, Yuan Liu, Rui Tang, Zihan Zhou, and Ping Tan. 2025. Spatialgen: Layout-guided 3d indoor scene generation. *arXiv preprint arXiv:2509.14981* 3 (2025).

Ruiqi Gao, Aleksander Holynski, Philipp Henzler, Arthur Brussee, Ricardo Martin-Brualla, Pratul Srinivasan, Jonathan T Barron, and Ben Poole. 2024. Cat3d: Create anything in 3d with multi-view diffusion models. *arXiv preprint arXiv:2405.10314* (2024).

Lukas Höllein, Ang Cao, Andrew Owens, Justin Johnson, and Matthias Nießner. 2023. Text2room: Extracting textured 3d meshes from 2d text-to-image models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 7909–7920.

Junjie Ke, Qifei Wang, Yilin Wang, Peyman Milanfar, and Feng Yang. 2021. Musiq: Multi-scale image quality transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*. 5148–5157.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 2023. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.* 42, 4 (2023), 139–1.

Geonung Kim, Janghyeok Han, and Sunghyun Cho. 2025. VideoFrom3D: 3D Scene Video Generation via Complementary Image and Video Diffusion Models. In *Proceedings of the SIGGRAPH Asia 2025 Conference Papers (SA Conference Papers '25)*. Association for Computing Machinery, New York, NY, USA, Article 110, 11 pages. doi:10.1145/3757377.3763871

LAION-AI. 2023. Aesthetic Predictor. https://github.com/LAION-AI/aesthetic-predictor. Accessed: 2025-05-01.

Yuheng Liu, Xinke Li, Yuning Zhang, Lu Qi, Xin Li, Wenping Wang, Chongshou Li, Xueting Li, and Ming-Hsuan Yang. 2025. Controllable 3D outdoor scene generation via scene graphs. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 28052–28062.

Yilin Liu, Liqiang Lin, Yue Hu, Ke Xie, Chi-Wing Fu, Hao Zhang, and Hui Huang. 2022. Learning reconstructability for drone aerial path planning. *ACM Transactions on Graphics (TOG)* 41, 6 (2022), 1–17.

Fan Lu, Kwan-Yee Lin, Yan Xu, Hongsheng Li, Guang Chen, and Changjun Jiang. 2024. Urban architect: Steerable 3d urban scene generation with layout prior. *arXiv preprint arXiv:2404.06780* (2024).

Gal Metzer, Elad Richardson, Or Patashnik, Raja Giryes, and Daniel Cohen-Or. 2023. Latent-nerf for shape-guided generation of 3d shapes and textures. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 12663–12673.

Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. 2021. Nerf: Representing scenes as neural radiance fields for view synthesis. *Commun. ACM* 65, 1 (2021), 99–106.

Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. 2022. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988* (2022).

Mike Roberts, Debadeepta Dey, Anh Truong, Sudipta Sinha, Shital Shah, Ashish Kapoor, Pat Hanrahan, and Neel Joshi. 2017. Submodular trajectory optimization for aerial 3d scanning. In *Proceedings of the IEEE International Conference on Computer Vision*. 5324–5333.

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

Sumantra Dutta Roy, Santanu Chaudhury, and Subhashis Banerjee. 2000. Isolated 3D object recognition through next view planning. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans* 30, 1 (2000), 67–76.

Nuri Ryu, Minsu Gong, Geonung Kim, Joo-Haeng Lee, and Sunghyun Cho. 2023. 360° Reconstruction From a Single Image Using Space Carved Outpainting. In *SIGGRAPH Asia 2023 Conference Papers* (Sydney, NSW, Australia) *(SA '23)*. Association for Computing Machinery, New York, NY, USA, Article 75, 11 pages. doi:10.1145/3610548.3618240

Manuel-Andreas Schneider and Angela Dai. 2026. WorldMesh: Generating Navigable Multi-Room 3D Scenes via Mesh-Conditioned Image Diffusion. *arXiv preprint arXiv:2603.22972* (2026).

Jonas Schult, Sam Tsai, Lukas Höllein, Bichen Wu, Jialiang Wang, Chih-Yao Ma, Kunpeng Li, Xiaofang Wang, Felix Wimbauer, Zijian He, et al. 2024. Controlroom3d: Room generation using semantic proxy rooms. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 6201–6210.

Ruoxi Shi, Hansheng Chen, Zhuoyang Zhang, Minghua Liu, Chao Xu, Xinyue Wei, Linghao Chen, Chong Zeng, and Hao Su. 2023. Zero123++: a single image to consistent multi-view diffusion base model. *arXiv preprint arXiv:2310.15110* (2023).

Jaidev Shriram, Alex Trevithick, Lingjie Liu, and Ravi Ramamoorthi. 2024. Realmdreamer: Text-driven 3d scene generation with inpainting and depth diffusion. *arXiv preprint arXiv:2404.07199* (2024).

Neil Smith, Nils Moehrle, Michael Goesele, and Wolfgang Heidrich. 2018. Aerial path planning for urban scene reconstruction: A continuous optimization method and benchmark. (2018).

Luming Tang, Menglin Jia, Qianqian Wang, Cheng Perng Phoo, and Bharath Hariharan. 2023. Emergent correspondence from image diffusion. *Advances in neural information processing systems* 36 (2023), 1363–1389.

Mingfeng Tang, Ningna Wang, Ziyuan Xie, Jianwei Hu, Ke Xie, Xiaohu Guo, and Hui Huang. 2025. Aerial Path Online Planning for Urban Scene Updation. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*. 1–11.

