ADAPTING 3D FOUNDATIONS TO SPIKE CAMERAS AT TEST TIME

Anonymous authors
Paper under double-blind review

ABSTRACT

3D foundations have substantially advanced novel-view synthesis (NVS) by pretraining on large-scale RGB datasets. However, RGB cameras commonly operate at 30–60 Hz, providing limited temporal resolution for rapid scene motion. This limited temporal resolution restricts the direct application of 3D foundations to high-speed scenes. Spike cameras provide an attractive alternative that captures high-speed scenes as binary streams at up to 40 kHz. To exploit these high-rate spike streams for NVS, we propose **SpikeTTT**, a framework that adapts 3D foundations to spike cameras at test time. During test-time training, SpikeTTT freezes the pretrained weights and optimizes only lightweight adapters through a tri-signal self-supervised objective. In this way, SpikeTTT adapts 3D foundations to the spike-camera distribution while retaining the benefits of large-scale RGB pretraining. We implement and evaluate SpikeTTT with diverse 3D foundations and NVS tasks. Our evaluation covers posed and pose-free settings on both static and dynamic datasets. With AnySplat, SpikeTTT improves rendering PSNR relative to the frozen cascade by 6.98 dB on Tanks and Temples and 4.54 dB on DL3DV. It also outperforms spike-specific baselines. SpikeTTT thus provides a practical route to high-quality novel-view synthesis in high-speed scenes by bringing large-scale RGB pretraining to spike-based 3D reconstruction.

1 INTRODUCTION

Novel-view synthesis (NVS) is a fundamental task in the field of 3D vision (Mildenhall et al., 2020; Kerbl et al., 2023). 3D foundations (Jiang et al., 2025; Szymanowicz et al., 2026; Lin et al., 2026) have substantially advanced NVS by learning geometric and appearance priors from large-scale RGB data. However, this capability is inherited from the pretraining distribution and therefore depends on the sharpness of RGB inputs. In practice, this assumption is often not satisfied because RGB cameras produce motion blur under rapid camera or scene motion.

Bio-inspired spike cameras provide an alternative sensing modality for high-speed scenes. Compared to RGB cameras that record images at 30–60 Hz, spike cameras operate at sampling rates of up to 40 kHz, as shown in Figure 1. Existing spike-based methods achieve novel-view synthesis with task-specific 3D pipelines, and further extend to dynamic or unposed settings (Guo et al., 2025; Chen et al., 2025; Ye et al., 2026; Guo et al., 2026). They demonstrate the viability of spike-based NVS in their respective settings. However, their task-specific and spike-tailored strategies are tightly coupled to particular scene representations and task configurations. This makes it difficult to apply spike cameras across diverse NVS tasks and to transfer advances in 3D foundations to these tasks.

We introduce **SpikeTTT** to address this limitation, as shown in Figure 1. SpikeTTT starts from a simple Spike NVS pipeline: a frozen cascade in which a spike-to-RGB reconstructor produces RGB proxies for a 3D foundation. In practice, the reconstructed images remain imperfect RGB proxies: temporal aggregation blurs spatial detail, while learning-based reconstructors introduce noise and photometric bias. These reconstructed images also fail to satisfy the sharpness assumption underlying the pretraining distribution of 3D foundations. Consequently, this frozen cascade *still faces a distribution mismatch, compromising the resulting 3D scene representation and rendering.*

To mitigate this mismatch, SpikeTTT adapts this cascade to spike-derived observations at test time. During test-time training, SpikeTTT optimizes lightweight adapters in the cascade using a tri-signal self-supervised objective. SpikeTTT first instantiates a lightweight adapter strategy according to

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

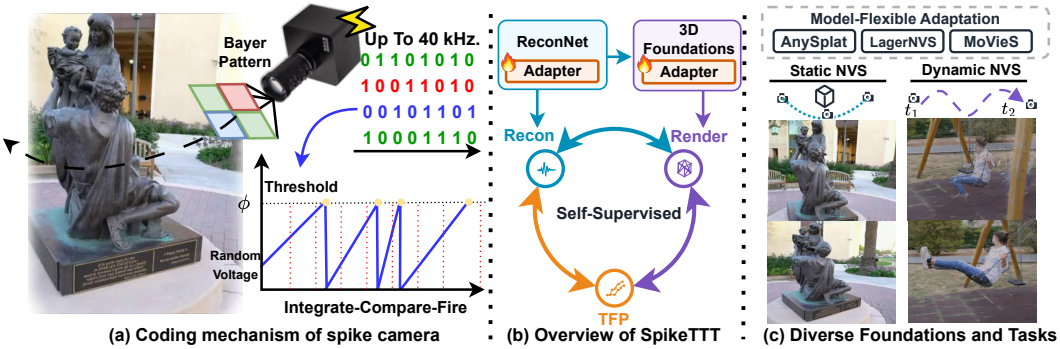


Figure 1: **Overview of SpikeTTT.** (a) Coding mechanism of the spike camera. (b) Test-time training of a pretrained reconstructor and a 3D foundation. (c) The same protocol instantiates diverse 3D foundations on static and dynamic NVS tasks.

the native computational structure: zero-initialized convolutional residuals are attached to convolutional paths, whereas low-rank updates are applied to Transformer blocks. Furthermore, a tri-signal self-supervised objective is introduced. It couples three signals: the reconstructed RGB proxies, the foundation rendering, and temporal filtering by photons (TFP) targets derived from the spike stream. These three signals play complementary roles during test-time training and help to mitigate the distribution mismatch by optimizing the adapters. Moreover, we highlight that SpikeTTT is a model-flexible protocol that allows diverse 3D foundations to adopt the same lightweight-adaptation strategy while retaining their frozen base rendering architectures. This makes it possible to transfer advances in diverse 3D foundations to spike-based NVS tasks.

We evaluate SpikeTTT in posed and pose-free settings across static scenes from Tanks and Temples and DL3DV-Evaluation (Knapitsch et al., 2017; Ling et al., 2024) and dynamic scenes from DAVIS and DyCheck (Perazzi et al., 2016; Gao et al., 2022). These results demonstrate improved reconstruction and novel-view quality over frozen cascades. SpikeTTT also achieves higher overall quality than task-specific spike baselines.

In summary, the contributions of SpikeTTT are as follows:

- We introduce **SpikeTTT**, a model-flexible framework that couples spike reconstructors with 3D foundations. It supports diverse 3D foundations and novel-view synthesis tasks in high-speed scenes.
- We formulate SpikeTTT as a test-time training protocol for adapting 3D foundations to spike-derived observations. It utilizes a tri-signal self-supervised objective to optimize lightweight adapters while retaining the pretrained weights.
- We evaluate SpikeTTT across diverse 3D foundations and task settings, covering posed and pose-free settings as well as static and dynamic scenes. With AnySplat, SpikeTTT raises rendering PSNR relative to its frozen cascade by 6.98 dB on Tanks and Temples and 4.54 dB on DL3DV. It also achieves higher quality than task-specific spike baselines.

2 Related Work

2.1 Spike Camera Imaging

Spike cameras are bio-inspired neuromorphic sensors that record pixel-wise irradiance as high-rate binary streams (Huang et al., 2023). Spike camera imaging methods consist of physics-based intensity estimation and learning-based reconstruction. Physics-based methods (Zhu et al., 2019) recover intensity from spike counts and timing cues, but temporal aggregation and spike fluctuations introduce blur and noise (Zhu et al., 2019; Zhao et al., 2024). Learning-based reconstructors improve sharpness through temporal fusion and joint demosaicing and denoising (Zhao et al., 2021; Chen et al., 2022; Dong et al., 2024), yet they retain noise and photometric biases. These reconstructed images serve as inputs or intermediate representations for downstream spike-vision tasks, such as depth estimation (Zhang et al., 2022; Gao et al., 2026), optical flow estimation (Hu et al., 2022) and 3D reconstruction (Guo et al., 2025; Chen et al., 2025; Guo et al., 2026).

2.2 SPIKE-BASED NOVEL VIEW SYNTHESIS

Pretrained 3D foundation models, such as AnySplat, LagerNVS, and MoVieS, have substantially advanced novel-view synthesis (NVS) by learning geometric and appearance priors from large-scale RGB data (Jiang et al., 2025; Szymanowicz et al., 2026; Lin et al., 2026). However, RGB cameras suffer from motion blur under rapid scene motion, limiting the direct application of 3D foundations in high-speed scenes. Spike cameras provide an alternative sensing modality that records high-speed scenes as dense binary streams. This motivates recent efforts to incorporate spike cameras into NVS (Guo et al., 2024; Dai et al., 2024; Guo et al., 2025; Chen et al., 2025). However, these methods rely on spike-specific reconstruction and optimization: SpikeGS utilizes accumulation rasterization and interval supervision, whereas USP-Gaussian jointly optimizes spike reconstruction, camera poses, and 3D Gaussians (Guo et al., 2025; Chen et al., 2025). Subsequent work further extends spike-based NVS to dynamic scenes and pose-free settings (Ye et al., 2026; Guo et al., 2026). While these methods have shown promise, they are limited by the spike-specific strategies for each task setting. To address this limitation, SpikeTTT proposes a model-flexible protocol that adapts 3D foundations to spike observations. This protocol supports diverse 3D foundation models and NVS settings, spanning static and dynamic scenes as well as posed and pose-free inputs.

