Scaffold3D: SfM-Conditioned Pointmap Prediction for Multi-View 3D Reconstruction

Frano Rajič[1], **Yutong Chen**[1], **Haofei Xu**[1], **Zador Pataki**[1], **Marc Pollefeys**[1,2], **Siyu Tang**[1]
[1]ETH Zurich [2]Microsoft

Abstract

Feed-forward models make multi-view 3D reconstruction fast and robust, but their predictions often remain locally inaccurate and geometrically inconsistent across views. Structure-from-motion (SfM) offers a complementary geometric signal through camera poses and 3D points estimated by correspondence filtering and optimization. To propagate this globally consistent *SfM scaffold* into dense geometry, we introduce **Scaffold3D**, an SfM-conditioned reconstruction framework that injects image patch-aligned SfM point tokens into a pairwise feed-forward pointmap predictor. Our approach preserves image-token reasoning while using explicit SfM structure to condition dense prediction. Because Scaffold3D predicts pairwise pointmaps in local frames, we fuse them with global alignment and an SfM anchoring term. Scaffold3D achieves stronger overall performance than feed-forward reconstruction models, point-token propagation, depth-completion baselines, and recent geometry-conditioned models on standard benchmarks (ScanNet++, ETH3D, and Tanks&Temples) and out-of-domain MV-dVRK, and is competitive on 4D-DRESS. On standard benchmarks, gains over point-token propagation are especially clear in low- and no-overlap regions. Our pipeline, built around a *two-view* SfM-conditioned pointmap predictor, outperforms several off-the-shelf *multi-view* geometry-conditioned baselines. Together, these results establish SfM-scaffold conditioning as a practical interface between learned pointmap prediction and classical geometric optimization. Project page: `https://ethz-vlg.github.io/scaffold3d`.

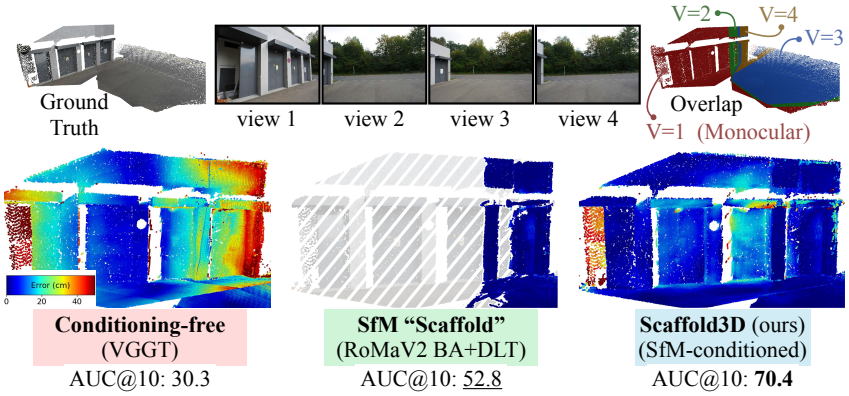

Figure 1: **Scaffold3D completes an SfM scaffold.** Top: ground-truth geometry, input views, and multi-view overlap. Bottom: a conditioning-free pointmap predictor is dense but inaccurate, while the SfM scaffold is accurate but incomplete. Scaffold3D conditions dense prediction on the SfM scaffold and improves AUC@10 cm on this ETH3D example from 30.3 and 52.8 to 70.4.

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

1 Introduction

Multi-view 3D reconstruction is a core component in robotics, AR/VR, digital content creation, and embodied perception. Modern feed-forward models have made dense reconstruction much faster and more robust, and now predict depth, pointmaps, and camera geometry directly from images [17, 19, 22, 39, 41, 43]. However, these models typically do not enforce explicit geometric constraints during prediction to improve multi-view consistency and precision. As a result, despite strong robustness and coverage, their dense outputs often remain geometrically misaligned across views and locally inaccurate, especially on fine structures and out-of-domain scenes.

Structure-from-motion (SfM) pipelines remain strong at enforcing geometric consistency through correspondence filtering and optimization [8, 23, 28, 32, 38]. Modern dense matchers and learned SfM pipelines have improved robustness to low texture and sparse, wide-baseline views [10, 28, 29]. SfM pipelines estimate camera poses and 3D points, either before dense prediction or as part of a reconstruction pipeline that also uses learned priors. The resulting *SfM scaffold* (*i.e.*, poses, intrinsics, and 3D points) covers only a subset of image regions, but it provides explicit multi-view constraints. We therefore frame dense reconstruction as a propagation problem: given a reliable SfM scaffold, how should a learned model propagate it into dense pointmaps without losing its global consistency?

A common approach refines feed-forward predictions with geometric optimization, such as bundle adjustment or global alignment [19, 39, 41]. When SfM geometry is used only for post-processing, the network predicts dense geometry without receiving the SfM cameras and points as inputs during its forward pass. Another option is monocular depth completion: complete each view from SfM depth samples and then fuse the views [15, 16, 35, 54]. This uses the scaffold points, but it treats dense prediction mostly as a per-image problem. Multi-view reconstruction should instead exploit cross-view consistency, so that observations from one view constrain another through camera geometry and correspondence.

Recent geometry-conditioned models move in this direction, but use the scaffold through different interfaces. Pow3R [14], Pi3X [43], and MapAnything [17] feed SfM-derived geometry as scaffold depth maps, together with intrinsics or pose conditioning, rather than explicit 3D point observations. GGPT [5] takes the opposite route: it preserves the scaffold as point tokens, but propagates them without access to image features, completely discarding appearance and matching cues that are useful for dense pointmap prediction. This motivates combining explicit SfM point observations with image features so that geometry and appearance jointly guide dense prediction.

We introduce **Scaffold3D**, an SfM-scaffold-conditioned pointmap prediction framework for multi-view reconstruction. Scaffold3D contains a feed-forward image encoder-decoder and augments the decoder with image patch-aligned SfM point tokens. For each view, SfM 3D points whose projections fall into the same image patch are encoded and aggregated into a single point token. Point tokens across patches interact through self-attention and are then injected into the image-token decoder. This allows partial but globally consistent SfM geometry to guide dense pairwise pointmap prediction beyond directly triangulated regions while preserving dense visual correspondence cues.

Since pairwise pointmaps are predicted in local coordinate frames, we further introduce an SfM-anchored global alignment formulation to fuse them into a globally consistent reconstruction. As a result, the SfM scaffold serves two complementary roles: locally, it guides the pairwise pointmap predictor toward geometrically informed dense reconstruction, and globally, it acts as a reliable anchor for fusing pairwise predictions into a coherent 3D scene.

Experiments on standard datasets [18, 31, 47] and out-of-domain data [4, 42] support four conclusions. First, conditioning on estimated SfM scaffolds improves over unconditioned feed-forward baselines [19, 39, 41, 43] and feed-forward baselines that use geometric optimization only after dense prediction [8]. Second, combining image and point tokens improves over point-only propagation [5], especially in low- and no-overlap regions. Third, our pipeline, built around a *two-view* Scaffold3D predictor, outperforms several off-the-shelf *multi-view* geometry-conditioned baselines [5, 17, 43]. Fourth, conditioning improves mean accuracy with all eleven scaffold sources, including those with under 3% pixel coverage. We release training and evaluation code, pre-trained checkpoints, and dataset preprocessing scripts to facilitate future research and reproducibility.

2 Related Work

SfM-Based 3D Reconstruction. COLMAP [32] remains a standard reconstruction approach for views with sufficient overlap. Modern SfM frameworks improve robustness to low overlap, large baselines, or low texture by incorporating learning-based matchers, learned initializations, or depth priors [8, 28, 29, 38]. GGPT [5] initializes bundle adjustment (BA) with feed-forward camera and point predictions, then constructs the scaffold from refined cameras and dense correspondences by direct linear triangulation (DLT). Despite improved matching robustness, triangulation still requires reliable cross-view correspondences and cannot constrain regions visible in only one view. Whereas prior-assisted SfM uses learned geometry to guide geometric optimization, Scaffold3D also conditions the image-token decoder on the SfM scaffold to complete geometry beyond triangulated support.