Glenn H Tarbox and Susan N Gottschlich. 1995. Planning for complete sensor coverage in inspection. *Computer vision and image understanding* 61, 1 (1995), 84–111.

Emanuele Trucco, Manickam Umasuthan, Andrew M Wallace, and Vito Roberto. 2002. Model-based planning of optimal sensor placements for inspection. *IEEE Transactions on Robotics and Automation* 13, 2 (2002), 182–194.

Dilin Wang, Hyunyoung Jung, Tom Monnier, Kihyuk Sohn, Chuhang Zou, Xiaoyu Xiang, Yu-Ying Yeh, Di Liu, Zixuan Huang, Thu Nguyen-Phuoc, Yuchen Fan, Sergiu Oprea, Ziyan Wang, Roman Shapovalov, Nikolaos Sarafianos, Thibault Groueix, Antoine Toisoul, Prithviraj Dhar, Xiao Chu, Minghao Chen, Geon Yeong Park, Mahima Gupta, Yassir Azziz, Rakesh Ranjan, and Andrea Vedaldi. 2025. WorldGen: From Text to Traversable and Interactive 3D Worlds. arXiv:2511.16825 [cs.CV] https://arxiv.org/abs/2511.16825

Qi Wang, Ruijie Lu, Xudong Xu, Jingbo Wang, Michael Yu Wang, Bo Dai, Gang Zeng, and Dan Xu. 2024a. Roomtex: Texturing compositional indoor scenes via iterative inpainting. In *European Conference on Computer Vision*. Springer, 465–482.

Zhenwei Wang, Tengfei Wang, Zexin He, Gerhard Hancke, Ziwei Liu, and Rynson WH Lau. 2024b. Phidias: A generative model for creating 3d content from text, image, and 3d conditions with reference-augmented diffusion. *arXiv preprint arXiv:2409.11406*

(2024).

Xiuyu Yang, Yunze Man, Jun-Kun Chen, and Yu-Xiong Wang. 2024. SceneCraft: Layout-guided 3D scene generation. *Advances in Neural Information Processing Systems* 37 (2024), 82060–82084.

Zhengyuan Yang, Jianfeng Wang, Zhe Gan, Linjie Li, Kevin Lin, Chenfei Wu, Nan Duan, Zicheng Liu, Ce Liu, Michael Zeng, et al. 2023. Reco: Region-controlled text-to-image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 14246–14255.

Botao Ye, Boqi Chen, Haofei Xu, Daniel Barath, and Marc Pollefeys. 2025. YoNoSplat: You Only Need One Model for Feedforward 3D Gaussian Splatting. *arXiv preprint arXiv:2511.07321* (2025).

Han Zhang, Yucong Yao, Ke Xie, Chi-Wing Fu, Hao Zhang, and Hui Huang. 2021. Continuous aerial path planning for 3D urban scene reconstruction. *ACM Trans. Graph.* 40, 6 (2021), 225–1.

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

Xiaohui Zhou, Ke Xie, Kai Huang, Yilin Liu, Yang Zhou, Minglun Gong, and Hui Huang. 2020. Offsite aerial path planning for efficient urban scene reconstruction. *ACM Transactions on Graphics (TOG)* 39, 6 (2020), 1–16.

Supplemental Document

For a more detailed inspection of our results, please refer to the supplemental video. We will release all code and datasets upon acceptance. This supplementary material provides additional implementation details, statistics, and analyses for SceneFrom3D. It covers:

- Input Geometry and Ground Mesh (Section S1)
- View Scheduling Details (Section S2)
- Anchor-view Generation Details (Section S3)
- Anchor-view Interpolation Details (Section S4)
- 3DGS Optimization Details (Section S5)
- Scene Statistics & Latency (Section S6)
- Comparison of View Scheduling (Section S7)
- Ablation of Soft Visibility Score (Section S8)
- Ablation of 3DGS Training Loss (Section S9)

S1 Input Geometry and Ground Mesh

The input geometry consists of controllable object meshes placed on a ground mesh. We construct the ground mesh to be substantially larger than the object region, covering an area roughly ten times larger than the region occupied by the objects. This prevents the boundary of a small ground mesh from appearing in the depth maps during view generation, which can otherwise cause artifacts such as floating patches of ground in the sky.

S2 View Scheduling Details

The numerical parameters used by the main-paper view-scheduling formulation and the supplementary scheduling procedure are summarized in Table S1.

S2.1 Surface Samples and Filtering

We draw surface samples from the object meshes and from a bounded region of the ground mesh. Because the ground mesh is intentionally larger than the object region, sampling the entire ground mesh would dominate the sample set with uninformative empty areas. We therefore treat the XY bounding box of the object meshes as the region of interest and expand it by a fixed factor. Ground samples are drawn only from the part of the ground mesh inside this expanded region.

Surface samples are generated by area-weighted barycentric sampling on mesh triangles followed by 3D Poisson-disk filtering [Cook 1986], which keeps the accepted samples approximately uniform over the mesh surface. For a sampled surface area A and surface-sample spacing h, we draw approximately $\lceil A/h^2 \rceil$ candidate samples before filtering. The Poisson-disk minimum distance is set to the same spacing h.

