

ChronoGS: Disentangling Invariants and Changes in Multi-Period Scenes

Zhongtao Wang¹ Jiaqi Dai² Qingtian Zhu³ Yilong Li¹ Mai Su¹
 Fei Zhu¹ Meng Gai¹ Shaorong Wang¹ Chengwei Pan⁴ Yisong Chen^{1*} Guoping Wang¹

¹Peking University ²Beijing Forestry University ³The University of Tokyo ⁴Beihang University

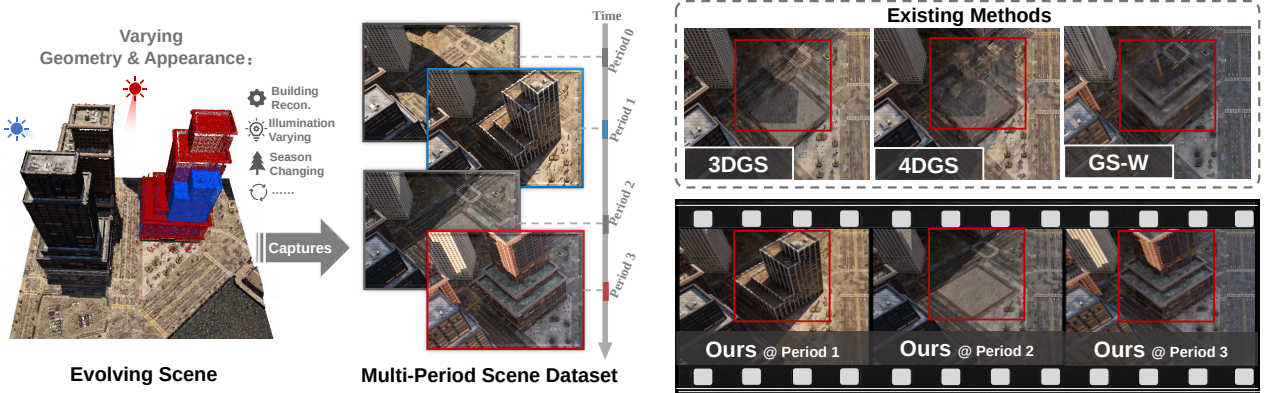


Figure 1. As time goes by, scenes naturally evolve — sunlight shifts, new structures emerge, seasons change, and so on. Cross-period captures therefore exhibit variations in both geometry and appearance. The red and blue overlays in the illustration simply exemplify the differences between the two periods. These coupled changes pose significant challenges for existing methods. Our approach, **ChronoGS**, robustly reconstructs and disentangles both geometry and appearance variations, delivering consistent and faithful results across time.

Abstract

Multi-period image collections are common in real-world applications. Cities are re-scanned for mapping, construction sites are revisited for progress tracking, and natural regions are monitored for environmental change. Such data form multi-period scenes, where geometry and appearance evolve. Reconstructing such scenes is an important yet underexplored problem. Existing pipelines rely on incompatible assumptions: static and in-the-wild methods enforce a single geometry, while dynamic ones assume smooth motion, both failing under long-term, discontinuous changes. To solve this problem, we introduce **ChronoGS**, a temporally modulated Gaussian representation that reconstructs all periods within a unified anchor scaffold. It’s also designed to disentangle stable and evolving components, achieving temporally consistent reconstruction of multi-period scenes. To catalyze relevant research, we release **ChronoScene** dataset, a benchmark of real and synthetic multi-period scenes, capturing geometric and appearance variation. Experiments demonstrate that ChronoGS consistently outperforms baselines

in reconstruction quality and temporal consistency. Our code and the ChronoScene dataset are publicly available at <https://github.com/ZhongtaoWang/ChronoGS>.

1. Introduction

Urban environments are periodically re-scanned for city mapping and management. Active construction sites are repeatedly surveyed to monitor progress and structural modifications. Regions affected by natural disasters are revisited over time to assess damage and document recovery. As a result, multi-period imagery of the same scene naturally accumulates—collected under different seasons, lighting, and even structural layouts. These multi-period captures contain valuable signals for long-term scene understanding, yet also bring unique challenges: geometry and appearance may evolve discontinuously due to construction, vegetation growth, or environmental variation.

Despite its clear importance, this multi-period scene reconstruction setting has remained largely unexplored. Existing pipelines are typically designed for two distinct settings. Static methods, including those designed for in-the-wild imagery, assume that all views share a time-invariant

*Corresponding author.

geometry. Dynamic methods, on the other hand, rely on smooth and continuous motion over time. Neither assumption holds when captures are seasons or even years apart. Consequently, current approaches fail to provide a consistent representation while flexibly modeling per-period variations. Our work aims to close this gap by introducing a framework that reconstructs all periods within a unified representation, disentangling what remains stable from what evolves.

We define a multi-period^{*} scene as a set of captures of the same spatial region acquired at discrete time intervals, where the scene may have undergone geometric or appearance changes between captures. Our key observation is that multi-period scene reconstruction is neither static nor smoothly dynamic — it is discrete in periods but shared in major structure. The majority of spatial content remains invariant across periods, while changes are independent, enabling the scene to be decoupleable. This property allows the scene to be factorized into a shared canonical geometry and period-specific variations, which can be independently modeled within a unified framework.

Therefore, we propose **ChronoGS**, a temporally modulated Gaussian representation that models multi-period scenes consistently. ChronoGS reconstructs all multi-period scenes within a unified anchor scaffold of union geometry across all periods, providing a consistent geometric backbone for cross-period reconstruction. Within this scaffold, each anchor maintains an invariant base feature and a pool of local period-varying features that capture temporal variations. In addition, a global period-varying feature is adopted to model scene-level changes. These features allow anchors to vary over time while remaining a unified structure, ensuring stable and consistent reconstructions across periods. A geometry activation mechanism further enables the scaffold to express geometry specific to each period while maintaining global spatial consistency. Through this unified design, ChronoGS captures complex temporal phenomena, including both geometry and appearance change, achieving consistent reconstruction on multi-period scenes.

To support our research, we introduce **ChronoScene**, a novel benchmark designed to capture both geometry and appearance changes of multi-period scenes. Unlike existing benchmarks that assume static geometry or short-term continuous motion, this setting exposes the real challenges of large-scale scene evolution—building construction, vegetation growth, seasonal, and lighting shifts, etc. ChronoScene aligns multi-period captures in a common world coordinate frame, and provides a principled testbed for temporally modulated 3D modeling approaches.

Our experiments demonstrate that ChronoGS outper-

^{*}We use the term **period** rather than **timestamp**. Because in our setting, each period corresponds to a time window during which images are captured at slightly different times.

forms existing methods across diverse scenes, delivering better renders and more accurate geometry for each period while synthesizing plausible intermediate states. Ablation studies prove the rationality of our design. Also, we hope that ChronoGS and our dataset establish a foundation for future research on multi-period scene understanding.

Our contributions are threefold:

- **Novelty.** To the best of our knowledge, this is the **first** work to formally address and benchmark 3D reconstruction on multi-period scenes, which undergo non-continuous geometry and appearance changes.
- **Dataset.** We introduce the **ChronoScene** dataset, a novel benchmark that aligns multi-period observations within a common coordinate and captures both geometric and appearance evolution across periods.
- **Method.** We propose **ChronoGS**, a temporally modulated Gaussian representation that effectively handles geometric and appearance variations that exist in multi-period scenes.

2. Related Work

2.1. Neural Radiance Fields

Neural rendering has advanced along two complementary tracks. Implicit approaches such as NeRFs [1, 17, 19, 20, 24] represent scenes as continuous volumetric fields, mapping 3D location and viewing direction to density and radiance via learned networks coupled with volumetric rendering. While achieving photorealistic novel views, dense ray sampling and repeated network evaluations are computationally expensive, prompting extensive work on acceleration [1, 4, 18]. In parallel, explicit point-based methods target real-time performance: 3D Gaussian Splatting [7] models scenes with rasterized Gaussian primitives for high-quality rendering, extended by hierarchical organization in Scaffold-GS [14], scalable variants for large scenes [8, 21, 23], and extract view-consistent surface representations [2, 6].

Despite these advances, both paradigms assume static scenes, forcing a single time-invariant representation of all observations. Jointly training all periods causes temporal averaging and artifacts, whereas per-period training avoids interference at the cost of heavier computation and storage without improving results. Our **ChronoGS** enables joint optimization across multiple periods, maintaining consistent long-term reconstruction in a unified model.

2.2. Dynamic Scene Representation

Building upon the success of NeRF in static scenes, several works have focused on extending it to dynamic environments. D-NeRF [19] models motion by constructing a canonical space and applying deformation MLPs, although they are constrained by long-sequence videos. K-

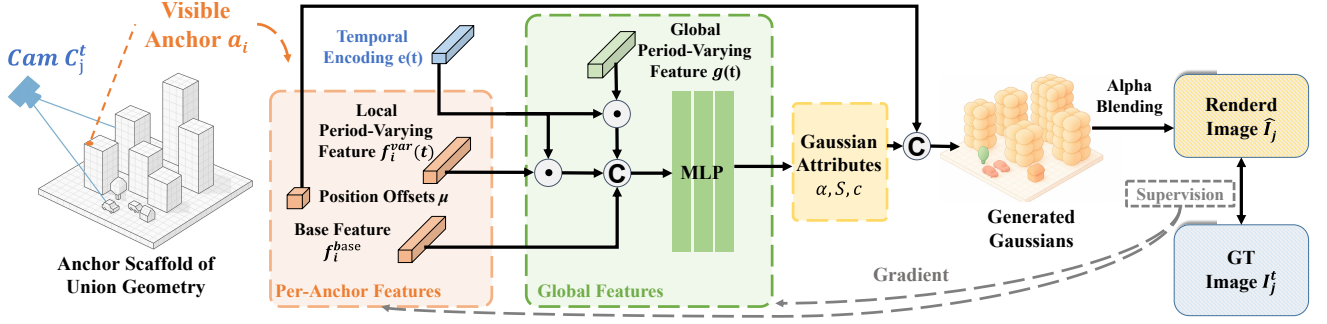


Figure 2. For a given camera C_j^t , we select visible anchors from our learned anchor scaffold of union geometry across periods. We use per-anchor’s features and global features to render the multi-period scene. Temporal behavior is modeled via the anchor’s local period-varying feature $f_i^{var}(t)$ and global period-varying feature $g(t)$ modulated by temporal encoding $e(t)$. After concatenating them with anchor’s period-invariant base feature f_i^{base} , a lightweight MLP predicts Gaussian attributes $\{\alpha, S, c\}$. These attributes, along with the learned Gaussian position offset μ , produce a small cluster of Gaussians per anchor. We render the Gaussians generated by all visible anchors via differentiable splatting with alpha blending to obtain the image $\hat{I}_j^t$, and optimize with photometric losses between the ground truth image. This design enables consistent reconstruction of both geometry and appearance evolution on multi-period scenes.

Planes [5] decompose features across multiple planes, enabling lightweight models to capture motion effectively. In parallel, 3DGS[7] has been extended to dynamic scenes. Dynamic 3DGS [15] optimizes Gaussians shared across time with regularizers, which rely on accurate point-cloud segmentation and additional temporal consistency terms. 4DGS [26] models dynamic scenes using canonical-space Gaussians combined with deformation MLPs. Real-time4DGS [28] and 4DRotor-GS [3] lift 3D Gaussians into 4D by adding a temporal dimension, and extract the corresponding 3D Gaussians at a given timestamp for rendering.

However, many of these methods assume temporally dense and smooth evolution, depend on precise masks or segmentations, and struggle with abrupt appearances and disappearances. These limitations are especially acute in multi-period data with long gaps and discontinuities. In contrast, our ChronoGS framework decouples the representation over periods, accommodating sudden shifts while preserving cross-period consistency.

