UltraVoxelGS: Voxel-First Feed-Forward Gaussian Splatting for 3D Ultrasound Reconstruction

Yue Wang, Haoxin Chen, Xiaoqing Guo
Department of Computer Science, Hong Kong Baptist University
Hong Kong SAR, China
25481983@life.hkbu.edu.hk, 25482661@life.hkbu.edu.hk
xiaoqingguo@hkbu.edu.hk

Abstract

We present **UltraVoxelGS**, a 3D ultrasound reconstruction method that achieves both ***fast reconstruction*** and physically ***faithful rendering*** for the first time, bringing live spatial feedback during 2D ultrasound scanning within reach. This is made possible by a feed-forward design that predicts a complete Gaussian scene representation from posed ultrasound images directly, entirely bypassing the per-scene optimization bottleneck that has confined prior methods to offline use. Realising such feed-forward prediction in this setting is non-trivial: unlike natural images, ultrasound slices share at most one-dimensional intersections and exhibit view-dependent appearance due to acoustic propagation and attenuation, violating the dense overlap and photometric consistency assumed by existing feed-forward Gaussian methods. Hence, to achieve fast reconstruction, we introduce a ***feed-forward voxel-to-Gaussian pass*** that decouples representation size from input count by lifting all posed slices into a fixed-size voxel support and predicting Gaussian parameters directly. For faithful rendering, we propose ***ultrasound-aware appearance rendering*** strategy factorize reflection from cumulative attenuation during training and exploit appearance continuity among spatially adjacent slices for efficient Render-time Appearance Adaptation. On both real-world and simulated datasets, UltraVoxelGS achieves best PSNR/SSIM compared with per-scene optimization baselines while reducing inference from 10-20 minutes to approximately 10 seconds—over 35× faster—and scales to sequences exceeding 1,000 frames. Code is available at https://github.com/YuebyYue/UltraVoxelGS-camera-ready.

1 Introduction

Despite being the most widely used diagnostic imaging modality [1, 2], ultrasound remains fundamentally limited by its 2D acquisition: sonographers scan a sequence of slices and must mentally reconstruct the underlying 3D anatomy to determine which regions remain unseen. Incomplete coverage is therefore common, especially among less experienced sonographers, and often goes unnoticed during scanning. While dedicated 3D probes exist, their limited field of view makes them inadequate for large-anatomy or extended-field-of-view examinations, leaving 2D freehand scanning the dominant acquisition mode in practice. These limitations motivate the need for *real-time automatic 3D reconstruction from 2D ultrasound sequences*[3–5]: during acquisition, a live 3D view can provide immediate spatial feedback to guide operators toward uncovered regions. After acquisition, a compact 3D representation also facilitates subsequent diagnosis and efficient archival. Realising this vision (Fig. 1) requires both ***fast reconstruction***—consolidating posed 2D slices into a compact 3D representation in seconds, not minutes—and ***faithful rendering***—synthesizing novel ultrasound views that respect the directional, cumulative-attenuation physics of acoustic imaging.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

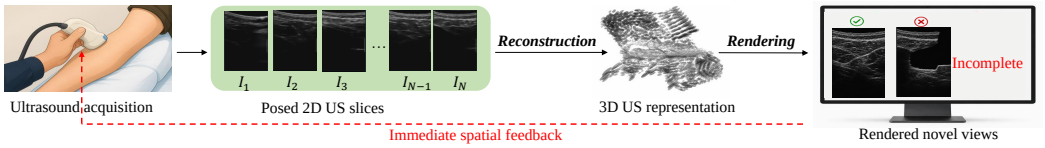


Figure 1: **Clinical workflow.** A sequence of posed 2D ultrasound slices are *reconstructed* into a 3D representation, from which novel views are *rendered* to guide the operator toward unseen regions.

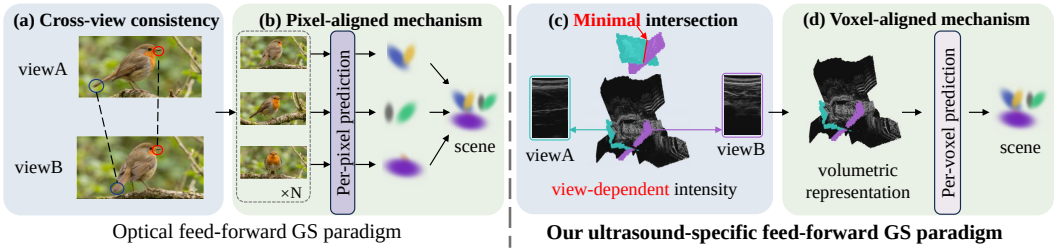


Figure 2: ***Optical feed-forward GS paradigm***: (a) Natural images provide dense cross-view overlap and photometric consistency. (b) Existing methods predict pixel-aligned primitives from individual views. ***Our ultrasound-specific feed-forward GS paradigm***: (c) Ultrasound slices share minimal intersection and exhibit view-dependent intensity. (d) We lift posed slices into a unified voxel support and predict Gaussians from the shared voxel support.

No existing approach satisfies both requirements. Classical interpolation-based methods [6, 7] are computationally cheap but treat ultrasound frames as generic intensity images, ignoring the directional propagation and cumulative attenuation of acoustic waves; they reproduce plausible intensities along scanned trajectories yet produce artifacts at novel viewpoints. Neural-field methods [8–11] achieve higher fidelity by learning a continuous scene representation, yet they rely on iterative per-scene optimization coupled with expensive volume rendering, making real-time reconstruction impractical. More recently, Gaussian splatting (GS)[12] has emerged as a compelling alternative: its explicit primitives support efficient rasterization-based rendering, and recent adaptations [13] show that the formulation can accommodate ultrasound imaging assumptions. However, these 3D GS methods still reconstruct each scene through iterative optimization, meaning that they enable fast rendering but not fast reconstruction. *The missing piece is not simply a faster renderer, but a feed-forward pathway from input slice sequences to a high-fidelity Gaussian representation.*

Feed-forward Gaussian prediction has been extensively explored in natural imaging [14–18], but those methods assume dense view overlap and photometric consistency across viewpoints (Fig. 2 (a)). Freehand ultrasound violates both assumptions: each frame captures only a thin 2D slice of the anatomy, so any two frames share at most a one-dimensional intersection (highlighted in Fig. 2 (c)), providing far too little overlap for dense cross-view matching; and acoustic anisotropy makes the appearance of the same tissue view-dependent, undermining photometric consistency even along that shared line. The central challenge is therefore clear: each ultrasound slice carries only a thin cross-section of the anatomy, so the information from any single frame or even a small group of frames is far too sparse to support reliable Gaussian prediction. Existing multi-view feed-forward methods partially address this by jointly processing several frames (Fig. 2 (b)), but their computational cost scales with the number of input views, making them impractical for the thousands of slices typical in freehand ultrasound. A scalable solution must therefore decouple representation size from input count (shown in Fig. 2 (d)): *first consolidate all posed slices into a fixed-size voxel support, then predict Gaussian parameters from this unified spatial context.*

We instantiate this idea in **UltraVoxelGS**, a feed-forward Gaussian splatting framework that, for the first time, enables both real-time reconstruction and physically faithful rendering for 3D ultrasound. Unlike prior feed-forward GS methods that generate pixel-aligned Gaussians from individual views and aggregate them afterward, we use probe poses to lift 2D ultrasound slices into 3D space and interpolate them into a unified voxel support, consolidating sparse observations into a coherent voxel support. We then predict Gaussian parameters directly from this shared voxel support, turning reconstruction into ***a feed-forward voxel-to-Gaussian pass***, entirely avoiding long per-scene optimization.

For rendering, a separate challenge arises from the physics of ultrasound image formation: ultrasound pixel intensity couples local reflection/scattering with cumulative attenuation, making appearance strongly direction-dependent, yet this coupling also means that slices acquired under nearby probe poses can share similar local appearance. We exploit both aspects through the proposed ***ultrasound-aware appearance rendering*** strategy, which decouples reflection/scattering from attenuation during training and leverages the local appearance similarity among neighboring slices for lightweight Render-time Appearance Adaptation. We demonstrate for the first time that feed-forward Gaussian splatting can match the reconstruction quality of per-scene optimization methods in ultrasound while reducing inference time from 10–20 minutes to 10 seconds, bringing 3D ultrasound reconstruction within reach of real-time clinical use.

In summary, our contributions are fourfold:

- We propose UltraVoxelGS, the first feed-forward Gaussian splatting framework for 3D ultrasound, enabling both real-time reconstruction and physically faithful rendering.
- We design a *feed-forward voxel-to-Gaussian pass* that lifts posed slices into a unified voxel support and predicts Gaussian parameters directly, achieving real-time 3D reconstruction.
- We introduce *ultrasound-aware appearance rendering* that decouples reflection/scattering from attenuation and exploits local appearance continuity for lightweight Render-time Appearance Adaptation, providing high-fidelity rendering at sub-second latency for operator inspection.
- We validate on two datasets spanning diverse anatomies, matching per-scene optimization baselines in quality with over $35\times$ speedup and scaling to sequences exceeding $1{,}000$ frames.

2 Related Works

3D Ultrasound Reconstruction Traditional methods depend on interpolation to aggregate images, such as kernel-based interpolation [7] to create an isotropic 3D volume. ImplicitVol [8, 19, 20] and RapidVol [9] extend this interpolation from discrete representation to continuous representation. However, all of these construct an isotropic volume, violating the anisotropic property of ultrasound. The emergence of NeRF-based methods, such as Ultra-NeRF [10, 21], formulate ultrasound parameters as continuous parameter field, then rendering images according to ultrasound imaging, following anisotropic property. These methods achieve high-quality synthesis, but suffer from low rendering speed coupled with dense neural field querying. As an alternative, GS methods model the scene as a set of Gaussian primitives, achieving fast rendering by rasterization. Recent methods focus on ultrasound imaging assumptions, assuming the imaging plane as a slice of the scene [13, 22–24] or adopting a variable frustum projection [25]. These methods render high-quality images efficiently in specific settings, but still belong to per-scene optimization.

Feed-Forward 3D Reconstruction Feed-forward methods [14] have emerged in natural imaging. These methods predict Gaussians in a single feed-forward pass. Most existing works focus on a pixel-aligned mechanism, directly predicting per-pixel Gaussian primitives and then aggregating them. More recent extensions improve scalability through structured aggregation or voxel-aligned prediction [16, 26]. Similar feed-forward ideas have also been explored in projection-based medical imaging such as sparse-view CT [27]. These methods rely on stronger cross-view correspondence but each observation in ultrasound is only a thin probe-aligned slice, cross-view overlap is extremely limited, and appearance is strongly view-dependent due to acoustic anisotropy. Therefore, existing feed-forward pipelines cannot be directly applied to freehand ultrasound.

3 Methodology

3.1 Problem Reformulation

Before presenting UltraVoxelGS, we formalize the mismatch between optical feed-forward GS paradigm and freehand ultrasound reconstruction. We first review the pixel-aligned feed-forward GS formulation used in natural imaging, then characterize ultrasound frames as sparse posed slice observations. This motivates a voxel-first reformulation that consolidates all slices into a coherent 3D support before predicting Gaussian primitives.

Pixel-aligned mechanism. Optical feed-forward GS paradigm formulates reconstruction as view-conditioned primitive prediction. Given a set of calibrated images $\{(I_i, T_i)\}_{i=1}^{N}$, a network predicts

Gaussian primitives from images and aggregates them into a continuous scene representation:

$$\mathcal{G} = \mathcal{A}\left(\{\phi_\theta(I_i, T_i)\}_{i=1}^{N}\right), \tag{1}$$

where ϕ_θ denotes the feed-forward Gaussian predictor, $\mathcal{A}$ denotes cross-view aggregation, and $\mathcal{G}$ is the Gaussian scene representation. This formulation implicitly assumes that different views provide sufficiently overlapping supports and approximately view-consistent appearance, so that primitives predicted in view-specific supports can be matched, fused, and regularized.