After drawing surface samples, we remove invalid samples whose normals are immediately occluded. Because the input object arrangement is user-specified, meshes can be placed in arbitrary contact configurations; for example, the floor surface of a house may directly touch the ground mesh. Such contact regions can produce surface samples whose outward normal is immediately blocked by nearby geometry, making them uninformative for camera placement. We therefore apply a normal-clearance filter that removes surface samples whose outward normal immediately intersects nearby geometry.

ALGORITHM 1: Node initialization

Input: Surface samples $\mathcal{P} = \{(\mathbf{p}_n, \mathbf{n}_n)\}_{n=1}^{N_\mathrm{p}}$
Output: Initial anchor-view set $\mathcal{V}$
Stage 1: Identify camera-sampleable samples;
$\mathcal{A} \leftarrow \emptyset$; `// camera-sampleable sample indices`
for $n \leftarrow 1$ **to** N_p **do**
  $c_n \leftarrow$ candidate camera from $(\mathbf{p}_n, \mathbf{n}_n)$;
  if $|\mathrm{pitch}(c_n)| < \theta_{\max}$ *and* $H(c_n; \mathcal{P}) > \delta_{\mathrm{cand}}$ **then**
    $\mathcal{A} \leftarrow \mathcal{A} \cup \{n\}$;
  end
end
Stage 2: Progressive initialization;
$\mathcal{V} \leftarrow \emptyset$;
while $\{n \in \mathcal{A} \mid \bar{V}_n(\mathcal{V}) < \delta_{\mathrm{vis}}\} \neq \emptyset$ **do**
  Sample n^* from $\{n \in \mathcal{A} \mid \bar{V}_n(\mathcal{V}) < \delta_{\mathrm{vis}}\}$;
  $\mathcal{V} \leftarrow \mathcal{V} \cup \{c_{n^*}\}$;
  $\mathcal{A} \leftarrow \mathcal{A} \setminus \{n^*\}$;
end
Stage 3: Refinement;
repeat
  $\mathcal{V}_{\mathrm{prev}} \leftarrow \mathcal{V}$;
  $(\mathcal{V}, \mathcal{A}) \leftarrow \mathrm{Add}(\mathcal{V}, \mathcal{A}, \mathcal{P})$;
  $\mathcal{V} \leftarrow \mathrm{Remove}(\mathcal{V}, \mathcal{P})$;
  $\mathcal{V} \leftarrow \mathrm{Merge}(\mathcal{V}, \mathcal{P})$;
until $\mathcal{V} = \mathcal{V}_{\mathrm{prev}}$;
return $\mathcal{V}$;

S2.2 Node Initialization

Algorithm 1 summarizes our node initialization procedure, which constructs an anchor-view set from the filtered surface samples before continuous pose optimization. We use $\mathcal{P}$ for the filtered surface-sample set and $\mathcal{V}$ for the anchor-view set, following the main paper notation.

The first stage identifies the subset of samples that can be used to initialize cameras. We introduce $\mathcal{A} \subseteq \mathcal{P}$ as the camera-sampleable subset used only for selecting new camera locations. For each surface sample $(\mathbf{p}, \mathbf{n}) \in \mathcal{P}$, we place a candidate camera along the outward normal $\mathbf{n}$ at the preferred distance d_0, shortening the distance when geometry blocks the normal ray. We denote by $B_{c,n} \in \{0, 1\}$ the visibility mask that is one when $\mathbf{p}_n$ lies inside the frustum of camera c and is not occluded by the input geometry. The visibility contribution of camera c is

$$H(c; \mathcal{P}) = \sum_{n=1}^{N_\mathrm{p}} B_{c,n} V_{c,n}, \tag{16}$$

where $V_{c,n}$ is the soft visibility score defined in the main paper. The sample is added to $\mathcal{A}$ only if the candidate camera has pitch below $\theta_{\max}$ and $H(c; \mathcal{P})$ is above δ_{cand}. Samples outside $\mathcal{A}$ remain in $\mathcal{P}$ for coverage evaluation, but are not used to initialize cameras.

The second stage progressively constructs the initial anchor-view set. Starting from an empty camera set, we repeatedly compute the

aggregated visibility of all surface samples and add a camera from the under-covered samples in $\mathcal{A}$. The under-covered set is defined by the initialization visibility threshold δ_{vis}. The new camera is placed along the selected sample normal and oriented to look back at the sample. This progressive initialization stops when no eligible under-covered sample remains.

The progressive procedure provides a coverage-oriented initial set, but it does not explicitly optimize the compactness of the camera set. The third stage therefore refines the initialized camera set before continuous pose optimization. It adds cameras for still under-covered samples in $\mathcal{A}$, removes cameras with small visibility contribution, and merges camera pairs with similar visibility vectors. Merging uses cosine similarity between the visibility rows $(V_{i,n})_{n=1}^{N_{\text{p}}}$ and combines poses by a visibility-weighted mean. The refinement thresholds are denoted by δ_{dense}, δ_{remove}, and δ_{merge}. The refinement loop stops when the camera set no longer changes. After node initialization, the camera poses are continuously optimized using the objective defined in the main paper while keeping the number of cameras fixed.