Feed-Forward 3D Reconstruction. Recent transformer-based models [11, 17, 37, 39–41, 43, 48–50] predict dense pointmaps or depth maps from RGB images, with cameras predicted directly or recovered geometrically. Feed-forward prediction is much faster than iterative SfM optimization. Large-scale training gives strong sparse-view robustness, but predictions can remain geometrically inconsistent across views, especially out of distribution [5]. Stereo and multi-view stereo (MVS) methods combine cross-view matching with learned priors, but require calibrated inputs [1, 13, 36]. Our pipeline takes only RGB images and estimates an SfM scaffold to condition dense prediction, achieving stronger overall reconstruction accuracy than the evaluated feed-forward baselines on standard benchmarks and MV-dVRK.

Depth Completion (DC). Depth completion predicts dense depth from depth values available at only a subset of image pixels, typically through *injection-based* methods [12, 21, 44] trained with depth inputs or *optimization-based* methods [15, 35] that fit predictions to depth samples at test time, including multi-view VGGT adaptation in CAPA [16]. Hybrid approaches [53, 54] predict depth gradients and integrate them into dense depth maps using a differentiable solver. However, independent per-view completion can produce inconsistent predictions in overlapping regions without depth samples, even with shared SfM guidance. Scaffold3D instead learns SfM-conditioned pairwise completion, jointly using both images without per-scene adaptation of the predictor.

Multi-View Geometry Conditioning. Multi-view models incorporate geometry through different interfaces. Pow3R [14] and Pi3X [43] encode depth or intrinsics as patch tokens, while MapAnything [17] fuses ray-geometry features with image embeddings. Align3R [25] encodes monocular depth as pointmaps and injects their features into a DUSt3R decoder for dynamic-video reconstruction. GGPT [5] propagates partial 3D structures via point tokens alone. Scaffold3D instead injects partial, triangulated SfM points into an image-token decoder, retaining appearance-based reasoning during completion and using the same SfM geometry to anchor the merging of pairwise predictions.

3 Method

Given N RGB input views $\{I_n \in \mathbb{R}^{H\times W\times 3}\}_{n=1}^{N}$, our goal is to predict dense 3D point maps $\{X_n \in \mathbb{R}^{H\times W\times 3}\}_{n=1}^{N}$ in a unified global coordinate system. As illustrated in Fig. 2, *Scaffold3D* operates in three stages. First, we leverage an SfM pipeline to estimate a geometric scaffold $\{S_n \in \mathbb{R}^{H\times W\times 3}\}_{n=1}^{N}$, which provides geometrically consistent 3D points wherever reliable cross-view correspondences are available, but leaves regions without such support incomplete. Second, an SfM-conditioned pointmap predictor predicts complete two-view pointmaps by conditioning on the RGB images and their respective SfM scaffolds. Finally, we introduce an SfM-anchored global alignment to aggregate these pairwise predictions into a globally consistent and dense 3D reconstruction.

3.1 SfM Scaffold

We adopt the SfM pipeline from GGPT [5] to generate our geometric scaffold. It establishes dense correspondences using RoMaV2 [10] and initializes the reconstruction with cameras and points predicted by a feed-forward reconstruction model, such as VGGT [39]. Global bundle adjustment (BA) refines these initial cameras and 3D points against the correspondences, yielding camera-to-world poses $\{T_n\}_{n=1}^{N}$ and intrinsics $\{K_n\}_{n=1}^{N}$. Matches are filtered by matching score and cycle consistency, then checked for epipolar consistency with the estimated cameras. Using the refined

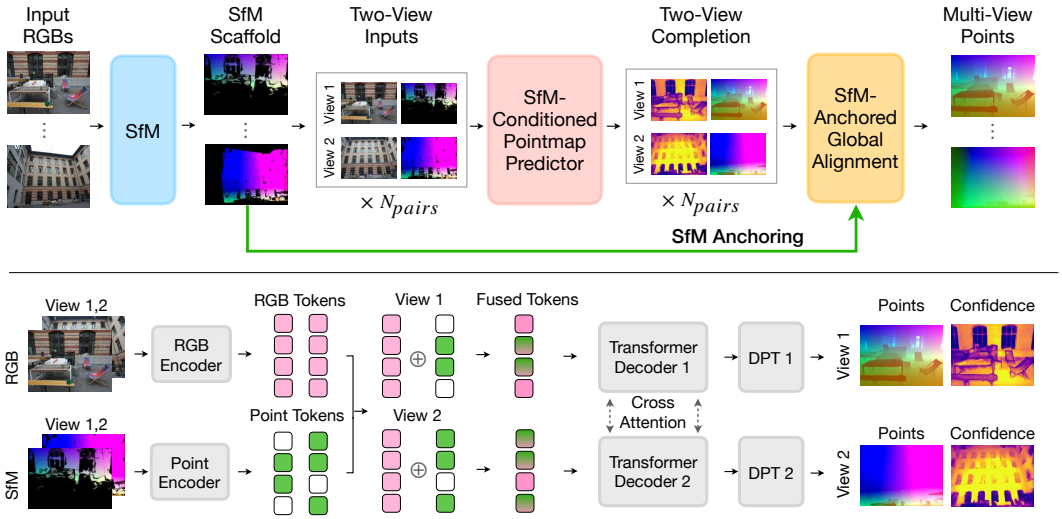

Figure 2: **Scaffold3D overview.** *Top*: Pipeline. First, we run SfM to triangulate a partial geometry scaffold. Second, we use an SfM-conditioned feed-forward pointmap predictor to predict complete pointmaps for view pairs, conditioning on both image tokens and point tokens. Finally, we fuse pairwise pointmaps with SfM-anchored global alignment. *Bottom*: Architecture of the predictor.

cameras, we triangulate tracks observed in at least two views using DLT and filter the resulting points by reprojection error and a minimum triangulation angle.

Using the resulting 3D-2D correspondences, we derive a multi-view SfM pointmap $\{S_n \in \mathbb{R}^{H\times W\times 3}\}_{n=1}^{N}$. Entries without valid triangulated support are left empty in S_n. We define the set of pixels with valid SfM support as $\Omega_{\mathcal{S}} = \{(n, p) \mid \text{pixel } p \text{ in view } n \text{ has a valid SfM estimate}\}$. These geometrically consistent yet incomplete points serve as a prior, which we propagate to regions lacking geometry support to achieve an accurate and complete reconstruction.

3.2 SfM-Conditioned Pointmap Predictor

To densify the partial reconstruction from $\Omega_{\mathcal{S}}$ across the entire image, Scaffold3D uses a two-view conditioned pointmap predictor. For a given image pair (a, b), the network takes the RGB images and normalized SfM points as input to predict complete 3D pointmaps $(\hat{X}_{a,a}, \hat{X}_{b,a})$ in the coordinate system of view a. We describe the architecture, normalization, and training objectives below.

3.2.1 Network Architecture

The architecture of our SfM-conditioned pointmap predictor (Fig. 2) comprises an RGB encoder and a point encoder to produce view-specific tokens. Spatially aligned RGB and point tokens are fused via element-wise addition before the first decoder layer at each view. These fused tokens are then processed by two cross-attended decoders to predict points and confidence maps. We instantiate the RGB encoder and decoders using the MASt3R architecture [19] and introduce a novel point encoder for aggregating SfM observations into patch-aligned conditioning tokens.

RGB encoder. Following [19, 41], for each view $v \in \{a, b\}$, the input image I_v is patchified and encoded by a ViT encoder [7]. We project the encoder features to the decoder width d and denote the resulting RGB tokens as $f_v^0 \in \mathbb{R}^{L\times d}$, where $L = HW/{P_{\text{patch}}}^2$ and P_{patch} is the patch size.

Point encoder. We map the SfM inputs to a sequence of point tokens spatially aligned with the RGB tokens. Given the normalized SfM input $\hat{S}_v \in \mathbb{R}^{H\times W\times 3}$, we partition the map into patches of size P_{patch}. For each pixel $(v, p) \in \Omega_{\mathcal{S}}$ with a valid SfM estimate, an MLP ϕ_{pt} encodes its 3D coordinates $\hat{S}_v(p)$ and its relative 2D location within the patch $\delta(p)$:

$$h_{v,p} = \phi_{\text{pt}}\Big(\hat{S}_v(p), \delta(p)\Big). \tag{1}$$

For each nonempty patch q, we average the features $h_{v,p}$ within the patch to compute a patch-level feature $g_{v,q}$:

$$g_{v,q} = \frac{1}{|\mathcal{K}_{v,q}|} \sum_{p \in \mathcal{K}_{v,q}} h_{v,p}, \qquad \mathcal{K}_{v,q} = \{p \mid (v,p) \in \Omega_{\mathcal{S}},\ p \text{ is in patch } q\}. \tag{2}$$

We collect features $g_{v,q} \in \mathbb{R}^d$ for all patches with valid SfM support to form a sequence $g_v \in \mathbb{R}^{L' \times d}$, where $L' \leq L$. A residual MLP and layer normalization refine each aggregated token, followed by lightweight self-attention blocks that exchange information among point tokens:

$$Z_v^{\ell+1} = \mathrm{PointBlock}^{\ell}\left(Z_v^{\ell}\right), \qquad Z_v^0 = \{\rho(g_{v,q}) \mid |\mathcal{K}_{v,q}| > 0\}. \tag{3}$$

Here, ρ denotes the per-token refinement. After processing, we scatter refined features onto the image-token grid as $\bar{g}_v \in \mathbb{R}^{L \times d}$, setting $\bar{g}_{v,q} = 0$ for empty patches. This ensures that point tokens $\bar{g}_v$ are spatially aligned with image tokens f_v^0.