Ultrasound slice observations. Freehand ultrasound has a fundamentally different observation geometry. Each frame is a thin 2D slice embedded in 3D space by its probe pose, rather than a perspective image with a wide viewing frustum. We denote a pixel on an ultrasound slice as $p = (u, z)$, where u is the lateral coordinate and z is the depth coordinate. For the i-th frame, the corresponding 3D position is

$$\mathbf{x}_i(p) = T_i[u, 0, z, 1]^\top, \qquad p = (u, z) \in \Omega_i, \tag{2}$$

where Ω_i is the image domain and T_i maps the probe coordinate system to the world coordinate system. The observed slice support is therefore

$$\Pi_i = \{\mathbf{x}_i(p) \mid p \in \Omega_i\}, \tag{3}$$

where Π_i can be viewed as the 2D surface formed by all pixels of frame i after pose lifting. Unlike a perspective image that covers a broad 3D viewing frustum, an ultrasound frame only contributes a thin slice of spatial evidence. For two frames i and j, the supports Π_i and Π_j typically share at most a line-like intersection within the anatomy (refer to Fig.2 (c)). Thus, ultrasound slices do not provide the broad shared support required by Eq. (1). Although probe poses lift slice pixels into 3D space, the lifted pixels remain irregular observations on sparse 2D slice surfaces rather than a complete voxel support. Constructing V therefore requires consolidating these posed observations onto a canonical voxel grid, where many voxel locations are only weakly constrained by nearby slice evidence due to the thin and weakly overlapping imaging planes. These observations suggest that the bottleneck is the absence of an explicit shared support before prediction.

Voxel-first Problem Reformulation. Instead of predicting Gaussians directly from images, to explicitly establish a shared support, we first consolidate all posed ultrasound slices into a unified voxel support:

$$\{(I_i, T_i)\}_{i=1}^{N} \longrightarrow V \longrightarrow \mathcal{G}, \tag{4}$$

where $V \in \mathbb{R}^{X \times Y \times Z}$ denotes a canonical voxel grid over the reconstruction volume. After this consolidation, Gaussian prediction is driven by the voxel support rather than by all input image pixels, so its scale is controlled by the voxel resolution XYZ instead of the raw observation count $N|\Omega|$.

We instantiate this reformulation in **UltraVoxelGS**. As shown in the overall pipeline (Fig. 3), we first lift posed slices $\{(I_i, T_i)\}_{i=1}^{N}$ into a unified voxel support V, then predict Gaussian primitives $\mathcal{G}$ from local voxel regions via a feed-forward voxel-to-Gaussian pass (Sec. 3.2), and render novel slices through the ultrasound-aware appearance rendering strategy (Sec. 3.3).

3.2 Feed-forward Voxel-to-Gaussian Pass

Our first step is to lift posed 2D slices $\{(I_i, T_i)\}_{i=1}^{N}$ into a unified voxel support V. Following Eq. 2, each pixel $p = (u, z)$ in frame I_i is lifted to its 3D world position $\mathbf{x}_i(p)$, producing a set of irregularly distributed 3D observations$\mathcal{P} = \{(\mathbf{x}_i(p), I_i(p)) \mid p \in \Omega_i,\ i = 1, \ldots, N\}$, where $\mathbf{x}_i(p)$ denotes location and $I_i(p)$ is the corresponding pixel value. We then construct V over the reconstruction volume. Let $\mathcal{X}_V$ denote the set of voxel centers. For a voxel center $\mathbf{x} \in \mathcal{X}_V$, we define $\mathcal{N}_k(\mathbf{x})$ as the up to k nearest lifted observations within a radius α:

$$\mathcal{N}_k(\mathbf{x}) \subset \{(i, p) \mid \|\mathbf{x} - \mathbf{x}_i(p)\|_2 \le \alpha\}. \tag{5}$$

The voxel intensity is computed by local inverse-distance interpolation:

$$V(\mathbf{x}) = \frac{\sum_{(i,p)\in\mathcal{N}_k(\mathbf{x})} \omega_{i,p}(\mathbf{x}) I_i(p)}{\sum_{(i,p)\in\mathcal{N}_k(\mathbf{x})} \omega_{i,p}(\mathbf{x}) + \epsilon}, \qquad \mathbf{x} \in \mathcal{X}_V, \tag{6}$$

where $\omega_{i,p}(\mathbf{x}) = (\|\mathbf{x} - \mathbf{x}_i(p)\|_2 + \delta)^{-1}$, and δ and ϵ are small constants for numerical stability. Restricting interpolation to a local top-k neighborhood avoids filling voxels from distant observations

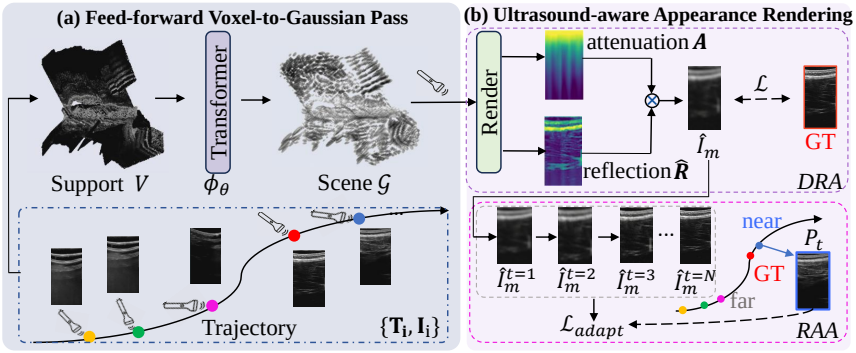


Figure 3: **Overview of UltraVoxelGS. (a)** Posed ultrasound slices $\{T_i, I_i\}$ are lifted into a shared voxel support V, which is mapped by a feed-forward Transformer ϕ_θ to the Gaussian scene representation $\mathcal{G}$. **(b)** Ultrasound-aware appearance rendering uses Decoupled Reflection and Attenuation Rendering (DRA) to render each target slice as cumulative attenuation A modulating anisotropic reflection $\hat{R}$ for image-level supervision. At render time, Render-time Appearance Adaptation (RAA) uses a nearby observed slice P_t as an appearance anchor to refine target-view appearance.

and limits the spread of unreliable measurements from shadowed regions. The resulting voxel support provides a shared voxel support for subsequent Gaussian inference.

However, for dense ultrasound sequences, the voxel support can remain high-resolution, making it computationally prohibitive to predict Gaussian primitives over the entire volume. Fortunately, doing so is also unnecessary: each frame only observes a thin imaging slice and its nearby acoustic slab, while voxels far from this slab contribute little to the rendering of that frame. We refer to this property as *thin-slab locality*, and use it to restrict voxel-to-Gaussian prediction to local sub-volumes. For each target frame, we extract a local block $\mathcal{B}_m$ from the consolidated voxel support that covers the imaging slice and its nearby acoustic slab, and predict Gaussian primitives only within this block. Because this block is carved from the full voxel support—which has already fused posed observations across the entire input sequence—prediction operates on dense, multi-view spatial context rather than sparse single-frame information.

Following this principle, for a supervised frame or target pose m, let $\mathcal{B}_m \subset \mathcal{X}_V$ denote the view-interacted local region associated with its imaging slice and nearby acoustic slab. We tokenize $\mathcal{B}_m$ into regular voxel patches $\{P_j\}_{j=1}^{J_m}$, where $P_j \in \mathbb{R}^{n\times n\times n}$. A Transformer-based predictor Φ_θ operates on these voxel tokens and predicts local voxel-aligned Gaussian primitives:

$$\{\mathcal{G}_j\}_{j=1}^{J_m} = \Phi_\theta(\{P_j\}_{j=1}^{J_m}), \qquad \widetilde{\mathcal{G}}_m = \bigcup_{j=1}^{J_m} \mathcal{G}_j. \tag{7}$$

Each predicted Gaussian primitive is parameterized as $g = \{\mathbf{c}_g, \Sigma_g, \boldsymbol{\rho}_g, \beta_g\}$, where $\mathbf{c}_g$ denotes the voxel-aligned Gaussian center, Σ_g denotes the covariance, $\boldsymbol{\rho}_g \in \mathbb{R}^9$ denotes second-order spherical-harmonic coefficients for anisotropic reflection, and β_g denotes the attenuation coefficient.

Since different view-interacted local regions can overlap, the same spatial location may receive multiple predicted Gaussian parameter vectors. Let $\mathbf{a}_k(\mathbf{x})$ denote the parameter vector predicted for location $\mathbf{x}$ by the k-th local prediction. We use a unified fusion rule: duplicated predictions are averaged, while non-duplicated predictions correspond to $K_\mathbf{x} = 1$ and are kept unchanged:

$$\bar{\mathbf{a}}(\mathbf{x}) = \frac{1}{K_\mathbf{x}} \sum_{k=1}^{K_\mathbf{x}} \mathbf{a}_k(\mathbf{x}). \tag{8}$$

During inference, UltraVoxelGS constructs the voxel support V once from the input sequence, predicts Gaussian primitives from the view-interacted local regions, and assembles them using Eq. 8. The fused and non-duplicated predictions together form the Gaussian scene representation $\mathcal{G}$. Importantly, this local-to-global assembly is not merely a computational shortcut: each local prediction is tied to the acoustic slab relevant to a target slice, while its input region is selected from the

consolidated voxel support that integrates posed observations from the sequence. UltraVoxelGS thus avoids redundant whole-support prediction while preserving the slice-relevant information needed for accurate ultrasound rendering, enabling scalable reconstruction for long ultrasound sequences.

3.3 Ultrasound-aware Appearance Rendering

Given the predicted Gaussian scene representation $\mathcal{G}$, faithful rendering requires preserving ultrasound-specific image formation rather than treating the target slice as a generic intensity image. As shown in Fig. 4, ultrasound appearance is governed by anisotropic reflection and cumulative attenuation[28]: even for the same anatomical plane, images can vary significantly with probe pose, as different poses induce different acoustic paths through the tissue. Only when probe poses are sufficiently close, and thus the acoustic paths are similar, do the resulting images exhibit similar appearance. Based on these properties, we introduce *ultrasound-aware appearance rendering* strategy, consisting of *Decoupled Reflection and Attenuation Rendering* for training supervision and *Render-time Appearance Adaptation* for refinement from nearby observed slices.

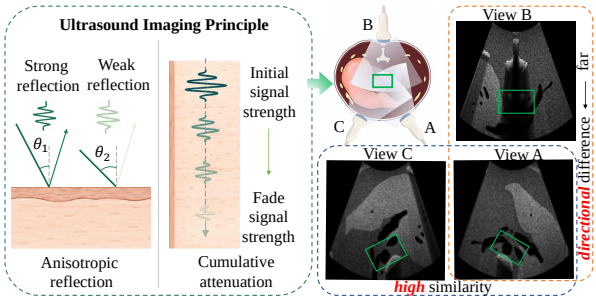


Figure 4: Ultrasound appearance modeling. Distant views are direction-dependent, while nearby views remain locally similar for Render-time Appearance Adaptation.

