# A SIMPLE TRANSFORMER BACKEND FOR FEED-FORWARD 3D RECONSTRUCTION

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Recent feed-forward 3D reconstruction methods such as DUSt3R and VGGT predict pointmaps and camera poses directly from images, yet their internal representations remain image-aligned and lack explicit cross-view geometric reasoning. AMB3R closes this gap by appending a sparse-voxel 3D backend to a frozen frontend, attributing its accuracy gains to the spatial compactness of voxelization. We hypothesize that a much simpler ingredient suffices: aggregating multi-view geometric features and injecting the refined features back into the frontend. We accordingly replace the voxel backend with a deliberately minimal alternative: a single pass of global self-attention over per-pixel geometric tokens, injected back into the frozen decoder through zero-initialized adapters, with no voxel grid, no space-filling-curve serialization, and no $K$-nearest-neighbor interpolation. Despite its simplicity, the proposed backend is comparable to a voxel backend trained under matched conditions across zero-shot monocular depth, camera pose, multi-view depth, multi-view 3D reconstruction, and video depth. Controlled ablations on 3D positional encoding, the backend-to-frontend injection operator, and iterative refinement suggest that a single feed-forward pass with minimal design machinery is sufficient. Finally, on three additional frontends ($\pi^3$, Déjà View, and Depth Anything 3), our backend performs comparably to or better than a voxel backend trained on the same frontend, suggesting that the mechanism studied in this paper transfers beyond VGGT.

## 1 INTRODUCTION

Feed-forward 3D models such as DUSt3R (Wang et al., 2024), VGGT (Wang et al., 2025a) and $\pi^3$ (Wang et al., 2026b) replace explicit correspondence and optimization over 3D scene representations (Hartley & Zisserman, 2003; Schönberger & Frahm, 2016) with single-pass prediction of depth, pointmaps, and cameras, amortizing much of the classical reconstruction pipeline into a general-purpose visual geometry model. Despite this progress, a gap remains between geometric outputs and internal representations. These models predict explicit 3D quantities, yet operate primarily on image-aligned tokens. Such tokens can encode substantial multi-view structure (Bratulić et al., 2026), but only implicitly: corresponding pixels may predict compatible 3D coordinates without sharing a scene-level state for the same physical point.

AMB3R (Wang & Agapito, 2026) interprets this absence of explicit shared state as a lack of *spatial compactness*: observations mapping to the same 3D location should share a compact scene state rather than remain separate image tokens. It instantiates this idea by training a sparse-voxel backend on top of a frozen VGGT frontend, substantially improving the frontend with comparatively modest additional training. Predicted pointmaps and geometric features are pooled into voxels, serialized with space-filling curves, processed by a Point-Transformer-style network (Wu et al., 2024), interpolated back to pixels, and injected into the frozen decoder through zero-initialized adapters (Zhang et al., 2023b).

This makes AMB3R an important case study: its backend exchanges information across views, routes it through a shared voxelized scene representation, and injects the result back into a strong pretrained frontend. AMB3R attributes its gains to spatial compactness, but these operations are coupled, leaving the central question unresolved: are shared compact states really necessary, or is cross-view geometric feature exchange followed by frontend injection sufficient?



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

In this paper, we answer this question with a deliberately minimal alternative. Our backend removes sparse voxels, voxel pooling, space-filling-curve serialization, voxel-level processing, and K-nearest-neighbor interpolation. It simply flattens per-pixel geometric features from all views into one token sequence, applies a standard Transformer (Vaswani et al., 2017), and injects the refined tokens through the same frontend interface as AMB3R. This preserves cross-view feature exchange and decoder injection while removing compact voxel fusion, the ingredient AMB3R identifies as central.

This design makes the backend a controlled diagnostic. If spatial compactness drives AMB3R's gains, removing voxels should substantially degrade performance. Instead, our evaluation shows that the gain of our plain Transformer backend over the frozen frontend is comparable to that of a voxel backend trained under matched conditions across depth and 3D reconstruction tasks. Applied to three additional frontends, $\pi^3$ (Wang et al., 2026b), Déjà View (Burzio et al., 2026), and Depth Anything 3 (Lin et al., 2026), the same backend performs comparably to or better than a voxel backend trained on the same frontend.

Our ablations further narrow the explanation. Absolute positional embeddings and 3D RoPE have only marginal effects; AdaLN-Zero performs similarly to zero-convolution; and both stacked and recurrent refinement fail to improve over a single backend pass. Thus, the evidence does not support voxel compactness, explicit 3D positional encoding, a specific injection operator, or iterative refinement as the critical factor. Our goal is not to surpass AMB3R as a full system, but to explain its backend gains and recover them with the simplest architecture possible. In the AMB3R-style frozen-frontend adaptation regime, spatial compactness is not required for backend gains on the tasks evaluated in this paper. A plain Transformer over per-pixel geometric tokens provides a simpler, more transparent, and easier-to-implement substrate for improving pretrained feed-forward geometry models, requiring only access to geometric tokens and a lightweight Transformer-and-injection module rather than a custom 3D backend.

Our contributions are fourfold. (i) we isolate AMB3R's spatial-compactness hypothesis and test whether sparse voxel fusion is necessary for backend gains. (ii) we propose a voxel-free backend that replaces voxelization, voxel pooling, space-filling-curve processing, and KNN interpolation with global self-attention over per-pixel geometric tokens. (iii) we show that voxel compactness is not necessary for these gains, and our results are consistent with cross-view geometric feature exchange followed by frontend injection being sufficient in the evaluated regime. (iv) we validate the backend across depth, pose, reconstruction, video depth, and three additional frontends, with training completed within a few hours on 8 H100 GPUs.

## 2 Related Work

Feed-forward 3D reconstruction replaces classical correspondence search, camera estimation, and optimization over explicit scene representations (Hartley & Zisserman, 2003; Schönberger & Frahm, 2016) with direct prediction of geometry from images. DUSt3R (Wang et al., 2024) established dense pointmaps as a general two-view reconstruction representation, and VGGT (Wang et al., 2025a) scaled this paradigm to multi-view inputs by jointly predicting cameras, depth, pointmaps, and tracks. Subsequent work improves feed-forward geometry by changing the frontend architecture, output representation, or temporal structure, for example MASt3R (Leroy et al., 2024), Spann3R (Wang & Agapito, 2025), CUT3R (Wang et al., 2025b), $\pi^3$ (Wang et al., 2026b), MapAnything (Keetha et al., 2026), Depth Anything 3 (Lin et al., 2026), and Déjà View (Burzio et al., 2026); recent extensions also target dynamic scenes and video reconstruction. Other works improve scalability or deployment through memory, streaming, sparse attention, or test-time adaptation, including LONG3R (Chen et al., 2025), WinT3R (Li et al., 2026), Sparse VGGT (Wang et al., 2026a), EVICT3R (Mahdi et al., 2025), VGG-T$^3$ (Elflein et al., 2026), and TTT3R (Chen et al., 2026). These methods are important baselines and system extensions, but our work studies a narrower regime: improving a frozen pretrained frontend without redesigning or retraining it.

AMB3R (Wang & Agapito, 2026) is the closest work in this regime because it improves a frozen VGGT frontend through a trainable backend rather than a new frontend architecture. Its central claim is that backend gains come from enforcing *spatial compactness* with a sparse voxel scene representation. Our work uses AMB3R as a controlled comparison point, not as a template: we keep the same frozen-frontend adaptation interface but remove the voxel representation, testing whether the gains are better explained by cross-view feature exchange followed by decoder injection.

# 3 METHOD

## 3.1 PRELIMINARIES: AMB3R

We follow the frozen-frontend adaptation setting of AMB3R (Wang & Agapito, 2026), which augments a pretrained feed-forward 3D frontend $\mathcal{F}_{\text{front}}$, such as VGGT (Wang et al., 2025a), with trainable backend components. We focus on the backend pathway, which refines frozen-frontend predictions and injects the refined features back into the decoder. For simplicity, consider a frontend that outputs pointmaps. Given $T$ images $\{I_t\}_{t=1}^{T}$, the frontend jointly predicts pointmaps in a shared coordinate system and extracts image-aligned geometric features:

$$\big\{(P_t,\, G_t)\big\}_{t=1}^{T} \;=\; \mathcal{F}_{\text{front}}\big(\{I_t\}_{t=1}^{T}\big)\,, \tag{1}$$

where $P_t[\mathbf{u}] \in \mathbb{R}^3$ is the predicted 3D point at pixel $\mathbf{u}$ in frame $t$, and $G_t[\mathbf{u}]$ is the corresponding geometric feature.

AMB3R quantizes the predicted pointmaps into a sparse voxel grid, averages the assigned per-pixel features $G_t[\mathbf{u}]$ within each occupied voxel, and processes the occupied voxels as a space-filling-curve-serialized sequence with a Point-Transformer-v3-style network (Wu et al., 2024). Then it maps the refined voxel features back to pixels by $K$-nearest-neighbor interpolation, yielding $\tilde{G}_t[\mathbf{u}]$. These backend features are injected into the frozen frontend decoder through zero-initialized convolutional adapters (Zhang et al., 2023b). For a decoder activation $d_{t,\mathbf{u}}^{(\ell)}$ at layer $\ell$, the injection can be written as

$$d_{t,\mathbf{u}}^{(\ell)} \;\leftarrow\; d_{t,\mathbf{u}}^{(\ell)} \;+\; \text{ZeroConv}_{\ell}\big(\tilde{G}_t[\mathbf{u}]\big), \tag{2}$$

where $\text{ZeroConv}_{\ell}$ is a $1\times1$ two-layer convolution with a zero-initialized final layer. Therefore, at initialization, the injected residual vanishes and the model behaves as the pretrained frontend; training then learns residual geometric corrections through the backend and injection layers.

## 3.2 VOXEL-FREE TRANSFORMER BACKEND

Our goal is to retain only the mechanisms needed to improve a frozen feed-forward geometry model. AMB3R is a strong backend-refinement baseline, but its voxel backend couples cross-view feature exchange with voxel quantization, feature pooling, spatial serialization, local voxel attention, KNN interpolation, and decoder injection. We ask how much of this structure can be removed while preserving the gain, and propose a *voxel-free* backend that refines image-aligned geometric tokens directly.