RGB and point token fusion. We fuse the RGB tokens (f_a^0, f_b^0) and point tokens $(\bar{g}_a, \bar{g}_b)$ for each view via element-wise addition:

$$\tilde{f}_a^0 = f_a^0 + \bar{g}_a, \quad \tilde{f}_b^0 = f_b^0 + \bar{g}_b. \tag{4}$$

Decoders. The fused tokens $\tilde{f}_a^0$ and $\tilde{f}_b^0$ are fed into decoders Dec_a and Dec_b, which employ mutual cross-attention:

$$\tilde{f}_a^{\ell+1} = \mathrm{Dec}_a^{\ell}(\tilde{f}_a^{\ell}, \tilde{f}_b^{\ell}), \qquad \tilde{f}_b^{\ell+1} = \mathrm{Dec}_b^{\ell}(\tilde{f}_b^{\ell}, \tilde{f}_a^{\ell}). \tag{5}$$

The features from the final attention layers are processed by two point heads to predict the dense geometry and associated confidence:

$$\left(\hat{X}_{a,a}, \hat{C}_{a,a}\right) = \psi_a(\{\tilde{f}_a^{\ell}\}_{\ell}), \qquad \left(\hat{X}_{b,a}, \hat{C}_{b,a}\right) = \psi_b(\{\tilde{f}_b^{\ell}\}_{\ell}). \tag{6}$$

Here, $\hat{X}_{a,a}$ and $\hat{X}_{b,a}$ represent 3D coordinates in the camera coordinate system of view a, following the convention in MAST3R [19].

Input normalization. We transform the original SfM points (S_a, S_b), initially in a global coordinate system with an arbitrary scale, into normalized local coordinates $(\hat{S}_a, \hat{S}_b)$ for view a. We first perform a conditioning-free pass (by setting $\bar{g}_a = \bar{g}_b = 0$) to obtain unconditional point predictions $(\hat{X}_{a,a}^{\text{uncond}}, \hat{X}_{b,a}^{\text{uncond}})$. Additionally, we transform S_a and S_b into camera a's coordinate system as $S_{a\to a}$ and $S_{b\to a}$ using the inverse SfM pose T_a^{-1}. We estimate the scale factor s from depth ratios across SfM-supported pixels in the pair:

$$s = \text{median} \left\{ \frac{\left[\hat{X}_{v,a}^{\text{uncond}}(p)\right]_z}{\left[S_{v\to a}(p)\right]_z} \mid (v,p) \in \mathcal{V}_{a,b} \right\}. \tag{7}$$

Here, $\mathcal{V}_{a,b} \subseteq \Omega_{\mathcal{S}}$ contains the pixels in views a and b where the predicted and scaffold depths are finite, nonzero, and have the same sign, ensuring positive scale ratios. We scale both pointmaps as $\hat{S}_a = s\, S_{a\to a}$ and $\hat{S}_b = s\, S_{b\to a}$. We then filter out points with nonfinite coordinates, nonpositive or near-zero depths, or extreme depths in camera a's frame. If points are removed, we recompute the scale once from the retained points when possible and reapply the filters (see Section C).

3.2.2 Training Loss

We supervise both the unconditional and SfM-conditioned forward passes with equal weight. Following MASt3R [19], each pass uses a confidence-weighted point map regression loss $\mathcal{L}_{\text{pm}}$ and an auxiliary InfoNCE matching loss $\mathcal{L}_{\text{match}}$:

$$\mathcal{L}_{\text{train}} = \frac{1}{2} \sum_{t \in \{\text{uncond},\text{cond}\}} \left(\mathcal{L}_{\text{pm}}^t + 0.075\, \mathcal{L}_{\text{match}}^t\right). \tag{8}$$

Training uses a staged mixture of ground-truth scaffold points sampled from overlapping regions and estimated SfM points, with inference-time scale alignment for the latter (Table C1). We randomly drop point tokens during training to ensure the decoder remains robust to varying scaffold densities.

3.3 SfM-Anchored Global Alignment

To aggregate pairwise predictions into a global 3D scene, we extend the global alignment framework from DUSt3R [41] by introducing an SfM-anchoring term. Let $\mathcal{E}$ be the image-pair graph used for both pointmap prediction and global alignment. We use the complete graph for up to 16 views. For larger inputs, we use a simple sparse graph that connects neighboring views in input order and uses two hubs, the first view and the middle view, each connected to every other view. We fix the camera parameters $\{K_n, T_n\}_{n=1}^{N}$ provided by the SfM scaffold and optimize a dense depth map D_v for each view along with an edge-local similarity transform (P_e, σ_e) for each pair. Here, $P_e \in SE(3)$ maps the pairwise frame to the SfM world frame, and $\sigma_e > 0$ scales the transformed point. The scene-frame point for pixel p is defined as:

$$\chi_v(p; D_v) = T_v\, \Pi^{-1}(p, D_v(p); K_v), \tag{9}$$

where Π^{-1} denotes pinhole back-projection. The pairwise loss $\mathcal{L}_{\text{pair}}$ enforces consistency between optimized scene points and pairwise predictions, using log-confidence weights and averaging over edges and pixels:

$$\begin{aligned}\mathcal{L}_{\text{pair}} = \frac{1}{|\mathcal{E}|HW} \sum_{e=(a,b)\in\mathcal{E}} \sum_{p} \Big[\log \hat{C}^e_{a,a}(p) \left\| \chi_a(p; D_a) - \sigma_e P_e \hat{X}^e_{a,a}(p) \right\|_2 \\ + \log \hat{C}^e_{b,a}(p) \left\| \chi_b(p; D_b) - \sigma_e P_e \hat{X}^e_{b,a}(p) \right\|_2 \Big].\end{aligned} \tag{10}$$

To ensure the dense maps remain anchored to the high-precision scaffold, we add a weighted anchoring term for SfM-supported pixels:

$$\mathcal{L}_{\text{anc}} = \frac{1}{\sum_{(v,p)\in\Omega_{\mathcal{S}}} w_{v,p}} \sum_{(v,p)\in\Omega_{\mathcal{S}}} w_{v,p} \left\| \chi_v(p; D_v) - S_v(p) \right\|_2. \tag{11}$$

The weights are DLT confidence scores $w_{v,p} = 1/(1 + e^{\text{reproj}}_{v,p})$, where $e^{\text{reproj}}_{v,p}$ is the reprojection error averaged over the visible 2D observations of the DLT track rasterized at pixel p. The final alignment is the result of:

$$\min_{\{D_v\},\{P_e\},\{\sigma_e>0\}} \mathcal{L}_{\text{pair}} + \lambda_{\text{anc}} \mathcal{L}_{\text{anc}}. \tag{12}$$

We initialize D_v from the prediction with the highest mean confidence among edges with view v as the reference, rescaled by that edge's initial alignment scale σ_e. To initialize (P_e, σ_e), we estimate camera b's pose in the predicted pair's reference frame using PnP, then align the predicted camera pair to the two fixed SfM cameras. At inference, the output is a set of dense point maps in the SfM world frame, with camera parameters preserved from the upstream SfM pipeline.

4 Experiments

Architecture and training. We train Scaffold3D on the 11 datasets as summarized in Table B2; full architecture, optimizer, schedule, and compute details are provided in Section C.

Evaluation metrics. We align each predicted reconstruction to the metric ground truth with robust Umeyama alignment [32], compute Euclidean point errors on ground-truth-covered pixels, and report the area under the accuracy curve (AUC) at the specified centimeter error thresholds. Our main tables use AUC@1/5/10, and AUC@M, where M is AUC@10 on monocular pixels visible in exactly one input view. All methods are evaluated on the same 512×384 image canvas. For MV-dVRK, the same numeric thresholds are measured in millimeters.