2.3 TEST-TIME ADAPTATION

Test-time adaptation addresses distribution shifts using unlabeled target data. Relevant studies investigate self-supervised test-time training, adaptation failure modes, and adaptation to changing data streams (Sun et al., 2020; Liu et al., 2021; Yuan et al., 2023). For 3D reconstruction, Test3R provides a scene-level example by freezing a pretrained point-map backbone and optimizing visual prompts using consistency across RGB image pairs (Jia et al., 2022; Yuan et al., 2026). Inspired by this scene-level adaptation paradigm, SpikeTTT applies test-time training to a cascade comprising a spike-to-RGB reconstructor and a pretrained RGB NVS model. It couples a spike reconstructor with a 3D foundation and applies a shared adaptation strategy with tri-signal self-supervision. It adapts the 3D foundation to spike-derived observations while retaining the pretrained geometric and appearance priors without modifying the base architecture for each task setting.

3 SPIKE CAMERA PRELIMINARIES

Spike cameras are bio-inspired neuromorphic sensors that record pixel-wise irradiance as a high-rate binary stream (Huang et al., 2023). These sensors synchronously read out binary spike frames at up to 40 kHz; each frame records whether a pixel fires within that sampling interval. Their sensing and coding processes are illustrated in Figure 1.

Specifically, each pixel follows an integrate–compare–fire mechanism. The pixel accumulates photocurrent from the incident irradiance until the accumulated charge reaches a firing threshold, emits a spike, and then resets its integrator. For pixel (u, v) at time t, let $L_C(u, v, t)$ denote the photocurrent on the corresponding Bayer color channel. We also model dark-current fluctuations as additive physical noise $\eta_{\mathrm{dark}}(u, v, t)$. The latent accumulated signal is

$$A(u, v, t) = \left(\int_0^t \left[L_C(u, v, \tau) + \eta_{\mathrm{dark}}(u, v, \tau) \right] d\tau \right) \bmod \phi, \tag{1}$$

where ϕ is the firing threshold. Whenever $A(u, v, t)$ crosses ϕ, the pixel outputs a binary spike and the accumulator is reset. This yields a spike stream $\mathcal{S} = \{\mathbf{S}_i\}_{i=1}^{T}$ of binary spike observations, where $\mathbf{S}_i \in \{0, 1\}^{H \times W}$ denotes the i-th binary spike observation. H and W denote spatial dimensions.

Temporal filtering by photons (TFP). For a timestamp t_i, TFP forms an image from the exposure window $\mathcal{T}_{t_i} := [t_i - \Delta T/2,\ t_i + \Delta T/2]$, whose duration is ΔT. The spike count at pixel (u, v)

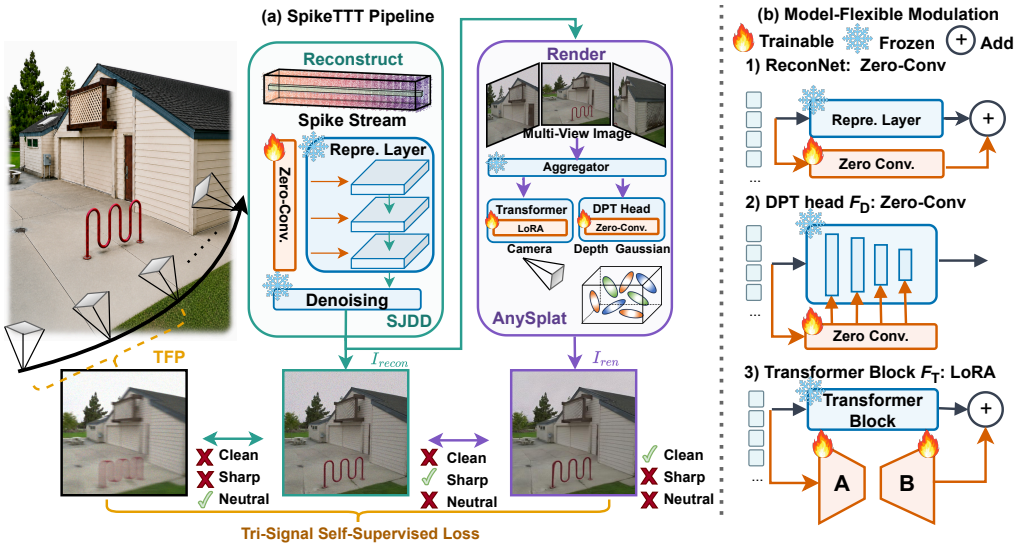


Figure 2: **Overview of SpikeTTT.** A pretrained spike reconstructor maps the spike stream to RGB proxies, and a 3D foundation renders novel views. Only lightweight adapters are updated, while all pretrained weights remain frozen. During test-time training, a tri-signal self-supervised loss couples reconstruction, rendering, and TFP to update the adapters.

within this window is $N_{u,v}(\mathcal{T}_{t_i}) := \sum_{k:\, t_k \in \mathcal{T}_{t_i}} \mathbf{S}_k(u,v)$.

$$
\begin{aligned}
I_{\mathrm{tfp}}(u,v,t_i) &:= \frac{C\, N_{u,v}(\mathcal{T}_{t_i})}{\Delta T} = \frac{C}{\Delta T} \sum_{k:\, t_k \in \mathcal{T}_{t_i}} \mathbf{S}_k(u,v) \\
&\approx \frac{1}{\Delta T} \int_{t_i - \Delta T/2}^{t_i + \Delta T/2} \left[L_C(u,v,\tau) + \eta_{\mathrm{dark}}(u,v,\tau) \right] d\tau,
\end{aligned}
\tag{2}
$$

where C is the sensor-dependent conversion constant. Thus, $I_{\mathrm{tfp}}(u,v,t_i)$ is the spike-count rate at pixel (u,v) and approximates the corresponding conventional long-exposure measurement over $\mathcal{T}_{t_i}$.

4 Method

SpikeTTT is a test-time training framework that adapts a pretrained spike reconstructor and a 3D foundation to spike-derived observations for NVS in high-speed scenes. The overall framework is shown in Figure 2. In this section, we first introduce the Spike NVS pipeline, the problem formulation and integration protocol in Section 4.1. Then, we present the test-time training protocol in Section 4.2, and define the self-supervised objective in Section 4.3.

4.1 Problem Formulation and Integration Protocol

We consider a 3D scene reconstruction task from spike-camera observations. The spike stream $\mathcal{S} = \{\mathbf{S}_i\}_{i=1}^{T}$ is represented as a sequence of binary spike observations, where $\mathbf{S}_i \in \{0,1\}^{H \times W}$ denotes the i-th binary spike observation, and H and W denote its spatial dimensions.

Spike NVS pipeline. The Spike NVS pipeline consists of two components: a spike-to-RGB reconstructor R_{θ_s} and a 3D foundation $\mathcal{F}_{\phi_s}$. This generic pipeline accommodates different 3D foundations for posed or pose-free NVS in static or dynamic scenes. Specifically, for each timestamp t_i, we form a temporal window $\mathcal{S}_{t_i}^{(N)} \in \{0,1\}^{N \times H \times W}$ from N consecutive spike observations centered at t_i. The reconstruction network maps this window to an RGB proxy $\mathbf{I}_{\mathrm{rec}}(t_i) = R_{\theta_s}\left(\mathcal{S}_{t_i}^{(N)}\right)$. The resulting T timestamp-indexed RGB proxies and a target camera pose $\mathbf{P}_{\mathrm{tar}}$ are supplied to the foundation as the multi-view input and query camera pose, respectively. A target camera pose is

provided externally or inferred by the 3D foundation. The foundation then renders the query image:

$$\mathbf{I}_{\mathrm{ren}}(\mathbf{P}_{\mathrm{tar}}) := \mathcal{F}_{\phi_s}\Big(\{\mathbf{I}_{\mathrm{rec}}(t_i)\}_{i=1}^{T}, \mathbf{P}_{\mathrm{tar}}\Big). \tag{3}$$

The Spike NVS pipeline is summarized by

$$\Big\{\mathcal{S}_{t_i}^{(N)}\Big\}_{i=1}^{T} \xrightarrow{R_{\theta_s}} \{\mathbf{I}_{\mathrm{rec}}(t_i)\}_{i=1}^{T} \xrightarrow{\mathcal{F}_{\phi_s}(\cdot, \mathbf{P}_{\mathrm{tar}})} \mathbf{I}_{\mathrm{ren}}(\mathbf{P}_{\mathrm{tar}}). \tag{4}$$