Given frontend geometric features $\{G_t\}_{t=1}^{T}$, we flatten the image-aligned tokens. Let $\mathcal{U}_t$ be the token locations in frame $t$, with $|\mathcal{U}_t| = N$. We write $X = \{G_t[\mathbf{u}]\}_{t,\mathbf{u}} \in \mathbb{R}^{(TN)\times C}$, where $C$ is the feature dimension. Thus, each image-token feature remains independent and is never merged by voxel quantization. We process $X$ with $L$ pre-norm Transformer blocks (Xiong et al., 2020; Vaswani et al., 2017), producing refined tokens $\hat{X} \in \mathbb{R}^{(TN)\times C}$. The backend therefore performs a single operation: global self-attention over geometric tokens from all views. Finally, $\hat{X}$ is reshaped to the original feature-map layout, yielding refined per-pixel features $\tilde{G}_t[\mathbf{u}]$ that are passed to the injection module as in the voxel backend (Section 3.4). The architecture is thus a voxel-free backend built around our hypothesized minimal structure: global geometric feature exchange followed by injection into the pretrained decoder.

## 3.3 3D POSITIONAL ENCODING VARIANTS

Removing AMB3R's voxel aggregation also removes explicit grouping by predicted 3D location. We therefore test whether our voxel-free Transformer must reintroduce 3D position, or whether the frontend features $G_t[\mathbf{u}]$ are already geometry-aware. We compare three variants.

**Absolute 3D Positional Encoding.** We add an MLP embedding of each token's predicted pointmap coordinate to the Transformer input: $x'_{t,\mathbf{u}} = x_{t,\mathbf{u}} + \phi_{\text{pe}}(P_t[\mathbf{u}])$, and feed $X' = \{x'_{t,\mathbf{u}}\}_{t,\mathbf{u}}$ to the Transformer. This provides absolute 3D position but no explicit pairwise relation.

**Relative 3D Positional Encoding (3D RoPE).** We extend RoPE (Su et al., 2024) to predicted pointmap coordinates. The Transformer input remains $X$, while query and key channels in each head are split into $x, y, z$ blocks and rotated axis-wise. For the $x$ block,

$$(q^{(x)}_{2k-1}, q^{(x)}_{2k}) \leftarrow \mathbf{R}(x\theta_k)(q^{(x)}_{2k-1}, q^{(x)}_{2k}), \qquad \theta_k = b^{-2(k-1)/(d_h/3)},$$

with analogous rotations for $y$ and $z$. As in standard RoPE, attention scores then depend on relative coordinate differences, here in predicted 3D pointmap space.

**No Positional Encoding.** We also feed $X$ to the Transformer without any positional signal, testing whether the frontend features already contain the spatial information needed for cross-view exchange. We compare all variants in Section 4.3.

### 3.4 Backend-to-Frontend Injection Variants

After cross-view exchange, refined backend features must enter the frozen decoder without disrupting its pretrained behavior. We use AMB3R's zero-initialized residual injection as the default and test an AdaLN-Zero alternative to check whether gains depend on the injection operator.

**Zero-convolution.** Following AMB3R (Wang & Agapito, 2026), we inject $\tilde{G}_t[\mathbf{u}]$ into decoder layer $\ell$ with zero-initialized $1\times1$ two-layer convolutions (Zhang et al., 2023b):

$$d^{(\ell)}_{t,\mathbf{u}} \leftarrow d^{(\ell)}_{t,\mathbf{u}} + \mathrm{ZeroConv}_\ell\big(\tilde{G}_t[\mathbf{u}]\big). \tag{3}$$

Since the convolution is zero-initialized, the residual vanishes at initialization and the model starts from the pretrained frontend.

**AdaLN-Zero.** As a modulation-based alternative inspired by DiT (Peebles & Xie, 2023), the backend feature predicts per-channel scale, shift, and gate parameters:

$$d^{(\ell)}_{t,\mathbf{u}} \leftarrow d^{(\ell)}_{t,\mathbf{u}} + \gamma_\ell(\tilde{G}_t[\mathbf{u}]) \odot \Big(\big(1+\alpha_\ell(\tilde{G}_t[\mathbf{u}])\big) \odot \mathrm{LN}(d^{(\ell)}_{t,\mathbf{u}}) + \beta_\ell(\tilde{G}_t[\mathbf{u}])\Big). \tag{4}$$

Here $\alpha_\ell$, $\beta_\ell$, and $\gamma_\ell$ are predicted from a small $1\times1$ convolutional stack with a zero-initialized final layer (Section A), so the injected residual vanishes at initialization. Comparing these variants in Section 4.3 tests whether stable decoder injection, rather than its specific operator, is the key ingredient.

### 3.5 Recurrent and Stacked Backend Refinement

The backend interface can be applied repeatedly: after one refinement pass, the updated predictions and features can be fed back into the backend. Since AMB3R uses only one pass, we test whether additional passes justify their inference cost.

Let $\mathcal{B}_\theta$ denote one backend-plus-injection pass and $G^{(0)}$ the initial frontend features. Recurrent refinement reuses the same backend, $G^{(r+1)} = \mathcal{B}_\theta(G^{(r)})$, increasing runtime but not parameter count. Stacked refinement uses stage-specific backends, $G^{(r+1)} = \mathcal{B}_{\theta_r}(G^{(r)})$, increasing both runtime and model size. We compare both against the single-pass model to test the accuracy-efficiency trade-off of iterative backend refinement.

### 3.6 Training Objective

We train our model in the frozen-frontend adaptation regime. The frontend parameters are kept fixed, and only the backend together with the backend-to-frontend injection layers is updated. This matches the backend training setting of AMB3R (Wang & Agapito, 2026), while our architectural change is restricted to the backend representation and feature-exchange module.

We use the same reconstruction supervision as AMB3R for backend training, namely the pointmap, depth, and camera losses inherited from VGGT (Wang et al., 2025a):

$$\mathcal{L} \;=\; \mathcal{L}_{\mathrm{pointmap}} \;+\; \mathcal{L}_{\mathrm{depth}} \;+\; \mathcal{L}_{\mathrm{camera}}. \tag{5}$$

This ensures that differences between the AMB3R-style voxel backend and our backend primarily reflect the backend architecture rather than a change in supervision. For $\pi^3$, we use its native training loss (Section D), and for Déjà View and Depth Anything 3, the same loss as for VGGT.

# 4 EXPERIMENTS

Our experiments address three questions. First, can a voxel-free backend recover the gains of an AMB3R-style backend across standard feed-forward 3D reconstruction benchmarks? Second, how sensitive is this backend to design choices such as 3D positional encoding, backend-to-frontend injection, and repeated refinement? Third, does the same backend transfer beyond VGGT to other feed-forward geometry frontends? We therefore evaluate on reconstruction and depth tasks, ablate the main design axes of the backend, and test the same design on three additional frontends.

## 4.1 EVALUATION PROTOCOL

**Tasks and Datasets.** We follow AMB3R's reconstruction and depth benchmarks (Wang & Agapito, 2026), but restrict the scope to frozen-frontend backend adaptation. AMB3R is a full metric-scale system with a scale head and downstream VO/SLAM and SfM extensions; these components introduce additional scale-estimation, mapping, and system-level design choices beyond the backend architecture studied here. We therefore evaluate tasks that directly measure the refined frontend predictions: monocular depth on NYUv2 (Silberman et al., 2012), KITTI (Geiger et al., 2012), ETH3D (Schöps et al., 2017), ScanNet (Dai et al., 2017), and DIODE (Vasiljevic et al., 2019); camera pose on RealEstate10K (Zhou et al., 2018); multi-view depth under the RMVDB protocol (Schröppel et al., 2022) on KITTI, ETH3D, ScanNet, DTU (Aanæs et al., 2016), and Tanks & Temples (Knapitsch et al., 2017); multi-view 3D reconstruction on ETH3D, DTU, and 7-Scenes (Brachmann et al., 2021); and video depth on Sintel (Butler et al., 2012), Bonn (Palazzolo et al., 2019), and KITTI.

**Metrics.** We use the AMB3R evaluation metrics (Wang & Agapito, 2026) for the tasks considered here. For depth estimation, we report absolute relative error (rel) and inlier ratios: monocular depth uses $\delta_{1.25}$, multi-view depth uses the stricter $\delta_{1.03}$, and video depth reports both. Camera pose estimation is evaluated by AUC@30. For multi-view 3D reconstruction, we report rel, accuracy, and completeness, where accuracy is the average nearest-neighbor distance from predicted points to the ground-truth point cloud, and completeness is the reverse direction. For scale-ambiguous depth and 3D reconstruction tasks, predictions are median-scale aligned to the ground truth following AMB3R; monocular depth uses per-image least-squares alignment of scale and shift following the Marigold (Ke et al., 2024) protocol. All geometric distances are reported in centimeters.

**Default Architecture.** Unless otherwise stated, our **default** voxel-free backend is built on a frozen VGGT frontend (Wang et al., 2025a) and uses 3D RoPE positional encoding, zero-convolution backend-to-decoder injection, and a single backend-to-frontend refinement pass. We first evaluate this default architecture on reconstruction and depth tasks, then analyze its main design axes in Section 4.3 (positional encoding, injection mechanism, and recurrent or stacked refinement) and its transfer to other frontends in Section 4.4.

**Baseline Protocol.** For controlled comparison, we retrain the AMB3R voxel backend with the same data mixture and training setup as ours, denoted as AMB3R-reproduced, and also report the public pretrained checkpoint when available, denoted as AMB3R-public. Although AMB3R-reproduced underperforms AMB3R-public on several benchmarks, it differs from our method only in the backend architecture, making it the appropriate control for isolating voxel-free versus voxel-based refinement. AMB3R-reproduced uses the official model code and training scripts released by the original authors, while the data preprocessing, which is not publicly available, follows our own implementation. We therefore report the public checkpoint alongside it as a reference for the absolute performance of the released system. Training details are in Sections A.2 and A.3. Unless otherwise stated, baseline numbers other than our reruns of VGGT (Wang et al., 2025a), $\pi^3$ (Wang et al., 2026b) for multi-view 3D reconstruction, and AMB3R (Wang & Agapito, 2026), as well as all rows of Table 7, are taken from AMB3R paper. VGGT, $\pi^3$, Déjà View (Burzio et al., 2026), and Depth Anything 3 (Lin et al., 2026) remain frozen during evaluation.