Evaluation datasets. We evaluate multi-view 3D reconstruction on five benchmarks. ScanNet++ [47] is an **in-domain** indoor benchmark, as it is included in the training data of Scaffold3D and all evaluated baselines (except MVSAnywhere, OMNI-DC, and Marigold-DC). ETH3D [31] and Tanks&Temples [18] test **cross-domain** generalization to real captures with different scene statistics. We also evaluate on two **out-of-domain** datasets: 4D-DRESS [42], which contains clothed humans with masked white background, and MV-dVRK [4], which contains synthetic robotic surgical scenes. Table B1 summarizes view count, GT coverage, view-overlap, and SfM-scaffold support statistics.

Baselines. We compare four groups of methods. (1) Feed-forward predictors include DUSt3R [41], MASt3R [19], MASt3R-SfM [8], VGGT [39], Pi3/Pi3X [43], Pow3R [14], and MapAnything [17].

Table 1: **Multi-view 3D reconstruction.** Scaffold3D achieves the highest all-pixel AUC@10 on the **in-domain** and **cross-domain** benchmarks and on **out-of-domain** MV-dVRK, while remaining competitive on 4D-DRESS, where the SfM scaffold already covers most evaluated pixels. On monocular pixels, it exceeds the strongest GGPT variant by 11.3 AUC points on ETH3D and 2.9 on Tanks&Temples. Each dataset block reports AUC@1/5/10/M. IT, PT, and D indicate image tokens, point tokens, and depth-map conditioning. Geometry-conditioned baselines use the same dataset-specific SfM scaffold. ‡Comparisons added after submission, shown in grey and excluded from the primary ranking marks. MASt3R† is the conditioning-free pass of our predictor (same network and weights). Rows labeled "+ SfM overwrite" replace completed depths with scaffold depths wherever available, preserving the scaffold's high-precision geometry. §MVSAnywhere uses the scaffold cameras and depth ranges derived from the scaffold points, but does not consume point tokens. Best and second-best results per metric and dataset are **bold** and underlined.

Method	IT	PT	D	ScanNet++$_4$ AUC@1/5/10/M *M pixels: 24.0%*				ETH3D$_4$ AUC@1/5/10/M *M pixels: 26.5%*				Tanks&Temples$_4$ AUC@1/5/10/M *M pixels: 10.2%*				4D-DRESS AUC@1/5/10/M *M pixels: 0.9%*				MV-dVRK AUC@1/5/10/M *M pixels: 0.4%*			
Feed-forward predictors																							
VGGT-Omega‡ [39]	✓	✗	✗	3.2	21.9	38.8	26.6	7.0	35.4	51.8	43.1	7.5	30.9	45.7	23.1	34.0	77.5	88.2	71.1	16.1	51.6	68.1	29.1
MASt3R-SfM [8]	✓	✗	✗	12.3	32.8	43.9	12.1	13.9	36.8	49.3	24.4	19.6	41.0	52.9	13.4	56.6	72.0	76.9	27.7	51.4	68.7	75.8	24.7
MASt3R [19]	✓	✗	✗	8.0	33.6	49.8	37.9	6.7	24.5	36.7	23.5	8.5	31.0	45.0	18.3	22.2	52.4	64.7	32.1	10.0	33.1	47.4	13.4
DUSt3R [41]	✓	✗	✗	9.4	34.9	50.4	39.6	5.1	21.1	33.3	21.7	5.8	24.3	37.5	14.2	7.8	31.6	46.1	19.1	14.0	41.6	56.4	19.1
Pow3R [14]	✓	✗	✗	10.2	36.7	52.4	42.9	6.3	20.4	31.8	18.4	5.2	22.1	34.8	12.1	7.6	32.2	45.9	20.6	9.8	32.4	47.1	15.5
VGGT [39]	✓	✗	✗	8.2	38.5	55.6	44.7	6.4	26.3	41.4	28.5	6.6	29.8	43.3	20.3	13.1	54.2	73.3	45.9	9.7	33.7	49.1	15.0
MapAnything [17]	✓	✗	✗	11.7	45.2	61.0	53.2	1.7	11.2	21.2	17.1	2.9	17.2	30.2	11.8	3.7	22.3	40.2	25.8	2.6	16.5	30.8	5.1
MASt3R†	✓	✗	✗	17.1	50.2	65.6	54.9	6.2	26.4	40.6	28.1	9.2	30.8	44.4	19.5	3.0	15.0	26.5	7.8	26.0	56.5	70.1	28.7
Pi3X [43]	✓	✗	✗	15.0	50.8	66.5	56.6	9.5	33.3	47.3	35.6	5.4	28.9	44.8	22.3	3.7	15.4	26.5	17.5	4.4	21.0	35.3	8.3
Pi3 [43]	✓	✗	✗	16.7	52.0	68.0	**57.8**	6.2	29.2	44.1	31.5	5.6	31.0	46.6	24.0	10.5	50.4	71.4	49.4	18.3	51.9	67.0	27.2
Monocular depth completion																							
OMNI-DC [54]	✓	✗	✓	19.3	41.6	54.2	20.1	17.0	36.8	48.6	12.6	16.3	37.3	49.4	10.9	50.0	78.8	87.7	44.7	50.2	75.0	83.5	32.7
OMNI-DC [54] + SfM overwrite	✓	✗	✓	22.5	44.5	56.1	20.0	25.6	47.0	57.1	12.7	26.1	48.8	59.5	11.0	74.8	88.4	92.7	45.5	66.1	82.9	88.0	32.9
Marigold-DC [35]	✓	✗	✓	19.7	42.9	55.3	27.8	17.4	39.1	52.2	20.9	8.7	28.1	43.4	15.7	69.8	87.8	92.6	51.5	64.3	83.2	88.9	40.7
Marigold-DC [35] + SfM overwrite	✓	✗	✓	21.8	44.4	56.3	27.7	25.6	48.5	59.6	20.8	25.2	47.1	58.1	15.3	76.2	89.1	93.1	50.4	68.6	84.7	89.7	40.4
CAPA (VGGT)‡ [16]	✓	✗	✓	22.6	52.1	66.4	55.0	20.0	46.3	60.0	41.1	13.2	38.0	52.7	27.9	38.7	68.1	79.8	55.9	45.5	72.6	82.9	39.8
CAPA (VGGT)‡ [16] + SfM overwrite	✓	✗	✓	25.1	54.2	67.9	56.3	27.3	54.0	66.0	41.3	27.6	49.3	60.9	27.6	75.3	88.0	92.4	55.4	68.8	84.5	89.8	41.8
Point-token propagation																							
GGPT (VGGT) [5]	✗	✓	✗	16.2	48.0	63.2	44.4	14.0	42.2	56.6	26.7	14.8	42.6	55.1	17.2	69.3	86.4	91.9	49.7	56.6	76.3	83.0	27.5
GGPT (Pi3X) [5]	✗	✓	✗	16.9	50.0	65.3	51.0	15.5	45.4	60.0	29.0	17.0	43.7	56.1	20.8	63.7	82.6	87.6	21.8	55.7	74.3	81.0	23.0
GGPT (Pi3) [5]	✗	✓	✗	14.0	50.2	66.1	54.9	7.4	33.3	50.0	27.3	10.4	36.6	51.9	23.4	72.4	87.8	92.7	53.4	61.7	81.4	87.9	35.8
GGPT (MASt3R†) [5]	✗	✓	✗	25.0	55.8	69.4	53.5	15.6	44.6	58.6	21.6	21.6	47.9	60.1	17.3	49.0	59.9	64.4	11.2	**68.8**	84.1	89.1	38.8
Geometry-conditioned predictors																							
MVSAnywhere [13]§	✓	✗	✗	10.2	27.5	40.1	23.6	11.3	30.3	41.8	21.5	9.9	29.8	42.9	16.4	30.2	63.1	77.4	**57.9**	48.9	71.0	79.8	31.2
MapAnything [17]	✓	✗	✓	12.4	37.9	50.4	30.9	6.7	26.3	39.5	21.4	9.3	31.1	44.8	15.5	39.0	74.5	82.9	37.3	49.8	72.9	79.2	17.6
Pow3R [14] (SfM-anchored DGA)	✓	✗	✓	23.4	49.4	61.7	42.1	**25.7**	50.9	62.3	27.2	26.1	48.1	59.3	21.0	**76.9**	**89.2**	**93.2**	41.2	65.2	83.5	88.6	36.5
Pow3R [14] (DGA)	✓	✗	✓	20.0	49.1	62.2	42.5	18.7	45.3	58.4	25.7	18.9	43.0	56.0	20.8	59.2	84.6	90.9	39.5	67.6	83.4	88.3	36.7
Pi3X [43]	✓	✗	✓	20.4	54.5	68.5	55.9	12.4	42.1	57.8	**40.9**	11.6	37.7	52.7	23.6	64.7	86.4	91.9	57.6	66.7	84.2	89.5	42.9
Scaffold3D (ours)	✓	✓	✗	**25.7**	**56.5**	**69.6**	56.3	25.6	**53.6**	**66.4**	40.3	**26.4**	**49.2**	**60.9**	**26.3**	74.4	88.6	92.9	51.2	68.4	**85.4**	**90.4**	**44.0**