2.3. Novel View Synthesis in the Wild

Methods targeting *in-the-wild* image collections relax controlled capture assumptions by handling unknown poses, illumination shifts, post-processing, and transient occluders. NeRF-W [16] models per-image appearance and a transient volume to explain lighting and clutter in Internet photos. NeRF-MS [10] regularizes appearance codes for cross-sequence consistency, while NeuSC [12] employs step functions and illumination embeddings to reconstruct long-term landmark evolution. 3DGS-based variants extend these ideas to explicit representations: WildGaussians [9] integrates feature priors and an appearance module for robust real-time rendering; Wild-GS [27] achieves similar performance on unstructured photos; VastGaussian[13],

GS-W [30] and WE-GS [25] separates intrinsic and dynamic appearance for stable reconstructions; and GaussianUpdate [29] incrementally updates a 3DGS model to incorporate new observations without storing original images.

However, most *in-the-wild* pipelines focus on appearance variability and lack mechanisms to model geometric change, typically collapsing all periods into a single representation. In contrast, our method explicitly disentangles temporal representations, reconstructing faithful geometry and appearance for each period while maintaining cross-period consistency.

3. Method

Given a multi-period scene’s captures $\{(\mathbf{I}_j^{(t)}, \mathbf{C}_j^{(t)})\}_{j=1}^N$ containing N images $\mathbf{I}_j^{(t)}$ and camera parameters $\mathbf{C}_j^{(t)}$ collected at $t = 1 \dots T$ discrete periods, our goal is to reconstruct all periods within a unified 3D representation. Different periods are often collected months or years apart, leading to substantial changes in both geometry and appearance.

We propose a temporally modulated Gaussian representation that can automatically distinguish period-varying and invariant components. Inspired by [14], which organizes scenes with anchor points that spawn local 3D Gaussians and predict their attributes, we adopt an anchor scaffold and render via per-anchor Gaussian decoding. Our framework learns an anchor scaffold of union geometry across periods, while temporally modulation and geometry activation mechanism identifies regions that change in appearance or geometry. The model thereby reconstructs each period faithfully and synthesizes plausible intermediate states. An overview of the pipeline is illustrated in Fig. 2.

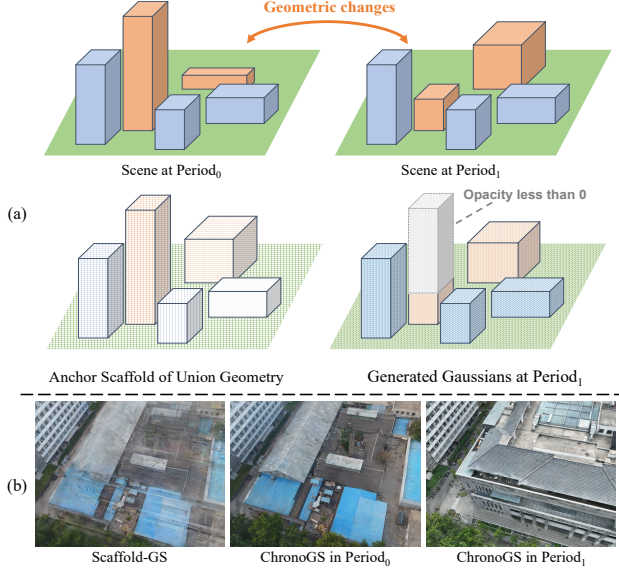


Figure 3. **Illustration of the temporal geometry activation mechanism.** (a). Anchor scaffold encodes the union of geometry across all periods, and generated Gaussians with negative opacity are deactivated at certain periods, enabling the model to adaptively represent geometry variations over time. (b). Comparison between Scaffold-GS and our ChronoGS trained on all periods’ images. Scaffold-GS entangles structures from different periods, while ChronoGS successfully distinguishes and reconstructs period-specific geometry at Period₀ and Period₁.

3.1. Temporally Modulated Feature Representation

Anchor feature decomposition. To jointly represent shared and period-specific properties, each anchor a_i maintains two features: a time-invariant base feature $\mathbf{f}_i^{\text{base}} \in \mathbb{R}^{d_b}$ that describes shared geometry and appearance, and a local time-varying feature pool $\mathbf{f}_i^{\text{var}} = [\mathbf{f}_i^{(1)}, \dots, \mathbf{f}_i^{(T)}] \in \mathbb{R}^{T \times d_v}$ that stores period-specific information.

Time encodings. We represent time using a unified T -dimensional encoding that supports both discrete and continuous timestamps. For integer timestamps, we adopt a one-hot encoding where each observed period corresponds to a unique basis vector. For fractional timestamps, we perform linear interpolation between adjacent integer embeddings to obtain a smooth, continuous representation:

$$\mathbf{e}(t) = (1 - w)\mathbf{e}_{\lfloor t \rfloor} + w\mathbf{e}_{\lceil t \rceil}, \quad w = t - \lfloor t \rfloor, \quad (1)$$

where $\mathbf{e}_{\lfloor t \rfloor}$ and $\mathbf{e}_{\lceil t \rceil}$ are the one-hot embeddings of neighboring integer periods. It produces exact one-hot encodings at observed periods while yielding smooth transitions for intermediate timestamps.

Global-Local Temporal Modulation. The i -th anchor maintains a time-invariant base feature $\mathbf{f}_i^{\text{base}}$ that encodes the

canonical geometry and appearance shared across all periods. The temporal encoding $\mathbf{e}(t)$ defined in Eq. (1) modulates the local period-varying feature $\mathbf{f}_i^{\text{var}}(t)$, capturing local temporal variations around the anchor. Also, a global period-varying feature $\mathbf{g}(t) \in \mathbb{R}^{T \times d_g}$ capturing scene-level factors such as illumination and seasonal appearance is modulated by $\mathbf{e}(t)$. Together, these components describe canonical, local, and global aspects of the scene, allowing the model to capture fine-grained dynamics while maintaining temporal consistency.

Explicit Gaussian Offsets for Geometric Stability. The i -th anchor stores K learnable but period-invariant Gaussian center offsets $\{\Delta\boldsymbol{\mu}_{ik}\}_{k=1}^K$, forming a fixed local arrangement around the anchor. Explicitly storing these offsets preserves local geometric stability across periods, so temporal geometry changes are mainly reflected through the Gaussians’ visibility. This fixed spatial layout ensures consistent geometry across periods, enabling reliable opacity-based filtering described in Sec. 3.2.

3.2. Modeling Cross-Period Unified Geometry

Unified scaffold. We aggregate multi-period geometry into a unified anchor scaffold. The scaffold is initialized by integrating sparse point clouds from all periods into a shared coordinate frame, providing uniform coverage of both stable and varying geometry. Rather than training separate models for each period, we optimize a single unified scaffold across time, allowing the learning process to automatically determine which anchors are active in each period. Moreover, this formulation supports continuous temporal reasoning, enabling the model to synthesize intermediate periods not observed during training.

Temporal geometry activation. Although the anchor scaffold encodes the union of geometry across all periods, not every anchor should contribute at every time step. We model geometry as locally stable around each anchor, while temporal changes are captured by modulating their attributes and visibility. As illustrated in Fig. 3, to adaptively express the geometry of a specific period, we introduce a simple but effective geometry activation mechanism based on the predicted opacity. During rendering, Gaussian primitives whose decoded opacity $\alpha_{ik}(t)$ falls below zero are excluded from both compositing and gradient backpropagation. Only Gaussians with an $\alpha_{ik}(t) > 0$ are activated at the current moment. This mechanism automatically deactivates anchors corresponding to geometry that is occluded in certain periods. Empirically, it also stabilizes training by preventing gradients from being propagated through inactive regions, while maintaining differentiability for regions that continuously evolve over time. Consequently, the scaffold functions as a global geometric prior that dynamically

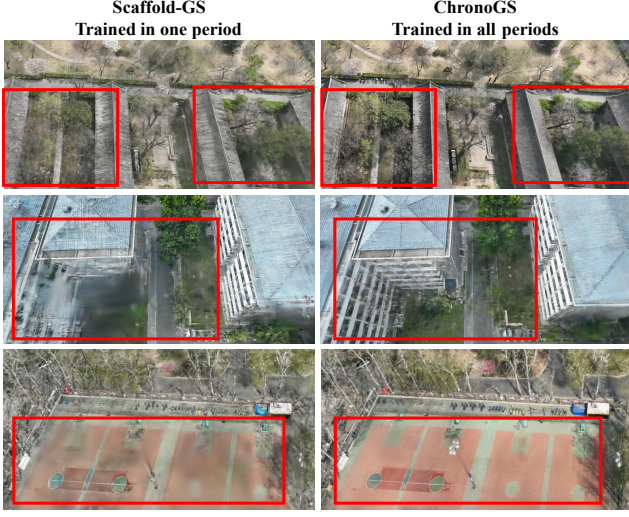


Figure 4. Comparison between Scaffold-GS trained in one period (left) and ChronoGS trained across all periods (right). Regions highlighted in red show that ChronoGS benefits from multi-period joint training, leveraging more observations to recover more accurate geometry and produce more complete reconstructions.

activates relevant anchors per period, preserving both adaptability and structural consistency.

Cross-period benefit. Training on multi-period imagery in a unified anchor scaffold also benefits geometrically stable regions. It allows complementary supervision on all periods’ capture to improve reconstruction quality. Anchors that persist across time receive denser supervision from multi-period observations, leading to improved photometric consistency and reduced overfitting to single-period noise. This cross-period aggregation acts as an implicit regularizer: it encourages shared anchors to capture canonical geometry. As shown in Fig. 4, such joint optimization produces sharper textures and a more reliable geometry in areas that remain static across periods. Overall, the unified scaffold serves as a structural backbone that enables change geometry modeling. It also take efficient use of all observations across periods in geometry-invariant areas, as opposed to training periods separately, which costs higher computation and storage.

3.3. Gaussian Decoding and Rendering

At each time t , we first fuse the features through temporal modulation:

$$\mathbf{h}_i(t) = \mathbf{g}(t) \odot \mathbf{e}(t) + \mathbf{f}_i^{\text{var}}(t) \odot \mathbf{e}(t) + \mathbf{f}_i^{\text{base}}, \quad (2)$$

where $\odot$ denotes channel-wise multiplication. Let $\mathbf{x}_i \in \mathbb{R}^3$ be the spatial position of anchor a_i , and $\mathbf{p}$ be the center of camera C_j^t . A lightweight MLP decoder Φ_θ maps $\mathbf{h}_i(t)$ to

the parameters of K Gaussians:

$$\left\{ \mathbf{S}_{ik}(t), \alpha_{ik}(t), \mathbf{c}_{ik}(t) \right\}_{k=1}^K = \Phi_\theta(\mathbf{h}_i(t), \hat{\mathbf{d}}_i), \quad (3)$$

where $\mathbf{S}_{ik}(t)$ denotes the shape (rotation and scale), $\alpha_{ik}(t)$ the opacity, $\mathbf{c}_{ik}(t)$ the color of the k -th Gaussian associated with anchor a_i , and $\hat{\mathbf{d}}_i = \mathbf{x}_i - \mathbf{p}$ is the viewing direction from the camera to the anchor. Each Gaussian’s 3D center is obtained as $\boldsymbol{\mu}_{ik} = \mathbf{x}_i + \Delta\boldsymbol{\mu}_{ik}$, where $\Delta\boldsymbol{\mu}_{ik}$ is a learnable offset explicitly stored within anchor a_i .

Finally, the rendered color $\mathbf{C}(\mathbf{r}, t)$ for a camera ray $\mathbf{r}$ follows the standard volumetric alpha compositing of 3DGS [7]. We first project 3D Gaussians to 2D with mean $\hat{\boldsymbol{\mu}}_{ik}$ and covariance $\hat{\boldsymbol{\Sigma}}_{ik}$. Then, a front-to-back alpha compositing yields the rendered color:

$$\hat{\mathbf{C}}(\mathbf{u}) = \sum_n \hat{\mathbf{c}}_n(\mathbf{u}) \hat{\alpha}_n(\mathbf{u}) \prod_{j < n} (1 - \hat{\alpha}_j(\mathbf{u})). \quad (4)$$

3.4. Optimization

We supervise the rendered images $\hat{\mathbf{I}}$ with ground-truth observations $\mathbf{I}$ using a hybrid photometric objective:

$$\mathcal{L} = \lambda \|\hat{\mathbf{I}} - \mathbf{I}\|_1 + (1 - \lambda)(1 - \text{SSIM}(\hat{\mathbf{I}}, \mathbf{I})). \quad (5)$$

Following [14], we track the accumulated gradient magnitude of each anchor feature over recent iterations, which reflects how much an anchor contributes to the reconstruction. Anchors with persistently small gradient magnitude are pruned to remove redundancy and reduce memory footprint. We also monitor the view-space positional gradients of the Gaussians emitted from each anchor. When these gradients exceed a threshold, we spawn new anchors at the corresponding 3D locations obtained by the positions of Gaussians from old anchors. Both pruning and growth operations are triggered at regular training intervals, allowing the scaffold to refine its structural support to cover the union of geometry across all periods while maintaining a consistent canonical representation.