## 4.2 RESULTS

**Monocular Depth Estimation.** Table 1 evaluates zero-shot monocular depth estimation. Ours improves over VGGT on all five datasets and remains competitive with both the original AMB3R

Table 1: **Monocular depth estimation (zero-shot).** AMB3R-public shows the original checkpoints while AMB3R-reproduced is our reproduction following the baseline protocol. Parentheses indicate trained using the same dataset. Datasets: NYUv2 (Silberman et al., 2012), KITTI (Geiger et al., 2012), ScanNet (Dai et al., 2017), DIODE (Vasiljevic et al., 2019), and ETH3D (Schöps et al., 2017).

| Method | NYUv2 | | KITTI | | ScanNet | | DIODE | | ETH3D | |
|---|---|---|---|---|---|---|---|---|---|---|
| | rel ↓ | $\delta_{1.25}$ ↑ | rel ↓ | $\delta_{1.25}$ ↑ | rel ↓ | $\delta_{1.25}$ ↑ | rel ↓ | $\delta_{1.25}$ ↑ | rel ↓ | $\delta_{1.25}$ ↑ |
| Omnidata (Eftekhar et al., 2021) | 7.4 | 94.5 | 14.9 | 83.5 | 7.5 | 93.6 | 33.9 | 74.2 | 16.6 | 77.8 |
| DepthAny V2 (Yang et al., 2024) | 4.3 | 97.9 | 8.0 | 94.4 | 4.2 | 97.9 | 21.6 | 75.2 | 5.7 | 98.3 |
| Marigold (Ke et al., 2024) | 5.5 | 96.4 | 9.9 | 91.6 | 6.4 | 95.1 | 30.8 | 77.3 | 6.5 | 96.0 |
| Diffusion-E2E (Garcia et al., 2025) | 5.2 | 96.6 | 9.6 | 91.9 | 5.8 | 96.2 | 30.2 | 77.9 | 6.2 | 95.9 |
| MoGe (Wang et al., 2025c) | 3.6 | 98.0 | 5.5 | 97.6 | 3.5 | 98.3 | 22.4 | 82.3 | 3.3 | 98.8 |
| VGGT (Wang et al., 2025a) | 3.6 | 97.9 | 10.8 | 88.5 | (2.7) | (98.8) | 27.1 | 79.0 | 3.8 | 97.9 |
| AMB3R-public (Wang & Agapito, 2026) | 3.0 | 98.9 | 7.2 | 95.3 | (2.7) | (98.9) | 24.8 | 80.7 | 3.2 | 98.8 |
| AMB3R-reproduced (Wang & Agapito, 2026) | 3.2 | 98.7 | 8.0 | 94.4 | (2.7) | (98.9) | 25.1 | 80.2 | 3.6 | 98.4 |
| **Ours (default)** | 3.2 | 98.7 | 8.4 | 93.7 | (2.7) | (98.9) | 24.9 | 80.5 | 3.6 | 98.3 |

Table 2: **Camera pose estimation.** AUC@30 over a random subset of 10 frames per scene in RealEstate10K (Zhou et al., 2018). Baselines: DUSt3R (Wang et al., 2024), MASt3R (Leroy et al., 2024), Fast3R (Yang et al., 2025), VGGT (Wang et al., 2025a), and AMB3R (Wang & Agapito, 2026).

| | DUSt3R | MASt3R | Fast3R | VGGT | AMB3R-public | AMB3R-reproduced | **Ours (default)** |
|---|---|---|---|---|---|---|---|
| Re10K | 67.7 | 76.4 | 72.7 | 81.7 | 86.2 | **86.0** | 85.6 |

Table 3: **Multi-view depth estimation** on RMVDB (Schröppel et al., 2022). Parentheses indicate trained using the same dataset.

| Approach | KITTI | | ScanNet | | DTU | | T&T | | ETH3D | | Average | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | rel ↓ | $\delta_{1.03}$ ↑ | rel ↓ | $\delta_{1.03}$ ↑ | rel ↓ | $\delta_{1.03}$ ↑ | rel ↓ | $\delta_{1.03}$ ↑ | rel ↓ | $\delta_{1.03}$ ↑ | rel ↓ | $\delta_{1.03}$ ↑ |
| **a) Classic approaches** | | | | | | | | | | | | |
| COLMAP (Schönberger & Frahm, 2016) | 12.0 | 58.2 | 14.6 | 34.2 | 0.7 | 96.5 | 2.7 | 95.0 | 16.4 | 55.1 | 9.3 | 67.8 |
| **b) Single-view depth** | | | | | | | | | | | | |
| UniDepthV2 (Piccinelli et al., 2026) | 4.0 | 55.3 | (2.1) | (82.6) | 3.2 | 72.3 | 3.6 | 68.4 | 3.7 | 66.2 | 3.3 | 68.9 |
| DepthAny V2 (Yang et al., 2024) | 6.6 | 38.6 | (4.0) | (58.6) | 2.6 | 74.7 | 4.5 | 57.5 | 4.7 | 56.5 | 4.8 | 54.1 |
| **c) Depth from frames and poses (w/ per-image range)** | | | | | | | | | | | | |
| Vis-MVSNet (Zhang et al., 2023a) | 9.5 | 55.4 | (8.9) | (33.5) | (1.8) | (87.4) | 4.1 | 87.2 | 10.8 | 43.3 | 7.0 | 61.4 |
| MVSFormer++ (Cao et al., 2024) | 4.4 | 65.7 | (7.9) | (39.4) | (0.9) | (95.3) | 3.2 | 88.1 | 7.8 | 50.4 | 4.8 | 67.8 |
| **d) Depth from frames and poses (w/o per-image range)** | | | | | | | | | | | | |
| Robust MVD (Schröppel et al., 2022) | 7.1 | 41.9 | 7.4 | 38.4 | 2.7 | 82.0 | 5.0 | 75.1 | 9.0 | 42.6 | 6.3 | 56.0 |
| MVSA (Izquierdo et al., 2025) | 3.2 | 68.8 | 3.7 | 62.9 | 1.3 | 95.0 | 2.1 | 90.5 | 3.2 | 68.0 | 2.7 | 77.0 |
| **e) Depth from frames (w/o poses)** | | | | | | | | | | | | |
| DeMoN (Ummenhofer et al., 2017) | 15.5 | 15.2 | 12.0 | 21.0 | 21.8 | 16.6 | 13.0 | 23.2 | 17.4 | 15.4 | 16.0 | 18.3 |
| DUSt3R (Wang et al., 2024) | 5.4 | 49.5 | (3.1) | (71.8) | 3.9 | 68.6 | 3.3 | 75.1 | 3.0 | 76.0 | 3.7 | 68.2 |
| Spann3R (Wang & Agapito, 2025) | 7.9 | 36.2 | (3.3) | (67.1) | 3.5 | 65.2 | 4.7 | 58.5 | 5.7 | 58.6 | 5.0 | 57.1 |
| Pow3R (Jang et al., 2025) | 5.7 | 45.7 | (3.2) | (68.8) | 3.0 | 74.3 | 3.3 | 76.6 | 3.0 | 74.7 | 3.6 | 68.0 |
| MUSt3R (Cabon et al., 2025) | 4.5 | 55.0 | (4.0) | (59.8) | 4.6 | 55.4 | (2.6) | (80.4) | 2.5 | 80.3 | 3.7 | 66.2 |
| VGGT (Wang et al., 2025a) | 4.5 | 59.5 | (2.3) | (80.9) | 0.9 | 95.6 | 2.4 | 84.1 | 1.8 | 85.9 | 2.4 | 81.2 |
| $\pi^3$ (Wang et al., 2026b) | 2.8 | 72.9 | (2.0) | (83.6) | 1.3 | 91.8 | 1.8 | 87.3 | 1.3 | 92.4 | 1.8 | 85.6 |
| MapAny (Keetha et al., 2026) | 4.0 | 59.4 | 4.0 | 60.5 | 3.9 | 63.7 | 3.3 | 73.0 | 2.8 | 73.2 | 3.6 | 66.0 |
| AMB3R-public (Wang & Agapito, 2026) | 2.8 | 74.1 | (1.9) | (85.6) | 0.9 | 95.1 | 1.7 | 90.2 | 1.4 | 90.8 | 1.7 | 87.2 |
| AMB3R-reproduced (Wang & Agapito, 2026) | 3.4 | 67.8 | (1.9) | (85.9) | 0.9 | 94.7 | 1.9 | 87.6 | 1.6 | 88.4 | 1.9 | 84.9 |
| **Ours (default)** | 3.4 | 66.6 | (1.9) | (85.8) | 0.8 | 95.1 | 2.0 | 86.6 | 1.6 | 88.9 | 1.9 | 84.6 |

checkpoint and our reproduced AMB3R baseline. In particular, our method matches the best score on ScanNet (Dai et al., 2017) and is within a small margin of reproduced AMB3R on the other datasets. These results indicate that removing voxel fusion does not compromise the monocular depth gains obtained by backend adaptation.

**Camera Pose Estimation.** Table 2 evaluates camera pose estimation on RealEstate10K (Zhou et al., 2018) using a random 10-frame subset of each scene, reporting AUC@30. Our model outperforms VGGT and performs comparably to the reproduced voxel backend (85.6 vs. 86.0).