S2.3 Edge Construction

For each edge selected by the scheduler, we store a smooth camera trajectory rather than only the two endpoint cameras. This is intended to mimic natural in-domain camera motion during video interpolation, where the camera usually moves along a gentle path around the scene instead of following a purely linear endpoint transition. For an accepted edge (v_i, v_k), let $\mathbf{q}_i$ and $\mathbf{q}_k$ be the endpoint camera centers, and let $\mathbf{f}_i$ and $\mathbf{f}_k$ be their forward directions. We define the trajectory $\Gamma_{i,k}$ by a cubic Bezier curve $\mathbf{q}(t)$, $t \in [0,1]$, whose endpoints are $\mathbf{q}_i$ and $\mathbf{q}_k$. The two interior control points are initialized along the chord between the endpoints and then offset away from a common focus region estimated from the endpoint viewing rays. The offset increases with the angular difference between $\mathbf{f}_i$ and $\mathbf{f}_k$, while being capped relative to the endpoint distance. Thus, small-baseline edges remain close to a straight path, whereas wider-baseline edges follow a smoother arc around the scene. If the sampled curve intersects the mesh, we fall back to the straight endpoint segment.

S2.4 Direction Construction and Generation Order

Let $\mathcal{E}$ denote the undirected edge set over the anchor views. To obtain a directed acyclic generation graph, we assign each anchor view a canonical order index $\rho(v_i) \in \{1, \dots, N_{\text{v}}\}$ and orient every undirected edge from the lower-order endpoint to the higher-order endpoint:

$$\mathcal{E}_{\rightarrow} = \{(v_i, v_k) \mid \{v_i, v_k\} \in \mathcal{E}, \rho(v_i) < \rho(v_k)\}. \tag{17}$$

Because ρ strictly increases along every directed edge, the resulting graph $\mathcal{G}_{\text{gen}} = (\mathcal{V}, \mathcal{E}_{\rightarrow})$ is a DAG. The anchor-view generation order is then any topological ordering of $\mathcal{G}_{\text{gen}}$. For a target anchor view v_k, its parent views are $\text{Pa}(v_k) = \{v_i \mid (v_i, v_k) \in \mathcal{E}_{\rightarrow}\}$ and are generated before v_k.

Table S1. View-scheduling parameter index.

Notation	Name	Value
h	Surface-sample spacing	1.0
d_0	Preferred camera distance	35.0
$\theta_{\max}$	Candidate camera pitch limit	10 degrees
δ_{cand}	Candidate visibility threshold	32.0
δ_{vis}	Initialization visibility threshold	0.001
δ_{dense}	Densification visibility threshold	0.1
δ_{remove}	Removal visibility threshold	80.0
δ_{merge}	Merge similarity threshold	0.1
δ_{shared}	Shared visibility threshold	0.007
β	Frustum sigmoid sharpness	5.0
λ_{front}	Front-facing weight	1.0
d_{safe}	Camera-mesh safety margin	6.0
λ_{rep}	Camera-mesh repulsion weight	1.0
λ_{tilt}	Pitch regularization weight	1.0

S3 Anchor-view Generation Details

To enable the anchor-view generation conditioning used by SceneFrom3D, we synthesize a task-specific training dataset and fine-tune a pretrained diffusion model. We describe the dataset synthesis process and the training setup below.

S3.1 Dataset Synthesis

Training requires paired examples that match the anchor-view generation input defined in the main paper. Using the same notation, each synthetic training example consists of a target anchor image $\mathbf{x}_j$ and its conditioning tuple

$$C_j = \Big(\mathbf{P}_j,\ \mathbf{O}_j,\ \hat{\mathbf{D}}_j,\ \mathbf{S}_j,\ \{(I_o, \mathbf{M}_{j,o})\}_{o \in O_j},\ \tau_j\Big). \tag{18}$$

The following paragraphs describe how each component of $(\mathbf{x}_j, C_j)$ is synthesized.

S3.1.1 Identity Images & Anchor Image. Fig. S1 summarizes the dataset synthesis pipeline. We first build an identity-image bank $\{I_o\}$ for the scene domain. For each identity, we first generate an object-specific identity-image text prompt using Qwen3-8B[3], so the prompt describes the object category together with appearance attributes such as material, structure, and visual style. For each object o, we use the FLUX.2 Klein base 9B[4] image diffusion model to generate an isolated identity image I_o on a clean background, with prompts that emphasize a single centered object, realistic materials, and no surrounding scene clutter. These identity images provide reusable appearance references for architecture and environment objects, while separate sky and ground images provide the background appearance references. In total, the identity bank contains 349 identity images across 22 identity categories.

We then use the same FLUX.2 Klein model to synthesize full outdoor anchor images from the identity bank. This synthesis is possible because FLUX.2 Klein supports multi-reference image generation: a single target anchor image can be conditioned on sky, ground, and multiple object references at once. Each synthetic case

[3] https://huggingface.co/Qwen/Qwen3-8B
[4] https://huggingface.co/black-forest-labs/FLUX.2-klein-9B