Observations. While the Spike NVS pipeline provides a straightforward interface, the spike-derived RGB proxies do not fully match the sharpness assumptions in the pretraining distribution of $\mathcal{F}_{\phi_s}$. The learning-based reconstructor retains comparatively sharp spatial details but introduces noise and color bias. TFP provides a photometric reference, but temporal integration blurs spatial details. TFP estimates are also affected by dark-current and spike fluctuations. These observations are illustrated in Figure 2. Consequently, this frozen cascade *still faces a distribution mismatch, compromising the resulting 3D scene representation and rendering.*

Scene-level adaptation. To address this distribution mismatch, SpikeTTT proposes test-time training that adapts the cascade to spike-derived observations. It freezes the pretrained parameters $\Theta_s = (\theta_s, \phi_s)$ and optimizes only lightweight adapters Θ_{adapt} attached to the reconstructor and foundation components, as detailed in Section 4.2. A tri-signal self-supervised objective in Section 4.3 drives the adapter optimization. In this way, our protocol adapts these two components to spike-derived observations while retaining the geometry and appearance priors. Finally, the test-time training paradigm is formulated as:

$$\Theta^*(\mathcal{S}) := \mathrm{SpikeTTT}(\Theta_s; \mathcal{S}) := \arg\min_{\Theta_{\mathrm{adapt}}} \mathcal{L}_{\mathrm{self}}(\mathcal{S}; \Theta_{\mathrm{adapt}}). \tag{5}$$

4.2 Test-Time Training Protocol

We instantiate the shared adaptation strategy according to the native computational structure of each pretrained module. Specifically, SpikeTTT adopts zero-initialized convolutions on convolutional paths (Zhang et al., 2023) and low-rank updates (Hu et al., 2021) for Transformer blocks.

Zero-initialized convolutions. SpikeTTT adopts zero-initialized convolutions on convolutional paths (Zhang et al., 2023). For a pretrained convolution F with input feature x,

$$\widetilde{F}(x) = F(x) + Z_\psi(x), \tag{6}$$

where Z_ψ is a 1×1 convolutional layer initialized to zero. This initialization introduces no residual change to the pretrained path at the start of test-time training. It avoids an abrupt initial perturbation while enabling scene-specific residual correction during adaptation.

Low-rank updates. SpikeTTT utilizes low-rank updates (Hu et al., 2021) for Transformer blocks. Given a pretrained block weight W and input feature x,

$$\widetilde{W}(x) = W(x) + \frac{\alpha}{r} BAx, \tag{7}$$

where the base weight W remains frozen and only the rank-r factor matrices A and B are optimized. This low-rank parameterization enables scene-specific adjustment of feature interactions while restricting adaptation capacity and parameter count.

In the default implementation, zero-initialized convolutions are attached to the representation layers of SJDDNet (Dong et al., 2024) and DPT-style heads of AnySplat (Jiang et al., 2025). LoRA is attached to the camera head. These two adapter families allow the pretrained backbone to adapt to spike-derived observations while keeping the pretrained base modules frozen. More details for LagerNVS (Szymanowicz et al., 2026) and MoVieS (Lin et al., 2026) are given in Appendix B.2.

4.3 TRI-SIGNAL SELF-SUPERVISION

We formulate a self-supervised objective that optimizes the lightweight adapters introduced in Section 4.2. The objective combines three complementary signals derived from $\mathcal{S}$: the reconstruction, the rendering, and TFP. Specifically, it consists of mutual supervision and TFP anchoring. Mutual supervision combines the reconstruction and rendering branches to refine texture and suppress reconstruction noise. TFP anchoring constrains both branches with the sensor-derived low-frequency photometric observation. Both components utilize the following image discrepancy:

$$\ell(\mathbf{I}_a, \mathbf{I}_b) = \lambda_1 \|\mathbf{I}_a - \mathbf{I}_b\|_1 + (1 - \lambda_1)\,(1 - \mathrm{SSIM}(\mathbf{I}_a, \mathbf{I}_b))\,. \tag{8}$$

The mutual-supervision objective aligns each reconstructed image with its corresponding rendering bidirectionally. Every pixel-wise discrepancy compares a co-view, co-temporal reconstruction–rendering pair. To reduce memory usage and decouple the two optimization paths, we detach the reconstructed inputs from the computation graph before passing them to the foundation. We then compute the bidirectional discrepancy with a stop-gradient target in each direction. Therefore, the mutual-supervision objective is:

$$\mathcal{L}_{\mathrm{mutual}} = \ell(\mathbf{I}_{\mathrm{rec}}, \mathrm{sg}(\mathbf{I}_{\mathrm{ren}})) + \ell(\mathrm{sg}(\mathbf{I}_{\mathrm{rec}}), \mathbf{I}_{\mathrm{ren}})\,, \tag{9}$$

where $\mathrm{sg}(\cdot)$ denotes stop-gradient.

For the TFP anchoring objective, we adopt the spike-accumulation correspondence inspired by SpikeGS (Guo et al., 2025). TFP is treated as an approximate temporal average in a fixed temporal window. Each TFP observation is constructed from a fixed spike window centered on the middle view of temporal group $\mathcal{G}_g$. The corresponding temporal average is approximated by the mean prediction over that group. This observation serves as a low-frequency photometric anchor for brightness and color. In mathematical terms, it is formulated as:

$$\begin{aligned} \mathbf{E}^g_{\mathrm{tfp}} &= \mathcal{T}_{\mathrm{tfp}}(\mathrm{center}(\mathcal{G}_g))\,, \qquad \bar{\mathbf{I}}^g_b = \frac{1}{|\mathcal{G}_g|} \sum_{i \in \mathcal{G}_g} \mathbf{I}_b(t_i), \quad b \in \{\mathrm{rec}, \mathrm{ren}\}, \\ \mathcal{L}_{\mathrm{tfp}} &= \frac{1}{G} \sum_{g=1}^{G} \left[\ell\Big(\bar{\mathbf{I}}^g_{\mathrm{rec}}, \mathbf{E}^g_{\mathrm{tfp}}\Big) + \ell\Big(\bar{\mathbf{I}}^g_{\mathrm{ren}}, \mathbf{E}^g_{\mathrm{tfp}}\Big) \right]. \end{aligned} \tag{10}$$

More details about the TFP anchoring objective are given in Appendix B.1. Finally, the complete self-supervised objective is

$$\mathcal{L}_{\mathrm{self}} := \mathcal{L}_{\mathrm{TTT}} = \mathcal{L}_{\mathrm{mutual}} + \mathcal{L}_{\mathrm{tfp}}. \tag{11}$$

5 EXPERIMENTS

5.1 EXPERIMENTAL SETTINGS

Datasets. The evaluation utilizes **Tanks and Temples** (Knapitsch et al., 2017) and **DL3DV-Evaluation** (Ling et al., 2024), which cover diverse static indoor and outdoor scenes. We additionally evaluate on the dynamic-scene datasets **DAVIS** (Perazzi et al., 2016) and **DyCheck** (Gao et al., 2022). Camera poses are supplied only to methods that explicitly require them. For each scene, adaptation and evaluation frames are strictly disjoint. We use 468 frames for scene-level adaptation and evaluate on 12 held-out views. More details are given in Appendix B.1.

Evaluation Metrics. We report scene-averaged **reconstruction (Recon)** and **rendering (Render)** results using PSNR↑, SSIM↑, and VGG-based LPIPS↓ (Zhang et al., 2018). We also report the training time measured on a single NVIDIA A800 GPU.

Baselines. We compare SpikeTTT with two families of baselines. The first comprises spike-specific 3DGS pipelines that optimize a scene-specific representation: SpikeGS (Guo et al., 2025) and USP-Gaussian (Chen et al., 2025). To also evaluate RGB reconstruction with USP-Gaussian, we construct the USP-Gaussian-RGB variant by integrating an SJDDNet-style encoder (Dong et al., 2024) into its reconstructor. Spike4DGS (Ye et al., 2026) and Nope-SGS (Guo et al., 2026) are not included in the quantitative comparison because public implementations are unavailable. The second family comprises frozen cascades, which feed either TFP or pretrained SJDDNet (Dong et al., 2024) reconstructions into the foundation.

Table 1: **Quantitative comparison on Tanks and Temples and DL3DV.** The Camera column marks whether a method requires camera-pose input (✓) or is pose-free (×). Color shading indicates the **best** and **second-best** result; GT-input rows are references and are excluded from ranking.