3.3.1 Decoupled Reflection and Attenuation Rendering

The predicted Gaussian representation is rendered into ultrasound slices for supervision. For a target frame m and pixel $p = (u, z) \in \Omega_m$, we model ultrasound intensity as anisotropic reflection modulated by cumulative attenuation:

$$I_m(p) = R_m(p, \mathbf{d}_m(p)) \cdot \exp\left(-\int_0^z \beta_m(u, \zeta)\, d\zeta\right), \tag{9}$$

where $\mathbf{d}_m(p)$ denotes the acoustic propagation direction under pose T_m, R_m denotes the anisotropic reflection response, and β_m denotes the attenuation coefficient.

Using the predicted Gaussian primitives, we instantiate this model with Decoupled Reflection and Attenuation Rendering. Let $\mathcal{G}_m(p)$ be the set of Gaussians contributing to pixel p under pose T_m, and let $w_{g,m}(p)$ be the contribution weight of Gaussian g. We compute the anisotropic reflection term as $\hat{R}_m(p) = \sum_{g\in\mathcal{G}_m(p)} S_{\mathrm{SH}}(\mathbf{d}_m(p); \boldsymbol{\rho}_g)\, w_{g,m}(p)$, where $S_{\mathrm{SH}}(\mathbf{d}_m(p); \boldsymbol{\rho}_g)$ denotes the second-order spherical-harmonic response parameterized by $\boldsymbol{\rho}_g$. The local attenuation coefficient is computed as $\hat{\beta}_m(p) = \sum_{g\in\mathcal{G}_m(p)} \beta_g\, w_{g,m}(p)$, and the accumulated transmittance along depth is $A_m(p) = \exp(-\sum_{\zeta<z} \hat{\beta}_m(p_\zeta)\Delta z)$ with $p_\zeta = (u, \zeta)$. The final rendered intensity is then $\hat{I}_m(p) = A_m(p)\hat{R}_m(p)$. This renderer is slice-consistent and physically decoupled: $\hat{R}_m(p)$ captures anisotropic reflection, while $A_m(p)$ captures cumulative attenuation.

During training, the fused Gaussian representation $\mathcal{G}$ is optimized with a rendering loss and an overlap consistency loss:

$$\mathcal{L} = \sum_{m\in\mathcal{M}} \sum_{p\in\Omega_m} \left\| \hat{I}_m(p) - I_m(p) \right\|_2^2 + \lambda \sum_{\mathbf{x}\in\mathcal{X}_{\mathrm{ov}}} \frac{1}{K_{\mathbf{x}}} \sum_{k=1}^{K_{\mathbf{x}}} \|\mathbf{a}_k(\mathbf{x}) - \bar{\mathbf{a}}(\mathbf{x})\|_2^2, \tag{10}$$

where $\mathcal{X}_{\mathrm{ov}}$ denotes spatial locations with multiple predictions. The first term supervises the final rendered intensity $\hat{I}_m(p)$ against the observed image $I_m(p)$, without requiring explicit reflection or attenuation annotations. The second term addresses a consistency issue: since thin-slab locality allows the same spatial location to be predicted from multiple overlapping slabs, the resulting Gaussian parameters may diverge across slabs. The overlap consistency loss penalizes the deviation of each predicted parameter vector $\mathbf{a}_k(\mathbf{x})$ from its fused average $\bar{\mathbf{a}}(\mathbf{x})$ over all $K_{\mathbf{x}}$ overlapping predictions at location $\mathbf{x}$, and λ balances rendering fidelity and overlap consistency.

Table 1: Quantitative comparison on real-world and simulated ultrasound datasets. The best and second-best results are highlighted in bold and underline, respectively.

Method	Type	Dataset A: Real-world			Dataset B: Simulated		
		PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Kernel Interpolation [6]	Interpolation	13.52	0.3952	0.5419	17.47	0.2391	0.6259
ImplicitVol [8]	Per-scene optimization	19.13	0.6128	0.5218	<u>18.97</u>	<u>0.2837</u>	0.7192
UltraNeRF [10]	Per-scene optimization	18.49	0.4056	0.5761	17.75	0.1975	**0.5985**
3DGS [12]	Per-scene optimization	<u>22.55</u>	<u>0.6695</u>	<u>0.3884</u>	18.04	0.2532	0.6702
UltraGauss [13]	Per-scene optimization	22.06	0.6220	0.4750	15.01	0.1838	0.7057
UltraVoxelGS w/o RAA	Feed-forward	19.76	0.5770	0.5130	18.25	0.2690	0.7860
UltraVoxelGS (Ours)	Feed-forward	**24.29**	**0.8329**	**0.2384**	**20.87**	**0.3138**	<u>0.6190</u>

3.3.2 Render-time Appearance Adaptation

The shared voxel support provides a common prediction domain, but voxel support interpolation can smooth local ultrasound appearance and reduce slice-level sharpness. Motivated by local appearance continuity, we use a nearby observed slice as an appearance anchor to provide sharper local appearance evidence for the target rendering. Given the initial Gaussian representation $\mathcal{G}_0 = \{(\mathbf{c}_g^0, \Sigma_g^0, \boldsymbol{\rho}_g^0, \beta_g^0)\}$ predicted by UltraVoxelGS, we freeze the geometric parameters $(\mathbf{c}_g^0, \Sigma_g^0)$ and update only a transient copy of the appearance parameters $(\boldsymbol{\rho}_g, \beta_g)$ for the target rendering. For a target pose T_t, we select the nearest observed source frame as the appearance anchor:

$$i^*(t) = \arg\min_{i \in \mathcal{S}} \|\mathbf{o}_i - \mathbf{o}_t\|_2, \qquad P_t = I_{i^*(t)}, \tag{11}$$

where $\mathcal{S}$ denotes the set of observed frames, $\mathbf{o}_i$ and $\mathbf{o}_t$ denote the probe centers of poses T_i and T_t, respectively, and P_t is the selected source frame used as the appearance anchor. Then, the adaptation objective, denoted as $\mathcal{L}_{\text{adapt}}$ in Fig. 3(c), optimizes the transient appearance copy while keeping geometry fixed:

$$(\boldsymbol{\rho}_t^*, \boldsymbol{\beta}_t^*) = \arg\min_{\boldsymbol{\rho}, \boldsymbol{\beta}} \underbrace{\left\| \mathcal{R}\left(\mathcal{G}\left(\mathbf{c}^0, \Sigma^0, \boldsymbol{\rho}, \boldsymbol{\beta}\right); T_{i^*(t)}\right) - P_t \right\|_2^2 + \gamma \left\| \boldsymbol{\rho} - \boldsymbol{\rho}^0 \right\|_2^2 + \eta \left\| \boldsymbol{\beta} - \boldsymbol{\beta}^0 \right\|_2^2}_{\mathcal{L}_{\text{adapt}}}. \tag{12}$$

where $\mathcal{R}(\mathcal{G}; T)$ denotes the DRA renderer, and γ, η regularize the adapted appearance parameters toward the feed-forward prediction. The first term uses the selected source frame P_t as an appearance anchor, while the regularization terms keep the transient update close to the feed-forward appearance. The adapted parameters are used only to render the target pose T_t and are discarded afterward. Because the adapted appearance parameters are discarded after rendering each target view, this stage only affects per-view rendering and leaves the reconstructed global representation unchanged.

4 Experiments

4.1 Datasets and implementation details

Dataset A: Real-world Robotic Ultrasound Video Sequences. This dataset consists of robot-assisted lumbar spine ultrasound scans with synchronized probe pose tracking [29], acquired from 63 healthy volunteers (used under CC BY 4.0). Each scan contains over 3K densely sampled frames. In the default setting, we uniformly sample every 5-th frame as input and supervision, and evaluate on frames with index $3 \bmod 5$ to avoid near-duplicate testing.

Dataset B: Simulated Liver Ultrasound Sequences. This dataset contains 100 simulated liver ultrasound sequences rendered from abdominal CT volumes from 3D-IRCADb [30] and CRLM-CT [31] based on LOTUS [32] (used under CC BY-NC-ND 4.0 and CRLM-CT under CC BY 4.0). We hold out every 7-th frame as novel views and use the remaining frames as input and supervision.

All datasets are used for non-commercial research purposes in accordance with their respective terms. Additional implementation details, baseline settings, and dataset/tool license information are provided in Appendix A.

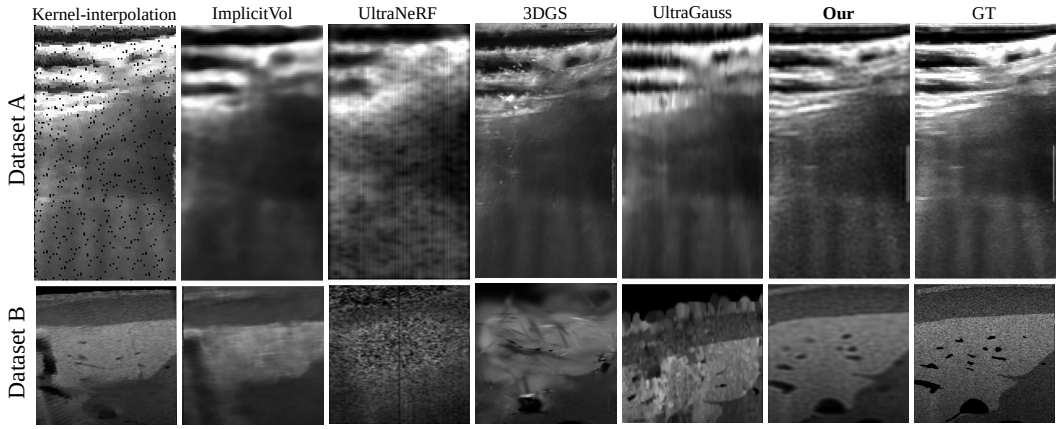


Figure 5: Qualitative comparison on real-world and simulated ultrasound datasets.

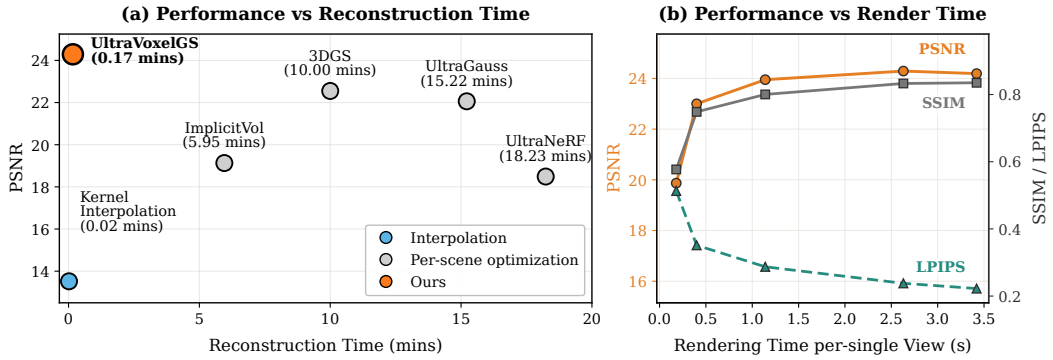


Figure 6: Quality–efficiency trade-off on Dataset A.