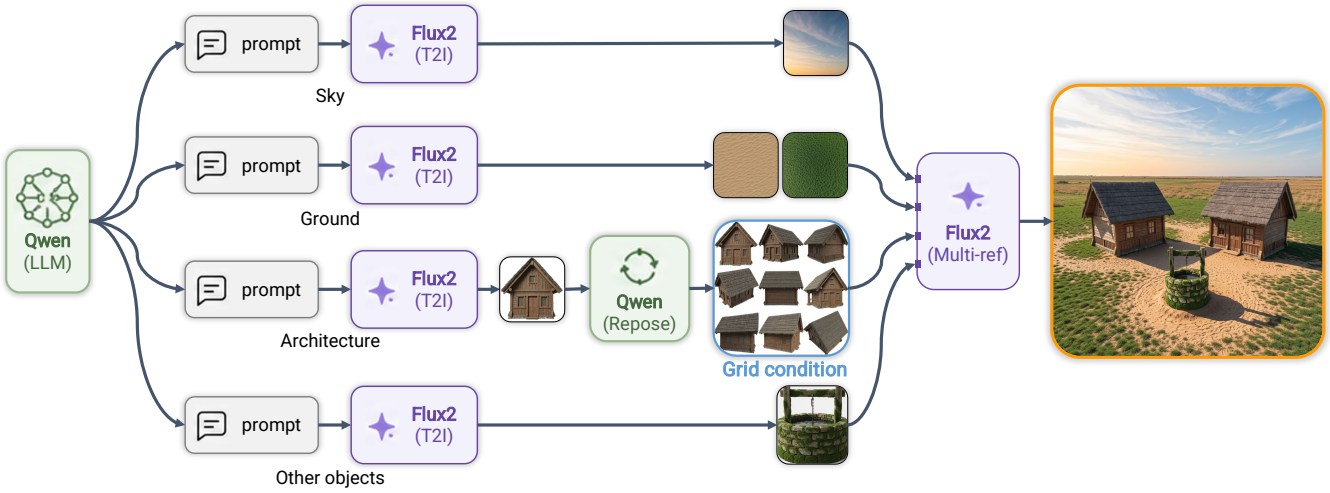


Fig. S1. Dataset synthesis pipeline for identity images and anchor images. Object-specific identity prompts are generated first, then used to synthesize reusable identity images. For architecture components, we use a grid condition of multiple identity views to reduce pose bias. The resulting identity conditions are composed with sky and ground references to create multi-reference anchor-image training pairs.

(a) W/o grid architecture condition

(b) W/ grid architecture condition

Fig. S2. Motivation for grid conditioning of architecture components. The anchor-image synthesis model shows strong pose bias for buildings when conditioned on a single architecture reference. Under the same conditioning with different random seeds, a grid of multiple identity views exposes the generator to the same component from different viewpoints and reduces this bias. We use this grid condition only for architecture components.

samples a sky reference, one or two ground references, a camera-height description, and a visible object set O_j with identity images $\{I_o\}_{o \in O_j}$. The scene-generation prompt specifies the role of each input image, asks the model to preserve the referenced object identities and object counts, and requests one coherent full-frame outdoor photograph rather than a collage or contact sheet. The result is the target anchor image $\mathbf{x}_j$, paired with metadata that records the semantic assignment of each reference image and is later used to construct the text prompt τ_j. This process yields approximately 17K identity-to-anchor-view training pairs.

For architecture components only, we use Qwen-Image-Edit-2511 with a multi-angle LoRA to generate multiple posed variants of the same identity and arrange them as a 3-by-3 grid condition. We apply this grid conditioning only to architecture components because the anchor-image synthesis model exhibits a particularly strong pose bias for buildings: when given a single building reference, it tends to reproduce the reference viewpoint rather than adapting the component to the target scene. As illustrated in Fig. S2, under the same conditioning but with different random seeds, the grid condition exposes the generator to the same architecture component from several viewpoints, while ordinary environment objects continue to use a single identity reference.

S3.1.2 Semantic Mask. From each synthetic anchor image $\mathbf{x}_j$, we estimate semantic masks for the visible sky, ground, and object regions. We use Grounded-SAM2[5] for this step. For every visible object $o \in O_j$, the mask $\mathbf{M}_{j,o}$ indicates the image region occupied by that object in $\mathbf{x}_j$. The identity-region conditions are therefore the paired inputs $\{(I_o, \mathbf{M}_{j,o})\}_{o \in O_j}$, so the generator receives both the object appearance and the region where that appearance should be expressed.

S3.1.3 Partial Observation. The pair $(\mathbf{P}_j, \mathbf{O}_j)$ represents the sparse RGB evidence available during sequential anchor-view generation. During training, we obtain it by applying random pattern masks to the target anchor image $\mathbf{x}_j$, such as stripe-like masks or block-shaped masks, and then dropping observed pixels with Bernoulli noise thresholding. The observed pixels form the partial observation $\mathbf{P}_j$, and the binary observation mask $\mathbf{O}_j$ records which pixels are retained. For no-observation training cases, both $\mathbf{P}_j$ and $\mathbf{O}_j$ are set to zero.

[5] https://github.com/IDEA-Research/Grounded-SAM-2

S3.1.4 Structure Condition. The pair $(\hat{\mathbf{D}}_j, \mathbf{S}_j)$ provides approximate geometric guidance without requiring exact target appearance. For each synthetic anchor image, we estimate a clean depth map $\mathbf{D}_j$ with MoGe-2[6] and construct a per-pixel corruption strength map $\mathbf{S}_j$. We then corrupt $\mathbf{D}_j$ according to $\mathbf{S}_j$ to obtain the structure condition $\hat{\mathbf{D}}_j$. We also use the HED detector[7] to extract boundary cues that are overlaid on the corrupted depth condition. This trains the generator to follow coarse structure while remaining robust to imperfect geometry cues.