Method	Camera	Tanks and Temples							DL3DV						
		Render			Recon			Time	Render			Recon			Time
		PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	mm:ss	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	mm:ss
Baseline															
SpikeGS	✓	24.90	0.731	0.275	–	–	–	38:08	16.46	0.283	0.561	–	–	–	52:37
USP-Gaussian	✓	20.74	0.578	0.490	22.90	0.487	0.599	278:52	16.96	0.447	0.606	17.85	0.317	0.680	270:58
USP-Gaussian-RGB	✓	21.81	0.671	0.254	26.36	0.801	0.328	420:11	18.21	0.524	0.381	19.71	0.571	0.525	442:10
LagerNVS-based															
GT RGB-LagerNVS	✓	25.74	0.804	0.125	–	–	–	–	18.79	0.556	0.290	–	–	–	–
TFP-LagerNVS	✓	24.45	0.727	0.354	26.71	0.720	0.379	–	18.99	0.531	0.552	20.20	0.499	0.554	–
SJDD-LagerNVS	✓	19.15	0.696	0.342	20.18	0.729	0.348	–	16.31	0.507	0.497	18.28	0.575	0.491	–
Ours-LagerNVS	✓	26.43	0.811	0.263	28.37	0.810	0.308	32:00	19.65	0.581	0.455	23.30	0.666	0.454	31:30
AnySplat-based															
GT RGB-AnySplat	×	25.15	0.821	0.154	–	–	–	–	20.11	0.649	0.271	–	–	–	–
TFP-AnySplat	×	24.45	0.768	0.318	26.71	0.720	0.379	–	18.04	0.514	0.537	20.20	0.484	0.554	–
SJDD-AnySplat	×	19.03	0.773	0.276	20.18	0.729	0.348	–	16.28	0.552	0.438	18.83	0.601	0.467	–
Ours-AnySplat	×	26.01	0.824	0.242	30.63	0.871	0.249	21:42	20.82	0.644	0.412	25.53	0.733	0.395	32:52

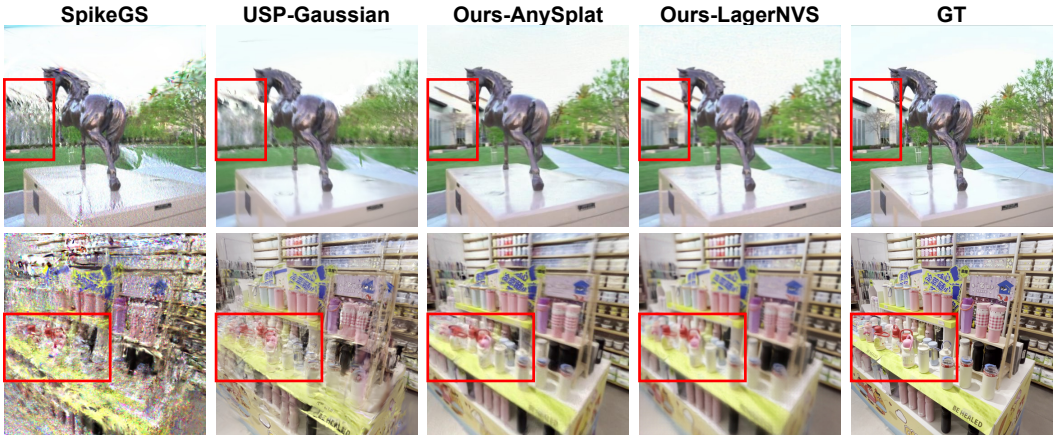


Figure 3: **Qualitative comparison on static scenes from Tanks and Temples and DL3DV.**

Implementation details. SpikeGS (Guo et al., 2025) and USP-Gaussian (Chen et al., 2025) are implemented with their official training recipes. We utilize pretrained SJDDNet (Dong et al., 2024) with AnySplat (Jiang et al., 2025) and LagerNVS (Szymanowicz et al., 2026) for static scenes and MoVieS (Lin et al., 2026) for dynamic scenes. SpikeTTT performs $K = 100$ update steps per scene. More details are given in Appendix B.2.

5.2 Experimental Results

Qualitative comparison. Figure 3 compares novel views from Tanks and Temples and DL3DV. The red boxes highlight the differences among the baselines and SpikeTTT variants. SpikeGS and USP-Gaussian introduce high-frequency speckle and grain-like artifacts, which break thin structures and obscure local surface textures. Both SpikeTTT variants recover more accurately localized boundaries and discernible fine-scale textures while maintaining photometric consistency. These differences are consistent across the Tanks and Temples and DL3DV datasets. Additional qualitative examples are shown in Figure 8.

Quantitative comparison. Table 1 reports scene-averaged results on Tanks and Temples and DL3DV. For AnySplat, SpikeTTT improves over the frozen SJDDNet+AnySplat cascade by 6.98 and 4.54 dB in rendering PSNR on Tanks and Temples and DL3DV, respectively. With LagerNVS, SpikeTTT also exhibits the same rendering PSNR improvement trend, and outperforms its frozen cascade on both datasets. More broadly, both SpikeTTT variants achieve higher rendering PSNR than spike-specific 3DGS baselines on both datasets while requiring less optimization time.

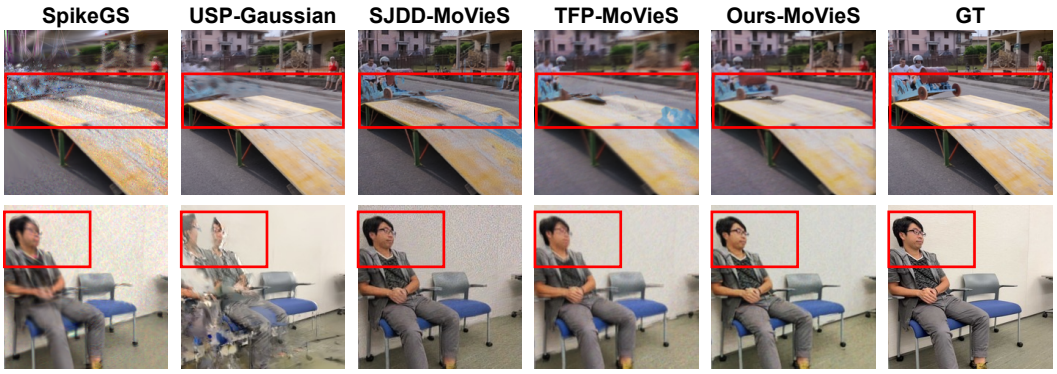


Figure 4: **Qualitative comparison on dynamic scenes from DAVIS and DyCheck.**

Table 2: **Quantitative comparison on DAVIS and DyCheck.**

Method	DAVIS							DyCheck						
	Render			Recon			Time	Render			Recon			Time
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	mm:ss	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	mm:ss
SpikeGS	18.27	0.457	0.480	–	–	–	52:23	22.81	0.644	0.328	–	–	–	21:24
USP-Gaussian-RGB	19.50	0.585	0.337	22.66	0.638	0.430	214:14	15.26	0.317	0.508	28.37	0.826	0.223	245:00
TFP-MoVieS	19.08	0.555	0.459	23.20	0.602	0.438	–	23.02	0.696	0.276	27.56	0.791	0.243	–
SJDD-MoVieS	16.57	0.561	0.387	20.53	0.742	0.316	–	18.39	0.625	0.289	19.93	0.812	0.233	–
Ours-MoVieS	20.22	0.663	0.379	26.91	0.803	0.299	21:11	23.41	0.747	0.260	28.44	0.866	0.209	19:46

5.3 Ablation Studies and Analysis

5.3.1 Framework Generalization

We evaluate the model flexibility of SpikeTTT across three 3D foundations and task settings. For static scenes, we instantiate the protocol with AnySplat (Jiang et al., 2025), which utilizes an explicit 3D Gaussian representation, and LagerNVS (Szymanowicz et al., 2026), which utilizes a latent neural renderer. For dynamic scenes, we instantiate the protocol with MoVieS (Lin et al., 2026), which utilizes an explicit Gaussian renderer. Across static scenes, both variants consistently improve rendering quality over their corresponding frozen cascades, as shown in Table 1. The shared trend across explicit-Gaussian and latent-neural-renderer paradigms shows that our protocol can be instantiated beyond a single NVS architecture. SpikeTTT also achieves dynamic-scene reconstruction, as shown in Table 2. Compared with the frozen cascades, SpikeTTT produces sharper and more faithful reconstructions, as highlighted by the red bounding boxes in Figure 4.

5.3.2 Ablation of Adaptation Modules and Self-Supervised Objectives

Table 3 and Figure 5 clarify the adaptation modules and their complementary roles within tri-signal self-supervision. We evaluate the contribution of adapting each component. Relative to the frozen cascade (I), adapting either the reconstructor (III) or the 3D foundation (IV) substantially improves rendering quality. Joint adaptation (V) further improves rendering, indicating complementary benefits from the two components. Mutual supervision alone (II) causes severe photometric drift. TFP stabilizes the photometry in III–V, thereby delivering a low-frequency photometric anchor. Given this anchor and mutual supervision, VI improves the reconstruction and rendering quality and yields cleaner error maps and sharper local structures. Overall, our tri-signal self-supervised objective utilizes TFP as a photometric anchor while leveraging complementary cross-branch supervision to recover high-frequency details and suppress branch-specific noise.