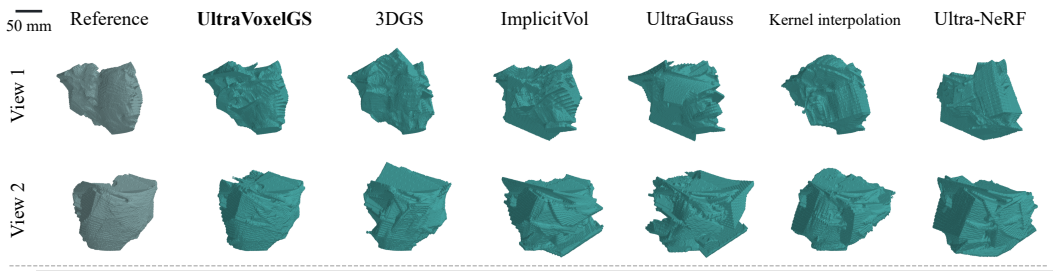
4.2 Main results on real-world and simulated data

Quantitative comparison. Tab. 1 compares UltraVoxelGS with interpolation-based and per-scene optimization baselines on Datasets A and B. UltraVoxelGS achieves the best overall performance on real tracked scans and remains competitive under controlled simulation, while using a feed-forward reconstruction pipeline for sub-second rendering rather than per-scene optimization. These results support the effectiveness of the voxel-first formulation and ultrasound-aware appearance rendering for held-out slice reconstruction. The contribution of RAA is further analyzed in Appendix B.5, highlighting its role in compensating for fine details lost during feed-forward reconstruction.

Qualitative comparison. Fig. 5 shows representative qualitative comparisons on Dataset A and B. Interpolation-based reconstruction produces visible interpolation artifacts and loses anatomical detail. ImplicitVol yields overly smoothed responses, while UltraNeRF exhibits noisy and unstable texture patterns. 3DGS preserves some high-frequency appearance but introduces unstable speckle-like artifacts and local structural distortions. UltraGauss recovers part of the anatomical structure, but still shows degraded contrast and less coherent local organization. In comparison, UltraVoxelGS reconstructs clearer layered structures and more faithful local appearance.

Segmentation-based anatomical consistency. To assess 3D anatomical consistency, we evaluate on Dataset B using CT-derived liver segmentation as the reference. For each method, we propagate an observed anchor-frame mask to the rendered novel views using the same frozen SAM2 model [33], and fuse the resulting masks into a 3D anatomical reconstruction using the corresponding poses. As shown in Fig. 7, UltraVoxelGS achieves the highest 2D Dice, 3D Dice, and 3D IoU, with reconstructed anatomy most closely matching the reference shape, supporting its improved preservation of 3D anatomical consistency.

Performance–efficiency trade-off. Fig. 6 (a) shows the relationship between quality and total per-scene reconstruction time on Dataset A. UltraVoxelGS remains in the seconds-level reconstruction



Method	UltraVoxelGS	3DGS	ImplicitVol	UltraGauss	Kernel interpolation	Ultra-NeRF
2D Dice ↑	**0.8469**	0.7441	0.6419	0.5153	0.6937	0.5215
3D Dice ↑	**0.8787**	0.8593	0.7283	0.6802	0.7683	0.5776
3D IoU ↑	**0.7866**	0.7557	0.5758	0.5188	0.6277	0.4115

Figure 7: 3D anatomical segmentation results on Dataset B. Top: one representative example visualized from two viewpoints; Bottom: quantitative evaluation across all test cases.

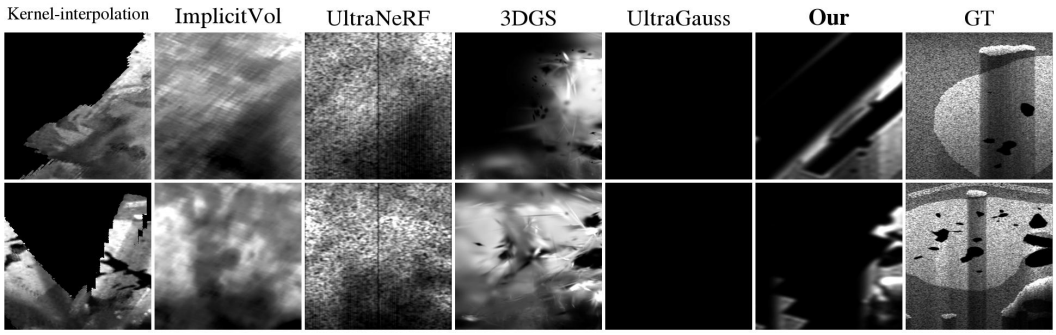


Figure 8: **Failure cases on Dataset B.** Visual comparisons under challenging scanning conditions, including sparse coverage, acoustic shadowing, and inaccurate probe poses.

regime while matching or surpassing per-scene optimization baselines, showing that our method improves the full reconstruction pipeline rather than merely accelerating rendering. Fig. 6 (b) further analyzes the Render-time Appearance Adaptation trade-off: increasing adaptation steps improves PSNR and SSIM, but also increases per-view rendering cost. We use 50 adaptation steps in the main experiments to balance reconstruction quality and rendering efficiency. A detailed runtime breakdown is provided in Appendix B.1.

4.3 Ablation and analysis

Ablation study. Tab. 2 isolates the contributions of feed-forward voxel prediction (FF), Decoupled Reflection and Attenuation Rendering (DRA), and Render-time Appearance Adaptation (RAA) on Dataset A. Starting from direct interpolation, introducing feed-forward voxel prediction improves PSNR but remains limited without a Gaussian scene representation and ultrasound-aware rendering. Adding DRA moves the model from voxel intensity prediction to slice-consistent Gaussian rendering with decoupled anisotropic reflection and cumulative attenuation, substantially improving PSNR, SSIM, and LPIPS. Finally, RAA provides the largest gain across all metrics by using nearby observed slices as appearance anchors for target-view refinement. Overall, the ablation supports the two main design choices of UltraVoxelGS: voxel-first feed-forward reconstruction and ultrasound-aware appearance rendering. We further compare UltraVoxelGS a generic feed-forward GS baseline, to assess the effectiveness of our ultrasound-specific design in Appendix C.4.

Scalability analysis. We evaluate long-sequence scalability by varying the input sampling stride on Dataset A at reconstruction time, without retraining the model. As shown in Tab. 3, UltraVoxelGS scales to over 1,000 input frames: denser inputs consistently improve reconstruction quality, while peak memory increases only mildly. This supports the effectiveness of the proposed local-to-global assembly for dense tracked ultrasound sequences.

Table 2: Ablation study on Dataset A. We isolate the contributions of feed-forward (FF), Decoupled Reflection and Attenuation Rendering (DRA), and Render-time Appearance Adaptation (RAA).

FF	DRA	RAA	PSNR↑	SSIM↑	LPIPS↓
✗	✗	✗	13.52	0.3950	0.5420
✓	✗	✗	14.46	0.3825	0.5735
✓	✓	✗	19.76	0.5770	0.5130
✓	✓	✓	**24.29**	**0.8330**	**0.2380**

Table 3: Long-sequence scalability on Dataset A. UltraVoxelGS remains stable as input density increases from sparse to dense tracked scans, without retraining. Note that 'PM' indicates peak memory (GB) during inference.

Stride	Frames	PSNR↑	SSIM↑	LPIPS↓	PM↓
10×	∼300	23.75	0.7940	0.2546	**3.66**
5×	∼600	24.28	0.8329	0.2386	3.80
2×	∼1600	**24.89**	**0.8777**	**0.2175**	3.94

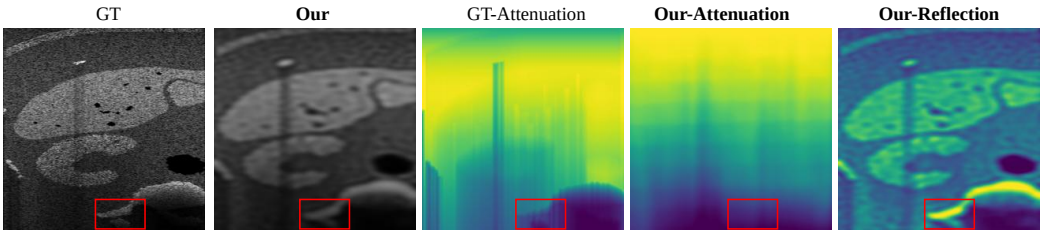


Figure 9: Decoupled Reflection and Attenuation Rendering visualization on Dataset B. Note that the ground-truth attenuation maps are obtained from the simulation pipeline [32] and serve solely as a visual reference; no attenuation supervision is used during training. The red boxes highlight a structure that is present in the reflection component but becomes weak in the final ultrasound appearance due to cumulative attenuation.

Attenuation and reflection representation. Dataset B provides simulator reference attenuation maps, which we use only as a visual reference rather than training supervision. Fig. 9 shows that UltraVoxelGS reconstructs the target slice while decomposing the rendered appearance into a cumulative attenuation component and an anisotropic reflection component. The learned attenuation component roughly follows the dominant depth-dependent trend in the simulator reference, while the reflection component captures complementary anatomical structures and local echo responses. Additionally, We quantitatively compare the learned attenuation component with the reference attenuation map: $\text{Pearson} = 0.6966 \pm 0.0259, \quad \text{Spearman} = 0.6699 \pm 0.0331$. Compared with a depth-linear decay baseline, whose slope and intercept are least-squares fitted to the reference ($\text{Pearson} = 0.6648$, $\text{Spearman} = 0.6545$), our results confirm that the factorization captures non-trivial attenuation structure despite receiving no supervision.

Accessibility of clinical workflow. Fig. 8 shows representative failure cases under extremely sparse coverage, strong acoustic shadowing, or noticeable probe tracking errors. Compared with ImplicitVol and UltraNeRF, which may produce dense but over-smoothed or noisy responses in weakly observed regions, UltraVoxelGS tends to preserve evidence-supported structures while leaving under-covered regions visually incomplete. Such incompleteness can also provide immediate spatial feedback, indicating regions that may require additional scanning, as illustrated in Fig. 1. To further validate clinical accessibility, we additionally evaluate our method on a handheld dataset in Appendix D.2, further demonstrating its value in real-world settings. Moreover, the periodic reconstruction mode presented in Appendix E demonstrates its potential for continuous scanning.

5 Conclusion

We have presented UltraVoxelGS, a feed-forward framework that, for the first time, achieves both fast reconstruction and physically faithful rendering for ultrasound imaging. The core insight is that when views are geometrically thin and photometrically inconsistent, the right strategy is to consolidate all observations into a shared volumetric representation first and predict locally. Paired with an ultrasound-aware appearance rendering strategy grounded in acoustic image formation, our design delivers reconstruction quality surpassing per-scene optimization baselines while reducing inference time from minutes to seconds and scaling to over 1,000 frames. Looking ahead, we hope UltraVoxelGS can enable real-time spatial feedback during ultrasound acquisition, guiding sonographers toward uncovered regions. Limitations are discussed in Appendix F.

6 Acknowledgement

This work is supported by Hong Kong Research Grants Council (RGC) Early Career Scheme grant 22203525. The work is also supported by Guangdong and Hong Kong Universities "1+1+1" Joint Research Collaboration Scheme. The authors declare no competing interests.

References

[1] A. Fenster, D. B. Downey, and H. N. Cardinal, "Three-dimensional ultrasound imaging," *Physics in medicine & biology*, vol. 46, no. 5, pp. R67–R99, 2001.

[2] B. R. Benacerraf, "Three-dimensional fetal sonography: use and misuse," 2002.

[3] C. A. Adriaans, M. Wijkhuizen, L. M. van Karnenbeek, F. Geldof, and B. Dashtbozorg, "Trackerless 3d freehand ultrasound reconstruction: a review," *Applied Sciences*, vol. 14, no. 17, p. 7991, 2024.

[4] H. Shi, Z. Zhang, B. Mailhe, Y. Liu, X. Chen, A. Mukherjee, T. Chen, and S. Sun, "Real-time novel-view freehand ultrasound imaging via point-cloud rendering and diffusion-bridge completion," in *Medical Imaging with Deep Learning*, 2026.

[5] H. Guo, S. Xu, B. Wood, and P. Yan, "Sensorless freehand 3d ultrasound reconstruction via deep contextual learning," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 463–472, Springer, 2020.