4. Experiments

4.1. Implementation Details

We set anchor’s base feature dimension to $d_b = 16$, the local period-varying feature dimension to $d_v = 16$, and the global feature dimension to $d_g = 32$. For each anchor, the lightweight MLP decoder predicts $K = 10$ Gaussians. All experiments are trained for 40k iterations to ensure sufficient convergence on large-scale multi-period scenes. Details and ablations on hyperparameters and training strategy are provided in the supplementary material.

Real Scenes																			
	Avg.	Lawncourt			Overstreet			BioBuilding			Canteen			Incubator			TriPlot		
Metrics	Mem.↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
3DGS[7]	1.01GB	18.29	0.4658	0.4862	18.79	0.6640	0.3627	17.37	0.5754	0.4402	19.64	0.5885	0.4150	15.97	0.6232	0.3874	20.17	0.6320	0.3793
Scaffold-GS[14]	0.36GB	16.27	0.3749	0.5363	15.20	0.5433	0.4390	16.09	0.4876	0.4923	17.70	0.5165	0.4528	15.72	0.5524	0.4291	18.04	0.5553	0.4280
GS-W[30]	0.23GB	20.33	0.4018	0.5638	21.64	0.6644	0.3974	21.15	0.5536	0.5053	21.24	0.5227	0.4780	22.39	0.6788	0.3922	22.57	0.6064	0.4436
4DGS[26]	0.18GB	19.03	0.4674	0.6085	19.12	0.5815	0.5110	17.18	0.5013	0.6211	21.67	0.5932	0.5345	18.39	0.5538	0.5333	19.35	0.5093	0.5691
realtime4DGS[28]	5.53GB	20.91	0.5124	0.4943	17.17	0.5087	0.4993	21.39	0.5801	0.4920	22.49	0.6181	0.4292	22.71	0.7070	0.3801	22.86	0.6681	0.3982
Ours	0.52GB	22.16	0.6533	0.3390	22.46	0.7736	0.2838	22.87	0.7221	0.3274	25.15	0.7538	0.2647	23.79	0.7852	0.2721	24.16	0.7474	0.2901

Synthetic Scenes																			
	Avg.	Aerial1			Aerial2			Aerial3			Aerial4			Street1			Street2		
Metrics	Mem.↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
3DGS[7]	1.05GB	20.79	0.7831	0.3179	19.95	0.7877	0.2252	23.20	0.8250	0.2682	21.41	0.7934	0.2631	15.40	0.6763	0.4065	18.77	0.7225	0.3580
Scaffold-GS[14]	0.30GB	19.52	0.7273	0.3605	18.35	0.7199	0.2738	19.51	0.7485	0.3327	18.38	0.7020	0.3272	14.49	0.5905	0.5252	16.86	0.5856	0.5279
GS-W[30]	0.22GB	25.11	0.7581	0.3754	23.21	0.7005	0.2510	28.79	0.7958	0.3186	26.13	0.7826	0.2644	15.73	0.6328	0.4847	18.93	0.7075	0.3873
4DGS[26]	0.18GB	20.69	0.7140	0.4375	19.62	0.6325	0.4324	22.47	0.7359	0.3971	20.67	0.6675	0.4369	13.53	0.5440	0.5635	16.03	0.5310	0.5928
realtime4DGS[28]	7.41GB	22.29	0.7774	0.3351	20.04	0.7088	0.3268	23.93	0.8044	0.3075	21.60	0.7443	0.3341	18.88	0.7121	0.3944	20.53	0.7451	0.3514
Ours	0.65GB	28.80	0.8562	0.2509	26.50	0.8467	0.1854	30.11	0.8751	0.2266	28.10	0.8350	0.2305	24.70	0.8071	0.3241	26.56	0.8220	0.2946

Table 1. Quantitative comparison on ChronoScene dataset. The best, second-best, and third-best results in each column are highlighted by red, orange, and yellow, respectively. The upper part shows results on real scenes, and the lower part on synthetic scenes.

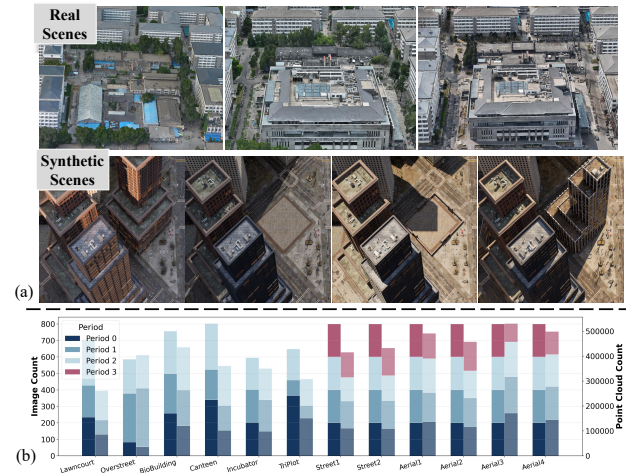


Figure 5. (a). Examples from the ChronoScene dataset. Top: real-world aerial captures showing natural evolution across long temporal intervals (e.g., construction and season change). Bottom: our synthetic scenes edited to simulate realistic changes with controllable geometry and appearance changes. (b). Number distribution of image and point cloud across periods for 6 real-world and 6 synthetic scenes. Left bars show image counts, right bars show sparse point cloud counts.

4.2. ChronoScene Dataset

We introduce *ChronoScene*, a multi-period dataset designed to challenge long-term 3D reconstruction under both *geometric* and *appearance* evolution. As illustrated in Fig. 5, scenes in ChronoScene exhibit coupled changes such as building construction or demolition, vegetation growth, and variations in season, lighting, and weather. ChronoScene contains a total of **8,891 images** across **12 scenes** and **42 sub-scenes** (each sub-scene corresponds to one specific scene at one period). All images are registered into a *common world coordinate frame* in COLMAP [22] format, ensuring that cameras across periods are spatially comparable.

Real data. The real-world scenes spans time gaps ranging from one year to three years between periods, reflecting nat-

ural changes in urban environments. Images are geo-tagged at capture time and globally aligned using GPS priors, followed by human-in-the-loop refinement to ensure consistent cross-period camera registration.

Synthetic data. To complement real data with controlled, precisely known temporal changes, we build synthetic scenes based on Matrix City [11]. We manually edit urban assets to simulate realistic evolutions—such as new constructions, façade refurbishments, or vegetation changes—and render both aerial and street-level views.

We release all RGB images, calibrated camera poses, and per-image period identifiers. More statistics, licensing, and details are provided in the supplementary material.

4.3. Evaluation

All experiments are reported on **ChronoScene**, which, to our knowledge, is the only available multi-period benchmark that couples geometric and appearance evolution.

Baselines and setup. We compare several representative Gaussian-based paradigms covering static, structured, in-the-wild, and 4D dynamic modeling: 3DGS [7] as the canonical static baseline, Scaffold-GS [14] introducing anchor-based structured Gaussians, GS-W [30] which targets uncontrolled photo collections, as well as 4DGS [26] and Realtime4DGS [28] that extend Gaussians to temporally continuous motion.

Protocol. Camera trajectories are independently captured at different periods, so we assess reconstruction quality *within each period*: for each test set of a given period, we render from its held-out cameras and compute PSNR, SSIM, and LPIPS with ground-truth images; metrics are then averaged over periods. We ensure that all methods use the same train-test splits and data augmentations.

4.4. Results

Quantitative performance. We report results separately for the real and synthetic portions of ChronoScene. Tab. 1 summarizes image-level metrics and overall storage cost in

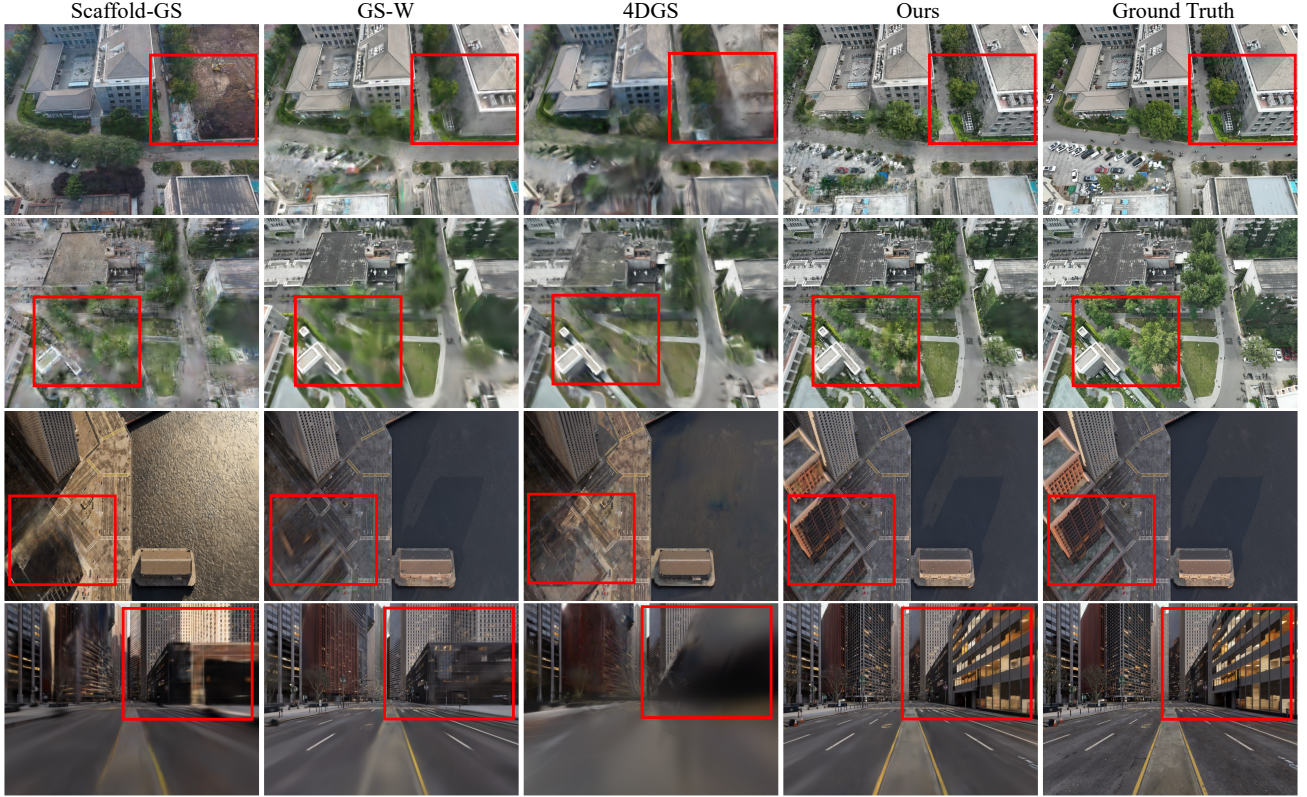


Figure 6. **Qualitative comparison on ChronoScene.** We compare ChronoGS with representative baselines. Each row shows novel-view renderings of different methods, along with ground truth. Static models trained on mixed multi-period data produce ghosting and appearance blending due to inconsistent geometry, while dynamic methods that assume continuous motion fail under the large temporal gaps, causing incorrect geometry interpolation. **ChronoGS** faithfully reconstructs period-specific geometry and appearance, preserving sharp, temporally consistent details across long-term changes.