5.3.3 Real-World Experiments

Figure 6 compares rendering quality under rapid camera motion in a static scene and rapid object motion in a dynamic scene. Because ground-truth RGB images are unavailable, we qualitatively compare the baselines with the SpikeTTT variants and use TFP as a reference. In the static scene, SpikeGS introduces floaters around the cow's glasses and face, while USP-Gaussian distorts these facial details. The frozen cascade also retains visible noise and color bias. The SpikeTTT variants faithfully reconstruct the glasses and facial contours. In the dynamic scene, the baselines over-

Table 3: **Ablation of adaptation modules and self-supervised objectives.**

ID	Updated modules		Configuration				Tanks and Temples						DyCheck					
							Render			Recon			Render			Recon		
	Recon	Foundation	Recon		Render													
			$\mathcal{L}_{\text{tfp}}$	$\mathcal{L}_{\text{mutual}}$	$\mathcal{L}_{\text{tfp}}$	$\mathcal{L}_{\text{mutual}}$	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
I							19.03	0.773	0.276	20.18	0.729	0.348	18.39	0.625	0.289	19.93	0.812	0.233
II	✓	✓		✓		✓	9.81	0.429	0.797	9.85	0.432	0.782	9.10	0.326	0.870	9.14	0.331	0.854
III	✓		✓		✓		25.04	0.795	0.241	30.17	0.807	0.281	22.93	0.692	0.275	28.07	0.812	0.231
IV		✓			✓		25.39	0.818	0.243	20.18	0.729	0.348	20.29	0.634	0.291	19.90	0.812	0.233
V	✓	✓	✓		✓		25.68	0.807	0.240	30.12	0.806	0.281	23.27	0.706	0.272	28.16	0.814	0.230
VI	✓	✓	✓	✓	✓	✓	26.01	0.824	0.242	30.63	0.871	0.249	23.41	0.747	0.260	28.44	0.866	0.209

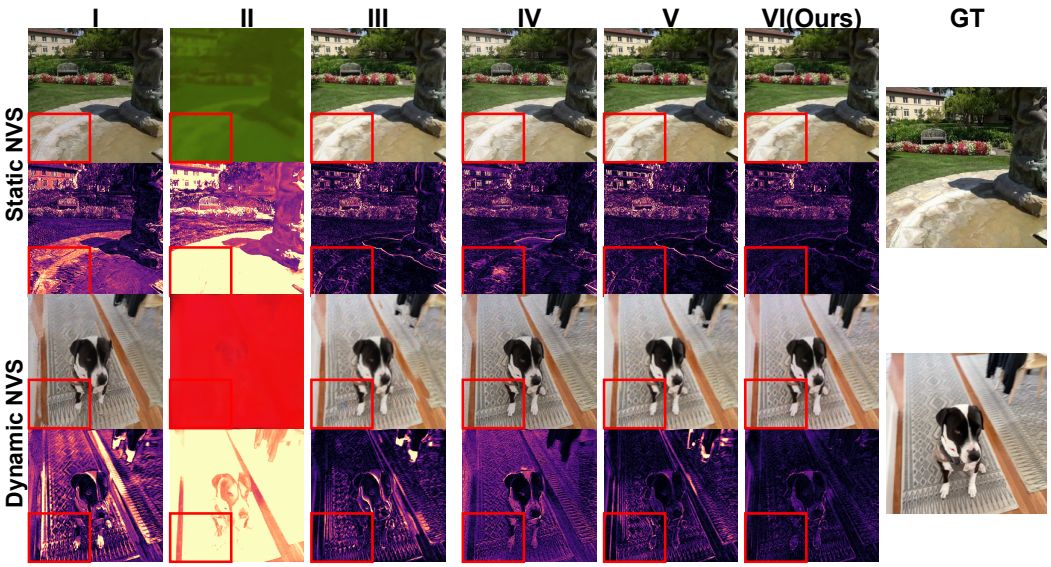


Figure 5: **Ablation study on the tri-signal loss design.**



Figure 6: **Qualitative comparison on real-world static and dynamic scenes.**

smooth the structure of the rotating object, while the frozen cascade introduces floaters and color bias. Overall, SpikeTTT preserves sharper geometry and clearer textures across the two settings.

6 Conclusion

We present **SpikeTTT**, a model-flexible test-time training protocol that adapts pretrained 3D foundations to spike-derived observations without relying on external supervision. It freezes the pretrained modules and updates only lightweight adapters under a tri-signal self-supervised objective. Across posed and pose-free settings and static and dynamic scenes, SpikeTTT improves reconstruction and rendering quality over frozen cascades and spike-specific 3DGS methods. In pose-free novel-view synthesis settings, with AnySplat, rendering PSNR rises by 6.98 dB on Tanks and Temples and 4.54 dB on DL3DV. SpikeTTT thus provides a practical route for bringing large-scale RGB pretraining to spike-based 3D reconstruction.

REPRODUCIBILITY STATEMENT

Our method and the complete test-time training protocol for the tri-signal objective are provided in Sections 4.1–4.3. Section 5.1 specifies the datasets, baselines, evaluation metrics, pretrained backbones, and hardware used in our experiments. Appendix B provides the scene-level test-time training algorithm, the construction of the training and test splits, camera-pose estimation, optimizer and loss settings, adapter placements, and learning rates.

AI USE STATEMENT

We used GPT-5.6 solely to polish the manuscript text and refine its descriptions, including improvements to grammar, wording, clarity, and readability. It was not used to generate research ideas, methods, code, experimental results, analyses, or scientific claims. All AI-assisted edits were reviewed and verified by the authors, who take full responsibility for the final content of the paper.

REFERENCES

Kang Chen, Jiyuan Zhang, Zecheng Hao, Yajing Zheng, Tiejun Huang, and Zhaofei Yu. USP-Gaussian: unifying spike-based image reconstruction, pose correction and gaussian splatting. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16609–16618. IEEE, 2025.

Shiyan Chen, Chaoteng Duan, Zhaofei Yu, Ruiqin Xiong, and Tiejun Huang. Self-supervised mutual learning for dynamic scene reconstruction of spiking camera. In *IJCAI*, pp. 2859–2866, 2022.

Gaole Dai, Zhenyu Wang, Qinwen Xu, Ming Lu, Wen Chen, Boxin Shi, Shanghang Zhang, and Tiejun Huang. SpikeNVS: Enhancing novel view synthesis from blurry images via spike camera. *arXiv preprint arXiv:2404.06710*, 2024.

Yanchen Dong, Ruiqin Xiong, Jing Zhao, Jian Zhang, Xiaopeng Fan, Shuyuan Zhu, and Tiejun Huang. Joint demosaicing and denoising for spike camera. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 1582–1590, 2024.

Hang Gao, Ruilong Li, Shubham Tulsiani, Bryan Russell, and Angjoo Kanazawa. Monocular dynamic view synthesis: A reality check. *Advances in Neural Information Processing Systems*, 35: 33768–33780, 2022.

Zhuoheng Gao, Yihao Li, Jiyao Zhang, Rui Zhao, Tong Wu, Hao Tang, Zhaofei Yu, Hao Dong, Guozhang Chen, and Tiejun Huang. SpikeStereoNet: A brain-inspired framework for stereo depth estimation from spike streams. In *International Conference on Learning Representations*, volume 2026, pp. 11457–11487, 2026.

Yijia Guo, Yuanxi Bai, Liwen Hu, Mianzhi Liu, Ziyi Guo, Lei Ma, and Tiejun Huang. Spike-NeRF: Neural radiance field based on spike camera. In *2024 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 1–6. IEEE, 2024.

Yijia Guo, Liwen Hu, Yuanxi Bai, Jiawei Yao, Lei Ma, and Tiejun Huang. SpikeGS: Reconstruct 3D scene captured by a fast-moving bio-inspired camera. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 3293–3301, 2025.

Yijia Guo, Tong Hu, Liwen Hu, Lei Ma, and Tiejun Huang. 3D gaussian splatting from unposed spike stream. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 41002–41011, 2026.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Liwen Hu, Rui Zhao, Ziluo Ding, Lei Ma, Boxin Shi, Ruiqin Xiong, and Tiejun Huang. Optical flow estimation for spiking camera. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17823–17832. IEEE, 2022.

Liwen Hu, Lei Ma, Yijia Guo, and Tiejun Huang. SCSim: a realistic spike cameras simulator. In *2024 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 1–6. IEEE, 2024.

Tiejun Huang, Yajing Zheng, Zhaofei Yu, Rui Chen, Yuan Li, Ruiqin Xiong, Lei Ma, Junwei Zhao, Siwei Dong, Lin Zhu, et al. 1000× faster camera and machine vision with ordinary devices. *Engineering*, 25:110–119, 2023.

Zhewei Huang, Tianyuan Zhang, Wen Heng, Boxin Shi, and Shuchang Zhou. Real-time intermediate flow estimation for video frame interpolation. In *European conference on computer vision*, pp. 624–642. Springer, 2022.

Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. In *European conference on computer vision*, pp. 709–727. Springer, 2022.

Lihan Jiang, Yucheng Mao, Linning Xu, Tao Lu, Kerui Ren, Yichen Jin, Xudong Xu, Mulin Yu, Jiangmiao Pang, Feng Zhao, et al. AnySplat: Feed-forward 3D gaussian splatting from unconstrained views. *ACM Transactions on Graphics (TOG)*, 44(6):1–16, 2025.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, George Drettakis, et al. 3D gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):1–14, 2023.

Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics (ToG)*, 36(4):1–13, 2017.