S3.1.5 Prompt Condition. The text condition τ_j is constructed by a rule-based template from the semantic assignments of the conditioning images. The template first describes the fixed slot layout: image 1 contains $(\mathbf{P}_j, \mathbf{O}_j)$, image 2 contains $(\hat{\mathbf{D}}_j, \mathbf{S}_j)$, and each subsequent image contains an identity-region pair $(I_o, \mathbf{M}_{j,o})$. It then appends object-specific clauses using the visible object names. For example, when image 3 contains a house reference and image 4 contains a tree reference, the resulting prompt is:

> image1 stacks the downsampled partial observation on top of its observed-region mask. image2 stacks the downsampled corrupted depth map on top of its per-pixel corruption strength. image3 provides the house reference together with its region mask. image4 provides the tree reference together with its region mask. Generate one coherent scene that follows all image conditions, stays faithful to the depth in image2, and does not add extra objects.

S3.2 Training

We fine-tune FLUX.2 Klein with LoRA adapters on the transformer attention projections, while keeping the text encoder and VAE frozen. All conditioning images are encoded by the VAE and concatenated as image tokens after the noisy target latent. Separate image-token coordinates distinguish the target latent from each paired condition. These paired conditions are $(\mathbf{P}_j, \mathbf{O}_j)$, $(\hat{\mathbf{D}}_j, \mathbf{S}_j)$, and the identity-region set $\{(I_o, \mathbf{M}_{j,o})\}_{o \in O_j}$. We use LoRA rank 128, alpha 128, and dropout 0.05. The model is trained with AdamW using learning rate 1e-4 and weight decay 1e-4, with batch size 16 for 1500 iterations.

S3.3 Anchor-view Verification

During sequential anchor-view inference, stochastic generation can occasionally place object content outside the intended semantic regions, especially when several object references are conditioned together. To detect these failures, we evaluate a semantic overflow ratio for each generated anchor-view candidate. We segment the generated image for the visible object categories, excluding sky and ground, and compare the union of the predicted object masks with the union of the target object regions in the conditioning semantic map. The overflow ratio is the image-area fraction of predicted object pixels that lie outside the target object union.

If the overflow ratio is larger than 0.01, we discard the candidate and resample the same conditioning with the next random seed. We evaluate at most 6 candidates in total and accept the first candidate whose overflow ratio is below the threshold. If no candidate satisfies the threshold, we keep the candidate with the lowest overflow ratio. This verification step only filters stochastic anchor-view failures; it does not change the scheduled view order or the conditioning inputs.

[6] https://github.com/microsoft/moge
[7] https://github.com/s9xie/hed

Table S2. 3DGS optimization parameter index.

Notation	Name	Value
λ_{r}	RGB reconstruction weight	0.8
λ_{s}	DSSIM weight	0.2
λ_{d}	Metric-depth loss weight	5.0
λ_{p}	Maximum LPIPS weight	0.10
T_{p}	LPIPS start iteration	5000

S4 Anchor-view Interpolation Details

After the anchor images are generated, each scheduled edge is converted into a Wan2.1-VACE-14B interpolation input. We use 49 frames for each edge. The first and last frames are fixed to the two generated anchor-view images, while the 47 intermediate control frames are normalized depth renderings along the scheduled edge trajectory. Video diffusion is run for 30 denoising steps. The same fixed prompt is used for all edges:

> Generate a smooth, temporally coherent interpolation video between the first and last anchor-view frames. Treat the first and last frames as strong appearance references, follow the intermediate control frames closely for geometry and camera motion, preserve the scene layout and object identity, keep textures and lighting consistent, and avoid flicker, deformation, new objects, or abrupt transitions.

S5 3DGS Optimization Details

We optimize the 3DGS model with the standard RGB reconstruction and DSSIM losses, plus metric-depth and LPIPS terms. The numerical parameters used in this stage are summarized in Table S2. Following CAT3D [Gao et al. 2024], the LPIPS coefficient is not fixed across all training views. Following the main paper notation, let $\mathcal{J}$ denote the training-view index set. Let $\mathcal{J}_{\mathrm{edge}} \subset \mathcal{J}$ denote the set of interpolated edge frames. For an edge frame $m \in \mathcal{J}_{\mathrm{edge}}$ with frame index $r_m \in \{0, \ldots, 48\}$, we define its normalized position along the edge as

$$t_m = \frac{r_m}{48}. \tag{19}$$

We then use the midpoint-normalized edge weight

$$w_m = 4t_m(1 - t_m), \tag{20}$$

which equals zero at the two endpoints and one at the midpoint. The effective LPIPS weight is

$$\lambda_{\mathrm{p}}^{\mathrm{eff}}(m, \ell) = \begin{cases} \lambda_{\mathrm{p}} w_m, & m \in \mathcal{J}_{\mathrm{edge}},\ \ell \geq T_{\mathrm{p}}, \\ 0, & \text{otherwise.} \end{cases} \tag{21}$$

This weighting gives zero LPIPS weight to anchor views and reaches its maximum at the midpoint of an interpolation edge.

Table S3. Scene statistics and runtime summary of our method across different layouts. We report the number of objects, graph nodes, graph edges, and generated views for each layout. Runtime is measured in minutes and includes view scheduling, multi-view generation, 3DGS training, and the total pipeline runtime.