Our conditioning-free reference, MASt3R†, uses the same learned weights as Scaffold3D but sets $\bar{g}_a = \bar{g}_b = 0$. (2) Monocular depth-completion baselines, OMNI-DC [54] and Marigold-DC [35], complete each view independently from SfM scaffold depth samples. We evaluate both alone and with *SfM overwrite*, which restores scaffold depths at supported pixels to preserve high-precision SfM geometry that completion may alter. Unconditioned two-view predictors merge pairwise outputs through global alignment with estimated cameras. Scaffold3D uses SfM-anchored alignment, as does conditioned Pow3R unless labeled *DGA*. (3) For point-token propagation, we evaluate GGPT [5] with different feed-forward initializers. (4) Geometry-conditioned baselines MVSAnywhere [13], MapAnything, Pi3X, and Pow3R are conditioned on the same dataset-specific SfM scaffold as Scaffold3D. Camera diagnostics are in Table F2. For scaffold construction, RoMaV2 [10] uses the same 512×384 images as the reconstruction models (see Table A6 for full-resolution matching).

4.1 Main Results

Comparison to feed-forward and conditioned baselines. Scaffold3D achieves the highest all-pixel AUC@10 on the three standard benchmarks in Table 1: 69.6 on ScanNet++$_4$, 66.4 on ETH3D$_4$, and 60.9 on Tanks&Temples$_4$. Its gains over point-token propagation are especially clear on monocular pixels, where geometry must be inferred beyond direct multi-view triangulation. Compared with the strongest GGPT variant on each dataset, Scaffold3D improves AUC@M from 29.0 to 40.3 on ETH3D and from 23.4 to 26.3 on Tanks&Temples. On Tanks&Temples, it also exceeds the best feed-forward predictor (24.0) and conditioned Pi3X (23.6). Pi3 and Pi3X lead on in-domain ScanNet++, while conditioned Pi3X reaches 40.9 AUC@M on ETH3D and CAPA reaches 27.9 on Tanks&Temples.

Table 2: **View-count scaling.** We evaluate the same scenes with 4, 8, and 16 input views. Each dataset block reports AUC@10/M. Additional views reduce the fraction of no-overlap pixels. Scaffold3D remains best or comparable as the number of views varies.

Method	ScanNet++						ETH3D						Tanks&Temples					
	All points			Monocular			All points			Monocular			All points			Monocular		
	4	8	16	4	8	16	4	8	16	4	8	16	4	8	16	4	8	16
Monocular pixels (%)	–	–	–	*24.0*	*9.1*	*2.9*	–	–	–	*26.5*	*12.9*	*3.9*	–	–	–	*10.2*	*4.7*	*2.7*
Feed-forward predictors																		
DUSt3R [41]	50.4	49.6	51.0	39.6	36.7	38.9	33.3	27.6	25.9	21.7	14.5	9.2	37.5	34.6	30.9	14.2	6.1	5.5
MASt3R [19]	49.8	49.8	50.0	37.9	40.7	39.3	36.7	31.1	28.9	23.5	15.3	19.0	45.0	43.4	40.8	18.3	8.9	9.8
Pow3R [14]	52.4	52.3	53.6	42.9	39.6	42.2	31.8	27.4	26.6	18.4	13.3	12.8	34.8	32.5	28.6	12.1	7.0	7.8
MapAnything [17]	61.0	60.0	61.3	53.2	53.1	52.1	21.2	17.0	13.5	17.1	12.4	9.9	30.2	26.9	24.9	11.8	8.0	5.2
VGGT [39]	55.6	56.6	57.9	44.7	48.1	44.4	41.4	37.1	31.2	28.5	19.9	19.8	43.3	44.5	41.1	20.3	14.7	12.9
VGGT-Omega‡ [39]	38.8	38.2	37.6	26.6	25.3	28.1	51.8	48.3	42.9	43.1	35.3	29.7	45.7	45.6	41.3	23.1	18.6	13.9
Pi3 [43]	68.0	70.2	72.4	**57.8**	61.7	61.8	44.1	35.7	34.9	31.5	21.2	20.3	46.6	47.0	44.0	24.0	18.1	15.8
Pi3X [43]	66.5	70.1	72.0	56.6	**62.1**	61.7	47.3	42.3	38.9	35.6	27.5	23.0	44.8	44.6	42.6	22.3	18.6	16.4
MASt3R†	65.6	66.4	66.8	54.9	57.0	52.8	40.6	36.4	34.8	28.1	19.8	15.9	44.4	44.7	39.7	19.5	10.2	13.2
Point-token propagation																		
GGPT (VGGT) [5]	63.2	70.2	74.8	44.4	51.6	48.6	56.6	58.0	62.8	26.7	20.0	22.8	55.1	61.2	59.7	17.2	17.2	17.7
GGPT (Pi3) [5]	66.1	71.0	74.6	54.9	58.7	58.6	50.0	49.5	52.4	27.3	22.6	28.4	51.9	57.4	57.9	23.4	19.6	19.7
GGPT (Pi3X) [5]	65.3	71.2	75.4	51.0	53.3	52.0	60.0	60.6	63.8	29.0	24.0	25.3	56.1	61.7	61.1	20.8	20.6	21.7
GGPT (MASt3R†) [5]	69.4	74.9	78.8	53.5	55.7	54.6	58.6	60.2	66.9	21.6	19.9	25.9	60.1	67.6	67.0	17.3	17.0	19.2
Geometry-conditioned predictors																		
MVSAnywhere [13]	40.1	51.5	58.1	23.6	31.5	35.7	41.8	48.4	52.5	21.5	21.5	27.5	42.9	47.2	47.7	16.4	17.7	20.2
MapAnything [17]	50.4	63.5	69.3	30.9	35.6	40.8	39.5	43.5	40.0	21.4	16.0	15.3	44.8	53.5	51.0	15.5	15.1	14.4
Pi3X [43]	68.5	73.8	77.4	55.9	61.7	61.3	57.8	56.1	59.2	**40.9**	34.3	**37.3**	52.7	56.7	55.3	23.6	**23.0**	21.8
Pow3R [14]	61.7	72.3	77.1	42.1	49.8	50.4	62.3	64.9	**67.6**	27.2	24.3	28.0	59.3	66.9	66.4	21.0	15.2	20.0
Scaffold3D (ours)	**69.6**	**75.7**	**79.0**	56.3	60.6	**62.7**	**66.4**	**67.4**	67.0	40.3	**36.1**	36.6	**60.9**	**69.5**	**68.2**	**26.3**	21.3	**25.1**

On out-of-domain data, behavior depends on the quality and coverage of the SfM scaffold. Scaffold3D is best on most MV-dVRK metrics, including AUC@M, AUC@10, and AUC@5. On 4D-DRESS, only 0.9% of pixels are monocular and 92.6% have scaffold support (Table B1). Its clothed humans are absent from our training data, and conditioning lifts our backbone from 26.5 to 92.9 AUC@10, close to the best result (93.2).

Effect of the number of views. Table 2 evaluates the same scenes with 4, 8, and 16 views. More views reduce the monocular fraction, e.g., from 26.5% to 3.9% on ETH3D. Scaffold3D is best on all-pixel AUC@10 in eight of nine dataset/view-count combinations (second on ETH3D with 16 views) and best or second-best on monocular pixels in seven of nine. On ETH3D, its all-pixel AUC@10 lead over the best GGPT variant narrows from 6.4 to 0.1 points between 4 and 16 views. At 16 views, Scaffold3D exceeds every GGPT variant on monocular pixels across all datasets.