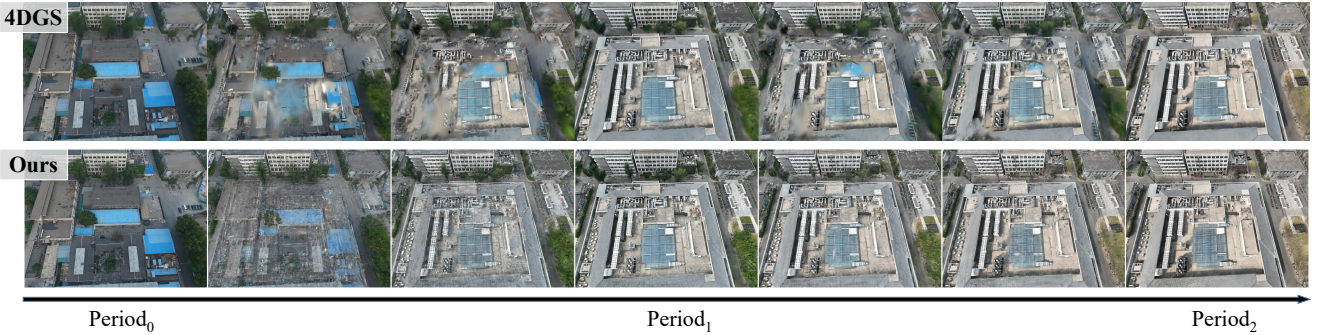


Figure 7. **Temporal evolution at a test viewpoint.** Rendered views from the same camera across periods demonstrate ChronoGS’s ability to model continuous temporal changes from discontinuous data. Compared to dynamic baselines such as 4DGS that assume smooth motion, which often hallucinate intermediate structures or blend neighboring periods, ChronoGS cleanly switches between distinct time states, recovering accurate geometry and appearance at each period.

all 12 scenes of real and synthetic data, where geometry and appearance changes across periods. Our method consistently outperforms existing baselines, achieving the best scores across all metrics and datasets, while maintaining a compact memory footprint. These results confirm the effectiveness of our method in modeling both geometry and

appearance variations across periods.

Qualitative comparisons. Fig. 6 presents a side-by-side comparison of our renderings, ground truth, and representative baselines for held-out views in each period. Static models tend to blur conflicting configurations and dynamic models produce temporal smearing; in contrast, our ap-

	3DGS [7]			Scaffold-GS [14]			Ours		
Period	PSNR↑	Mem.↓	Iters.	PSNR↑	Mem.↓	Iters.	PSNR↑	Mem.↓	Iters.
Period 0	21.97	1.6GB	40k	22.04	0.19GB	40k	22.64	N/A	N/A
Period 1	22.31	2.0GB	40k	21.88	0.25GB	40k	22.50	N/A	N/A
Period 2	22.23	1.3GB	40k	21.81	0.21GB	40k	22.88	N/A	N/A
Overall	22.23	4.9GB	120k	21.87	0.65GB	120k	22.66	0.57GB	40k

Table 2. Quantitative comparison of static methods’ per-period independent training and our joint multi-period optimization on the *Overstreet* scene. Reported metrics include PSNR, memory footprint, and training iterations. ChronoGS achieves the **highest** quality across all periods while markedly reducing training iterations and storage requirements compared with static baselines.

	Lawncourt			Canteen		
Settings	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
w/o Var.&Global.	18.18	0.4609	0.4928	19.88	0.5959	0.4107
w/o Base.	21.60	0.6055	0.3918	24.57	0.7289	0.2970
w/o Var.	21.96	0.6440	0.3526	24.62	0.7426	0.2801
w/o Global.	22.11	0.6466	0.3497	24.94	0.7502	0.2716
Ours (full)	22.16	0.6533	0.3390	25.15	0.7538	0.2647

Table 3. Quantitative ablation results on two real scenes *Lawncourt* and *Canteen*. Removing the base, local period-varying, or global period-varying feature leads to performance degradation, confirming that all three are essential in our design.

proach maintains sharp, period-correct details. Our method preserves period-specific appearance and suppresses cross-period ghosting that appears in static models.

Temporal interpolation. To evaluate whether our model can synthesize intermediate periods, in Fig. 7 we show the renderings at the *same* test viewpoint across time and compare our method to dynamic baselines. Methods designed for continuous dynamics often interpolate through non-existent intermediate states under ChronoScene’s discontinuous region, leading to hallucinated transitions or residuals from neighboring periods. Our model cleanly switches between periods with more reasonable intermediate states.

Compare with Per-Period Training. To evaluate the effect of multi-period joint optimization, we compare ChronoGS with static baselines trained independently for each period under the same evaluation setting on *Overstreet* scene. As summarized in Tab. 2 and shown in Fig. 4, compared with training separately in each period’s capture, our approach, which trains data from all periods together, receives denser and more complementary supervision—leading to the same or improved quality and significant reductions in both training and storage cost.

4.5. Ablation Study

To evaluate the role of each component in ChronoGS, we perform ablations under identical training settings. We compare the full model with several degraded variants: **w/o Base.**, removing the base feature $\mathbf{f}_i^{\text{base}}$ and forcing each anchor to rely only on time-dependent representations; **w/o Var.**, removing the local period-varying feature $\mathbf{f}_i^{\text{var}}(t)$ and

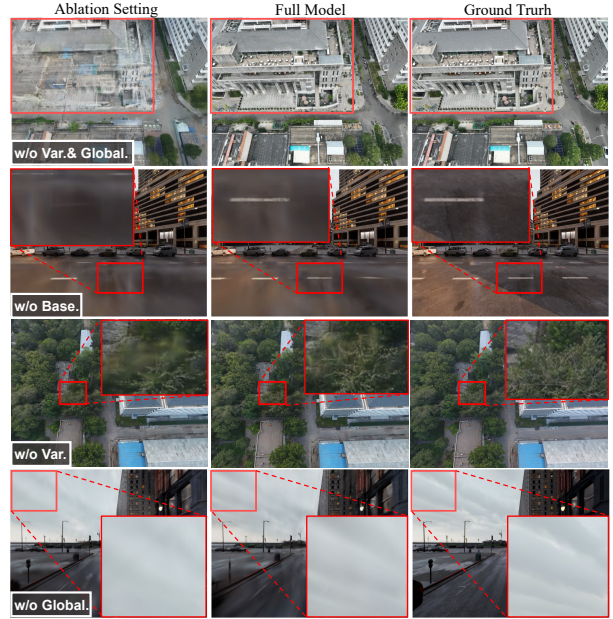


Figure 8. **Qualitative Ablation results.** Removing the base, local time-varying, or global time-varying feature leads to degraded geometry stability, local temporal fidelity, and scene-level appearance consistency, respectively, compared with the full model.

disabling local temporal adaptation; **w/o Global.**, removing the global period-varying feature $\mathbf{g}(t)$ thus losing global appearance coherence; and **w/o Var.&Global.**, removing $\mathbf{g}(t)$ and $\mathbf{f}_i^{\text{var}}(t)$ to disable model’s temporal behavior.

As shown in Tab. 3, the full ChronoGS achieves the best quantitative and perceptual scores. Our design disentangles changes from period-invariant structure to achieve consistent multi-period scene reconstruction. Without the base feature, the shared geometry becomes unstable. Without the local period-varying feature, local temporal changes cannot be expressed, and without the global period-varying feature, scene-level illumination and color consistency degrade noticeably. These can be clearly observed in Fig. 8.

5. Conclusion

We presented **ChronoGS**, a temporally modulated Gaussian representation for reconstructing multi-period scenes. ChronoGS represents a multi-period scene with a unified anchor scaffold that aggregates geometry shared across periods. Temporally modulated features and a geometry activation mechanism jointly model both invariant structures and period-varying attributes. This formulation effectively handles geometric and appearance variations that arise when imagery is collected in different periods. In parallel, we introduced **ChronoScene**, a novel benchmark that collects multi-period observations and captures both geometric and appearance evolution across periods. Extensive experiments demonstrate that ChronoGS achieves sharper renderings, more accurate geometry, and improved cross-

period consistency compared to existing baselines. Ablation studies validate the contribution of each component. To our knowledge, this is the first work to formally address and benchmark 3D reconstruction on multi-period scenes, undergoing non-continuous geometry and appearance changes. We provide both a simple but strong baseline and a valuable public benchmark for future research.

References

- [1] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5470–5479, 2022. 2
- [2] Danpeng Chen, Hai Li, Weicai Ye, Yifan Wang, Weijian Xie, Shangjin Zhai, Nan Wang, Haomin Liu, Hujun Bao, and Guofeng Zhang. Pgsr: Planar-based gaussian splatting for efficient and high-fidelity surface reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, 2024. 2
- [3] Yuanxing Duan, Fangyin Wei, Qiyu Dai, Yuhang He, Wenzheng Chen, and Baoquan Chen. 4d-rotor gaussian splatting: towards efficient novel view synthesis for dynamic scenes. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–11, 2024. 3
- [4] Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5501–5510, 2022. 2
- [5] Sara Fridovich-Keil, Giacomo Meanti, Frederik Rahbæk Warburg, Benjamin Recht, and Angjoo Kanazawa. K-planes: Explicit radiance fields in space, time, and appearance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12479–12488, 2023. 3
- [6] Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2d gaussian splatting for geometrically accurate radiance fields. In *ACM SIGGRAPH 2024 conference papers*, pages 1–11, 2024. 2
- [7] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023. 2, 3, 5, 6, 8, 1
- [8] Bernhard Kerbl, Andreas Meuleman, Georgios Kopanas, Michael Wimmer, Alexandre Lanvin, and George Drettakis. A hierarchical 3d gaussian representation for real-time rendering of very large datasets. *ACM Transactions on Graphics (TOG)*, 43(4):1–15, 2024. 2
- [9] Jonas Kulhanek, Songyou Peng, Zuzana Kukelova, Marc Pollefeys, and Torsten Sattler. Wildgaussians: 3d gaussian splatting in the wild. *arXiv preprint arXiv:2407.08447*, 2024. 3
- [10] Peihao Li, Shaohui Wang, Chen Yang, Bingbing Liu, Weichao Qiu, and Haoqian Wang. Nerf-ms: Neural radiance fields with multi-sequence. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 18591–18600, 2023. 3
- [11] Yixuan Li, Lihan Jiang, Linning Xu, Yuanbo Xiangli, Zhenzhi Wang, Dahua Lin, and Bo Dai. Matrixcity: A large-scale city dataset for city-scale neural rendering and beyond. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3205–3215, 2023. 6
- [12] Haotong Lin, Qianqian Wang, Ruojin Cai, Sida Peng, Hadar Averbuch-Elor, Xiaowei Zhou, and Noah Snavely. Neural scene chronology. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20752–20761, 2023. 3
- [13] Jiaqi Lin, Zhihao Li, Xiao Tang, Jianzhuang Liu, Shiyong Liu, Jiayue Liu, Yangdi Lu, Xiaofei Wu, Songcen Xu, Youliang Yan, et al. Vastgaussian: Vast 3d gaussians for large scene reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5166–5175, 2024. 3
- [14] Tao Lu, Mulin Yu, Linning Xu, Yuanbo Xiangli, Limin Wang, Dahua Lin, and Bo Dai. Scaffold-gs: Structured 3d gaussians for view-adaptive rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20654–20664, 2024. 2, 3, 5, 6, 8, 1
- [15] Jonathon Luiten, Georgios Kopanas, Bastian Leibe, and Deva Ramanan. Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis. In *2024 International Conference on 3D Vision (3DV)*, pages 800–809. IEEE, 2024. 3
- [16] Ricardo Martin-Brualla, Noha Radwan, Mehdi SM Sajjadi, Jonathan T Barron, Alexey Dosovitskiy, and Daniel Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7210–7219, 2021. 3
- [17] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021. 2
- [18] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM transactions on graphics (TOG)*, 41(4):1–15, 2022. 2
- [19] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10318–10327, 2021. 2
- [20] Christian Reiser, Rick Szeliski, Dor Verbin, Pratul Srinivasan, Ben Mildenhall, Andreas Geiger, Jon Barron, and Peter Hedman. Merf: Memory-efficient radiance fields for real-time view synthesis in unbounded scenes. *ACM Transactions on Graphics (TOG)*, 42(4):1–12, 2023. 2
- [21] Kerui Ren, Lihan Jiang, Tao Lu, Mulin Yu, Linning Xu, Zhangkai Ni, and Bo Dai. Octree-gs: Towards consistent real-time rendering with lod-structured 3d gaussians. *arXiv preprint arXiv:2403.17898*, 2024. 2
- [22] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 6, 1