	Scene statistics				Latency (minutes)			
Layout name	Object	Node	Edge	Views	View Scheduling	Multi-view Generation	3DGS Training	Total
Village	16	21	22	1099	4.1	316.8	39.7	360.6
Tribe Town	16	12	12	600	17.0	372.5	37.3	426.7
Desert	12	12	12	600	8.0	238.9	36.2	283.1
Cherry Blossom	9	10	10	500	8.0	185.2	35.8	229.0
Christmas	11	15	17	848	2.3	291.9	43.7	337.9
Ferris Wheel	10	12	14	698	3.5	286.7	38.3	328.5
Court Yard	10	6	6	300	1.1	115.6	28.1	144.8
Back Yard	11	11	11	550	6.2	165.3	28.6	200.1
Small Village	9	4	4	200	9.6	53.5	23.7	86.8

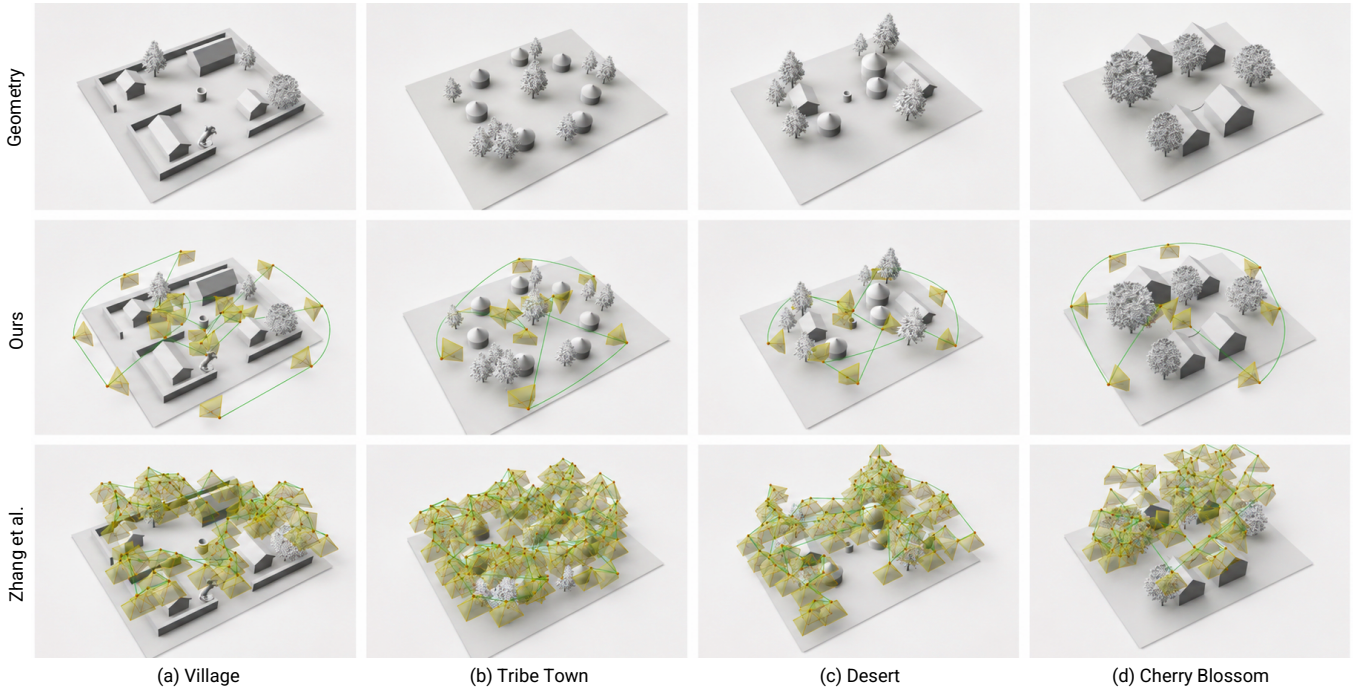


Fig. S3. Qualitative comparison of view scheduling between our method and Zhang et al. [2021]. Yellow frustums denote anchor views, and green edges denote interpolation trajectories.

S6 Scene Statistics & Latency

Table S3 reports the scene statistics and runtime of SceneFrom3D across different layouts. For each input layout, the table shows the number of objects, graph nodes, graph edges, and generated multi-view observations determined by our automatic view scheduling algorithm, together with the latency of each stage and the total pipeline runtime. All experiments were conducted on NVIDIA A100-80G GPUs. We used a single GPU for view scheduling and 3DGS training, while multi-view generation used two GPUs for parallel video inference. On average, a single anchor-view generation takes approximately 4 minutes and a single interpolation takes approximately 7 minutes. The reported anchor-view generation time includes retry attempts caused by verification failures.

S7 View Scheduling Comparison

Fig. S3 provides qualitative comparisons of view scheduling across four input layouts. Given the input geometry shown in the first row, SceneFrom3D produces a sparse set of anchor views and interpolation trajectories that cover the scene structure, as visualized by the yellow frustums and green edges in the second row. In contrast, Zhang et al. [2021] generates substantially denser camera placements and trajectories, with many views biased toward overhead viewpoints. This results in redundant observations and less effective coverage of object-level scene structures. The comparison shows that SceneFrom3D achieves more compact and structured view scheduling while maintaining coverage of the input layout.