Zhengqi Li, Richard Tucker, Forrester Cole, Qianqian Wang, Linyi Jin, Vickie Ye, Angjoo Kanazawa, Aleksander Holynski, and Noah Snavely. MegaSaM: Accurate, fast, and robust structure and motion from casual dynamic videos. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10486–10496. IEEE, 2025.

Chenguo Lin, Yuchen Lin, Panwang Pan, Yifan Yu, Tao Hu, Honglei Yan, Katerina Fragkiadaki, and Yadong Mu. MoVieS: Motion-aware 4D dynamic view synthesis in one second. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 295–306, 2026.

Lu Ling, Yichen Sheng, Zhi Tu, Wentian Zhao, Cheng Xin, Kun Wan, Lantao Yu, Qianyu Guo, Zixun Yu, Yawen Lu, et al. DL3DV-10K: A large-scale scene dataset for deep learning-based 3D vision. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 22160–22169. IEEE, 2024.

Yuejiang Liu, Parth Kothari, Bastien Van Delft, Baptiste Bellot-Gurlet, Taylor Mordan, and Alexandre Alahi. TTT++: When does self-supervised test-time training fail or thrive? *Advances in Neural Information Processing Systems*, 34:21808–21820, 2021.

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. In *Computer Vision – ECCV 2020*, pp. 405–421. Springer International Publishing, 2020. doi: 10.1007/978-3-030-58452-8_24.

Federico Perazzi, Jordi Pont-Tuset, Brian McWilliams, Luc Van Gool, Markus Gross, and Alexander Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 724–732, 2016.

Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei Efros, and Moritz Hardt. Test-time training with self-supervision for generalization under distribution shifts. In *International conference on machine learning*, pp. 9229–9248. PMLR, 2020.

Stanislaw Szymanowicz, Minghao Chen, Jianyuan Wang, Christian Rupprecht, and Andrea Vedaldi. LagerNVS: Latent geometry for fully neural real-time novel view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15443–15453, 2026.

Jianyuan Wang, Minghao Chen, Shangzhan Zhang, Nikita Karaev, Johannes Schönberger, Patrick Labatut, Piotr Bojanowski, David Novotny, Andrea Vedaldi, and Christian Rupprecht. VGGT-Ω. *arXiv preprint arXiv:2605.15195*, 2026.

Qinghong Ye, Yiqian Chang, Jianing Li, Haoran Xu, Xuan Wang, Wei Zhang, Yonghong Tian, and Peixi Peng. Spike4DGS: Towards high-speed dynamic scene rendering with 4D gaussian splatting via a spike camera array. *Advances in Neural Information Processing Systems*, 38:28707–28727, 2026.

Longhui Yuan, Binhui Xie, and Shuang Li. Robust test-time adaptation in dynamic scenarios. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15922–15932. IEEE, 2023.

Yuheng Yuan, Qiuhong Shen, Shizun Wang, Xingyi Yang, and Xinchao Wang. Test3R: Learning to reconstruct 3D at test time. *Advances in Neural Information Processing Systems*, 38:100445–100464, 2026.

Jiyuan Zhang, Lulu Tang, Zhaofei Yu, Jiwen Lu, and Tiejun Huang. Spike transformer: Monocular depth estimation for spiking camera. In *European Conference on Computer Vision*, pp. 34–52. Springer, 2022.

Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 3813–3824. IEEE, 2023.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *2018 IEEE/CVF conference on computer vision and pattern recognition*, pp. 586–595. IEEE, 2018.

Jing Zhao, Ruiqin Xiong, Hangfan Liu, Jian Zhang, and Tiejun Huang. Spk2ImgNet: Learning to reconstruct dynamic scene from continuous spike stream. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11991–12000. IEEE, 2021.

Rui Zhao, Ruiqin Xiong, Jing Zhao, Jian Zhang, Xiaopeng Fan, Zhaofei Yu, and Tiejun Huang. Boosting spike camera image reconstruction from a perspective of dealing with spike fluctuations. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 24955–24965. IEEE, 2024.

Lin Zhu, Siwei Dong, Tiejun Huang, and Yonghong Tian. A retina-inspired sampling method for visual texture reconstruction. In *2019 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 1432–1437. IEEE, 2019.

A Limitations and Future Work

While SpikeTTT improves spike-based novel-view synthesis through scene-level adaptation, several limitations remain. First, the attainable quality of reconstruction and rendering is bounded by the pretrained modules. Limited foundation geometry and novel-view fidelity directly constrain the performance ceiling. Second, the current formulation and evaluation focus on NVS. Extending the framework to other spike-vision tasks will require task-specific self-supervised signals. Finally, although SpikeTTT updates only lightweight adapters, per-scene optimization still requires repeated forward and backward passes, which may limit large-scale, high-resolution, or online use.

B Additional Details

B.1 Dataset Construction

For spike evaluation, we construct per-scene novel-view synthesis datasets from RGB sequences. Static scenes come from Tanks and Temples (Knapitsch et al., 2017) and DL3DV-Evaluation (Ling et al., 2024), while dynamic scenes come from DAVIS (Perazzi et al., 2016) and DyCheck (Gao et al., 2022).

We adopt the two-stage conversion protocol of USP-Gaussian (Chen et al., 2025). First, we temporally densify each RGB sequence by interpolating intermediate frames with RIFE (Huang et al., 2022). We then convert the resulting intensity sequence into a spike stream using a spike-camera simulator (Hu et al., 2024). This procedure yields paired RGB and spike sequences for each scene. The spike streams serve as model inputs, and the corresponding RGB frames provide ground truth for quantitatively evaluating reconstruction and novel-view synthesis quality.

RGB-to-spike conversion. Algorithm 1 details the simulator used in the second stage. Each 8-bit RGB frame $\mathbf{I}_{\mathrm{rgb}}(t)$ is normalized to $[0, 1]$ and mosaicked according to the Bayer pattern $\mathcal{M}$: Pixels at even–even, mixed-parity, and odd–odd coordinates retain the red, green, and blue channels, respectively. The selected Bayer intensity determines the photocurrent $L_C(u, v, t)$, which is integrated with the dark-current term $\eta_{\mathrm{dark}}(u, v)$ in the accumulator $A(u, v, t)$ to yield the spike stream $\mathcal{S}(u, v, t)$ described in Section 3. Pixel-level capacitance and voltage variations define a nonuniform firing threshold $\phi(u, v)$. Once $A(u, v, t)$ reaches $\phi(u, v)$, the pixel emits $\mathcal{S}(u, v, t) = 1$, one threshold is subtracted from the accumulator, and the next integration increment is attenuated by a factor of 0.998. In practice, the initial accumulator is randomized to avoid synchronized firing. The binary samples are traversed in temporal and raster order and packed eight samples per byte. For each timestamp t_i, N consecutive spike observations centered at t_i form the temporal window $\mathcal{S}_{t_i}^{(N)} \in \{0, 1\}^{N\times H\times W}$ used in Section 4.1.

In our experiments, we use the same hyperparameter settings as SCSim (Hu et al., 2024): we set $\Delta t = 0.05$, $V_{\mathrm{rst}} = 3.3$, $V_{\mathrm{ref}} = 2.2$, $C_0 = 4.348$, $V_0 = -0.0076$, and a photocurrent scale of 20. The simulator samples $\eta_{\mathrm{dark}}(u, v) \sim \mathcal{N}(0.4442, 0.1731^2)$, $C(u, v) \sim \mathcal{N}(4.348, 0.0128^2)$, and $V(u, v) \sim \mathcal{N}(-0.0076, 0.0497^2)$ once for each pixel, and defines $\phi(u, v) = C(u, v)[V_{\mathrm{rst}} - V_{\mathrm{ref}} + V(u, v)]$. These hyperparameters are estimated from measurements of a real spike camera and remain fixed across all scenes.

Real-world evaluation. For real-world evaluation, we utilize spike-camera recordings from two sources. For scenes with moving objects, we use the real Bayer spike streams released with SpikeGS (Guo et al., 2025). These recordings comprise rotating objects captured at 700, 1,000, and 1,400 revolutions per minute using a 1000×1000 Spike M1K40-H4-Gen3 camera. For static scenes, we also collect additional spike streams using the Spike M1K40-H4-Gen3 camera. During acquisition, the camera follows casual trajectories. We use these recordings to evaluate reconstruction under non-curated camera motion.

We will release the derived datasets and real spike-camera recordings once the paper is accepted.

Dataset split. We distinguish the held-out target view from the raw spike samples that support centered reconstruction windows. Each $T = 936$ sequence is divided into 12 non-overlapping chunks with local indices $0, \ldots, 77$. Test-time training uses the 39 context-view centers $\mathcal{C}_{\mathrm{train}} =$

Table 4: **Details of the selected synthetic evaluation scenes.** Sequence length denotes the number of derived spike frames per scene. The adaptation/evaluation column reports the numbers of frames used for scene-level adaptation and held-out evaluation, respectively.