- [23] Mai Su, Zhongtao Wang, Huishan Au, Yilong Li, Xizhe Cao, Chengwei Pan, Yisong Chen, and Guoping Wang. Hug: Hierarchical urban gaussian splatting with block-based reconstruction for large-scale aerial scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 28839–28848, 2025. [2](#)
- [24] Matthew Tancik, Ethan Weber, Evonne Ng, Ruilong Li, Brent Yi, Terrance Wang, Alexander Kristoffersen, Jake Austin, Kamyar Salahi, Abhik Ahuja, et al. Nerfstudio: A modular framework for neural radiance field development. In *ACM SIGGRAPH 2023 conference proceedings*, pages 1–12, 2023. [2](#)
- [25] Yuze Wang, Junyi Wang, and Yue Qi. We-gs: An in-the-wild efficient 3d gaussian representation for unconstrained photo collections. *arXiv preprint arXiv:2406.02407*, 2024. [3](#)
- [26] Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Xinggang Wang. 4d gaussian splatting for real-time dynamic scene rendering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 20310–20320, 2024. [3](#), [6](#)
- [27] Jiacong Xu, Yiqun Mei, and Vishal Patel. Wild-gs: Real-time novel view synthesis from unconstrained photo collections. *Advances in Neural Information Processing Systems*, 37:103334–103355, 2024. [3](#)
- [28] Zeyu Yang, Hongye Yang, Zijie Pan, and Li Zhang. Real-time photorealistic dynamic scene representation and rendering with 4d gaussian splatting. *arXiv preprint arXiv:2310.10642*, 2023. [3](#), [6](#)
- [29] Lin Zeng, Boming Zhao, Jiarui Hu, Xujie Shen, Ziqiang Dang, Hujun Bao, and Zhaopeng Cui. Gaussianupdate: Continual 3d gaussian splatting update for changing environments. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 25800–25809, 2025. [3](#)
- [30] Dongbin Zhang, Chuming Wang, Weitao Wang, Peihao Li, Minghan Qin, and Haoqian Wang. Gaussian in the wild: 3d gaussian splatting for unconstrained image collections. In *European Conference on Computer Vision*, pages 341–359. Springer, 2024. [3](#), [6](#)

ChronoGS: Disentangling Invariants and Changes in Multi-Period Scenes

Supplementary Material

6. More Implementation Details

To help readers better understand our method and reproduce our results, we provide additional implementation details in this section. For clarity and completeness, we only summarize the most relevant aspects here. Please refer to our code for further details and full implementation.

6.1. Environment Setup

All experiments are conducted on a Linux workstation equipped with NVIDIA RTX A6000 48GB GPUs and Intel Xeon Gold 6330 (112) @ 3.1 GHz cpus. We use PyTorch 2.4.1 with CUDA 12.1 and Python 3.8.19. We use the gsplat backend and differentiable splatting kernels provided therein. Training and evaluation on large images are performed with the default downsampling strategy of 3DGS[7].

6.2. Anchor Scaffold Construction

We follow the anchor-based scene parameterization introduced in Scaffold-GS[14]. Given a COLMAP[22] sparse reconstruction, let $\mathcal{P} = \{\mathbf{p}_j\}$ denote the sparse SfM point cloud. The goal is to construct a compact yet spatially uniform set of *anchors* $\mathcal{A} = \{a_i\}_{i=1}^{N_a}$ that provide stable geometric support for subsequent Gaussian decoding.

Spatial Grid Generation. The SfM point cloud is first enclosed by an axis-aligned bounding box. We discretize the bounding box into a 3D rectilinear grid with cell size $\Delta = 0.001$ unless otherwise specified. For each grid cell $\mathcal{G}_u$, we check whether it contains at least one SfM point:

$$\mathcal{G}_u \text{ is occupied} \iff \exists \mathbf{p}_j \in \mathcal{P} \text{ s.t. } \mathbf{p}_j \in \mathcal{G}_u.$$

Anchor Feature Initialization. Each anchor a_i maintains three types of learnable feature vectors: a period-invariant base feature $f_i^{\text{base}} \in \mathbb{R}^{d_b}$, a set of local period-varying features $f_i^{\text{var}}(t) \in \mathbb{R}^{T \times d_v}$. And a global period feature $g(t) \in \mathbb{R}^{T \times d_g}$ shared across anchors. Unless otherwise specified, all anchors' feature vectors are initialized with zero initialization, which allows the model to learn meaningful structure purely from data. At initialization, the anchor distribution mainly captures the coarse occupied volume of the scene, while the detailed geometry and appearance are progressively refined through the Gaussian clusters decoded from these features during training.

Anchor Filter. To accelerate rendering, we precompute a coarse visibility mask for each anchor. For every camera

C_j , an anchor a_i is marked as potentially visible if it lies inside the camera frustum and in front of the camera center. This frustum-based filtering significantly reduces runtime culling overhead and is reused by our method without modification.

Anchor Growth and Pruning. We maintains a dynamic anchor set instead of a fixed scaffold. During training, anchors are periodically added or removed based on simple reconstruction statistics. In the growth step, regions with large residuals or strong gradients are identified in a coarse-to-fine hierarchy, and new anchors are inserted to increase local capacity. In the pruning step, anchors with consistently low visibility, opacity, or gradient magnitude are removed as redundant. This anchor-level adaptation keeps the scaffold compact while allocating more anchors to challenging regions.

6.3. Gaussian Cluster Prediction

Each anchor decodes a local cluster of Gaussian primitives, which collectively represent local shape and appearance. Instead of using a single large MLP to predict all Gaussian parameters, we use three lightweight MLPs to decode geometry, opacity, and appearance separately. We find this design more stable in practice. We use $K = 10$ Gaussians per anchor. Our MLPs employ a lightweight two-layer architecture: a fully connected layer with ReLU activation mapping the input to a hidden representation of dimension d_f , followed by an output layer producing task-specific parameters. The opacity MLP uses tanh activation, the color MLP uses sigmoid activation, and the covariance MLP has no final activation to allow unconstrained geometric parameters.

6.4. Training Strategy

We train our model using the Adam optimizer and employing differentiated learning rates for different parameter groups. Anchor positions remain fixed throughout training, while offset parameters use an exponential decay schedule from $0.01 \times \text{spatial_lr_scale}$ to $0.0001 \times \text{spatial_lr_scale}$, where spatial_lr_scale corresponds to the scale of scene. MLP networks for opacity, covariance, and color prediction use exponential decay schedules with initial learning rates of 0.002, 0.004, and 0.008, decaying to 0.00002, 0.004, and 0.00005, respectively. Geometric parameters, including anchor features, scaling, and rotation use fixed learning rates of 0.0075, 0.007, and 0.002. Our training process consists of three distinct phases: during the initialization phase (iterations 0–500), we initialize model pa-

	Scene Street1			
Settings	PSNR↑	SSIM↑	LPIPS↓	Mem.↓
$\dim(g(t)) = 16 * T$	24.55	0.7991	0.3333	0.42GB
$\dim(g(t)) = 24 * T$	24.25	0.7948	0.3389	0.36GB
$\dim(g(t)) = 32 * T$	24.57	0.7972	0.3241	0.52GB
$\dim(g(t)) = 48 * T$	23.94	0.7790	0.3623	0.58GB
$\dim(g(t)) = 64 * T$	24.25	0.7921	0.3426	0.43GB

Table 4. **Ablation on the dimension $T \times d_g$ of the global feature $g(t)$.** We evaluate on Scene *Street1*. A small dimension limits scene-level change modeling, while a large one increases memory with no consistent quality gain. The best balance is achieved at $d_g = 32$.

	Scene Aerial1			
Settings	PSNR↑	SSIM↑	LPIPS↓	Mem.↓
$\dim(f_i^{\text{var}}(t)) = 8 * T$	28.35	0.8478	0.2631	0.49GB
$\dim(f_i^{\text{var}}(t)) = 16 * T$	28.80	0.8562	0.2509	0.63GB
$\dim(f_i^{\text{var}}(t)) = 24 * T$	28.62	0.8521	0.2573	0.90GB
$\dim(f_i^{\text{var}}(t)) = 32 * T$	28.61	0.8512	0.2572	0.74GB

Table 5. **Ablation on the dimension $T \times d_v$ of the global feature $f_i^{\text{var}}(t)$.** We evaluate on Scene *Aerial1*. A small dimension limits anchor-level structure modeling, while a large one increases memory with no consistent quality gain. So we choose $d_v = 16 * T$.

rameters and begin standard gradient-based optimization; in the densification phase (iterations 500–20,000), we first collect gradient and opacity statistics from iterations 500–1,500, then perform active densification every 100 iterations from iteration 1,500 onwards, where we grow new anchors at offsets with gradient norms exceeding the threshold $\tau_g = 0.0002$ and visibility above $0.5 \times \text{success_threshold} \times \text{update_interval}$, while pruning anchors with accumulated opacity below $\text{min_opacity} \times \text{anchor_demon}$ and sufficient training statistics; finally, in the pure optimization phase (iterations 20,000–40,000), we disable densification, clean up accumulated statistics, and continue pure parameter optimization without structural changes to the anchor set.

7. More Ablation Studies

7.1. Ablation on Feature Dimensions

We further ablate the dimensions of the three feature components in our period-aware representation: the global period-varying feature $g(t)$, the local period-varying feature $f_i^{\text{var}}(t)$, and the base feature f_i^{base} . These dimensions control the capacity allocated to scene-level period-varying changes, local period-specific variations, and stable, period-invariant structure.

For $g(t)$, as shown in Tab. 4, we sweep $d_g \in \{16, 24, 32, 48, 64\}$ and observe that $d_g = 32$ offers the best quality. For the period-varying feature $f_i^{\text{var}}(t)$, as shown in

	Scene Aerial2			
Settings	PSNR↑	SSIM↑	LPIPS↓	Mem.↓
$\dim(f_i^{\text{base}}) = 8$	26.00	0.8340	0.2011	0.87GB
$\dim(f_i^{\text{base}}) = 16$	26.50	0.8467	0.1854	1.10GB
$\dim(f_i^{\text{base}}) = 24$	26.41	0.8478	0.1850	1.20GB
$\dim(f_i^{\text{base}}) = 32$	26.38	0.8452	0.1867	1.30GB

Table 6. **Ablation on the dimension d_b of the global feature f_i^{base} .** We evaluate on Scene *Aerial2*. A small dimension limits scene-level appearance modeling, while a large one exhibits no dominant quality gain but costs more storage. We use $d_b = 16$ for the trade-off.

	Scene Street2			
Settings	PSNR↑	SSIM↑	LPIPS↓	
Uni. Init.	26.47	0.8182	0.3011	
Gaussian Init.	26.11	0.8155	0.3024	
Zero Init.	26.56	0.8220	0.2946	

Table 7. **Ablation on anchor feature initialization.** Evaluated on *Street2*. We compare three initialization strategies for anchor features: uniform sampling, Gaussian sampling, and zero initialization. Zero initialization yields the best overall reconstruction quality, suggesting that our framework does not require pre-imposed feature diversity to achieve stable optimization.

Tab. 5, we test $d_v \in \{8, 16, 24, 32\}$, where small d restricts temporal expressiveness, and large d increases cost with diminishing returns. For the base feature f_i^{base} , as shown in Tab. 6, we evaluate $d_b \in \{8, 16, 24, 32\}$ and find that moderate dimensions (e.g., $d_b = 16$) provide the best trade-off between accuracy and memory cost.

Overall, the configuration $(d_b, d_v, d_g) = (16, 16, 32)$ provides the best balance between reconstruction quality and storage cost.