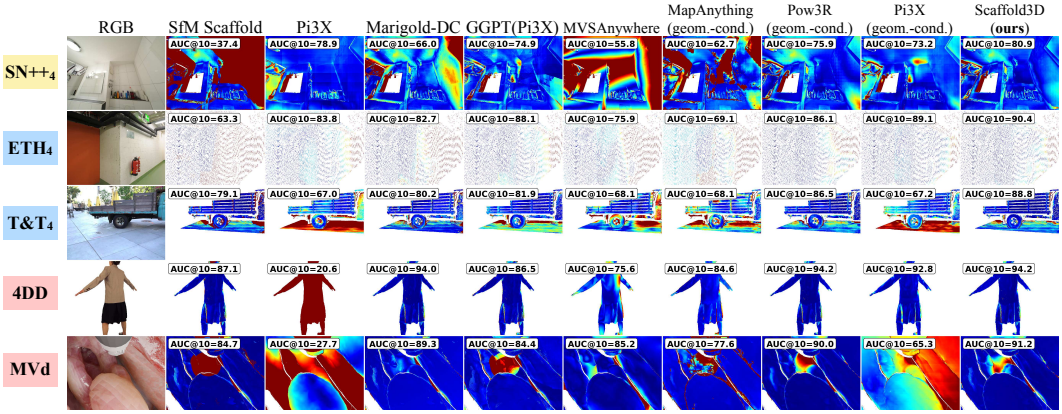

Figure 3: **Qualitative reconstruction comparison.** Rows show individual scenes and their first input view. Scaffold-conditioned methods share the scaffold. Reconstruction tiles report AUC@10.

Qualitative comparison. Figure 3 compares representative reconstructions across in-domain, cross-domain, and out-of-domain scenes. These examples show that Scaffold3D preserves accurate

Table 3: **SfM conditioning ablation.** We compare simple, training-free ways to use the SfM scaffold with MASt3R† (the conditioning-free pass of Scaffold3D) against our learned Scaffold3D predictor. Each dataset block reports AUC@10 overall, outside SfM support (NS), on monocular pixels (M), and on non-monocular pixels (NM). M/NM labels are approximated from GT reprojection. N/A is used for methods not using SfM. DGA denotes DUSt3R global alignment. Unanchored DGA removes the final SfM anchoring term. See Table B1 for the overlap and SfM-support ratios.

Method	ScanNet++$_4$				ETH3D$_4$				Tanks&Temples$_4$				4D-DRESS				MV-dVRK			
	AUC@10	NS	M	NM	AUC@10	NS	M	NM	AUC@10	NS	M	NM	AUC@10	NS	M	NM	AUC@10	NS	M	NM
MASt3R† feed-forward	65.6	N/A	54.9	68.9	40.6	N/A	28.1	44.1	44.4	N/A	19.5	46.2	26.5	N/A	7.8	26.9	70.1	N/A	28.7	70.3
SfM scaffold	25.2	0.0	0.2	33.1	42.0	0.0	0.3	56.7	52.0	0.0	0.4	56.2	87.6	0.0	4.8	88.7	83.1	0.0	11.3	83.4
GGPT (MASt3R†)	69.1	64.9	53.7	**74.0**	56.3	33.2	21.7	67.1	58.5	29.4	18.4	62.0	83.5	23.2	13.6	84.4	89.1	56.5	38.8	89.4
Naive depth completion (MASt3R†)	68.4	64.1	**57.1**	72.1	61.4	39.7	33.6	69.9	58.5	27.2	20.9	61.8	90.4	34.0	27.0	91.1	88.1	45.9	33.6	88.3
MASt3R† + SfM-anchored DGA	68.0	63.8	56.4	71.7	61.9	39.8	32.5	70.5	59.8	31.0	24.0	62.8	86.7	45.8	34.2	87.3	85.7	49.0	34.5	86.0
Scaffold3D (Unanchored DGA)	69.1	65.3	54.5	73.7	61.2	47.8	39.2	68.0	55.2	34.9	26.2	57.9	61.5	55.3	39.3	61.8	88.4	**81.0**	**44.3**	88.6
Scaffold3D (ours)	**69.6**	**65.8**	56.3	**74.0**	**66.4**	**49.1**	**40.3**	**74.7**	**60.9**	**35.5**	**26.3**	**64.1**	**92.9**	**72.7**	**51.2**	**93.5**	**90.4**	66.4	44.0	**90.6**

scaffold-supported geometry while improving dense prediction beyond directly triangulated regions. Our project page provides interactive comparisons with selected baselines for all scenes in Table 1.

4.2 Ablations and Analysis

How should the scaffold be used? Table 3 compares ways to use the same SfM scaffold with our MASt3R† backbone. The scaffold-only row evaluates the triangulated points themselves and therefore has 0.0 AUC@NS outside scaffold support. Naive depth completion uses scaffold depth samples to complete each view independently, reaching 39.7 AUC@NS on ETH3D$_4$. SfM-anchored DGA uses the scaffold only after dense prediction, fitting dense depth maps to fixed scaffold cameras, and reaches 39.8. GGPT propagates scaffold point tokens without refining image-token features and reaches 33.2. Scaffold3D reaches 49.1, improving over all three alternatives. It also improves over the strongest of these alternatives by 0.9 AUC@NS points on ScanNet++$_4$ and 4.5 on Tanks&Temples$_4$. On this shared backbone, conditioning the image-token decoder on SfM points is more effective for completing geometry beyond scaffold support than these alternatives.

Additional cross-backbone comparisons in Table A8 show that simple scaffold fusion with stronger predictors trained on broader data mixtures, such as Pi3, can approach Scaffold3D's accuracy and even exceed it on 4D-DRESS, where our unconditioned backbone is substantially weaker, highlighting that scaffold conditioning and the backbone's learned prior play complementary roles.

Role of SfM anchoring. The unanchored-DGA ablation isolates the SfM anchoring term's contribution in Eq. (12). Anchoring improves all-pixel AUC@10 on every dataset, supporting its role in combining pairwise predictions into a reconstruction consistent with the scaffold. On MV-dVRK, removing the anchor improves accuracy outside scaffold support, but reduces overall accuracy.

Architecture and training choices. The architecture ablations (Table A1) support fusing point tokens with image tokens by addition at the decoder input and letting them exchange information through self-attention. Pretraining with ground-truth scaffolds also improves reconstruction accuracy. To compare our conditioning interface with existing alternatives, we fine-tune Pow3R on SfM scaffolds and use our SfM-anchored DGA, obtaining performance comparable to Scaffold3D (Table A3), suggesting that several conditioning interfaces can learn to use the scaffold effectively.

4.3 Scaffold Quality, Reliability, and Scale

How sensitive is Scaffold3D to the scaffold source? Table 4 (left) rebuilds the scaffold from eleven sources, rerunning the full construction from each matcher's correspondences. In mean AUC@10, every source improves over the conditioning-free pass (49.4), and seven of the eleven sources improve over the per-dataset best ranked feed-forward baseline (64.6). Very sparse scaffolds already help: SuperPoint+LightGlue [6, 24] covers 0.3–2.6% of pixels and lifts the mean to 58.3, and LoMa reaches 67.0 at similar coverage. RoMa v2 at full resolution reaches the highest mean accuracy, 76.8.

When does conditioning help, and when is a feed-forward model preferable? Conditioning improves the conditioning-free pass on all ten evaluation splits, by 3.3 to 66.4 AUC@10 (Tables 1 and E1). On the hand-object and human datasets DexYCB and EgoBody, however, the scaffold covers only 5.5–13.4% and 31.0% of pixels (30.8–92.6% on the main benchmarks), and Scaffold3D stays below Pi3, whose prior there is 12.0 to 19.6 AUC@10 stronger than our backbone's on DexYCB.

Table 4: **Scaffold source and view count.** *Left:* AUC@10 of Scaffold3D with scaffolds from eleven sources, using resized and center-cropped 512×384 images (C) or full-resolution 4:3 center crops (F) for matching. Cov.: scaffold pixel coverage range; best FF: per-dataset best feed-forward baseline ranked in Table 1; cyan: default configurations. *Right:* AUC@10, pair counts, and total runtime (s) on ETH3D for both pair graphs. Details in Tables D1, D2 and F1.