Dataset	Scene	Scene regime	Motion setting	Seq. length	Adapt./Eval.	Pattern
Tanks and Temples (Knapitsch et al., 2017)	ballroom barn church family francis horse ignatius	Static	Moving camera	936	468/12	RGGB
DL3DV-Evaluation (Ling et al., 2024)	20 selected sequences	Static	Moving camera	936	468/12	RGGB
DAVIS (Perazzi et al., 2016)	surf bmx-bumps car-turn bmx-trees soapbox kite-walk rollerblade stroller motocross-bumps blackswan swing dance-twirl tennis horsejump-low goat drift-straight scooter-gray motocross-jump scooter-black rhino	Dynamic	Monocular video	936	468/12	RGGB
DyCheck (Gao et al., 2022)	apple backpack block creeper handwavy haru-sit mochi-high-five paper-windmill pillow space-out spin sriracha-tree teddy wheel	Dynamic	Moving camera	936	468/12	RGGB

Table 5: **DL3DV scenes used for the averages reported in Table 1.** Each identifier corresponds to a selected DL3DV-10K sequence.

#	Scene ID	#	Scene ID
1	ae90ea23130506b51f0aa19e518d2cc920ebb8c9a2b89a2178c00d3f5959440b	11	9bc29652c28b8999a86d0fc012d66c440ec8985f8f5544c7accc9c08ff70b09d
2	7ab720a2e21e729185ab0147e2aeda65d0ae2c0434641f6b13102b01e927919b	12	d2ce0396dc6df5a579dbd8189885ce978c19490709875043e3566efba054236f
3	9d746529972b9f3957bb5a4895984946638bd1945880934c448c2431523dd1d1	13	309a93c9e8aeeb1f7851b23da892aaa750c7072d50297d835613df2ee601bf2c
4	a126efe829a3872bb092e09b58ece5c87d464dd3c3ec5665b1b5fddb696d55ff	14	924228bfa8983b5aa7921a609d5046036c96022f8458c037460386e74f3ffec6
5	086736c55127909088cfba10a7a8cf300604723c25f1370109550302ae19093e	15	f47839e25e89f44f1741d3fffd8674544893f2d704f7082b30cce77847bf0a45
6	b93f5691ba1cc8890d0b0fb5792668d8c8e084e3b3ae1a476fc5e2c1e38b79ae	16	f8a2ebdff72059df052f1aae39d1fb613d4167a24d1e7e22f37f94bc96da8f3b
7	5f57615993d0136efc3628e41994c4cec16779a2b31a47f0a5b8a31637927491	17	f2ee054873ffbfae0269060f956ef713579852b6a94ded2faceee282baf3448a
8	ec0d0f130fbe697564dbac8df4b8a07665868a6ed0441cd29458607c67f61836	18	5ad1e824cc04de35689a995671675dc195f7fbb0947d7feeaa6040a1a235f0c7
9	05cd8f26e146d8fee12e1f393ba17fd76c3de796dcac9bb5573d9ee508a3c765	19	5019ee7f71f5367a6f6c21248051519d1247eb17f5e1c6bcfd40c8f6ad047aaa
10	18ae5d387baba346017f225c7661c885ab4ce1ef7dd028a9ad80773002528557	20	599b9dc15771ada7ce5eed6c2bc69c90977a2fa072a82f7121714f4a882f6ee5

$\{19, \ldots, 57\}$. For a center c, SJDDNet receives the 39-sample window $\mathcal{W}(c) = \{c-19, \ldots, c+19\}$; consequently, the union of the adaptation-window supports is $\bigcup_{c \in \mathcal{C}_{\text{train}}} \mathcal{W}(c) = \{0, \ldots, 76\}$. The view centered at $q = 77$ is held out. During adaptation, we access neither its RGB target, its

Algorithm 1 Conversion of an RGB video sequence to an RGGB spike stream

Require: Ordered 8-bit RGB frames $\{\mathbf{I}_{\mathrm{rgb}}(t)\}_{t=1}^{F}$ of size $H \times W$, Bayer pattern $\mathcal{M}$, brightness factor d, and an optional random seed
Ensure: Bit-packed spike stream $\mathcal{S}$

Sample $\eta_{\mathrm{dark}}(u,v)$, $C(u,v)$, and $V(u,v)$ once for all pixels using the distributions above
$\phi(u,v) \leftarrow C(u,v)[V_{\mathrm{rst}} - V_{\mathrm{ref}} + V(u,v)]$
Initialize $A(u,v,0) \sim \mathcal{U}(0, 0.9C_0(V_{\mathrm{rst}} - V_{\mathrm{ref}} + V_0))$ and set the post-spike flag $q(u,v) \leftarrow 0$
for $t = 1, \ldots, F$ **do**
    Normalize $\mathbf{I}_{\mathrm{rgb}}(t) \leftarrow \mathbf{I}_{\mathrm{rgb}}(t)/255$
    for each pixel (u,v) in raster order **do**
        Select the Bayer intensity $I_{\mathcal{M}}(u,v,t)$ from $\mathbf{I}_{\mathrm{rgb}}(t)$ using $\mathcal{M}(u,v)$
        $L_C(u,v,t) \leftarrow 20C_0(V_{\mathrm{rst}} - V_{\mathrm{ref}} + V_0)I_{\mathcal{M}}(u,v,t)d$
        $J(u,v,t) \leftarrow \min(L_C(u,v,t) + \eta_{\mathrm{dark}}(u,v), \phi(u,v)/\Delta t)$
        $\Delta A(u,v,t) \leftarrow \Delta t\, J(u,v,t)$
        if $q(u,v) = 1$ **then**
            $\Delta A(u,v,t) \leftarrow 0.998\Delta A(u,v,t)$; $q(u,v) \leftarrow 0$
        end if
        $A(u,v,t) \leftarrow A(u,v,t-1) + \Delta A(u,v,t)$
        if $A(u,v,t) \geq \phi(u,v)$ **then**
            $\mathcal{S}(u,v,t) \leftarrow 1$; $A(u,v,t) \leftarrow A(u,v,t) - \phi(u,v)$; $q(u,v) \leftarrow 1$
        else
            $\mathcal{S}(u,v,t) \leftarrow 0$
        end if
        Append $\mathcal{S}(u,v,t)$ to the output, packing each consecutive group of eight samples into one byte
    end for
end for
return $\mathcal{S}$

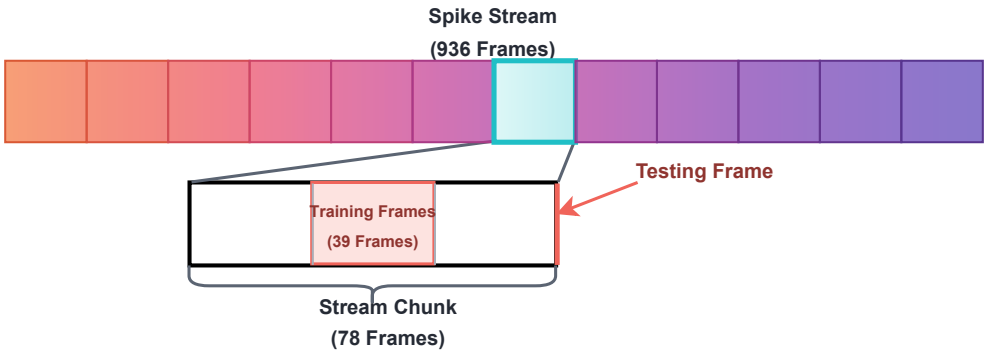


Figure 7: **Per-scene split of a spike stream containing $T = 936$ frames.** The stream is partitioned into 12 non-overlapping chunks of $2N = 78$ frames.

reconstruction window $\mathcal{W}(q) = \{58, \ldots, 96\}$, nor any loss or model-selection signal computed at q. For NVS evaluation, the adapted model receives the context views and query camera pose, but not the query appearance. The split is therefore disjoint at the target-view and optimization-signal levels. This protocol provides 468 adaptation views and 12 held-out queries per scene, as summarized in Figure 7.

Camera poses. We estimate camera poses from TFP using VGGT-Ω (Wang et al., 2026) for static scenes and MegaSaM (Li et al., 2025) for dynamic scenes. *These estimated poses are supplied only to methods that require camera-pose input*, including SpikeGS (Guo et al., 2025), USP-Gaussian (Chen et al., 2025), USP-Gaussian-RGB (Chen et al., 2025; Dong et al., 2024), LagerNVS (Szymanowicz et al., 2026), and MoVieS (Lin et al., 2026). None of the evaluation scenes were used to pretrain these modules.