7.2. Effect of Anchor Feature Initialization

To study the impact of anchor feature initialization, we evaluate three alternatives to the feature initialization: uniform sampling, Gaussian sampling, and zero initialization. As shown in Tab. 7, zero initialization achieves the best performance across all metrics, indicating that our model can effectively learn discriminative anchor features even when starting from a fully collapsed initialization. Uniform initialization provides competitive performance, while Gaussian initialization shows slightly degraded accuracy. These results demonstrate that our approach is robust to the choice of initialization and does not rely on carefully crafted feature priors.

Scene	Period 0		Period 1		Period 2		Period 3		Total	
	Images	Points	Images	Points	Images	Points	Images	Points	Images	Points
Lawncourt	234	85,655	193	57,884	277	117,595	N/A	N/A	704	261,134
Overstreet	83	35,513	295	235,537	208	132,957	N/A	N/A	586	404,007
BioBuilding	258	120,122	240	143,694	258	171,462	N/A	N/A	756	435,278
Canteen	340	101,523	183	99,915	279	159,250	N/A	N/A	802	360,688
Incubator	202	98,073	199	126,315	194	125,461	N/A	N/A	595	349,849
TriPlot	365	150,896	94	50,081	189	106,937	N/A	N/A	648	307,914
Street1	200	110,591	200	108,932	200	95,038	200	101,074	800	415,635
Street2	200	108,621	200	112,293	200	101,061	200	110,714	800	432,689
Aerial1	200	136,491	200	116,790	200	136,624	200	101,494	800	491,399
Aerial2	200	116,443	200	115,481	200	109,213	200	116,434	800	457,571
Aerial3	200	171,323	200	145,781	200	139,733	200	73,597	800	530,434
Aerial4	200	144,446	200	133,058	200	129,169	200	92,044	800	498,717
Total	2,682	1,379,697	2,404	1,445,761	2,605	1,524,500	1,200	595,357	8,891	4,945,315

Table 8. Point Cloud and Image Counts by Scene and Period

8. More Qualitative Results

In this section, we provide additional qualitative comparisons that complement the results shown in the main paper. First, we include comparisons with 3DGS[7] and realtime4DGS[28], which are omitted from the qualitative figures in the main paper due to space constraints. These results further demonstrate the advantages of our method in handling multi-period geometry and appearance variations.

Image Demonstrations. We additionally present figures that contain representative qualitative results for every scene. These visualizations provide a complete overview of reconstruction quality and temporal consistency across the entire dataset. Please refer to Figure 9 and Figure 10 for the qualitative results.

Video Demonstrations. To provide a more intuitive comparison between our method and static or dynamic baselines, we also include video demonstrations in our supplementary material. These videos show how our method faithfully reconstructs complex geometry and appearance changes across periods. However, static methods, when trained on multi-period datasets that contain both geometric and appearance changes, tend to produce blurred reconstructions in the varying regions and fail to model the distinct scene states of each period. Dynamic methods, on the other hand, rely on assumptions of temporal continuity; when applied to discrete periods, they often yield poor intermediate states or introduce significant artifacts.

9. More Information About ChronoScene

To provide a comprehensive overview of the ChronoScene dataset, we present detailed statistics and visualizations that

characterize the temporal distribution of both images and point clouds across different scenes. Tab. 8 summarizes the quantitative distribution of images and point clouds for each scene across different time periods, revealing the scale and temporal distributions of our dataset.

To further illustrate the spatial-temporal structure of the point cloud data, we visualize the **sparse point cloud distributions** projected onto three orthogonal planes (XY, XZ, and YZ) for each scene, as shown in Figs. 11 to 22. These visualizations reveal how point clouds are distributed across different time periods within the same coordinate system, with each period represented by a distinct color (red, yellow, blue, and green for periods 0, 1, 2, and 3, respectively). The projections highlight both the spatial coverage and temporal variations in the point cloud density. These figures demonstrate that our dataset encompasses a diverse range of scenes with varying temporal characteristics, from street scenes to aerial views.

These analyses collectively demonstrate that ChronoScene provides a rich and diverse dataset for temporal 3D scene understanding, with balanced temporal distributions and comprehensive spatial coverage across multiple scene types.

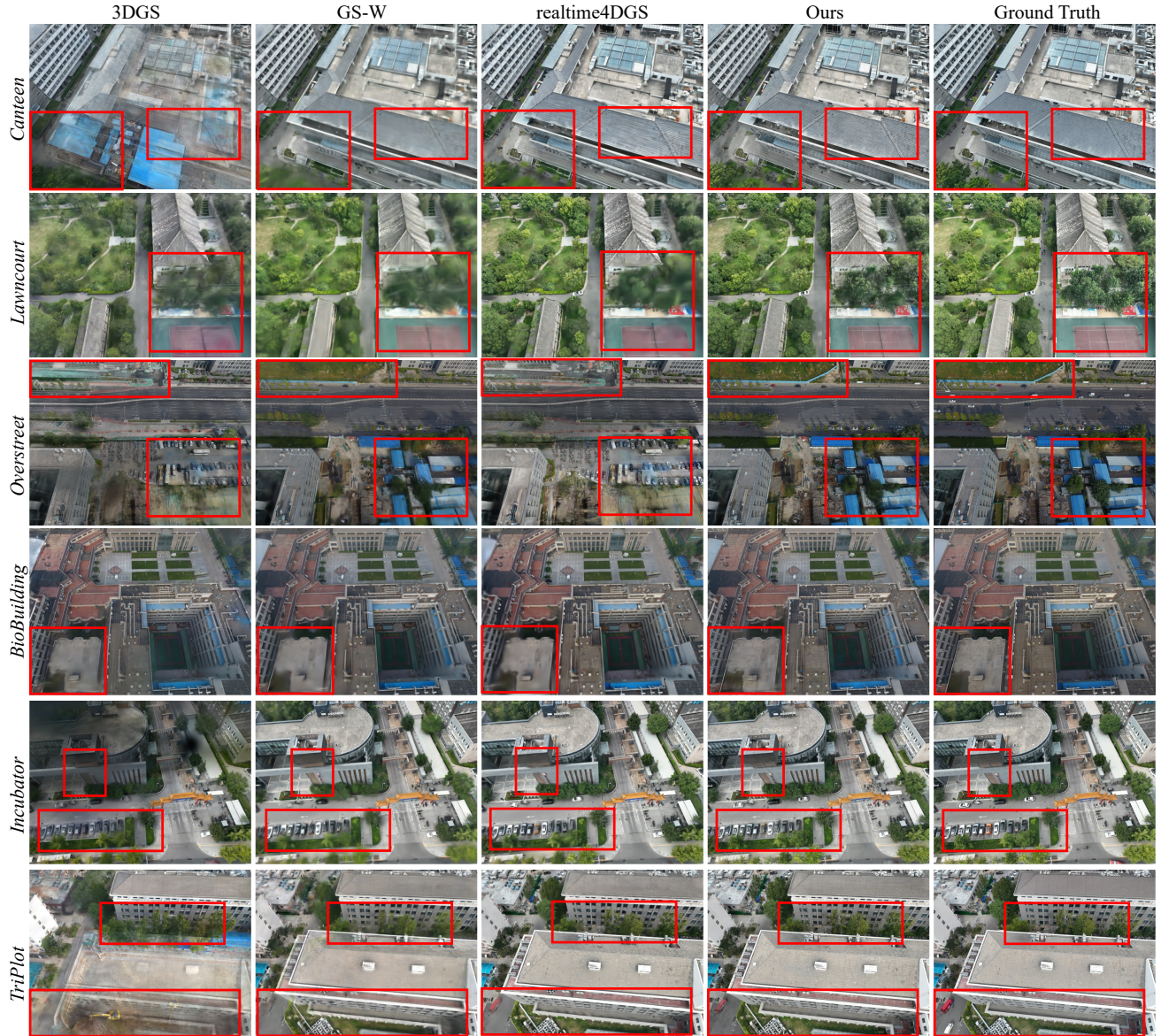


Figure 9. **Qualitative comparison on real scenes of ChronoScene.** We compare ChronoGS with representative baselines. Each row shows novel-view renderings of different methods, along with ground truth. Static models trained on mixed multi-period data produce ghosting and appearance blending due to inconsistent geometry, while dynamic methods that assume continuous motion fail under the large temporal gaps, causing incorrect geometry interpolation. **ChronoGS** faithfully reconstructs period-specific geometry and appearance, preserving sharp, temporally consistent details across long-term changes.

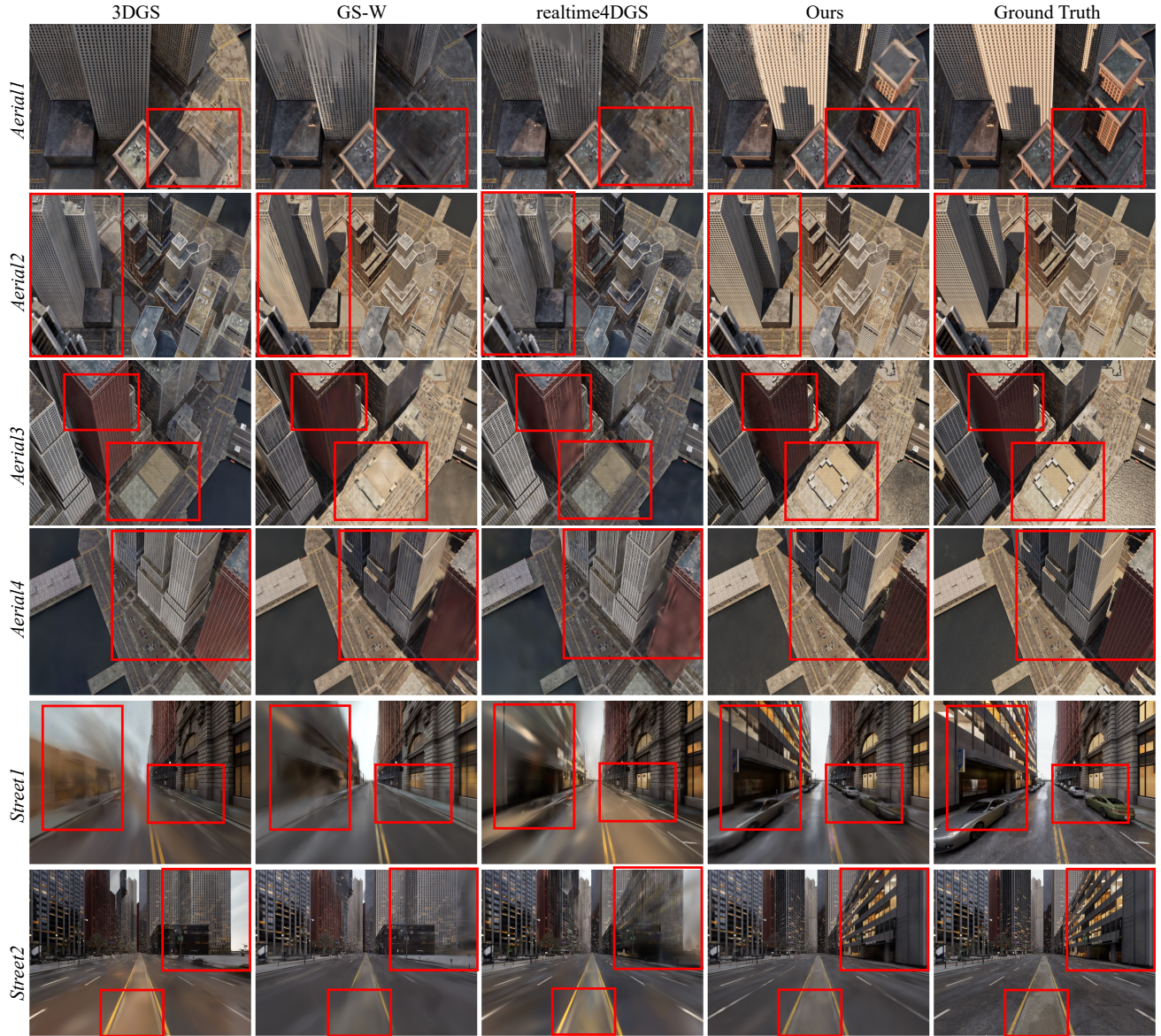


Figure 10. **Qualitative comparison on synthetic scenes of ChronoScene.** We compare ChronoGS with representative baselines. Each row shows novel-view renderings of different methods, along with ground truth. Static models trained on mixed multi-period data produce ghosting and appearance blending due to inconsistent geometry, while dynamic methods that assume continuous motion fail under the large temporal gaps, causing incorrect geometry interpolation. **ChronoGS** faithfully reconstructs period-specific geometry and appearance, preserving sharp, temporally consistent details across long-term changes.