**Multi-View Depth Estimation.** Table 3 evaluates multi-view depth estimation on the RMVDB benchmark (Schröppel et al., 2022). Under matched training conditions, our method performs on par

Table 4: **Multi-view 3D reconstruction** following AMB3R.

| | Method | DTU | | | 7-Scenes | | | ETH3D | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | rel↓ | Acc↓ | Cp↓ | rel↓ | Acc↓ | Cp↓ | rel↓ | Acc↓ | Cp↓ |
| Onl. | Spann3R (Wang & Agapito, 2025) | 4.96 | 1.89 | 0.64 | 8.58 | 4.81 | 5.48 | 24.96 | 47.63 | 31.87 |
| | MUSt3R (Cabon et al., 2025) | 5.26 | 1.68 | 1.09 | 11.57 | 8.94 | 14.97 | 19.91 | 45.19 | 82.72 |
| | CUT3R (Wang et al., 2025b) | 9.11 | 3.75 | 0.96 | 6.32 | 2.88 | 3.26 | 18.83 | 38.90 | 22.93 |
| Offl. | **VGGT (Wang et al., 2025a)** | 0.84 | 0.24 | 0.17 | 5.42 | 3.44 | 3.70 | 6.04 | 13.68 | 13.02 |
| | $\pi^3$ (Wang et al., 2026b) | 1.52 | 0.50 | 0.34 | 5.90 | 3.46 | 4.15 | 5.84 | 11.40 | 11.73 |
| | MapAny (Keetha et al., 2026) | 10.37 | 4.35 | 1.54 | 7.61 | 3.48 | 3.33 | 11.20 | 21.87 | 19.76 |
| | AMB3R-public (Wang & Agapito, 2026) | 0.81 | 0.25 | 0.17 | 4.52 | 2.54 | 2.95 | 4.63 | 10.88 | 10.80 |
| | AMB3R-reproduced (Wang & Agapito, 2026) | 0.84 | 0.26 | 0.18 | 5.55 | 3.06 | 3.75 | 7.39 | 16.71 | 16.70 |
| | **Ours (default)** | 0.79 | 0.24 | 0.16 | 4.48 | 2.36 | 2.91 | 6.54 | 15.66 | 15.75 |

Table 5: **Video depth estimation.** We run zero-shot inference over every scene of each split. We did not finetune on dynamic-scene data, unlike $\pi^3$.

| Method | Sintel | | | Bonn | | | KITTI | | |
|---|---|---|---|---|---|---|---|---|---|
| | rel↓ | $\delta_{1.03}$↑ | $\delta_{1.25}$↑ | rel↓ | $\delta_{1.03}$↑ | $\delta_{1.25}$↑ | rel↓ | $\delta_{1.03}$↑ | $\delta_{1.25}$↑ |
| Spann3R (Wang & Agapito, 2025) | 55.2 | 7.3 | 46.3 | 5.5 | 40.9 | 97.8 | 24.7 | 10.4 | 64.4 |
| CUT3R (Wang et al., 2025b) | 58.4 | 9.6 | 46.6 | 5.1 | 47.4 | 98.0 | 13.6 | 18.0 | 81.0 |
| VGGT (Wang et al., 2025a) | 28.3 | 13.5 | 61.8 | 3.6 | 63.1 | 98.4 | 8.2 | 40.4 | 90.7 |
| $\pi^3$ (Wang et al., 2026b) | 31.9 | 13.3 | 66.8 | 3.0 | 72.0 | 98.8 | 3.6 | 59.5 | 98.8 |
| AMB3R-public (Wang & Agapito, 2026) | 23.5 | 14.2 | 62.8 | 3.3 | 66.0 | 98.5 | 4.1 | 53.9 | 98.3 |
| AMB3R-reproduced (Wang & Agapito, 2026) | 23.1 | 16.5 | 66.8 | 3.2 | 68.8 | 98.5 | 5.4 | 48.1 | 96.9 |
| **Ours (default)** | 24.8 | 14.5 | 63.5 | 3.3 | 65.6 | 98.7 | 5.7 | 44.6 | 96.7 |

with our reproduction of AMB3R (Wang & Agapito, 2026). Both methods clearly improve over the VGGT (Wang et al., 2025a) baseline.

**Multi-View 3D Reconstruction.** Table 4 evaluates multi-view 3D reconstruction on three datasets. Our method outperforms the reproduced AMB3R voxel backend on all three datasets, and on DTU and 7-Scenes it also slightly improves over the public AMB3R checkpoint (0.79 vs. 0.81 and 4.48 vs. 4.52 in rel). On ETH3D, both our reproduced AMB3R baseline and our method degrade relative to VGGT and the official AMB3R checkpoint, likely due to non-architectural reproduction factors such as preprocessing or sampling ratios. We therefore treat ETH3D as a reproduction caveat (Section C). These results support the claim that compact voxel fusion is not required for backend gains. Qualitative comparisons of per-pixel pointmap errors are provided in Section G.

**Video Depth Estimation.** Table 5 evaluates zero-shot per-frame depth on video benchmarks. Ours improves over VGGT on all metrics across Sintel (Butler et al., 2012), Bonn (Palazzolo et al., 2019), and KITTI (Geiger et al., 2012), and remains close to our reproduced AMB3R baseline. Relative to the original AMB3R checkpoint, the gains are mixed: our method is competitive on Sintel and Bonn, while the public checkpoint remains stronger on KITTI. Notably, despite not being trained on dynamic-scene data, Ours outperforms $\pi^3$ (Wang et al., 2026b) on Sintel in rel and $\delta_{1.03}$, while $\pi^3$ remains stronger on Bonn and KITTI.

### 4.3 Backend Design Analysis

We next analyze the design sensitivity of the voxel-free backend. Because the backend is a plain Transformer inserted into a fixed frozen-frontend pipeline, we can vary its main design axes while keeping the frontend, supervision, and evaluation protocol unchanged. We study 3D positional encoding, backend-to-frontend injection, and repeated backend refinement on multi-view 3D reconstruction, where changes in the backend directly affect the predicted geometry.

**3D Positional Encoding.** Table 6a compares the three positional encoding variants introduced in Section 3.3: no positional encoding, absolute 3D positional encoding from predicted pointmaps, and 3D RoPE. All three variants perform within a narrow range on DTU (Aanæs et al., 2016) and 7-Scenes (Brachmann et al., 2021). In particular, removing explicit 3D positional encoding does not noticeably degrade performance, and is even slightly better on some 7-Scenes metrics. This suggests that the geometric features produced by the frontend already contain sufficient spatial information for the backend, and that explicitly reintroducing pointmap coordinates is not the dominant source of

Table 6: **Ablations on our backend.** We evaluate multi-view reconstruction on DTU (Aanæs et al., 2016) and 7-Scenes (Brachmann et al., 2021). **(a)** Positional encoding methods. **(b)** Backend-to-frontend injection methods. **(c)** The middle group stacks newly-trained backend stages on top of the stage-1 model (doubling/tripling parameters); the bottom group reuses the stage-1 checkpoint and feeds it its own previous-pass features for $N{=}2{-}3$ rounds (no new parameters). All variants, including the default, are trained with the same shorter 6-epoch schedule for a controlled comparison, so the default row differs from Table 4, which uses the full 30-epoch schedule.

(a) **Positional encoding**.

| PE | DTU | | | 7-Scenes | | |
|---|---|---|---|---|---|---|
| | rel↓ | Acc↓ | Cp↓ | rel↓ | Acc↓ | Cp↓ |
| No PE | 0.79 | **0.23** | **0.16** | **4.33** | 2.34 | **2.64** |
| Absolute PE | **0.78** | **0.23** | **0.16** | 4.39 | **2.33** | 2.74 |
| 3D RoPE (default) | **0.78** | **0.23** | **0.16** | 4.43 | 2.39 | 2.74 |

(b) **Backend-to-frontend injection**.

| Injection | DTU | | | 7-Scenes | | |
|---|---|---|---|---|---|---|
| | rel↓ | Acc↓ | Cp↓ | rel↓ | Acc↓ | Cp↓ |
| ZeroConv (default) | **0.78** | **0.23** | **0.16** | 4.43 | **2.39** | **2.74** |
| AdaLN-Zero | 0.79 | 0.24 | **0.16** | **4.27** | **2.39** | 2.90 |

(c) **Refinement: stacked vs. recurrent inference**.

| Refinement | DTU | | | 7-Scenes | | |
|---|---|---|---|---|---|---|
| | rel↓ | Acc↓ | Cp↓ | rel↓ | Acc↓ | Cp↓ |
| Single backend (default) | | | | | | |
| 1 stage / $\times 1$ | **0.78** | **0.23** | **0.16** | **4.43** | **2.39** | **2.74** |
| Stacked stages (extra params) | | | | | | |
| 2 stages | 0.80 | **0.23** | **0.16** | 4.62 | 2.53 | 2.96 |
| 3 stages | 0.81 | 0.24 | 0.17 | 4.56 | 2.50 | 2.94 |
| Recurrent inference (no extra params) | | | | | | |
| $\times 2$ iters | 0.79 | **0.23** | **0.16** | 6.68 | 3.84 | 4.65 |
| $\times 3$ iters | 0.80 | **0.23** | **0.16** | 9.00 | 5.48 | 6.65 |

Table 7: **Multi-view 3D reconstruction across frontends** comparing frozen frontends ($\pi^3$, Déjà View, and Depth Anything 3) against the same frontends wrapped with the AMB3R voxel backend or our backend. Bold marks the best result within each frontend. DVLT denotes Déjà View.

| | Method | DTU | | | 7-Scenes | | | ETH3D | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | rel↓ | Acc↓ | Cp↓ | rel↓ | Acc↓ | Cp↓ | rel↓ | Acc↓ | Cp↓ |
| $\pi^3$ | $\pi^3$ (Wang et al., 2026b) | 1.52 | 0.50 | 0.34 | 5.90 | 3.46 | 4.15 | 5.84 | 11.40 | 11.73 |
| | $\pi^3$ + AMB3R (Wang & Agapito, 2026) | 2.23 | 0.74 | 0.45 | 5.53 | 3.17 | 3.65 | 8.36 | 18.06 | 16.22 |
| | $\pi^3$ + **Ours (default)** | **1.48** | **0.47** | **0.33** | **5.33** | **3.04** | **3.61** | **4.89** | **11.19** | **11.47** |
| DVLT | Déjà View (Burzio et al., 2026) | **2.54** | **0.87** | 0.47 | 5.17 | 1.87 | 3.02 | 4.96 | 12.13 | 12.29 |
| | Déjà View + AMB3R | 2.70 | 0.99 | 0.43 | **4.95** | **1.80** | **2.45** | 4.96 | 11.37 | **10.74** |
| | Déjà View + **Ours (default)** | 2.55 | 0.91 | **0.42** | **4.95** | **1.80** | 2.53 | **4.92** | **11.34** | 11.12 |
| DA3 | Depth Anything 3 (Lin et al., 2026) | **1.91** | **0.61** | 0.26 | 5.43 | 2.12 | 3.01 | 12.53 | 26.98 | 17.13 |
| | Depth Anything 3 + AMB3R | 2.00 | 0.67 | **0.24** | 5.21 | 1.89 | 2.94 | **9.30** | 20.77 | 15.88 |
| | Depth Anything 3 + **Ours (default)** | 2.01 | 0.67 | **0.24** | **5.19** | **1.88** | **2.86** | 9.31 | **20.63** | **15.82** |

improvement.