[6] O. V. Solberg, F. Lindseth, H. Torp, R. E. Blake, and T. A. N. Hernes, "Freehand 3d ultrasound reconstruction algorithms—a review," *Ultrasound in medicine & biology*, vol. 33, no. 7, pp. 991–1009, 2007.

[7] X. Chen, T. Wen, X. Li, W. Qin, D. Lan, W. Pan, and J. Gu, "Reconstruction of freehand 3d ultrasound based on kernel regression," *Biomedical engineering online*, vol. 13, no. 1, p. 124, 2014.

[8] P.-H. Yeung, L. Hesse, M. Aliasi, M. Haak, W. Xie, A. I. Namburete, I. 21st Consortium, *et al.*, "Implicitvol: Sensorless 3d ultrasound reconstruction with deep implicit representation," *arXiv preprint arXiv:2109.12108*, 2021.

[9] M. C. Eid, P.-H. Yeung, M. K. Wyburd, J. F. Henriques, and A. I. Namburete, "Rapidvol: rapid reconstruction of 3d ultrasound volumes from sensorless 2d scans," in *2025 IEEE 22nd International Symposium on Biomedical Imaging (ISBI)*, pp. 1–5, IEEE, 2025.

[10] M. Wysocki, M. F. Azampour, C. Eilers, B. Busam, M. Salehi, and N. Navab, "Ultra-nerf: Neural radiance fields for ultrasound imaging," in *Medical Imaging with Deep Learning*, pp. 382–401, PMLR, 2024.

[11] T. Grutman, M. Bismuth, B. Glickstein, and T. Ilovitsh, "Implicit neural representation for scalable 3d reconstruction from sparse ultrasound images," *npj Acoustics*, vol. 1, no. 1, p. 14, 2025.

[12] B. Kerbl, G. Kopanas, T. Leimkühler, G. Drettakis, *et al.*, "3d gaussian splatting for real-time radiance field rendering.," *ACM Trans. Graph.*, vol. 42, no. 4, pp. 139–1, 2023.

[13] M. C. Eid, A. I. Namburete, and J. F. Henriques, "Ultragauss: Ultrafast gaussian reconstruction of 3d ultrasound volumes," *arXiv preprint arXiv:2505.05643*, 2025.

[14] D. Charatan, S. L. Li, A. Tagliasacchi, and V. Sitzmann, "pixelsplat: 3d gaussian splats from image pairs for scalable generalizable 3d reconstruction," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 19457–19467, 2024.

[15] Y. Chen, H. Xu, C. Zheng, B. Zhuang, M. Pollefeys, A. Geiger, T.-J. Cham, and J. Cai, "Mvsplat: Efficient 3d gaussian splatting from sparse multi-view images," in *European conference on computer vision*, pp. 370–386, Springer, 2024.

[16] L. Jiang, Y. Mao, L. Xu, T. Lu, K. Ren, Y. Jin, X. Xu, M. Yu, J. Pang, F. Zhao, *et al.*, "Anysplat: Feed-forward 3d gaussian splatting from unconstrained views," *ACM Transactions on Graphics (TOG)*, vol. 44, no. 6, pp. 1–16, 2025.

[17] B. Ye, S. Liu, H. Xu, X. Li, M. Pollefeys, M.-H. Yang, and S. Peng, "No pose, no problem: Surprisingly simple 3d gaussian splats from sparse unposed images," *arXiv preprint arXiv:2410.24207*, 2024.

[18] H. Xu, S. Peng, F. Wang, H. Blum, D. Barath, A. Geiger, and M. Pollefeys, "Depthsplat: Connecting gaussian splatting and depth," in *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 16453–16463, 2025.

[19] H. Chen, L. Kumaralingam, S. Zhang, S. Song, F. Zhang, H. Zhang, T.-T. Pham, K. Punithakumar, E. H. Lou, Y. Zhang, *et al.*, "Neural implicit surface reconstruction of freehand 3d ultrasound volume with geometric constraints," *Medical Image Analysis*, vol. 98, p. 103305, 2024.

[20] S. Song, Y. Chen, D. Xu, S. Ge, Y. Huang, J. Shi, M. Chen, H. Chen, and R. Zheng, "Implicitcell: Resolution cell modeling of joint implicit volume reconstruction and pose refinement in freehand 3d ultrasound," *arXiv preprint arXiv:2503.06686*, 2025.

[21] R. Dagli, A. Hibi, R. G. Krishnan, and P. N. Tyrrell, "Nerf-us: Removing ultrasound imaging artifacts from neural radiance fields in the wild," *arXiv preprint arXiv:2408.10258*, 2024.

[22] X. Guo, M. Alsharid, H. Zhao, Y. Wang, J. Lander, A. T. Papageorghiou, and J. A. Noble, "A visually grounded language model for fetal ultrasound understanding," *Nature Biomedical Engineering*, pp. 1–17, 2026.

[23] M. Krähenmann, S. Tascon-Morales, F. Laumer, J. E. Vogt, and E. Ozkan, "From slices to structures: Unsupervised 3d reconstruction of female pelvic anatomy from freehand transvaginal ultrasound," *arXiv preprint arXiv:2508.14552*, 2025.

[24] D. Kong, Y. Wang, W. Guo, Y. Bu, B. Zhang, Y. Duan, X. Yue, W. Du, Y. Zhong, Y. Chen, *et al.*, "Gaussianpile: A unified sparse gaussian splatting framework for slice-based volumetric reconstruction," *arXiv preprint arXiv:2603.20611*, 2026.

[25] Y. Yang, W. Cai, D. Yang, Y. Dong, X. Dong, and Z. Jin, "Ultrags: Gaussian splatting for ultrasound novel view synthesis," *arXiv preprint arXiv:2511.07743*, 2025.

[26] W. Wang, Y. Chen, Z. Zhang, H. Liu, H. Wang, Z. Feng, W. Qin, F. Chen, Z. Zhu, D. Y. Chen, *et al.*, "Volsplat: Rethinking feed-forward 3d gaussian splatting with voxel-aligned prediction," *arXiv preprint arXiv:2509.19297*, 2025.

[27] Y. Liu, W. Li, W. Yu, C. Li, A. Alahi, M. Meng, and Y. Yuan, "X-grm: Large gaussian reconstruction model for sparse-view x-rays to computed tomography," *arXiv preprint arXiv:2505.15235*, 2025.

[28] J. A. Jensen, "Ultrasound imaging and its modeling," in *Imaging of complex media with acoustic and seismic waves*, pp. 135–166, Springer, 2002.

[29] N. A. Cavalcanti, R. Li, L. Arango, A. Davoodi, K. Van Assche, Y. Ao, A. Massalimova, M. Salehi, L. Zingg, T. Götschi, *et al.*, "A large, paired dataset of robotic and handheld lumbar spine ultrasound with ground-truth ct benchmarking," *Scientific Data*, vol. 12, no. 1, p. 1766, 2025.

[30] L. Soler, A. Hostettler, V. Agnus, A. Charnoz, J.-B. Fasquel, J. Moreau, A.-B. Osswald, M. Bouhadjar, and J. Marescaux, "3d image reconstruction for comparison of algorithm database," *URL: https://www. ircad. fr/research/data-sets/liver-segmentation-3d-ircadb-01*, 2010.

[31] A. L. Simpson, J. Peoples, J. M. Creasy, G. Fichtinger, N. Gangai, K. N. Keshavamurthy, A. Lasso, J. Shia, M. I. D'Angelica, and R. K. Do, "Preoperative ct and survival data for patients undergoing resection of colorectal liver metastases," *Scientific data*, vol. 11, no. 1, p. 172, 2024.

[32] Y. Velikova, M. F. Azampour, W. Simson, V. Gonzalez Duque, and N. Navab, "Lotus: learning to optimize task-based us representations," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 435–445, Springer, 2023.

[33] N. Ravi, V. Gabeur, Y.-T. Hu, R. Hu, C. Ryali, T. Ma, H. Khedr, R. Rädle, C. Rolland, L. Gustafson, *et al.*, "Sam 2: Segment anything in images and videos," in *International Conference on Learning Representations*, vol. 2025, pp. 28085–28128, 2025.

Appendix

We provide additional experimental details, runtime and RAA analyses, ablation and robustness studies, reconstruction results, clinical workflow discussions, and limitations to supplement the main paper.

A Experimental Details

A.1 Implementation details.

UltraVoxelGS first constructs a unified voxel support from posed ultrasound slices and then predicts voxel-aligned Gaussian primitives from view-interacted local regions. Unless otherwise specified, the input image resolution is 176×88, the voxel grid spacing is $0.00075\,\mathrm{m}$, and each voxel patch has size $8 \times 8 \times 8$. For voxel support construction, each voxel uses up to $k = 3$ nearest lifted observations. The Gaussian primitive is parameterized as $(\mathbf{c}_g, \Sigma_g, \boldsymbol{\rho}_g, \beta_g)$, where $\boldsymbol{\rho}_g \in \mathbb{R}^9$ corresponds to second-order spherical-harmonic coefficients for anisotropic reflection and β_g denotes the attenuation coefficient.

We train UltraVoxelGS using Adam with learning rate 1×10^{-5} for 300k iterations on one NVIDIA H20 GPU. The batch size is varied between 2 and 12 according to the number of selected view-interacted local regions and GPU memory. The overlap consistency weight is set to $\lambda = 0.01$. Render-time Appearance Adaptation (RAA) is disabled during training and is applied only during target-view rendering.

Table 4: **Implementation details of UltraVoxelGS.**

Item	Setting
Input image resolution	176×88
Voxel grid spacing	$0.00075\,\mathrm{m}$
Voxel patch size	$8 \times 8 \times 8$
Top-k neighbors	3
SH order	2
Gaussian parameters	$(\mathbf{c}_g, \Sigma_g, \boldsymbol{\rho}_g, \beta_g)$
Optimizer	Adam
Learning rate	1×10^{-5}
Training steps	300k
Batch size	2–12
Overlap consistency weight λ	0.01
Training GPU	$1\times$ NVIDIA H20

A.2 Baseline Settings

We compare UltraVoxelGS with representative interpolation-based, continuous-field, neural rendering, and optimization-based Gaussian reconstruction methods. All methods use the same input and

supervision frames and are evaluated on the same held-out target views whenever applicable. For optimization-based baselines, we follow their default reconstruction pipelines and report total per-scene reconstruction time under the same evaluation protocol.

Table 5: **Baseline settings.** We summarize the main baseline configurations used for comparison.

Method	Type	Input Setting	Optimization / Inference Setting
Kernel Interpolation	Interpolation	Same sparse input	No optimization
ImplicitVol	Per-scene optimization	Same sparse input	50k steps
UltraNeRF	Per-scene optimization	Same sparse input	10k steps
3DGS	Per-scene optimization	Same sparse input	10k steps
UltraGauss	Per-scene optimization	Same sparse input	10k steps
UltraVoxelGS w/o RAA	Feed-forward	Same sparse input	Feed-forward only
UltraVoxelGS	Feed-forward	Same sparse input	Feed-forward + RAA

A.3 Additional Details for Long-sequence Evaluation

For the long-sequence scalability experiment in the main paper, we vary the input sampling stride on Dataset A without retraining the model. The original robotic scans contain over 3K tracked frames. We use $10\times$, $5\times$, and $2\times$ sampled inputs, corresponding to approximately 300, 600, and 1,600 input frames, respectively. All settings are evaluated on the same held-out subset with view index $3 \bmod 10$, containing 2,047 novel views from 8 scans. This protocol isolates reconstruction-time scalability under increasingly dense inputs while keeping the trained model fixed.