Algorithm 2 Scene-Level Test-Time Training for SpikeTTT

Require: Scene-specific training set $\mathcal{D}_{\text{train}} = \{\mathcal{S}_{t_i}^{(N)}\}_{i\in\mathcal{I}_{\text{train}}}$, camera poses $\{\mathbf{P}_i\}_{i\in\mathcal{I}_{\text{train}}}$ (required only for pose-conditioned foundations), pretrained $(R_{\theta_s}, \mathcal{F}_{\phi_s})$, and K update steps
Ensure: Scene-adapted parameters (θ^*, ϕ^*)
1: Freeze all pretrained parameters and enable only Θ_{adapt}.
2: Organize only $\mathcal{D}_{\text{train}}$ into temporal groups $\{\mathcal{G}_g\}_{g=1}^{G}$ and compute $\mathbf{E}_{\text{tfp}}^{g} \leftarrow \mathcal{T}_{\text{tfp}}(\text{center}(\mathcal{G}_g))$.
3: **for** $k = 1, \ldots, K$ **do**
4: Sample a clip $\mathcal{B}$ from the training groups $\{\mathcal{G}_g\}_{g=1}^{G}$.
5: $\mathbf{I}_{\text{rec}} \leftarrow R_\theta(\mathcal{S}_{\mathcal{B}}, \mathcal{M}_{\mathcal{B}})$.
6: $\mathbf{I}_{\text{ren}} \leftarrow \mathcal{F}_\phi(\text{sg}(\mathbf{I}_{\text{rec}}), \mathbf{P}_{\mathcal{B}}$ if required).
7: Evaluate $\mathcal{L}_{\text{TTT}}$ in Equation (11) using only samples from $\mathcal{B}$.
8: Update Θ_{adapt} using the optimizer in Appendix B.2.
9: **end for**
10: $\theta^*, \phi^* \leftarrow \theta, \phi$.
11: **return** (θ^*, ϕ^*).

Scene-level test-time training. Algorithm 2 summarizes the scene-level test-time training under the training–test separation defined in Section B.1. Camera poses are optional inputs and are supplied only to pose-conditioned foundations. Only the adapter subset is optimized; all remaining pretrained parameters are kept frozen.

B.2 Experimental Protocol

Implementation environment and backbone instantiation. We implement SpikeTTT in PyTorch and run all methods on a single NVIDIA A800 GPU. SpikeGS (Guo et al., 2025) and USP-Gaussian (Chen et al., 2025) use their official training recipes on the same device. Unless otherwise stated, the spike reconstructor is a pretrained SJDDNet (Dong et al., 2024) model. We pair it with AnySplat (Jiang et al., 2025) or LagerNVS (Szymanowicz et al., 2026) for static scenes and with MoVieS (Lin et al., 2026) for dynamic scenes.

SpikeTTT optimizes Θ_{adapt} with AdamW for $K = 100$ steps per scene, and we use $\lambda_1 = 0.8$ in Equation (8). The two mutual-supervision terms and two TFP-anchoring terms are equally weighted.

Modulation details. We restrict scene-level adaptation to lightweight modulation parameters while freezing the remaining pretrained weights. **The following learning rates and adapter placements are shared across all datasets.**

SJDDNet. We insert zero-initialized convolutional adapters into the representation layers and optimize them with a learning rate of 1×10^{-3}.

AnySplat. We adapt the camera head using LoRA with a learning rate of 2×10^{-5}. The camera-head LoRA uses rank $r = 16$ and scaling $\alpha = 16$. For the DPT-style depth and Gaussian heads, we instead use zero-initialized convolutional adapters with a learning rate of 1×10^{-4}.

LagerNVS. We apply LoRA to the Transformer blocks in the decoder and optimize the adapters with a learning rate of 1×10^{-4}. This decoder LoRA uses rank $r = 16$ and scaling $\alpha = 16$.

MoVieS. We equip the DPT-style depth and splat heads with zero-initialized convolutional adapters and use a learning rate of 1×10^{-4}. To stabilize dynamic-scene adaptation, we use a smaller learning rate of 3×10^{-6} for motion-related parameters.

We use these settings without dataset-specific tuning, providing evidence that the same adaptation hyperparameters generalize across the evaluated datasets.

Frozen-cascade baselines keep both pretrained modules fixed and perform no adapter updates.

B.3 Adaptation Parameter Budget

We measure the FP32 storage of each foundation instantiation used by SpikeTTT. Table 6 reports the full model storage, the storage of parameters updated during test-time training, and the corresponding adapted components.

Table 6: **FP32 parameter storage and adapted components of the three SpikeTTT foundation instantiations.** Trainable storage is the adapter subset optimized for 100 steps. Zero-conv denotes a zero-initialized convolutional adapter, and LoRA denotes a LoRA adapter.

Foundation	Total (MB)	Trainable (MB)	Trainable (%)	Adapted components
AnySplat (Jiang et al., 2025)	4,765.04	21.52	0.4516	Gaussian-head zero-conv; depth-head zero-conv; camera-head LoRA
LagerNVS (Szymanowicz et al., 2026)	4,416.83	3.44	0.0779	Renderer attention-projection LoRA
MoVieS (Lin et al., 2026)	2,947.10	38.84	1.3180	Depth-, splat-, motion-, and motion–splat-head zero-conv

Across the three instantiations, trainable storage is 3.44–38.84 MB, or 0.0779%–1.3180% of the corresponding full model. LagerNVS is the smallest case: LoRA on the renderer attention projections accounts for 3.44 MB (0.0779%). AnySplat, which uses zero-initialized convolutions in the Gaussian and depth heads and LoRA in the camera head, requires 21.52 MB (0.4516%). MoVieS is the largest trainable case at 38.84 MB (1.3180%), with zero-initialized convolutions in the depth, splat, motion, and motion–splat heads. These fractions show that test-time training updates only a small subset of each foundation's parameters.

C Additional Experiments

C.1 Additional Qualitative Results on Static Scenes

Figure 8 provides additional novel-view comparisons on static scenes, complementing Figure 3. In the highlighted regions, SpikeGS introduces grain-like artifacts, whereas USP-Gaussian fails at direct novel-view reconstruction. Compared with the baselines, both SpikeTTT variants recover sharper structures and an appearance that more closely matches the ground truth.

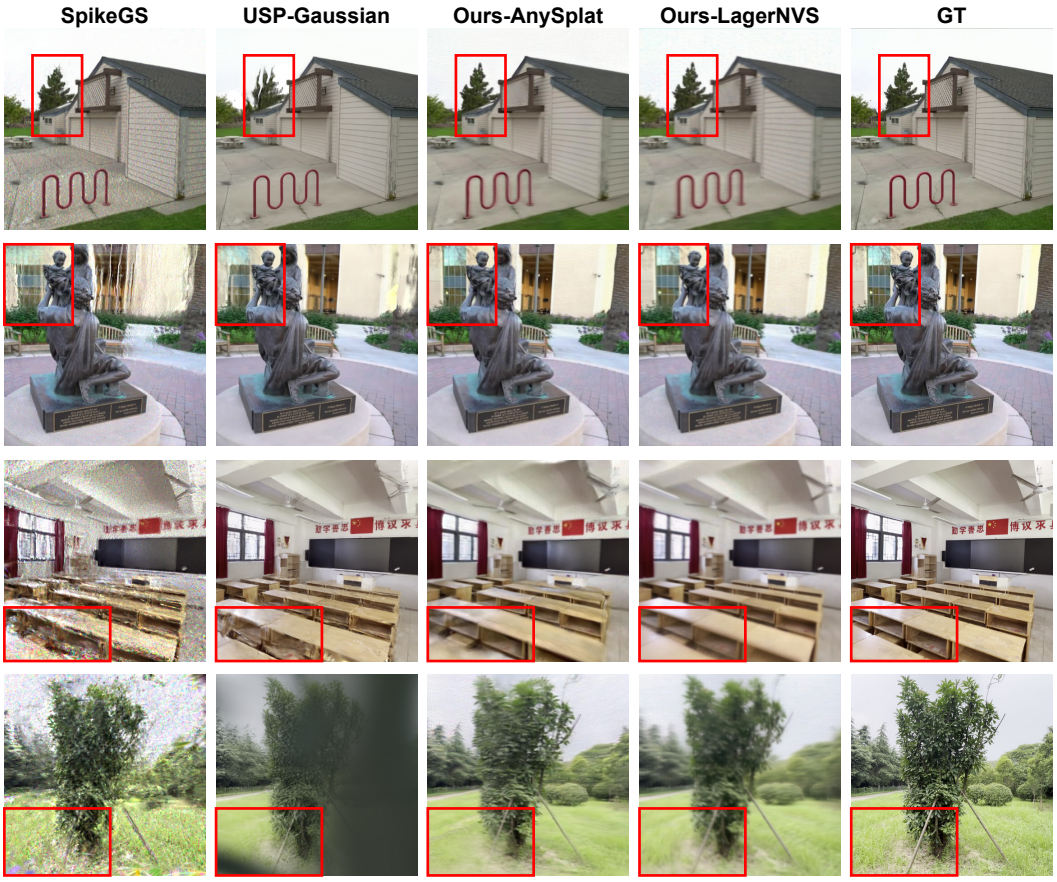


Figure 8: **Additional qualitative comparison on static scenes from Tanks and Temples (Knapitsch et al., 2017) and DL3DV (Ling et al., 2024).** Columns show SpikeGS, USP-Gaussian, SpikeTTT with AnySplat, SpikeTTT with LagerNVS, and ground truth (GT). Red boxes mark regions used for detailed comparison.