Point Cloud Projections by Period: Lawncourt

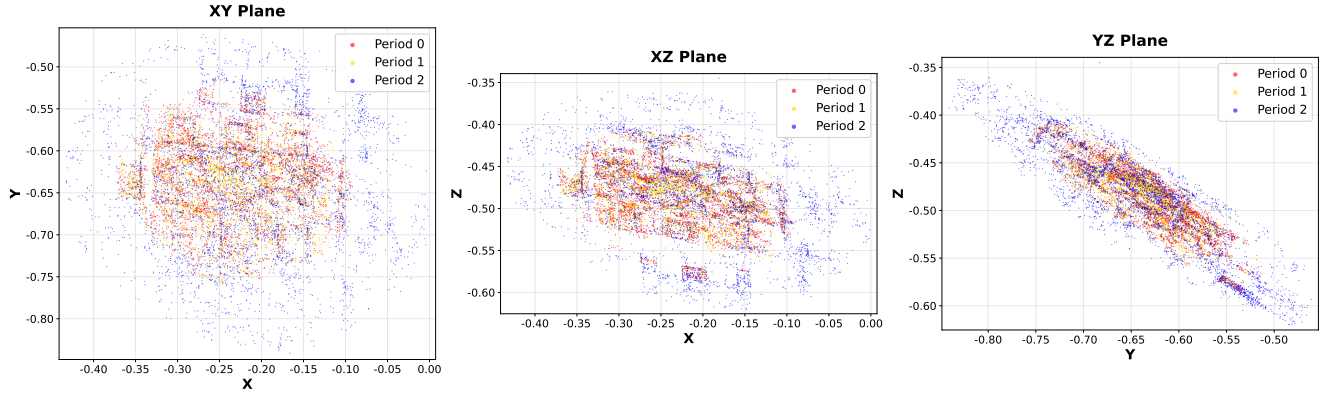


Figure 11. Point cloud distribution of scene *Lawncourt*.

Point Cloud Projections by Period: Overstreet

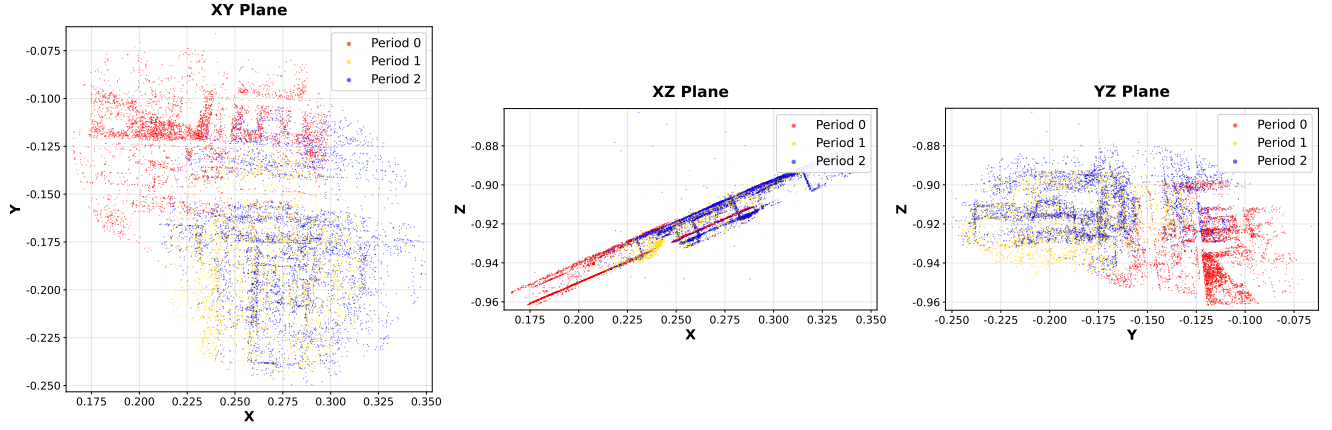


Figure 12. Point cloud distribution of scene *Overstreet*.

Point Cloud Projections by Period: BioBuilding

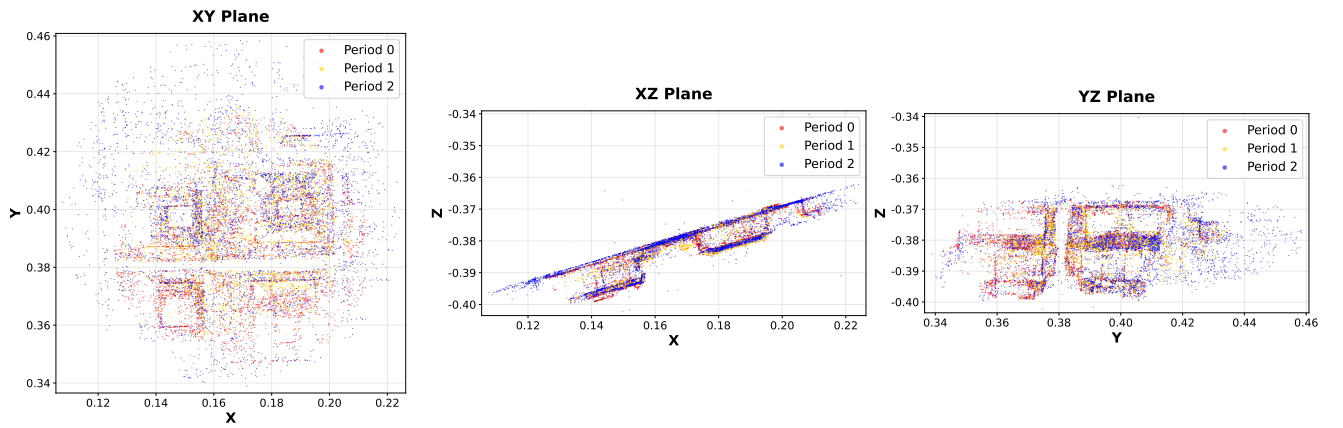


Figure 13. Point cloud distribution of scene *BioBuilding*.

Point Cloud Projections by Period: Canteen

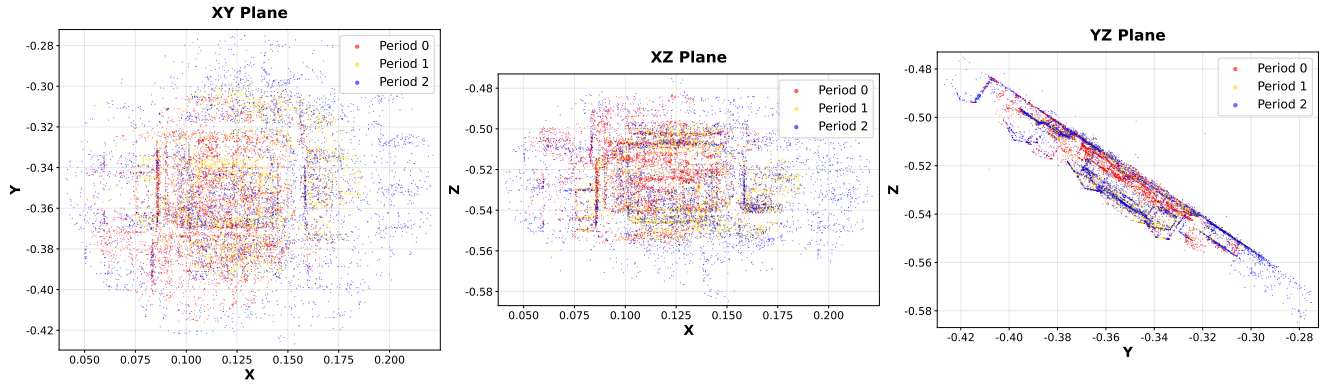


Figure 14. Point cloud distribution of scene *Canteen*.

Point Cloud Projections by Period: Incubator

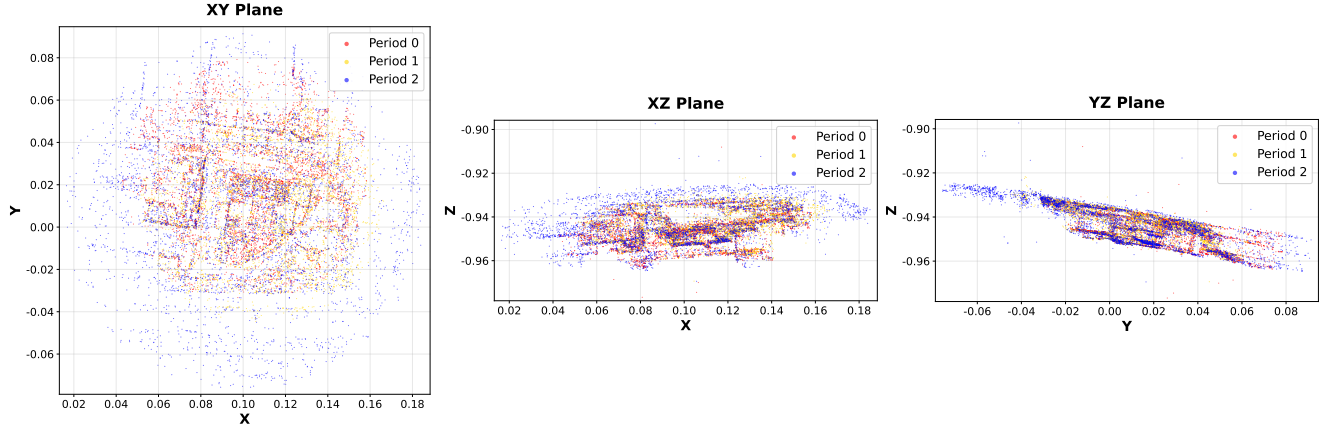


Figure 15. Point cloud distribution of scene *Incubator*.

Point Cloud Projections by Period: TriPlot

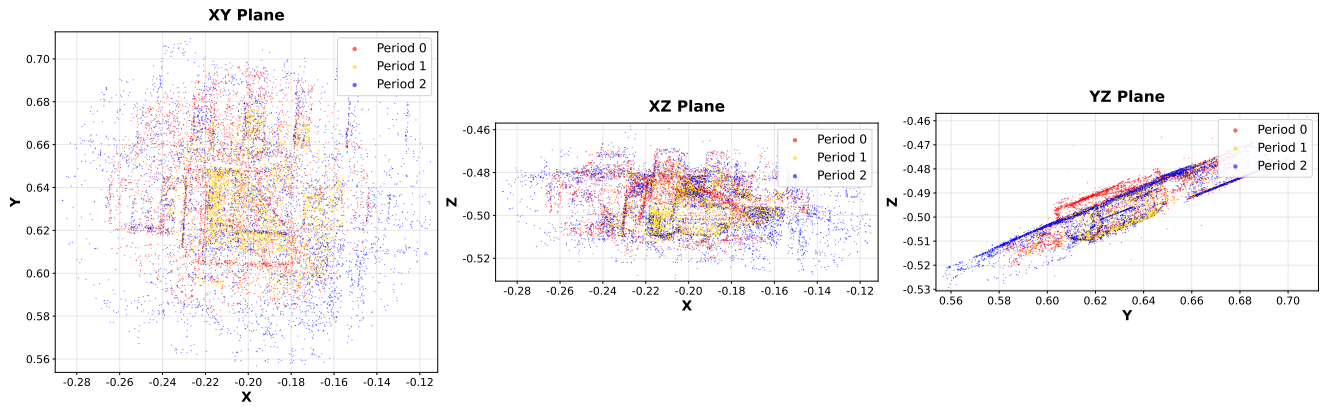


Figure 16. Point cloud distribution of scene *TriPlot*.