B Runtime and Render-time Adaptation Analysis

B.1 Runtime Breakdown

Table 6: **Runtime breakdown of UltraVoxelGS.** We report per-scene representation construction time and average per-view rendering time on Dataset A.

Stage	Time (s) $\downarrow$
Voxel support construction	1.12
View-interacted block extraction	1.96
Network inference	4.83
Gaussian fusion	2.13
Representation construction	10.04
Rendering per target view w/o RAA	0.00362
Rendering per target view w/ 50-step RAA	0.623

UltraVoxelGS constructs the Gaussian representation in 10.04 s per case. The representation construction time includes voxel support construction, view-interacted block extraction, feed-forward network inference, and Gaussian fusion. Render-time Appearance Adaptation does not change the reconstructed global representation; it only increases the per-view rendering cost. The average rendering time is 3.62 ms per target view without RAA (2.53 ms rasterization + 1.09 ms filtering) and 623 ms per target view with 50-step RAA. Rendering time is averaged over 651 held-out target views, sharded across three V100 GPUs for throughput measurement.

B.2 Render-time Appearance Adaptation Step Ablation

Effect of adaptation steps. We evaluate the quality–efficiency trade-off of Render-time Appearance Adaptation (RAA) by varying the number of update steps. For all variants, the feed-forward geometry is fixed, and only a transient copy of the appearance parameters is updated for each target rendering and discarded afterward. As shown in Tab. 7, more adaptation steps generally improve reconstruction quality at the cost of higher per-view rendering time. Fig. 10 further shows the corresponding visual trend, where RAA progressively sharpens local structures and contrast toward the ground truth.

Table 7: **Render-time Appearance Adaptation step ablation.** We evaluate the quality–efficiency trade-off of different adaptation steps on Dataset A.

Variant	Steps	PSNR↑	SSIM↑	LPIPS↓	Render / View (s)↓
w/o RAA	0	19.76	0.5770	0.5130	**0.00362**
RAA	10	23.00	0.7484	0.3513	0.140
RAA	20	23.95	0.8003	0.2872	0.259
RAA	50	**24.29**	0.8329	0.2380	0.623
RAA	100	24.19	**0.8350**	**0.2224**	1.212

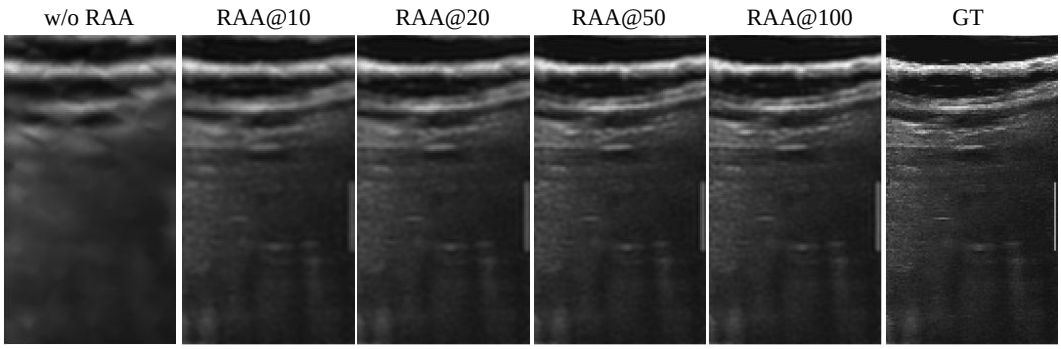


Figure 10: **Visual effect of Render-time Appearance Adaptation.** RAA@K denotes applying K steps of Render-time Appearance Adaptation. Increasing the number of RAA steps progressively improves local sharpness and contrast toward the ground truth.

Increasing the number of adaptation steps improves local appearance fidelity but also increases per-view rendering time. We use 50 adaptation steps in the main experiments to balance reconstruction quality and rendering efficiency.

B.3 RAA Sensitivity to Target–Anchor Discrepancy

To analyze the sensitivity of RAA to pose variation, we rank all candidate anchors for each target view by either translational distance or rotational discrepancy, and apply RAA using anchors at different ranks (Top-1, Top-3, Top-5, and Top-10). As shown in Tab. 8, RAA maintains strong refinement within a small spatial and angular neighborhood, and although the benefit gradually decreases as the anchor's translational or rotational discrepancy grows, it consistently outperforms UltraVoxelGS w/o RAA even at Top-10.

B.4 Comparison with Per-scene Optimization

RAA differs from short per-scene optimization in two aspects (shown in Tab. 9). First, they differ in *what is optimized*: RAA adapts transient appearance parameters while keeping the reconstructed geometry frozen, whereas per-scene optimization updates the full scene representation itself. Second, they differ in *how the cost scales*: RAA is a query-time refinement with a fixed per-query cost that is independent of the number of source views, whereas per-scene optimization scales with the length of the input sequence, making it ill-suited to long ultrasound acquisitions. Overall, RAA is not a shortened per-scene reconstruction but a query-time appearance adaptation, offering a more practical refinement strategy for interactive ultrasound inspection.

B.5 Role of RAA Across Different Acquisition Densities

The feed-forward pass and RAA are complementary: whenever interpolation smooths out detail, RAA restores it—decisively when the effect is severe (Dataset A) and as a consistent refinement when it is mild (Dataset B). RAA is thus an integral component, not an optional add-on, keeping UltraVoxelGS robust across acquisition densities.

Table 8: RAA refinement under translation- and rotation-ranked anchors at different Top-k ranks on Dataset B.

Strategy	Anchor	Translation distance (mm)	Rotation angle (°)	PSNR↑	SSIM↑	LPIPS↓
UltraVoxelGS	Top-1	7.87 ± 3.52	2.78 ± 2.31	20.87	0.3138	0.6190
Translation	Top-3	14.09 ± 5.56	5.16 ± 3.87	19.89	0.2889	0.6919
	Top-5	19.09 ± 6.83	6.80 ± 4.47	19.39	0.2813	0.7018
	Top-10	28.86 ± 8.27	10.28 ± 6.12	18.76	0.2719	0.7136
Rotation	Top-3	16.98 ± 10.25	4.11 ± 1.95	19.91	0.2889	0.6922
	Top-5	22.24 ± 10.46	5.59 ± 2.39	19.41	0.2812	0.7020
	Top-10	34.32 ± 14.24	8.45 ± 3.10	18.68	0.2707	0.7156
UltraVoxelGS w/o RAA	-	-	-	18.25	0.2690	0.7860

Table 9: Per-scene optimization versus RAA.

	Per-scene optimization	RAA
Target	Complete scene	Current queried view
Updated variables	Geometry and appearance	Transient ρ, β
Global representation	Optimized per scene	Feed-forward and fixed
Scope	Entire scene	Query-relevant local support
Persistence	Stored in the scene	Discarded after rendering
Runtime	Increases with scene scale and input views	Independent of source views (623 ms / 50 steps)

On Dataset A, the dense, closely-spaced poses cause interpolation to oversmooth fine structures and degrade the feed-forward prediction; as the dense acquisition also places anchors close to the target, RAA effectively recovers the lost detail, yielding a large gain of about 4 dB (shown in Tab. 1). On Dataset B, the sparse poses induce far less smoothing and the feed-forward prediction is already sharp, matching the other baselines; RAA still compensates for the residual interpolation-induced detail loss, contributing a smaller but still substantial gain of about 2 dB.

C Additional Ablation and Robustness Studies

C.1 Supporting Analysis of Decoupled Reflection and Attenuation Rendering

We compare UltraGauss with an optimized variant of our representation, UltraVoxelGS-opt, to examine the proposed rendering parameterization under per-scene optimization. UltraGauss uses a coupled ultrasound rendering model, while UltraVoxelGS-opt decouples reflection and cumulative attenuation and models anisotropic reflection with SH coefficients. Since the two methods differ in representation and initialization, this comparison is intended as supporting evidence rather than an isolated causal ablation.

Table 10: **Supporting analysis of Decoupled Reflection and Attenuation Rendering.** We compare UltraGauss with an optimized variant of our representation under per-scene optimization.

Rendering Variant	Decoupled (ρ, β)	SH Reflection	PSNR↑	SSIM↑
UltraGauss	✗	✗	22.06	0.6220
UltraVoxelGS-opt	✓	✓	**22.55**	**0.6245**

Tab. 10 shows that UltraVoxelGS-opt achieves comparable or slightly better reconstruction quality than UltraGauss, suggesting that the decoupled and direction-aware appearance parameterization remains competitive under optimization.

C.2 Overlap Consistency Ablation

Effect of overlap consistency. In local-to-global reconstruction, the same spatial support can be selected by multiple supervised views and thus receive repeated Gaussian predictions. We introduce overlap consistency to regularize these duplicated predictions before fusion. Tab. 11

Table 11: **Overlap consistency ablation.** We evaluate whether overlap consistency stabilizes repeated local predictions under the proposed local-to-global assembly. RAA is disabled in this ablation.

Variant	Consistency Loss	RAA	PSNR↑	SSIM↑
Block-wise only	✗	✗	19.85	0.5742
Block-wise + Consistency	✓	✗	**19.76**	**0.5770**

compares block-wise UltraVoxelGS with and without this loss, both without Render-time Appearance Adaptation. With overlap consistency, both PSNR and SSIM improve slightly, indicating that regularizing repeated local predictions makes the fused Gaussian representation more stable without degrading reconstruction fidelity. This supports its role in stabilizing global fusion across overlapping local predictions.

C.3 Voxel Support Resolution Analysis

To examine whether a sharper spatial support can reduce the effect of RAA, we increase the voxel resolution at inference time without changing the model or retraining it. As shown in Tab. 12, the denser support provides a modest improvement for UltraVoxelGS w/o RAA, but increases the Gaussian count from approximately 38.0M to 90.5M. Meanwhile, the complete UltraVoxelGS (w/ RAA) still achieves substantially better performance.

Table 12: Expressive support-density analysis.

Support resolution	Method	PSNR↑	SSIM↑	LPIPS↓
0.75 mm	UltraVoxelGS w/o RAA	19.76	0.5770	0.5130
0.75 mm	UltraVoxelGS	**24.29**	**0.8329**	**0.2384**
0.50 mm	UltraVoxelGS w/o RAA	20.14	0.5909	0.4719
0.50 mm	UltraVoxelGS	24.22	0.8200	**0.2131**

C.4 Comparison with Generic Feed-Forward GS

To evaluate the effectiveness of our ultrasound-specific feed-forward reconstruction paradigm, we introduce VolSplat[26], a generic feed-forward GS baseline, for comparison. However, directly applying VolSplat to long ultrasound sequences is computationally impractical because its native formulation jointly processes a limited number of input views before lifting their features into a 3D feature space and decoding the scene.

We therefore adapt VolSplat to our long-sequence setting while preserving its original training paradigm. During training, randomly sampled source views are used as input, their features are lifted into a 3D feature space, and the complete scene is decoded. During inference, we divide all source views into groups, process each group independently to obtain lifted 3D features, fuse the features from all groups into a common global feature space, and decode the scene representation once. This allows VolSplat to use the full source sequence.

The results on Dataset A are shown in Tab. 13. UltraVoxelGS w/o RAA substantially outperforms the adapted VolSplat baseline. This gap reflects the distinct reconstruction paradigms. VolSplat must process limited view subsets independently, losing the global scene context and making it difficult to aggregate information distributed across different subset views. In contrast, UltraVoxelGS first consolidates the complete sequence into a unified 3D support and then predicts Gaussians only from the view-interacted local regions. Owing to ultrasound thin-slab locality, each such local region contains the relevant context from the globally constructed support without requiring repeated whole-scene decoding. The result therefore supports the contribution of the ultrasound-specific voxel-first formulation and local-to-global reconstruction strategy.