Scaffold source	Res.	Cov. (%)	**Mean**	$SN++_4$	$ETH3D_4$	$T\&T_4$	4D-DR.	MV-dVRK
None (MASt3R†)	–	–	49.4	65.6	40.6	44.4	26.5	70.1
None (best FF)	–	–	64.6	68.0	49.3	52.9	76.9	75.8
SP+LG [6, 24]	–	0.3–2.6	58.3	64.6	42.7	49.4	68.2	66.5
UFM [51]	C	19.6–83.5	61.2	58.9	49.7	49.5	77.9	70.0
UFM [51]	F	20.7–83.1	62.1	59.9	52.1	49.8	77.5	71.4
LoMa [27]	C	0.2–2.7	64.3	65.8	47.9	50.8	81.1	75.8
RoMa [9]	F	4.9–70.2	65.4	58.1	55.7	55.9	83.7	73.4
MASt3R [19]	F	13.0–36.7	66.5	64.5	50.5	52.4	83.7	81.4
RoMa [9]	C	6.8–72.1	66.5	60.8	54.7	56.9	86.2	74.0
MASt3R [19]	C	13.1–36.9	66.8	64.9	50.8	52.8	82.0	83.4
LoMa [27]	F	0.3–3.5	67.0	66.1	53.1	51.2	86.2	78.3
RoMa v2 [10]	C	30.8–92.6	76.0	**69.3**	66.1	**60.5**	93.2	**90.7**
RoMa v2 [10]	F	32.4–92.4	**76.8**	69.2	**71.0**	60.0	**93.5**	90.4

Views		4	8	16	32	50
AUC@10 on ETH3D						
Scaffold3D (sparse)		**66.4**	66.9	**69.9**	**67.8**	**60.3**
Scaffold3D (complete)		**66.4**	**67.4**	67.0	51.3	24.0
Pi3X [43]		47.3	42.3	38.9	32.7	32.9
Pi3 [43]		44.1	35.7	34.9	26.5	25.3
VGGT [39]		41.4	37.1	31.2	26.0	26.1
MapAnything [17]		21.2	17.0	13.5	11.6	11.2
Pairs and total runtime						
Sparse graph	Pairs	6	17	41	89	143
	Time (s)	35.9	57.6	152.2	220.5	361.6
Complete graph	Pairs	6	28	120	496	1225
	Time (s)	35.9	86.0	312.9	786.0	1876.5

With ground-truth cameras and point coordinates at the scaffold support (Table E3), Scaffold3D overtakes Pi3 on EgoBody-Exo (45.3 vs. 41.8) but not on DexYCB, supporting the complementary role of a strong backbone prior when scaffold coverage is sparse. A plain feed-forward model is thus preferable when its domain prior is much stronger and the scaffold covers little of the image.

Scaling to more views. On ETH3D with 8 to 50 views (Table 4, right), the sparse graph cuts total runtime by 33% to 81%. It matches the complete graph at 8 views (-0.5 AUC@10) and exceeds it at 16, 32, and 50 views ($+2.9$, $+16.5$, $+36.3$), consistent with instability in global alignment when many low-overlap pairs are included. With the sparse graph, Scaffold3D outperforms the feed-forward baselines at every view count from 4 to 50, and we use it beyond 16 views.

4.4 Discussion and Limitations

Scaffold3D assumes that the SfM scaffold is reliable but incomplete. We consider this realistic with modern dense matchers and SfM pipelines [10, 28, 29]: when matching becomes hard, the scaffold mostly loses coverage rather than accuracy on our main benchmarks, and conditioning still helps (Table 4). However, on textureless surfaces under extreme perspective change, we observed that matches can drift along the epipolar line, survive filtering, and triangulate to incorrect depths. Scaffold3D has no explicit mechanism for such points, and they influence the prediction. Conditioning still improves the same backbone there, but a feed-forward model with a stronger prior for the domain (such as Pi3) starts higher. These limitations motivate conditioning on per-point uncertainty to reduce reliance on unreliable observations and extending scaffold conditioning to stronger multi-view backbones. The method is also not real time (Table D1).

5 Conclusion

Scaffold3D uses SfM cameras and triangulated points as explicit conditioning signals, injects their observations as patch-aligned point tokens into image-token pointmap prediction, and fuses pairwise outputs with SfM-anchored global alignment. It achieves leading reconstruction accuracy on standard sparse-view benchmarks and MV-dVRK, with particularly clear gains over point-token propagation in low- and no-overlap regions. Together, these results show how explicit SfM geometry and learned image priors can work together to complete dense geometry beyond triangulated support.

Acknowledgements

We thank Luigi Piccinelli, Yiming Wang, Sergey Prokudin, Nikita Karaev, and Xudong Jiang for valuable discussions and support. This study was conducted within the national "Proficiency" research project (No. PFFS-21-19), funded by the Swiss Innovation Agency Innosuisse in 2021 as one of 15 flagship initiatives. This work was supported by the Swiss AI Initiative under project IDs A43, A44, and A129, with computational resources provided by the Swiss National Supercomputing Centre on the Alps infrastructure. This work was also supported by the ETH Future Computing Laboratory. Yutong Chen is supported by a Google PhD Fellowship. The authors declare no competing interests.

References

[1] L. Bartolomei, F. Tosi, M. Poggi, and S. Mattoccia. Stereo anywhere: Robust zero-shot deep stereo matching even where either stereo or mono fail. In *CVPR*, 2025.

[2] G. Baruch, Z. Chen, A. Dehghan, Y. Feigin, P. Fu, T. Gebauer, D. Kurz, T. Dimry, B. Joffe, A. Schwartz, and E. Shulman. ARKitScenes: A diverse real-world dataset for 3D indoor scene understanding using mobile RGB-D data. In *NeurIPS*, 2021.

[3] Y. Cabon, N. Murray, and M. Humenberger. Virtual KITTI 2. *arXiv preprint arXiv:2001.10773*, 2020.

[4] G. Caccianiga, S. Prokudin, Y. Chen, B. Javot, R. L'Orsa, O. B. Aladağ, Y. Sharon, J. Rolinger, I. Capobianco, A. Deguet, S. Tang, and K. J. Kuchenbecker. MV-dVRK: A multi-viewpoint benchmark for spatial surgical perception. arXiv preprint arXiv:2609.02717, 2026.

[5] Y. Chen, Y. Wang, X. Zhang, S. Prokudin, and S. Tang. GGPT: Geometry-grounded point transformer. In *CVPR*, 2026.

[6] D. DeTone, T. Malisiewicz, and A. Rabinovich. SuperPoint: Self-supervised interest point detection and description. *CVPRW*, 2018.

[7] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.

[8] B. P. Duisterhof, L. Zust, P. Weinzaepfel, V. Leroy, Y. Cabon, and J. Revaud. MASt3R-SfM: a fully-integrated solution for unconstrained structure-from-motion. In *3DV*, 2025.

[9] J. Edstedt, Q. Sun, G. Bökman, M. Wadenbäck, and M. Felsberg. RoMa: Robust Dense Feature Matching. In *CVPR*, 2024.

[10] J. Edstedt, D. Nordström, Y. Zhang, G. Bökman, J. Astermark, V. Larsson, A. Heyden, F. Kahl, M. Wadenbäck, and M. Felsberg. RoMa v2: Harder Better Faster Denser Feature Matching. In *ECCV*, 2026.

[11] X. Fang, J. Gao, Z. Wang, Z. Chen, X. Ren, J. Lyu, Q. Ren, Z. Yang, X. Yang, Y. Yan, and C. Lyu. Dens3R: A foundation model for 3D geometry prediction. In *ICLR*, 2026.

[12] H. Guo, H. Zhu, S. Peng, H. Lin, Y. Yan, T. Xie, W. Wang, X. Zhou, and H. Bao. Multi-view reconstruction via SfM-guided monocular depth estimation. In *CVPR*, 2025.

[13] S. Izquierdo, M. Sayed, M. Firman, G. Garcia-Hernando, D. Turmukhambetov, J. Civera, O. Mac Aodha, G. J. Brostow, and J. Watson. MVSAnywhere: Zero shot multi-view stereo. In *CVPR*, 2025.

[14] W. Jang, P. Weinzaepfel, V. Leroy, L. Agapito, and J. Revaud. Pow3R: Empowering unconstrained 3D reconstruction with camera and scene priors. In *CVPR*, 2025.

[15] C. Jeong, I. Bae, J.-H. Park, and H.-G. Jeon. Test-time prompt tuning for zero-shot depth completion. In *ICCV*, 2025.