**Backend-to-Frontend Injection.** Table 6b compares zero-convolution, our default injection module following AMB3R (Wang & Agapito, 2026), with the AdaLN-Zero variant described in Section 3.4. The two interfaces give similar results, with no consistent advantage across all metrics. This indicates that the gains are not tied to a particular injection operator. In both cases, the refined cross-view features enter the pretrained decoder through a stable zero-initialized interface.

**Repeated Backend Refinement.** Table 6c evaluates the recurrent and stacked refinement strategies from Section 3.5. Stacked stages add parameters without improving over a single pass, while recurrent reuse degrades substantially on 7-Scenes. This suggests that most useful geometric exchange occurs in the first global-attention pass, and further iterations mainly add cost or amplify errors.

### 4.4 Transfer to Other Frontends

We test whether the proposed backend is specific to VGGT by applying it to three frontends with different architectures, $\pi^3$ (Wang et al., 2026b), Déjà View (Burzio et al., 2026), and Depth Anything 3 (Lin et al., 2026). For a controlled comparison, we train both our voxel-free backend and an AMB3R-style voxel backend on the same frozen frontend. Table 7 shows that on $\pi^3$, the AMB3R-

style voxel backend is not a reliable drop-in replacement: it degrades $\pi^3$ on ETH3D and DTU, and improves it only on 7-Scenes. This suggests that hard voxel fusion can be sensitive to frontend. By contrast, our voxel-free backend improves over $\pi^3$ alone on all three datasets and outperforms the AMB3R-style voxel backend on all reported metrics. On Déjà View (Burzio et al., 2026) and Depth Anything 3 (Lin et al., 2026), our backend improves the frozen frontend on ETH3D and 7-Scenes and performs comparably to the voxel backend. On DTU, both backends slightly increase rel and accuracy error on these two frontends while improving completeness, and remain comparable to each other. This supports the mechanism studied in this paper: cross-view geometric feature exchange followed by decoder injection transfers beyond VGGT, while voxel aggregation is not required.

## 5 Discussion

The experiments support a mechanism-level interpretation of backend-based refinement. In the controlled AMB3R-style setting, replacing sparse voxel fusion with a simple Transformer over per-pixel geometric tokens preserves much of the benefit of backend adaptation: it is competitive on monocular depth, pose, multi-view depth, and video depth, improves over the reproduced voxel-backend baseline on all three reconstruction benchmarks, and performs comparably to or better than the voxel backend on three additional frontends. This does not claim uniform superiority over the official AMB3R checkpoint, especially given the ETH3D reproduction caveat. Rather, the controlled comparisons show that spatially compact voxel fusion is not necessary for backend gains on the tasks studied here.

The ablations further narrow the explanation. Pointmap-based positional encodings, including absolute 3D embeddings and 3D RoPE, have only marginal effects; zero-convolution and AdaLN-Zero perform similarly; and recurrent or stacked backend refinement adds cost without improving over a single pass. Thus, the gains are not explained by explicit 3D positional signals, a particular injection operator, or iterative refinement. Instead, the results are consistent with the view that a single global exchange of geometric features across views, followed by stable injection into the pretrained decoder, is sufficient.

This conclusion defines the scope of the work. We do not argue that compact 3D structures are unnecessary in general. Sparse voxels, point clouds, hash grids, or other spatial representations may be preferable for long sequences, unbounded scenes, memory-limited inference, or system-level reconstruction. Our claim is narrower: in the frozen-frontend backend-adaptation regime, the core benefit of the backend can be obtained without voxelization, space-filling-curve serialization, voxel pooling, or KNN interpolation. The proposed backend is therefore a minimal, transparent, and easy-to-implement substrate for improving pretrained feed-forward geometry models after pretraining.

## 6 Conclusion

We presented a voxel-free backend for frozen-frontend feed-forward 3D reconstruction. Our goal is not to claim a new state-of-the-art system, but to provide a simple, easy-to-implement alternative that isolates what is necessary for backend-based refinement. By replacing AMB3R's voxelization, space-filling-curve processing, and $K$-nearest-neighbor interpolation with global self-attention over per-pixel geometric tokens, we show that much of the backend benefit can be obtained without compact voxel fusion. The results and ablations are consistent with cross-view feature exchange followed by stable injection into the pretrained decoder being sufficient in the evaluated regime.

**Limitations.** This work studies a narrow controlled setting. Our primary comparison is against a reproduced AMB3R-style voxel backend trained with the same data and setup, not against the official AMB3R checkpoint as a full system. On ETH3D (Schöps et al., 2017), a gap to the public AMB3R checkpoint remains for both reproduced backends, although our backend still outperforms the voxel backend trained under the same conditions. We attribute this gap to non-architectural reproduction factors such as preprocessing or sampling ratios. We provide a more detailed discussion in Section C. We also do not evaluate AMB3R's metric-scale head, VO/SLAM, SfM, or long-sequence settings. Compact 3D representations may still be preferable for scalability, persistent memory, or system-level reconstruction.

### AI USE STATEMENT

In this work, we used generative AI tools to propose and refine hypotheses, to design and provide feedback on the research methodology and experiments, to implement methods, to assist with translation, to clean and reformat datasets, to support qualitative and thematic analysis of experimental data, and to interpret results. We have not used generative AI tools to generate synthetic data sets or to develop theoretical models or conceptual frameworks, and formulating or proving mathematical claims is not applicable to this work. Additionally, we used generative AI tools to create and modify figures, suggest experimental parameters, create and edit software code, draft and edit parts of the paper for readability, summarize and analyze existing literature, identify research topics and gaps, brainstorm, search for information, identify relevant literature, format references, and suggest the structure, title, and keywords of the paper. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI. All AI-assisted code was manually reviewed by the authors.

### REPRODUCIBILITY STATEMENT

We describe the backend architecture, positional encodings, and backend-to-frontend injection in Section 3 and Section A, and the transfer to other frontends, including the injection sites and training losses for each frontend, in Section D. The training data mixture and training configuration are given in Sections A.2 and A.3, and the evaluation datasets, metrics, and protocols in Section 4.1 and Section B. The AMB3R baseline is trained with the official model code and training scripts released by the original authors.

## REFERENCES

Henrik Aanæs, Rasmus Ramsbøl Jensen, George Vogiatzis, Engin Tola, and Anders Bjorholm Dahl. Large-scale data for multiple-view stereopsis. *International Journal of Computer Vision*, 120(2): 153–168, 2016.

Eduardo Arnold, Jamie Wynn, Sara Vicente, Guillermo Garcia-Hernando, Áron Monszpart, Victor Adrian Prisacariu, Daniyar Turmukhambetov, and Eric Brachmann. Map-free visual relocalization: Metric pose relative to a single image. In *Proceedings of the European Conference on Computer Vision*, pp. 690–708, 2022.

Eric Brachmann, Martin Humenberger, Carsten Rother, and Torsten Sattler. On the limits of pseudo ground truth in visual camera re-localisation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6218–6228, 2021.

Jelena Bratulić, Sudhanshu Mittal, Thomas Brox, and Christian Rupprecht. On geometric understanding and learned priors in feed-forward 3D reconstruction models. *arXiv preprint arXiv:2512.11508*, 2026.

Alessandro Burzio, Tobias Fischer, Sven Elflein, Qunjie Zhou, Riccardo de Lutio, Jiawei Ren, Jiahui Huang, Shengyu Huang, Marc Pollefeys, Laura Leal-Taixé, Zan Gojcic, and Haithem Turki. Déjà View: Looping transformers for multi-view 3D reconstruction. *arXiv preprint arXiv:2605.30215*, 2026.

Daniel J. Butler, Jonas Wulff, Garrett B. Stanley, and Michael J. Black. A naturalistic open source movie for optical flow evaluation. In *Proceedings of the European Conference on Computer Vision*, pp. 611–625, 2012.

Yohann Cabon, Naila Murray, and Martin Humenberger. Virtual KITTI 2. *arXiv preprint arXiv:2001.10773*, 2020.

Yohann Cabon, Lucas Stoffl, Leonid Antsfeld, Gabriela Csurka, Boris Chidlovskii, Jerome Revaud, and Vincent Leroy. MUSt3R: Multi-view network for stereo 3D reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1050–1060, 2025.

Chenjie Cao, Xinlin Ren, and Yanwei Fu. MVSFormer++: Revealing the devil in transformer's details for multi-view stereo. In *International Conference on Learning Representations*, 2024.

Xingyu Chen, Yue Chen, Yuliang Xiu, Andreas Geiger, and Anpei Chen. TTT3R: 3D reconstruction as test-time training. In *International Conference on Learning Representations*, 2026.

Zhuoguang Chen, Minghui Qin, Tianyuan Yuan, Zhe Liu, and Hang Zhao. LONG3R: Long sequence streaming 3D reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5273–5284, 2025.

Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. ScanNet: Richly-annotated 3D reconstructions of indoor scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5828–5839, 2017.

Ainaz Eftekhar, Alexander Sax, Jitendra Malik, and Amir Zamir. Omnidata: A scalable pipeline for making multi-task mid-level vision datasets from 3D scans. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10786–10796, 2021.

Sven Elflein, Ruilong Li, Sérgio Agostinho, Zan Gojcic, Laura Leal-Taixé, Qunjie Zhou, and Aljosa Osep. VGG-T$^3$: Offline feed-forward 3D reconstruction at scale. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 36464–36474, 2026.

Gonzalo Martin Garcia, Karim Abou Zeid, Christian Schmidt, Daan De Geus, Alexander Hermans, and Bastian Leibe. Fine-tuning image-conditional diffusion models is easier than you think. In *IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 753–762. IEEE, 2025.

Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the KITTI vision benchmark suite. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3354–3361, 2012.

Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2 edition, 2003.

Po-Han Huang, Kevin Matzen, Johannes Kopf, Narendra Ahuja, and Jia-Bin Huang. DeepMVS: Learning multi-view stereopsis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2821–2830, 2018.

Sergio Izquierdo, Mohamed Sayed, Michael Firman, Guillermo Garcia-Hernando, Daniyar Turmukhambetov, Javier Civera, Oisin Mac Aodha, Gabriel Brostow, and Jamie Watson. MVSAnywhere: Zero-shot multi-view stereo. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11493–11504, 2025.