C.5 Controlled pose-noise Analysis

The purpose of this experiment is to evaluate whether UltraVoxelGS remains stable when the input poses contain tracking errors. Since training data can be curated with offline pose refinement, we

Table 13: Feed-forward comparison on Dataset A.

Method	Regime	PSNR↑	SSIM↑	LPIPS↓
VolSplat	Feed-forward	15.70	0.4522	0.5884
UltraVoxelGS w/o RAA	FF + DRA	19.76	0.5770	0.5130
UltraVoxelGS	FF + DRA + RAA	24.29	0.8329	0.2384

focus on the clinically relevant scenario of inference-time tracking noise. Pose information is used in three stages of our framework: constructing the global representation (fusion), selecting the observed anchor (anchor), and querying the target view (query).

To isolate the sensitivity of each stage, we decompose the effect across four conditions: (A) clean setting; (B) noisy poses are used only for scene fusion; (C) noisy poses are used for both scene fusion and anchor selection; and (D) noisy poses are used for all three stages. Specifically, we inject zero-mean Gaussian noise independently for each frame and for each translational and rotational axis at two levels ($\sigma_t = 8\,\text{mm}/\sigma_r = 4^\circ$ and $\sigma_t = 12\,\text{mm}/\sigma_r = 6^\circ$), and report the mean and standard deviation over three independent random seeds. As shown in Tab. 14, fusion is essentially unaffected (B remains at the clean-pose level, within run-to-run variation), and the degradation is concentrated in anchor selection, while target querying adds little. Notably, even under such large perturbations (realized RMS $20.8\,\text{mm}/10.3^\circ$), PSNR drops by only $\sim$1.38 dB, demonstrating that UltraVoxelGS is robust to realistic tracking errors.

Table 14: Reproducibility of controlled pose experiment on dataset B.

σ_t (mm)	σ_r (°)	Condition	PSNR↑	SSIM↑	LPIPS↓
0	0	A	20.86	0.3158	0.6206
8	4	B	20.92 ± 0.01	0.3177 ± 0.0001	0.6199 ± 0.0001
		C	20.11 ± 0.08	0.3017 ± 0.0015	0.6336 ± 0.0013
		D	19.92 ± 0.08	0.2985 ± 0.0016	0.6384 ± 0.0043
12	6	B	20.87 ± 0.10	0.3165 ± 0.0027	0.6232 ± 0.0052
		C	19.62 ± 0.01	0.2920 ± 0.0007	0.6423 ± 0.0004
		D	19.48 ± 0.03	0.2900 ± 0.0002	0.6448 ± 0.0016

D Additional Reconstruction Results

D.1 Additional Qualitative Results

Fig. 11 and Fig. 12 provides additional qualitative comparisons on held-out slice reconstruction. Compared with interpolation-based and optimization-based baselines, UltraVoxelGS better preserves layered anatomical structures and local appearance details under sparse ultrasound observations.

D.2 Validation on Handheld Ultrasound

To directly evaluate the handheld setting in real-world, we test the trained model on tracked-handheld Dataset C—paired with Dataset A but differing in acquisition trajectory—without retraining and under the same protocol as Dataset A (Tab. 15). No baseline is optimized for this handheld setting, yet UltraVoxelGS remains close to its own robotic performance and still outperforms the per-scene UltraGauss, confirming that it generalizes to the more complex trajectories of real-world handheld scanning.

Table 15: Subject-level average on robotic and tracked-handheld test cases.

Setting	PSNR↑	SSIM↑	LPIPS↓
UltraGauss (Handheld acquisition)	20.05	0.5858	0.5580
UltraVoxelGS (Robotic acquisition)	24.29	0.8329	0.2384
UltraVoxelGS (Handheld acquisition)	23.96	0.7703	0.3356

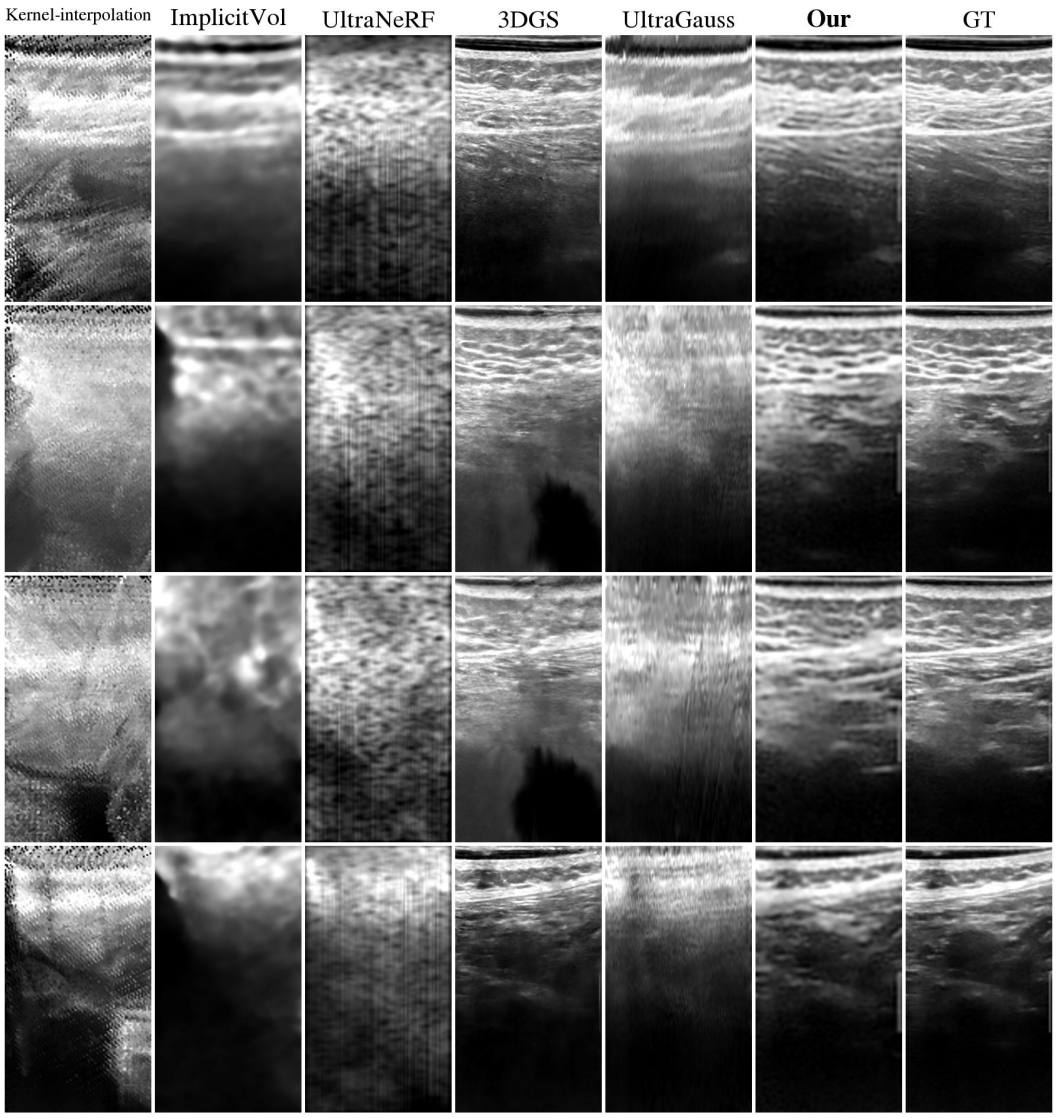


Figure 11: **Additional qualitative comparison on dataset A.** UltraVoxelGS better preserves anatomical structures, local sharpness, and cross-view consistency under sparse ultrasound observations.

E Streaming and Clinical Workflow

E.1 Streaming / periodic reconstruction

As a natural extension of UltraVoxelGS, we have implemented and evaluated Streaming/periodic reconstruction. For each voxel $\mathbf{x}$ we maintain two persistent fusion accumulators, the weighted numerator $A(\mathbf{x})$ and denominator $W(\mathbf{x})$:

$$A_t(\mathbf{x}) = \sum \omega_{i,p}(\mathbf{x})\, I_i(p), \qquad W_t(\mathbf{x}) = \sum \omega_{i,p}(\mathbf{x}), \tag{13}$$

where the summation aggregates the contributions of all frames processed so far. When a new chunk of frames arrives, they add their contributions $(\Delta A_t, \Delta W_t)$ only to the voxels they affect, and the

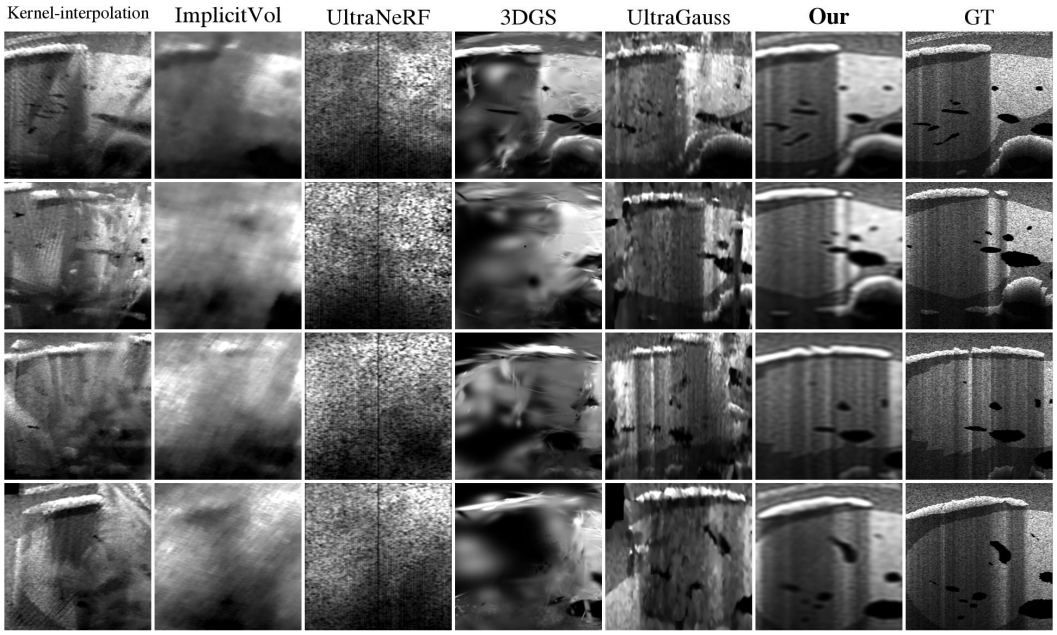


Figure 12: **Additional qualitative comparison on datasetB.** UltraVoxelGS better preserves anatomical structures, local sharpness, and cross-view consistency under sparse ultrasound observations.

voxel intensity is the updated ratio:

$$A_{t+1} = A_t + \Delta A_t, \qquad W_{t+1} = W_t + \Delta W_t, \qquad V_{t+1}(\mathbf{x}) = \frac{A_{t+1}(\mathbf{x})}{W_{t+1}(\mathbf{x}) + \epsilon}. \tag{14}$$

This incremental fusion follows directly from Eq. 6. For the Gaussian stage, only the blocks whose voxel support has been updated by the current chunk, $\mathcal{U}_t = \{\mathbf{x} \mid \Delta W_t(\mathbf{x}) > 0\}$, are re-processed by the voxel-to-Gaussian network, since these are the only blocks that have received new information; all remaining blocks keep their previous Gaussians unchanged:

$$\mathbf{G}_{t+1}^{b} = \begin{cases} f_{\mathrm{V2G}}\left(V_{t+1}^{b}\right), & b \in \mathcal{B}_t, \\ \mathbf{G}_{t}^{b}, & b \notin \mathcal{B}_t. \end{cases} \tag{15}$$

We evaluate the streaming mode in two ways. Compared with direct reconstruction, which builds the voxel support from the full sequence, generates the Gaussians block-wise, and evaluates all novel views at once, the two streaming settings proceed incrementally. In *progressive streaming*, every L incoming frames update the current voxel support (following the equations above), and only the blocks whose support has changed are re-processed by the feed-forward model to refresh the corresponding Gaussians. Right after each update, we evaluate the novel views whose target planes fall within the spatial extent of the newly updated blocks; each novel view is evaluated exactly once, immediately after the update that covers it, using only the information available at that point.