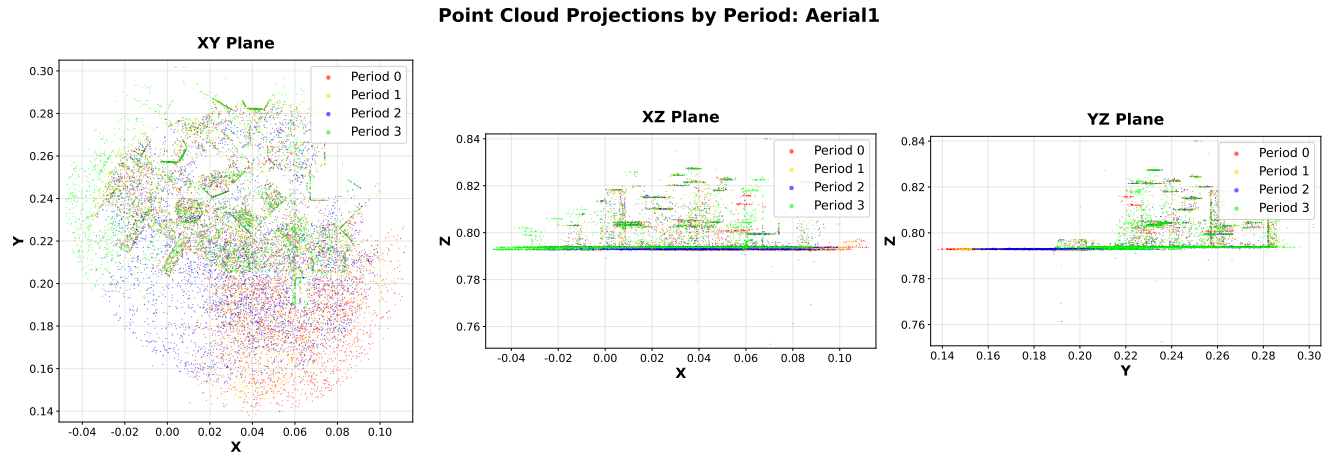


Figure 17. Point cloud distribution of scene *Aerial1*.

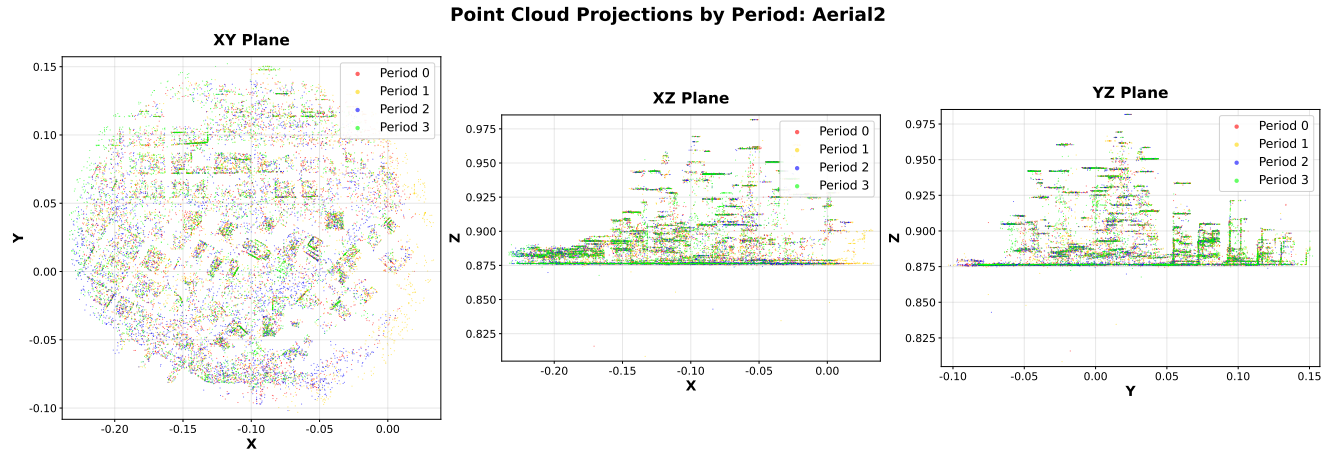


Figure 18. Point cloud distribution of scene *Aerial2*.

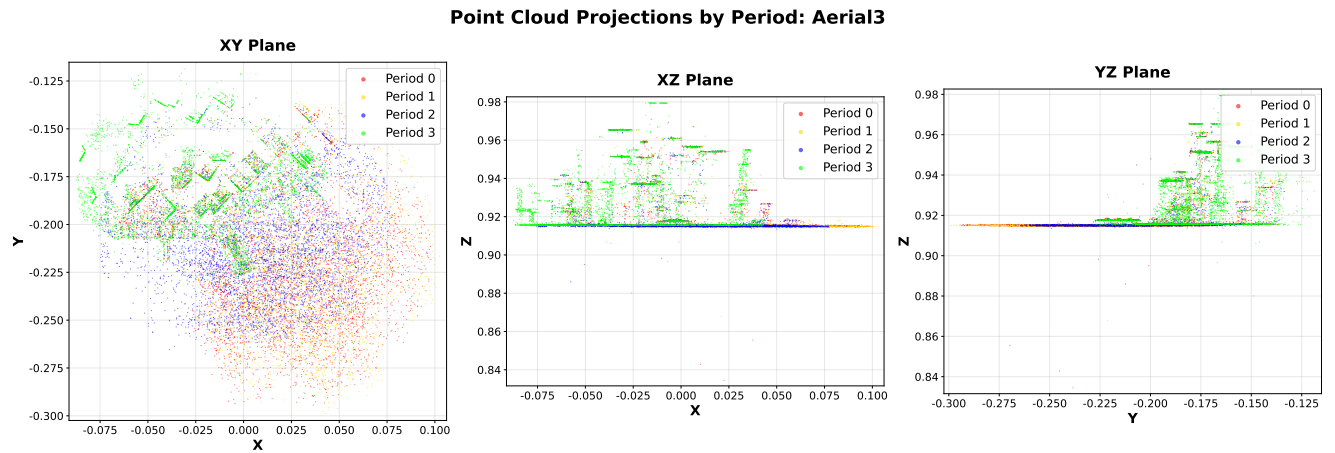


Figure 19. Point cloud distribution of scene *Aerial3*.

Point Cloud Projections by Period: Aerial4

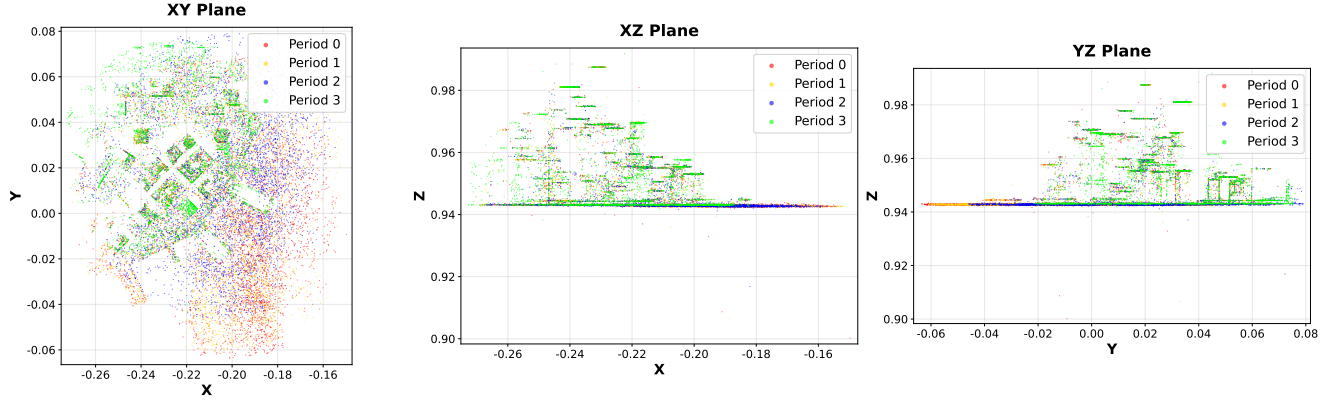


Figure 20. Point cloud distribution of scene *Aerial4*.

Point Cloud Projections by Period: Street1

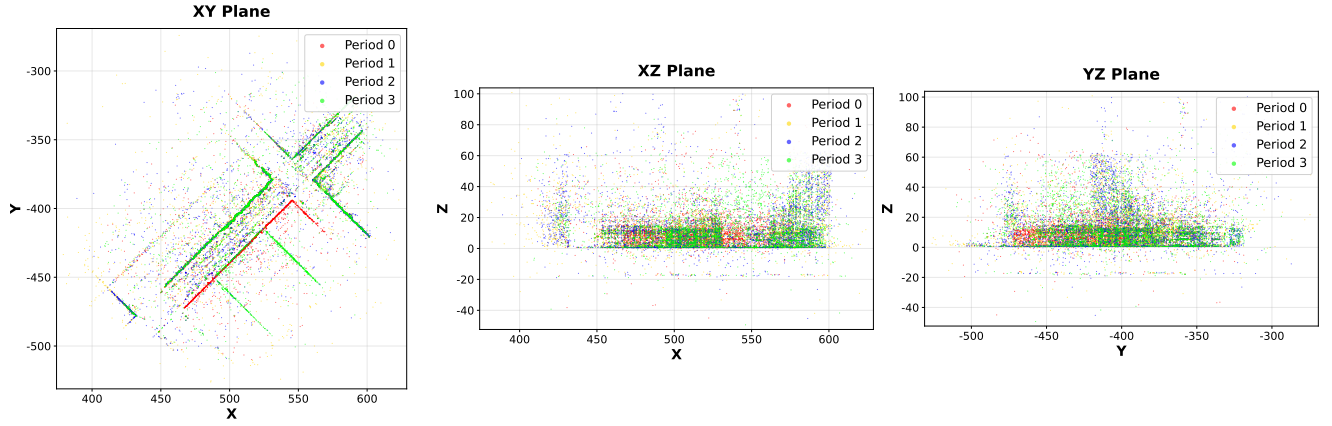


Figure 21. Point cloud distribution of scene *Street1*.

Point Cloud Projections by Period: Street2

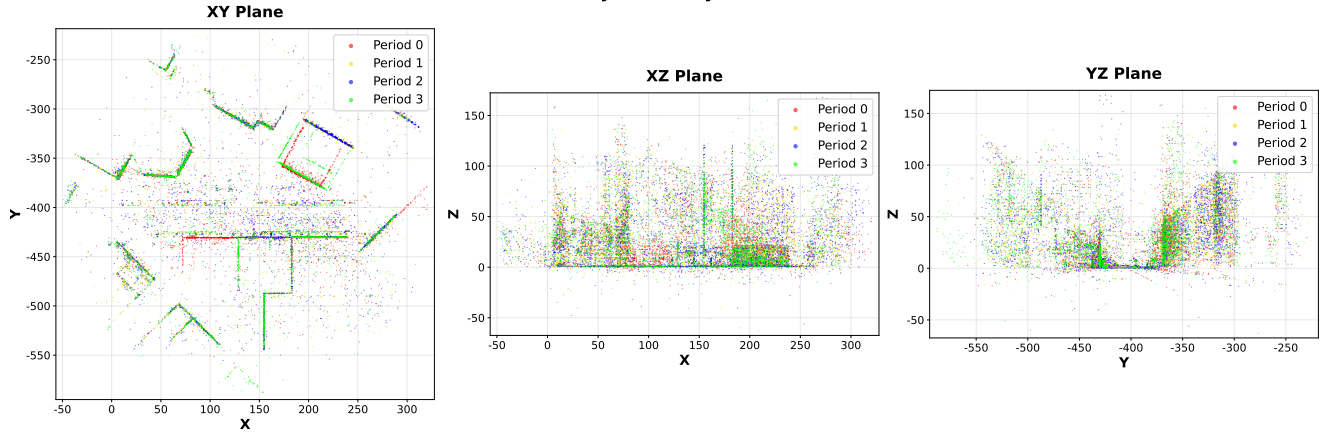


Figure 22. Point cloud distribution of scene *Street2*.