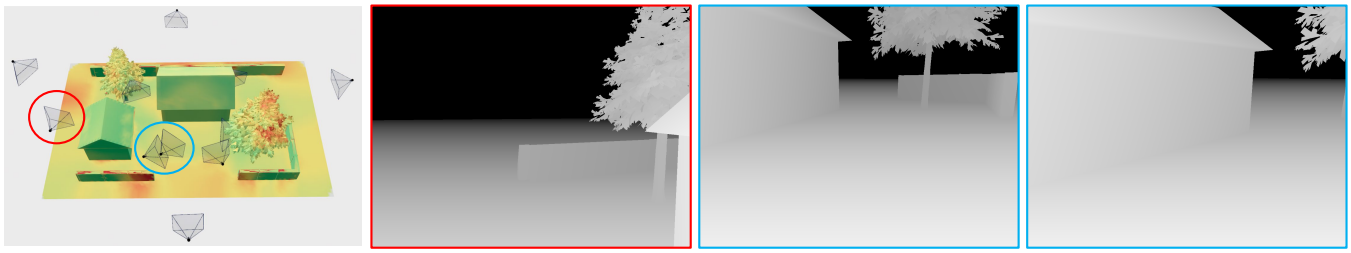


Fig. S4. Ablation example without the FOV term V^{fov}. (a) Optimized camera poses without V^{fov}. (b) View rendered from the red-circled camera in (a), which observes little informative scene content. (c) Views rendered from the blue-circled cameras in (a), which are redundant with nearby views.

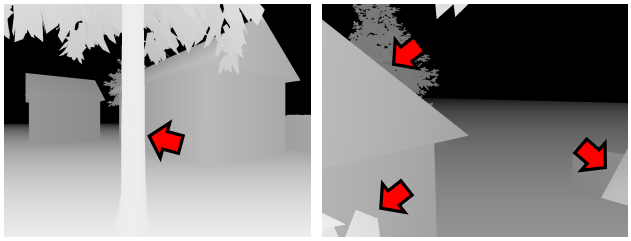


Fig. S5. Ablation examples without the distance term V^{dist}. The rendered depth views show optimized cameras whose observations are largely blocked by nearby foreground geometry. Red arrows indicate occluding objects placed close to the camera.

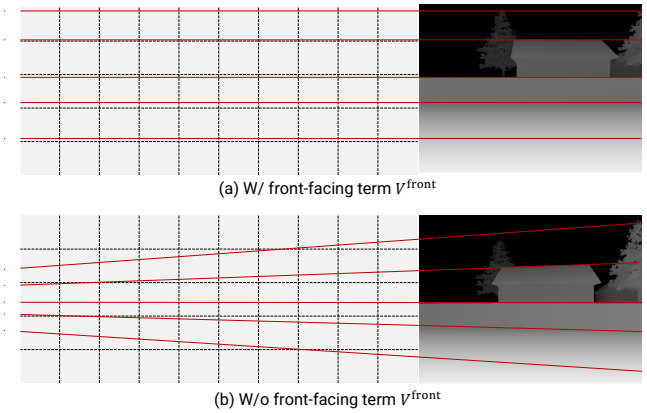


Fig. S6. Visualization of depth maps and perspective lines from optimized camera poses with and without the front-facing term V^{front}.

S8 Analysis on Soft Visibility Score

We discuss the effect of each geometric term, including field-of-view, distance, and front-facing terms, used in our visibility score.

Effect of field-of-view term. Fig. S4 shows the result of removing the FOV term V^{fov}, which assigns visibility only to surface samples inside the camera frustum. This term is the most critical component because, without it, surface samples outside the camera view can receive the same visibility score as visible samples. As a result, the view optimization becomes almost random with respect to actual view coverage. This can produce uninformative views that observe little useful scene content, as shown in Fig. S4(b), or near-duplicate views that redundantly observe almost the same region, as shown in Fig. S4(c).

Effect of distance term. Fig. S5 shows the results of removing the distance term V^{dist}. Without this term, surface samples are scored similarly regardless of their distance from the camera. Therefore, if a sufficient number of background surface samples fall within the view, the optimization can accept undesirable cameras even when a foreground object severely blocks the scene. As shown in Fig. S5, this often results in views where nearby geometry occludes the target regions, leading to poor and less useful observations for generation.

Effect of front-facing term. Fig. S6 shows the effect of the front-facing term V^{front}. This term encourages cameras to observe surfaces from a more frontal direction rather than from oblique angles. Such front-facing observations are generally more favorable for image generation because the target structure is clearer and less distorted in the rendered condition. In contrast, without the front-facing term, the optimized camera can observe surfaces at a slanted angle, which can degrade the quality of the generated views.

S9 Ablation on 3DGS Training Loss

Effect of depth loss. Fig. S7(a,b) shows the effect of the depth loss during 3DGS optimization. Without the depth loss, the optimized Gaussians can drift away from the input geometry and produce floaters, as shown in Fig. S7(b). These floaters can incorrectly appear in front of scene elements, occluding the tree in this example. By enforcing consistency with the rendered depth from the input geometry, the depth loss suppresses such geometric artifacts and keeps the optimized 3DGS better aligned with the intended 3D structure, as shown in Fig. S7(a).

Effect of LPIPS loss. Fig. S7(c,d) shows the effect of the LPIPS loss during 3DGS optimization. Without the LPIPS loss, high-frequency appearance details from the generated training views can be overly smoothed, resulting in blurry textures, as shown in Fig. S7(d). This is particularly noticeable in the ground region highlighted by the orange inset. By encouraging perceptual similarity to the generated

(a) W/ depth loss (b) W/o depth loss (c) W/ LPIPS loss (d) W/o LPIPS loss

Fig. S7. Ablation examples of the 3DGS training losses. (a,b) Results with and without the depth loss. (c,d) Results with and without the LPIPS loss. The arrows indicate structural differences, and the orange insets highlight texture differences.

views, the LPIPS loss helps preserve sharper and more detailed textures during 3DGS optimization, as shown in Fig. S7(c).