Final streaming shares this exact construction but evaluates all novel views once at the end, using the representation obtained after the full sequence is processed.

In streaming experiments we set the chunk size to $L = 10$ frames. As shown in Tab. 16, both settings are comparable to direct reconstruction, confirming that the online paradigm is practical without sacrificing fidelity. Fig. 13 further shows that both streaming modes preserve comparable anatomical structures and visual details. Final streaming is slightly better than progressive streaming, as it evaluates every view using the complete sequence, whereas progressive streaming can only use the frames seen up to each point. Direct reconstruction performs slightly better than final streaming because the feed-forward model is trained under the direct (full-volume) setting; the streaming pipeline **reuses the same model without any retraining or fine-tuning**, so the minor gap is expected and reflects only the mismatch between training and inference procedures rather than a fundamental limitation of the streaming paradigm.

Table 16: Performance of streaming reconstruction on dataset A.

Setting	PSNR↑	SSIM↑	LPIPS↓
Progressive streaming	24.08	0.8240	0.2555
Final streaming	24.11	0.8280	0.2476
Direct reconstruction	**24.29**	**0.8329**	**0.2384**

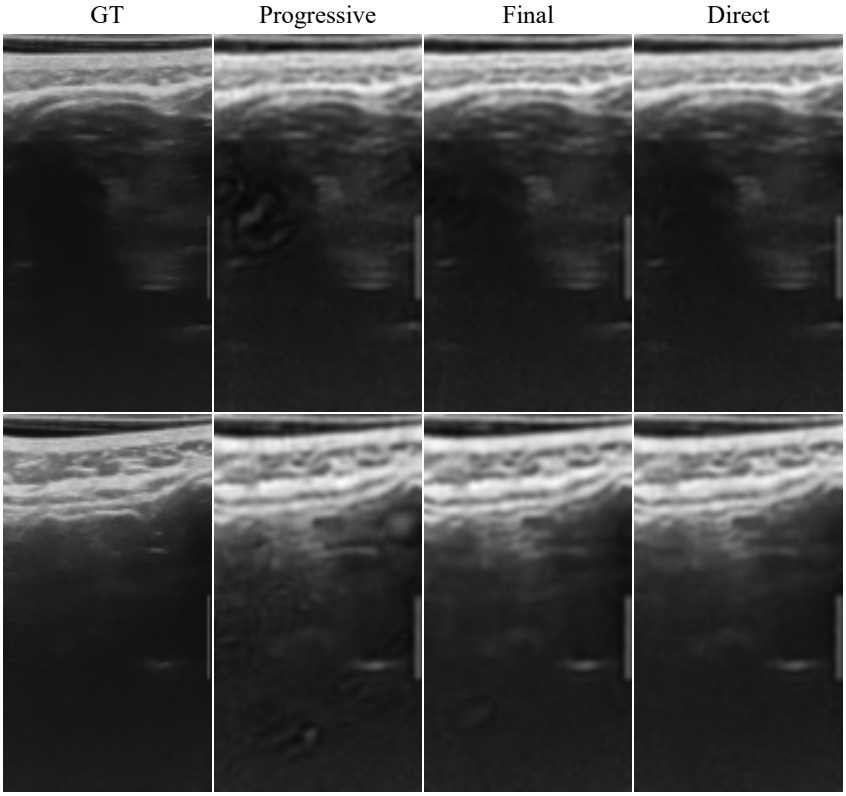


Figure 13: **Qualitative comparison of reconstruction modes on Dataset B.** Progressive and final streaming achieve visual quality comparable to direct reconstruction.

E.2 Comparison with Commercial Reconstruction Systems

The key difference lies in the underlying scene representation. UltraVoxelGS uses a direction-aware Gaussian representation, which allows it to model ultrasound-specific appearance factors such as anisotropic reflection and cumulative attenuation. This representation better accommodates flexible scanning trajectories, including local revisits and changes in scanning direction.

In contrast, many conventional or commercial reconstruction systems form an isotropic voxel volume by lifting registered 2D slices into 3D space and compounding their intensities, producing a tomographic-style representation similar to CT or MRI volumes. Such reconstruction commonly favors smooth, approximately uniform sweeps with limited dwelling or repeated sampling of the same region, because redundant observations are often averaged during voxel compounding, which may smooth local details. This can constrain the acquisition trajectory and, consequently, the applicable scanning scenarios.

This difference is particularly relevant near interfaces with large acoustic-impedance differences, where strong specular reflection or acoustic shadowing may hide structures from a single insonification direction. Additional observations from different probe directions can therefore be important.

UltraVoxelGS is designed to preserve and render such direction-dependent observations within a unified representation rather than collapsing them into a single isotropic intensity value.

E.3 Visual Feedback and Scan Guidance

Real-time scan guidance requires two capabilities: (i) fast 3D reconstruction and rendering that keeps pace with the ongoing scan, and (ii) a mechanism to identify under-reconstructed regions and translate them into actionable probe-pose suggestions. This work addresses capability (i). UltraVoxelGS enables a live reconstruction workflow: as the operator scans, the system simultaneously streams acquired frames into a voxel support and, at fixed intervals, generates local GS representations and fuses them into the global scene. This construction proceeds in the background, overlapping with the scan rather than waiting for it. When inspection is needed, the operator can render corresponding views in real time.

Regarding capability (ii), reconstruction incompleteness is already visually observable in the rendered views (see Fig. 11), but it has not yet been converted into explicit guidance. One natural extension is to associate uncertainty with the Gaussian representation and compute a pose-dependent cumulative uncertainty score for candidate poses, directing the operator toward under-observed regions. Beyond visual inspection, the reconstructed scene also supports downstream analysis such as 3D segmentation , which can further assist the operator in assessing anatomical coverage. These extensions are enabled by, and fundamentally depend on, the fast reconstruction and rendering prerequisite addressed in this work.

F Limitations and Broader Impact

Limitations. UltraVoxelGS relies on sufficiently accurate probe tracking to lift posed 2D ultrasound slices into a unified voxel support. The method may degrade when the observed coverage is extremely sparse, when severe acoustic shadowing removes local structural evidence, or when probe poses contain noticeable errors. In addition, our evaluation focuses on held-out slice reconstruction and representation quality, which does not directly establish clinical diagnostic performance. Bridging this gap would likely require explicit uncertainty estimation to flag unreliable regions and probe-pose guidance to direct operators toward uncovered anatomy, both of which we leave to future work.

UltraVoxelGS also adopts a simplified ultrasound-aware appearance rendering model. Specifically, Decoupled Reflection and Attenuation Rendering focuses on anisotropic reflection and cumulative attenuation, which capture the main direction- and path-dependent factors in B-mode ultrasound intensity formation: local reflection determines the generated echo response, while cumulative attenuation controls signal decay along the acoustic path. This formulation is compatible with Gaussian scene representation, where Gaussian appearance parameters model local reflection/scattering and accumulated transmittance models attenuation. Our results on real-world and simulated datasets suggest that this approximation is effective for the reconstruction tasks studied in this work, but it is not a complete acoustic simulation. Other effects, such as speed-of-sound variation, refraction, speckle decorrelation, beamforming artifacts, and device-specific post-processing, are not explicitly modeled and may become important under large view changes, strong specular interfaces, severe shadowing, or cross-device settings. Future work may incorporate richer acoustic effects while preserving the efficiency of the feed-forward voxel-to-Gaussian pass.

Broader impact. Fast 3D ultrasound reconstruction can support low-latency scan guidance, spatial feedback, and more efficient review of ultrasound acquisitions. By reducing reconstruction time and hardware requirements, feed-forward 3D ultrasound reconstruction may also help extend volumetric imaging capabilities to resource-limited clinical settings where dedicated 3D probes are unavailable. However, the current method should be viewed as a reconstruction and visualization tool rather than a standalone clinical decision system. Future work should further study uncertainty estimation, pose-noise robustness, broader freehand ultrasound validation, and prospective clinical evaluation.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the main contributions, including the voxel-first feed-forward Gaussian formulation, ultrasound-aware appearance rendering, Render-time Appearance Adaptation, and scalability to long ultrasound sequences. The claims are supported by the quantitative comparisons, ablation studies, runtime analysis, and the stated limitations.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper includes a dedicated limitations discussion in the appendix. It discusses the dependence on accurate probe tracking, degradation under sparse coverage or severe acoustic shadowing, the simplified acoustic rendering model, and the fact that held-out slice reconstruction does not directly establish clinical diagnostic performance.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: The paper does not present formal theoretical results, theorems, or proofs. The equations in the methodology define the reconstruction formulation, rendering model, and optimization objectives.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification:The paper describes the model formulation, rendering objective, training loss, datasets, evaluation splits, baseline settings, implementation details, and runtime protocol. Additional experimental details are provided in the appendix to support reproducibility of the reported comparisons.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: Code and data will be released upon the acceptance of the paper.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The main paper specifies the datasets and held-out evaluation protocols, and the appendix provides implementation details including image resolution, voxel spacing, patch size, top-k interpolation, SH order, optimizer, learning rate, training iterations, batch size, overlap consistency weight, and GPU setting.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: The paper reports average quantitative metrics over held-out target views and cases, but does not currently include error bars, confidence intervals, or statistical significance tests. The evaluation uses consistent splits and case-level aggregation, but uncertainty estimates are not reported.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The appendix reports the training GPU, training configuration, baseline optimization steps, runtime breakdown, per-view rendering cost with and without RAA, and peak memory for long-sequence scalability. These details describe the main compute resources required for reproducing the reported experiments.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The work uses previously collected/cited ultrasound datasets and simulated ultrasound data, does not introduce new human-subject data collection, and does not deploy the method as a clinical decision system. The submission is anonymized and the paper discusses limitations and broader impact.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.

- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The appendix discusses the potential positive impact of fast 3D ultrasound reconstruction for low-latency scan guidance, spatial feedback, and resource-limited settings. It also notes that the method should not be treated as a standalone clinical diagnostic system and requires further robustness, uncertainty, and clinical validation.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper poses no such risks

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: The paper credits the original datasets, simulation tools, and baseline methods through citations, but the current manuscript does not exhaustively list the license names and terms of use for every existing asset. These details will be added to the release documentation and should be included in the final version where applicable.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [N/A]

Justification: The submission does not introduce or release a new dataset at submission time. If code, trained models, or processed data splits are released after acceptance, they will be accompanied by documentation, usage instructions, licenses, and limitations.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: The paper does not conduct crowdsourcing experiments or new human-subject studies. It uses previously collected/cited ultrasound data and simulated data, so no participant instructions, screenshots, or compensation details are applicable.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.

- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The authors do not collect new human-subject data in this work. The real ultrasound data are previously collected/cited data, and the simulated dataset is generated from existing resources; therefore, new IRB approval for participant recruitment is not applicable to this submission.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs are not used as an important, original, or non-standard component of the proposed method, experiments, or scientific claims. Any use of LLMs, if any, is limited to writing, editing, or formatting assistance and does not affect the core methodology or results.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.