Wonbong Jang, Philippe Weinzaepfel, Vincent Leroy, Lourdes Agapito, and Jerome Revaud. Pow3R: Empowering unconstrained 3D reconstruction with camera and scene priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1071–1081, 2025.

Bingxin Ke, Anton Obukhov, Shengyu Huang, Nando Metzger, Rodrigo Caye Daudt, and Konrad Schindler. Repurposing diffusion-based image generators for monocular depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9492–9502, 2024.

Nikhil Keetha, Norman Müller, Johannes Schönberger, Lorenzo Porzi, Yuchen Zhang, Tobias Fischer, Arno Knapitsch, Duncan Zauss, Ethan Weber, Nelson Antunes, Jonathon Luiten, Manuel Lopez-Antequera, Samuel Rota Bulò, Christian Richardt, Deva Ramanan, Sebastian Scherer, and Peter Kontschieder. MapAnything: Universal feed-forward metric 3D reconstruction. In *International Conference on 3D Vision*, pp. 499–509. IEEE, 2026.

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

Vincent Leroy, Yohann Cabon, and Jérôme Revaud. Grounding image matching in 3D with MASt3R. In *Proceedings of the European Conference on Computer Vision*, pp. 71–91, 2024.

Zizun Li, Jianjun Zhou, Yifan Wang, Haoyu Guo, Wenzheng Chang, Yang Zhou, Haoyi Zhu, Junyi Chen, Chunhua Shen, and Tong He. WinT3R: Window-based streaming reconstruction with camera token pool. In *International Conference on Learning Representations*, 2026.

Haotong Lin, Sili Chen, Jun Hao Liew, Donny Y. Chen, Zhenyu Li, Yang Zhao, Sida Peng, Hengkai Guo, Xiaowei Zhou, Guang Shi, Jiashi Feng, and Bingyi Kang. Depth Anything 3: Recovering the visual space from any views. In *International Conference on Learning Representations*, 2026.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

Soroush Mahdi, Fardin Ayar, Ehsan Javanmardi, Manabu Tsukada, and Mahdi Javanmardi. EVICT3R: Training-free token eviction for memory-bounded streaming visual geometry transformers. *arXiv preprint arXiv:2509.17650*, 2025.

Emanuele Palazzolo, Jens Behley, Philipp Lottes, Philippe Giguere, and Cyrill Stachniss. ReFusion: 3D reconstruction in dynamic environments for RGB-D cameras exploiting residuals. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 7855–7862, 2019.

Xiaqing Pan, Nicholas Charron, Yongqian Yang, Scott Peters, Thomas Whelan, Chen Kong, Omkar Parkhi, Richard Newcombe, and Yuheng Carl Ren. Aria Digital Twin: A new benchmark dataset for egocentric 3d machine perception. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 20133–20143, 2023.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4195–4205, 2023.

Luigi Piccinelli, Christos Sakaridis, Yung-Hsu Yang, Mattia Segu, Siyuan Li, Wim Abbeloos, and Luc Van Gool. UniDepthV2: Universal monocular metric depth estimation made simpler. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 48(3):2354–2367, 2026.

Jeremy Reizenstein, Roman Shapovalov, Philipp Henzler, Luca Sbordone, Patrick Labatut, and David Novotny. Common Objects in 3D: Large-scale learning and evaluation of real-life 3D category reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10901–10911, 2021.

Mike Roberts, Jason Ramapuram, Anurag Ranjan, Atulit Kumar, Miguel Angel Bautista, Nathan Paczan, Russ Webb, and Joshua M Susskind. Hypersim: A photorealistic synthetic dataset for holistic indoor scene understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10912–10922, 2021.

Johannes L. Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4104–4113, 2016.

Thomas Schöps, Johannes L. Schönberger, Silvano Galliani, Torsten Sattler, Konrad Schindler, Marc Pollefeys, and Andreas Geiger. A multi-view stereo benchmark with high-resolution images and multi-camera videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3260–3269, 2017.

Philipp Schröppel, Jan Bechtold, Artemij Amiranashvili, and Thomas Brox. A benchmark and a baseline for robust multi-view depth estimation. In *International Conference on 3D Vision*, pp. 637–645. IEEE, 2022.

Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from RGB-D images. In *Proceedings of the European Conference on Computer Vision*, pp. 746–760, 2012.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. RoFormer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Yu Zhang, Jonathon Shlens, Zhifeng Chen, and Dragomir Anguelov. Scalability in perception for autonomous driving: Waymo Open Dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2446–2454, 2020.

Benjamin Ummenhofer, Huizhong Zhou, Jonas Uhrig, Nikolaus Mayer, Eddy Ilg, Alexey Dosovitskiy, and Thomas Brox. DeMoN: Depth and motion network for learning monocular stereo. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5038–5047, 2017.

Igor Vasiljevic, Nick Kolkin, Shanyi Zhang, Ruotian Luo, Haochen Wang, Falcon Z. Dai, Andrea F. Daniele, Mohammadreza Mostajabi, Steven Basart, Matthew R. Walter, and Gregory Shakhnarovich. DIODE: A dense indoor and outdoor depth dataset. *arXiv preprint arXiv:1908.00463*, 2019.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Chung-Shien Brian Wang, Christian Schmidt, Jens Piekenbrinck, and Bastian Leibe. Block-sparse global attention for efficient multi-view geometry transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14546–14555, 2026a.

Hengyi Wang and Lourdes Agapito. 3D reconstruction with spatial memory. In *International Conference on 3D Vision*, pp. 78–89. IEEE, 2025.

Hengyi Wang and Lourdes Agapito. AMB3R: Accurate feed-forward metric-scale 3D reconstruction with backend. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14612–14625, 2026.

Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. VGGT: Visual geometry grounded transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5294–5306, 2025a.

Kaixuan Wang and Shaojie Shen. Flow-motion and depth network for monocular stereo and beyond. *IEEE Robotics and Automation Letters*, 5(2):3307–3314, 2020.

Qianqian Wang, Yifei Zhang, Aleksander Holynski, Alexei A. Efros, and Angjoo Kanazawa. Continuous 3D perception model with persistent state. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10510–10522, 2025b.

Ruicheng Wang, Sicheng Xu, Cassie Dai, Jianfeng Xiang, Yu Deng, Xin Tong, and Jiaolong Yang. MoGe: Unlocking accurate monocular geometry estimation for open-domain images with optimal training supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5261–5271, 2025c.

Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. DUSt3R: Geometric 3D vision made easy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20697–20709, 2024.

Yifan Wang, Jianjun Zhou, Haoyi Zhu, Wenzheng Chang, Yang Zhou, Zizun Li, Junyi Chen, Jiangmiao Pang, Chunhua Shen, and Tong He. $\pi^3$: Permutation-equivariant visual geometry learning. In *International Conference on Learning Representations*, 2026b.

Tong Wu, Jiarui Zhang, Xiao Fu, Yuxin Wang, Jiawei Ren, Liang Pan, Wayne Wu, Lei Yang, Jiaqi Wang, Chen Qian, Dahua Lin, and Ziwei Liu. OmniObject3D: Large-vocabulary 3D object dataset for realistic perception, reconstruction and generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 803–814, 2023.

Xiaoyang Wu, Li Jiang, Peng-Shuai Wang, Zhijian Liu, Xihui Liu, Yu Qiao, Wanli Ouyang, Tong He, and Hengshuang Zhao. Point Transformer V3: Simpler, faster, stronger. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4840–4851, 2024.

Hongchi Xia, Yang Fu, Sifei Liu, and Xiaolong Wang. RGBD objects in the wild: Scaling real-world 3D object learning from RGB-D videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22378–22389, 2024.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. On layer normalization in the transformer architecture. In *International Conference on Machine Learning*, pp. 10524–10533, 2020.

Jianing Yang, Alexander Sax, Kevin J. Liang, Mikael Henaff, Hao Tang, Ang Cao, Joyce Chai, Franziska Meier, and Matt Feiszli. Fast3R: Towards 3D reconstruction of 1000+ images in one forward pass. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21924–21935, 2025.

Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth Anything V2. *Advances in Neural Information Processing Systems*, 37:21875–21911, 2024.

Chandan Yeshwanth, Yueh-Cheng Liu, Matthias Nießner, and Angela Dai. ScanNet++: A high-fidelity dataset of 3D indoor scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 12–22, 2023.

Jingyang Zhang, Shiwei Li, Zixin Luo, Tian Fang, and Yao Yao. Vis-MVSNet: Visibility-aware multi-view stereo network. *International Journal of Computer Vision*, 131(1):199–214, 2023a.

Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3836–3847, 2023b.

Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. *ACM Transactions on Graphics*, 37(4), 2018.

# A Implementation Details

## A.1 Architecture

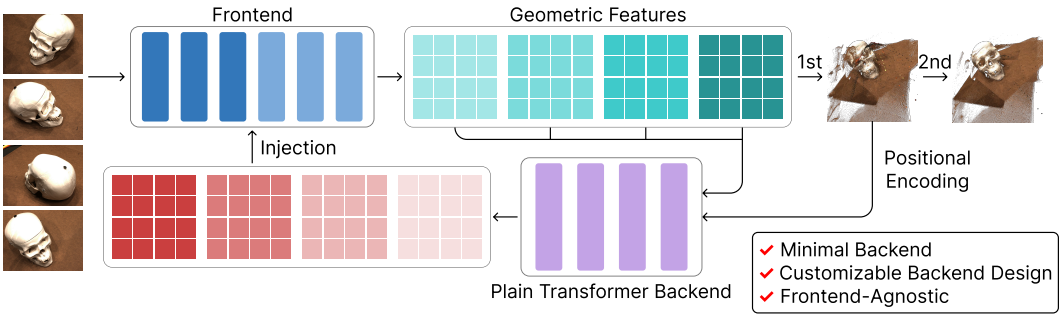



Figure 1: **Overview of Our Method.** A frontend extracts geometric features from multi-view images. Our plain Transformer backend processes these features with global self-attention, optionally using predicted points as positional encodings. The refined features are injected back to the frontend, producing the final 3D reconstruction.

**Backend.** The proposed backend (Fig. 1) is a plain pre-norm Transformer (Xiong et al., 2020) that consumes the per-pixel geometric tokens emitted by the frontend. Encoder and decoder features are concatenated channel-wise into an input tensor of shape $(B,\ T,\ C_{\text{in}},\ H,\ W)$, where $H$ and $W$ denote the spatial height and width of the frontend's final feature map rather than the input image resolution. Here $C_{\text{in}} = 3072$ (encoder $1024$ plus the concatenated frame/global decoder features $2048$). The tensor is passed through a small two-layer aligner MLP ($3072 \to 1536 \to 1024$ with GELU) to a common $1024$-dim representation, bilinearly upsampled by a factor of two as in AMB3R (with the predicted pointmaps resized to the same grid), and then permuted and flattened into a single sequence of length $N = T \cdot 2H \cdot 2W$ before a linear projection maps it to the backend's hidden dimension. After the backend, a strided convolution block maps the refined features back to the $H \times W$ grid before injection. Our default configuration uses 4 blocks, hidden dimension 384, 8 attention heads (head dimension 48), and an MLP ratio of 4, totalling roughly 8M parameters. This is more than an order of magnitude smaller than AMB3R's voxel backend (Wang & Agapito, 2026) ($\sim$153M). Training uses bf16 mixed precision throughout.

**3D RoPE.** Following Section 3.3, the per-head channel dimension is split into three equal axis groups; within each group, channel pairs are rotated by an angle proportional to the corresponding component of the predicted pointmap, with a log-spaced frequency base $b$=100. Because RoPE makes attention scores depend on coordinate differences, the scores are insensitive to translations of the coordinate origin within the chosen predicted pointmap frame, up to the periodicity of the embedding (Su et al., 2024).

**Absolute PE.** For the absolute-PE variant ablated in Section 4.3, we embed each token's predicted 3D coordinate $P_t[\mathbf{u}] \in \mathbb{R}^3$ with a two-layer MLP and add the result to the token sequence once, before the first transformer block. The MLP has the form $\mathbb{R}^3 \to \mathbb{R}^C \to \text{GELU} \to \mathbb{R}^C$ where $C$ is the backend hidden dimension; its output projection is zero-initialised, so absolute PE acts as a no-op at the start of training and the model is free to either use it or ignore it. Unlike 3D RoPE, this scheme injects the absolute pointmap value once and lets it propagate through self-attention as a regular feature.

**Backend-to-frontend injection.** The refined backend tokens are fed back to the frozen frontend through one zero-initialised injection module per decoder layer of the VGGT frontend. For zero-convolution (Zhang et al., 2023b) we use two-layer $1\times1$ Conv2d $C_{\text{dec}} \to C_{\text{dec}} \to C_{\text{dec}}$ with a GELU activation and a zero-initialised final layer, where $C_{\text{dec}}$ is the frontend's decoder hidden dimension; for AdaLN-Zero (Peebles & Xie, 2023) we predict a (gate, scale, shift) triplet from the same conv stack. In both cases the injected residual is exactly zero at initialisation, so the system reduces to the pretrained frontend at the start of training and the backend is free to learn how to perturb it.

### A.2 Training Datasets

We adopt the training-data mixture used by AMB3R (Wang & Agapito, 2026). The mixture contains 11 datasets covering indoor scenes, object-centric captures, driving scenes, and outdoor environments: ScanNet (Dai et al., 2017), ScanNet++ (Yeshwanth et al., 2023), WildRGBD (Xia et al., 2024), MapFree (Arnold et al., 2022), Aria (Pan et al., 2023), Waymo (Sun et al., 2020), Virtual KITTI 2 (Cabon et al., 2020), GTA-SfM (Wang & Shen, 2020), MVS-Synth (Huang et al., 2018), OmniObject3D (Wu et al., 2023), and Hypersim (Roberts et al., 2021). Following AMB3R, we retain only a subset of each dataset. Since AMB3R finds that dataset diversity matters more than raw sample count, we sample 2,000 sequences per epoch from the full mixture. We exclude Co3D (Reizenstein et al., 2021), consistent with AMB3R's report that including Co3D can make the VGGT frontend overfit to dataset-specific patterns.

Each training sequence contains 5–16 frames from a single scene. Images are rendered at one of six resolutions with a 518-pixel long edge: $518 \times \{392, 336, 294, 266, 210, 154\}$. We allocate the 2,000 sequences per epoch across four scene categories. The indoor category receives 700 sequences, drawn from ScanNet (315), ScanNet++ (180), Hypersim (160), and Aria (45). The object-centric category receives 300 sequences, drawn from WildRGBD (245) and OmniObject3D (55). The driving category receives 600 sequences, drawn from Waymo (570) and Virtual KITTI 2 (30). The outdoor category receives 400 sequences, drawn from MapFree (235), GTA-SfM (110), and MVS-Synth (55). Within each category, the allocation is proportional to the number of distinct scenes contributed by each dataset.

### A.3 Training Configuration

We optimise with AdamW (Loshchilov & Hutter, 2019) ($\beta_1$=0.9, $\beta_2$=0.95, weight decay 0.05) for 30 epochs (60,000 sequences in total). The learning rate is $10^{-6}$, kept constant after a 1-epoch linear warm-up. The ablations in Tables 6a to 6c use a shorter 6-epoch schedule with a base learning rate of $1.5 \times 10^{-4}$, scaled linearly with batch size, and a half-cycle cosine decay to $10^{-6}$ after a 1-epoch warm-up. Training runs on 8 H100 GPUs with a per-GPU batch of 1 and 6 gradient accumulation steps, giving an effective batch size of 48. The frontend is frozen throughout; only the backend (aligner MLP, input projection, transformer blocks, output projection, strided convolution block) and the per-decoder-layer injection modules receive gradient updates. End-to-end training takes about 4 hours (about 8 minutes per epoch) on this hardware.

## B Detailed Evaluation Protocol

This section supplements the protocol summary in Section 4.1 with implementation-level details.

**Monocular depth estimation.** We use the Marigold (Ke et al., 2024) split, and align predictions per image by least-squares scale and shift before computing the reported metrics.

**Camera pose estimation.** For each RealEstate10K (Zhou et al., 2018) scene we draw a uniformly random subset of 10 frames (fixed seed across all methods) and feed all 10 jointly to the model. Reported numbers therefore correspond to a 10-frame multi-view pose-from-images setting rather than a streaming or pairwise setting.

**Multi-view depth estimation.** We follow the RMVDB protocol (Schröppel et al., 2022) unchanged: per-scene depth predictions are evaluated against the RMVDB keyview–source samples (110 across 22 DTU (Aanæs et al., 2016) scans, 104 across 13 ETH3D (Schöps et al., 2017) scenes), with median-scale alignment to GT depth. The same sample lists are reused for multi-view 3D reconstruction on ETH3D and DTU, so the two tasks share identical test samples.

**Multi-view 3D reconstruction.** For each scene, VGGT (Wang et al., 2025a) predicts per-frame depth, camera intrinsics, and camera extrinsics. We recover world-space points by unprojecting the predicted depths with the corresponding intrinsics and transforming them with the estimated extrinsics. The resulting point cloud is then median-scale aligned to the ground-truth point cloud

before evaluation. For efficiency, we deterministically subsample up to 100,000 valid points per scene before computing accuracy and completeness with KD-tree nearest-neighbor queries.

For frontends whose camera predictions are not expressed in a frame-0 anchored world coordinate system, such as $\pi^3$ (Wang et al., 2026b), we re-anchor the predicted poses in a lightweight wrapper. Specifically, we left-multiply each predicted pose by $T_0^{-1}$, where $T_0$ is the predicted pose of the first frame. This operation only changes the global coordinate frame and leaves the $\pi^3$ supervision unchanged, since its training loss is defined on local points and pairwise relative poses, both of which are invariant to a global SE(3) transform. No retraining is therefore required. After re-anchoring, we recompute the world-space points and evaluate them with the same scale-aligned pipeline used for VGGT.

**Video depth estimation.** The model consumes the entire test sequence in a single forward pass, after which per-frame metrics are averaged across the sequence.

**Frame sampling.** For 7-Scenes (Brachmann et al., 2021) we follow the Spann3R (Wang & Agapito, 2025) keyframe schedule, taking every 40th frame from each test sequence (typically 25 keyframes; shorter sequences yield fewer). For DTU (Aanæs et al., 2016) we use the same RMVDB sample list as for multi-view depth. For ETH3D (Schöps et al., 2017) we use the RMVDB sample list (Schröppel et al., 2022).

## C Failure Modes and Limitations

**ETH3D and dataset-composition sensitivity.** The most visible failure case of our model is on ETH3D (Schöps et al., 2017) when using the VGGT (Wang et al., 2025a) frontend (Table 4). After 30 epochs of training, rel is 6.54 with our backend, compared with 6.04 for frozen VGGT. Our reproduction of AMB3R (Wang & Agapito, 2026), trained with the same data mixture as our model (Section A.2), reaches 7.39, which is substantially worse than the 4.63 reported by the public AMB3R checkpoint. This appendix expands on the limitation noted in the main paper's conclusion (Section 6).

We align the training setup with the official AMB3R repository where possible, but our reproduction does not exactly match the public checkpoint's full training-data recipe. In particular, although we follow AMB3R in retaining only a subset of each dataset, the per-dataset scene selection is not publicly available, and we use our reproduced per-category sampling budget. These differences are important because ETH3D appears to be especially sensitive to data composition and reproduction details. The fact that the public AMB3R checkpoint performs well on ETH3D shows that an AMB3R-style backend can handle this benchmark, but it does not by itself isolate the source of the gap in our reproduction.

Several observations suggest that the ETH3D degradation is unlikely to be caused solely by the choice between voxel and voxel-free backend representations. First, both reproduced backends fall behind the public checkpoint on ETH3D under the same training mixture, with 7.39 for the AMB3R-style voxel backend and 6.54 for our global self-attention backend. Thus, the gap is shared across backend architectures under our reproduced setup. Second, the same voxel-free backend improves $\pi^3$ (Wang et al., 2026b) on ETH3D, reducing rel from 5.84 to 4.89, and also improves Déjà View and Depth Anything 3 on ETH3D (Table 7). This indicates that the backend architecture can generalize to ETH3D when paired with a different frontend. Third, in preliminary experiments, we observed that re-balancing the VGGT-based training proportions can improve ETH3D, but at the cost of performance on other benchmarks.

Fig. 2 shows rel over the 30-epoch schedule. On ETH3D, our backend is lower than the voxel backend at most checkpoints, and at several checkpoints it also achieves lower rel than frozen VGGT (e.g., 4.58 at epoch 2 vs. 6.04). On 7-Scenes and DTU, our backend remains close to or below the voxel backend throughout training.

We therefore treat ETH3D as a reproduction caveat rather than as evidence for or against voxel compactness. The gap may reflect several non-architectural factors, including the per-dataset scene selection, per-category sampling ratios, frame sampling, and preprocessing. More broadly, the result suggests that VGGT-based backend adaptation is sensitive to the training-dataset composition, and that finding a single mixture that performs well across all benchmarks is difficult within our

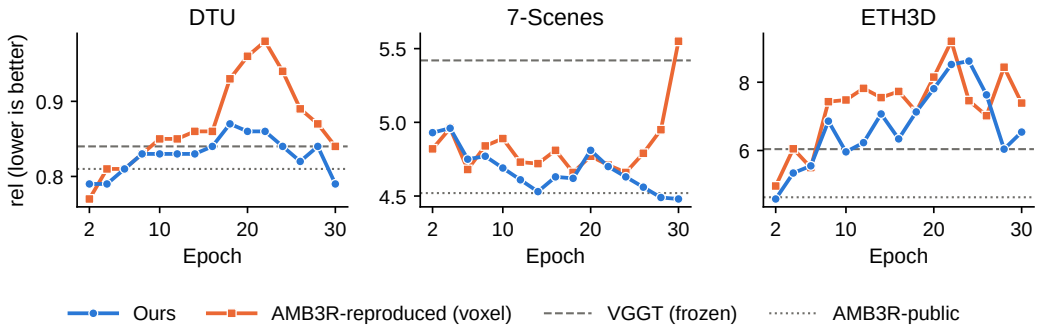



Figure 2: **Multi-view 3D reconstruction over training.** The rel of our backend and the reproduced AMB3R voxel backend at every second epoch of the 30-epoch schedule, both trained with the same data and frozen VGGT frontend. Dashed and dotted lines show frozen VGGT and the public AMB3R checkpoint.

reproduced setup. Closing the ETH3D gap therefore appears to require improved data curation, rather than a change in backend representation alone.

## D Transfer Details for Other Frontends

The proposed backend is not tied to the internal architecture of VGGT: it only requires per-pixel geometric tokens and per-pixel 3D coordinates from the frontend. We test this by attaching the same backend to three other frontends, $\pi^3$ (Wang et al., 2026b), Déjà View (Burzio et al., 2026), and Depth Anything 3 (Lin et al., 2026), and re-running multi-view 3D reconstruction (Table 7). With $\pi^3$, our backend improves over the frozen frontend on every benchmark we evaluate: ETH3D (Schöps et al., 2017) improves from 5.84 to 4.89, DTU (Aanæs et al., 2016) from 1.52 to 1.48, and 7-Scenes (Brachmann et al., 2021) from 5.90 to 5.33. It also outperforms the AMB3R-style voxel backend on all reported metrics in this setting. Our backend also improves Déjà View and Depth Anything 3 on ETH3D (from 4.96 to 4.92 and from 12.53 to 9.31), suggesting that the ETH3D regression observed with the VGGT (Wang et al., 2025a) frontend may be tied to the VGGT-based reproduction setup rather than to the voxel-free backend architecture alone (Section C).

**Backend-to-frontend injection.** The injection interface follows the block structure of each frontend. For $\pi^3$, the same zero-initialized residual injection module used with VGGT is inserted before each decoder block. Depth Anything 3 has no separate multi-view decoder, so we inject before each of the 24 transformer blocks of its backbone. Déjà View applies a single transformer block recurrently, and we inject only before the first recurrent step.

**Loss.** For $\pi^3$, we train the backend with its native training loss (Wang et al., 2026b): a depth-weighted $L_1$ loss on scale-aligned local points, a surface-normal loss on per-cell normal angles, a binary-cross-entropy confidence loss with a sky-mask term, and a relative-pose loss combining a geodesic rotation term with a Huber translation term. We attach our backend without modifying any of these loss components. For Déjà View and Depth Anything 3, we use the same AMB3R training loss as for VGGT.

## E Inference Cost

We also measure the end-to-end inference cost of the backend on DTU (Aanæs et al., 2016) and 7-Scenes (Brachmann et al., 2021). Wall-clock time is measured from the evaluation-log `tqdm` traces on a single H100 GPU with BF16 inference. The reported time includes the model forward pass as well as the downstream evaluation steps, including KD-tree subsampling, accuracy computation, and completeness computation. These numbers should therefore be interpreted as end-to-end per-sample evaluation time rather than pure model-forward latency.

Table 8: End-to-end inference cost on DTU. DTU contains 110 evaluation samples with approximately 5 frames per sample. Times include model forward and evaluation-side KD-tree computations.

| Method | Params | Time / sample | Total time |
|---|---|---|---|
| VGGT baseline | – | ∼0.83 s | – |
| AMB3R-public | ∼153M | 1.18 s | 02:09 |
| Ours (default) | ∼8M | 1.09 s | 01:59 |

Table 9: End-to-end inference cost on 7-Scenes. We evaluate 18 sequences with approximately 25 keyframes per sequence, corresponding to about 104k backend tokens. Times include model forward and evaluation-side KD-tree computations.

| Method | Params | Time / sequence | Total time |
|---|---|---|---|
| VGGT baseline | – | 1.51 s | 00:27 |
| AMB3R-public | ∼153M | 2.25 s | 00:40 |
| Ours (default) | ∼8M | 2.42 s | 00:43 |

On DTU, our default backend is slightly faster than the AMB3R voxel backend, reducing the average time from 1.18 s/sample to 1.09 s/sample. This corresponds to roughly an 8% reduction in end-to-end evaluation time. Since each DTU sample contains only about five frames, the flattened token sequence is short enough that the quadratic cost of global self-attention does not dominate. In this regime, the fixed overhead of the voxel backend, including space-filling-curve serialization and KNN interpolation, is comparatively costly.

On 7-Scenes, where each sequence contains about 25 keyframes, our default backend is slightly slower than the voxel backend: 2.42 s/sequence compared with 2.25 s/sequence. The overhead over the VGGT-only baseline is 0.91 s for our default backend and 0.74 s for AMB3R. This is consistent with the expected scaling of global self-attention: at roughly 104k tokens per sequence, the $O(N^2)$ attention cost begins to offset the simplicity of the voxel-free design.

Overall, the default backend is substantially smaller than the voxel backend, with roughly 8M parameters compared with AMB3R's approximately 153M parameters, while remaining comparable in wall-clock evaluation time on the tested settings. It is slightly faster on short DTU samples and within about 8% of AMB3R on the longer 7-Scenes sequences. These measurements also clarify the main scalability limitation of the current implementation: for substantially longer sequences, sparse, windowed, or tiled attention variants would likely be needed to preserve the simplicity of the voxel-free backend while avoiding the quadratic cost of global attention.

## F Voxel Pooling Ablation

To test whether voxel aggregation adds anything on top of global attention, we add AMB3R-style voxel pooling to our backend. Per-pixel tokens are average-pooled into voxels according to their predicted 3D coordinates, processed by the same Transformer, and mapped back to pixels by $K$-nearest-neighbor interpolation as in AMB3R (Wang & Agapito, 2026). All other settings follow our default configuration and the same 30-epoch schedule. As shown in Table 10, voxel pooling yields no improvement over our backend on any of the three datasets.

Table 10: **Voxel pooling ablation.** Multi-view 3D reconstruction (rel) after 30 epochs, with the frozen VGGT frontend and the same training data for all rows.

| Method | DTU | 7-Scenes | ETH3D |
|---|---|---|---|
| AMB3R-reproduced (voxel) | 0.84 | 5.55 | 7.39 |
| Ours + voxel pooling | 0.80 | 4.71 | 6.89 |
| **Ours (default)** | **0.79** | **4.48** | **6.54** |

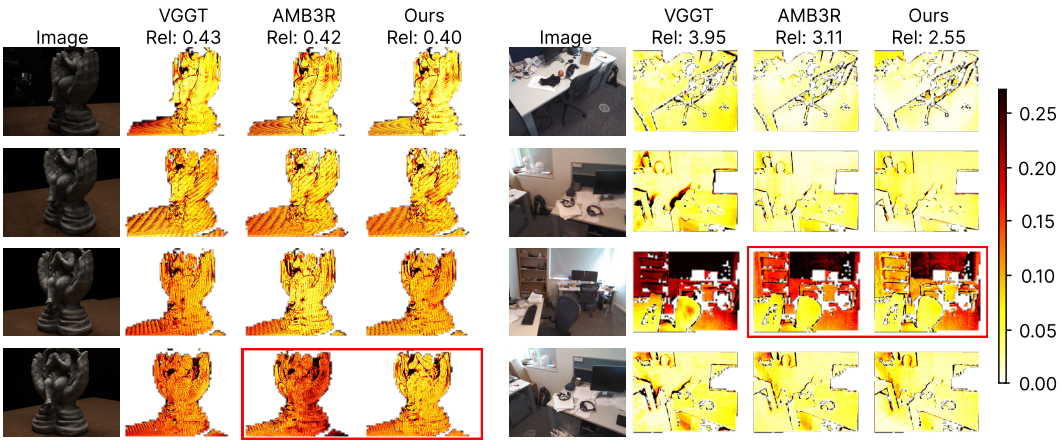



Figure 3: **Pointmap error visualization.** Per-pixel L2 error of predicted pointmaps for VGGT (Wang et al., 2025a), AMB3R (Wang & Agapito, 2026), and our method, shown on representative scenes from DTU (Aanæs et al., 2016) (left) and 7-Scenes (Brachmann et al., 2021) (right). Brighter colors indicate lower error. Red boxes highlight regions where our method produces visibly lower error than AMB3R.

## G Qualitative Comparison

Fig. 3 visualizes the predicted pointmaps as heatmaps of per-pixel L2 error for VGGT, AMB3R, and our method. Examples are drawn from DTU (Aanæs et al., 2016) and 7-Scenes (Brachmann et al., 2021). As highlighted by the red boxes, the difference in reconstruction quality between AMB3R and the proposed method is visually discernible, with our method producing visibly lower error in the highlighted regions.