[16] B. Ke, Q. Zhou, J. Huang, X. Ren, T. Shen, K. Schindler, L. Leal-Taixé, and S. Huang. Depth completion as parameter-efficient test-time adaptation. *arXiv preprint arXiv:2602.14751*, 2026.

[17] N. Keetha, N. Müller, J. Schönberger, L. Porzi, Y. Zhang, T. Fischer, A. Knapitsch, D. Zauss, E. Weber, N. Antunes, J. Luiten, M. Lopez-Antequera, S. R. Bulò, C. Richardt, D. Ramanan, S. Scherer, and P. Kontschieder. MapAnything: Universal feed-forward metric 3D reconstruction. In *3DV*, 2026.

[18] A. Knapitsch, J. Park, Q.-Y. Zhou, and V. Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM ToG*, 2017.

[19] V. Leroy, Y. Cabon, and J. Revaud. Grounding image matching in 3D with MASt3R. In *ECCV*, 2024.

[20] Z. Li and N. Snavely. MegaDepth: Learning single-view depth prediction from internet photos. In *CVPR*, 2018.

[21] H. Lin, S. Peng, J. Chen, S. Peng, J. Sun, M. Liu, H. Bao, J. Feng, X. Zhou, and B. Kang. Prompting Depth Anything for 4K resolution accurate metric depth estimation. In *CVPR*, 2025.

[22] H. Lin, S. Chen, J. H. Liew, D. Y. Chen, Z. Li, Y. Zhao, S. Peng, H. Guo, X. Zhou, G. Shi, J. Feng, and B. Kang. Depth Anything 3: Recovering the visual space from any views. In *ICLR*, 2026.

[23] P. Lindenberger, P.-E. Sarlin, V. Larsson, and M. Pollefeys. Pixel-Perfect Structure-from-Motion with Featuremetric Refinement. In *ICCV*, 2021.

[24] P. Lindenberger, P.-E. Sarlin, and M. Pollefeys. LightGlue: Local feature matching at light speed. In *ICCV*, 2023.

[25] J. Lu, T. Huang, P. Li, Z. Dou, C. Lin, Z. Cui, Z. Dong, S.-K. Yeung, W. Wang, and Y. Liu. Align3R: Aligned monocular depth estimation for dynamic videos. In *CVPR*, 2025.

[26] N. Mayer, E. Ilg, P. Hausser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *CVPR*, 2016.

[27] D. Nordström, J. Edstedt, G. Bökman, J. Astermark, A. Heyden, V. Larsson, M. Wadenbäck, M. Felsberg, and F. Kahl. LoMa: Local feature matching revisited. In *ECCV*, 2026.

[28] Z. Pataki, P.-E. Sarlin, J. L. Schönberger, and M. Pollefeys. MP-SfM: Monocular Surface Priors for Robust Structure-from-Motion. In *CVPR*, 2025.

[29] Z. Pataki, P.-E. Sarlin, and M. Pollefeys. VidMap: Exploiting temporal structure for video-based structure-from-motion. In *ECCV*, 2026.

[30] J. Reizenstein, R. Shapovalov, P. Henzler, L. Sbordone, P. Labatut, and D. Novotny. Common objects in 3D: Large-scale learning and evaluation of real-life 3D category reconstruction. In *ICCV*, 2021.

[31] T. Schöps, J. L. Schönberger, S. Galliani, T. Sattler, K. Schindler, M. Pollefeys, and A. Geiger. A multi-view stereo benchmark with high-resolution images and multi-camera videos. In *CVPR*, 2017.

[32] J. L. Schönberger and J.-M. Frahm. Structure-from-motion revisited. In *CVPR*, 2016.

[33] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine, V. Vasudevan, W. Han, J. Ngiam, H. Zhao, A. Timofeev, S. Ettinger, M. Krivokon, A. Gao, A. Joshi, Y. Zhang, J. Shlens, Z. Chen, and D. Anguelov. Scalability in perception for autonomous driving: Waymo open dataset. In *CVPR*, 2020.

[34] A. Szot, A. Clegg, E. Undersander, E. Wijmans, Y. Zhao, J. Turner, N. Maestre, M. Mukadam, D. Chaplot, O. Maksymets, A. Gokaslan, V. Vondrus, S. Dharur, F. Meier, W. Galuba, A. Chang, Z. Kira, V. Koltun, J. Malik, M. Savva, and D. Batra. Habitat 2.0: Training home assistants to rearrange their habitat. In *NeurIPS*, 2021.

[35] M. Viola, K. Qu, N. Metzger, B. Ke, A. Becker, K. Schindler, and A. Obukhov. Marigold-DC: Zero-shot monocular depth completion with guided diffusion. In *ICCV*, 2025.

[36] F. Wang, Q. Xu, Y.-S. Ong, and M. Pollefeys. Lightweight and accurate multi-view stereo with confidence-aware diffusion model. *TPAMI*, 2025.

[37] H. Wang and L. Agapito. 3D reconstruction with spatial memory. In *3DV*, 2025.

[38] J. Wang, N. Karaev, C. Rupprecht, and D. Novotny. VGGSfM: Visual geometry grounded deep structure from motion. In *CVPR*, 2024.

[39] J. Wang, M. Chen, N. Karaev, A. Vedaldi, C. Rupprecht, and D. Novotny. VGGT: Visual geometry grounded transformer. In *CVPR*, 2025.

[40] Q. Wang*, Y. Zhang*, A. Holynski, A. A. Efros, and A. Kanazawa. Continuous 3D perception model with persistent state. In *CVPR*, 2025.

[41] S. Wang, V. Leroy, Y. Cabon, B. Chidlovskii, and J. Revaud. DUSt3R: Geometric 3D vision made easy. In *CVPR*, 2024.

[42] W. Wang, H.-I. Ho, C. Guo, B. Rong, A. Grigorev, J. Song, J. J. Zarate, and O. Hilliges. 4D-DRESS: A 4D dataset of real-world human clothing with semantic annotations. In *CVPR*, 2024.

[43] Y. Wang, J. Zhou, H. Zhu, W. Chang, Y. Zhou, Z. Li, J. Chen, J. Pang, C. Shen, and T. He. π^3: Permutation-equivariant visual geometry learning. In *ICLR*, 2026.

[44] Z. Wang, S. Chen, L. Yang, J. Wang, Z. Zhang, H. Zhao, and Z. Zhao. Depth Anything with any prior. *arXiv preprint arXiv:2505.10565*, 2025.

[45] H. Xia, Y. Fu, S. Liu, and X. Wang. RGBD objects in the wild: Scaling real-world 3D object learning from RGB-D videos. *arXiv preprint arXiv:2401.12592*, 2024.

[46] Y. Yao, Z. Luo, S. Li, J. Zhang, Y. Ren, L. Zhou, T. Fang, and L. Quan. BlendedMVS: A large-scale dataset for generalized multi-view stereo networks. In *CVPR*, 2020.

[47] C. Yeshwanth, Y.-C. Liu, M. Nießner, and A. Dai. ScanNet++: A high-fidelity dataset of 3D indoor scenes. In *CVPR*, 2023.

[48] C. Zhang, G. Le Moing, S. Koppula, I. Rocco, L. Momeni, J. Xie, S. Sun, R. Sukthankar, J. K. Barral, R. Hadsell, Z. Ghahramani, A. Zisserman, J. Zhang, and M. S. M. Sajjadi. Efficiently reconstructing dynamic scenes one D4RT at a time. In *CVPR*, 2026.

[49] J. Zhang, C. Herrmann, J. Hur, V. Jampani, T. Darrell, F. Cole, D. Sun, and M.-H. Yang. MonST3R: A simple approach for estimating geometry in the presence of motion. In *ICLR*, 2025.

[50] S. Zhang, J. Wang, Y. Xu, N. Xue, C. Rupprecht, X. Zhou, Y. Shen, and G. Wetzstein. FLARE: Feed-forward geometry, appearance and camera estimation from uncalibrated sparse views. In *CVPR*, 2025.

[51] Y. Zhang, N. Keetha, C. Lyu, B. Jhamb, Y. Chen, Y. Qiu, J. Karhade, S. Jha, Y. Hu, D. Ramanan, S. Scherer, and W. Wang. UFM: A simple path towards unified dense correspondence with flow. In *NeurIPS*, 2025.

[52] Y. Zhou, Y. Wang, J. Zhou, W. Chang, H. Guo, Z. Li, K. Ma, X. Li, Y. Wang, H. Zhu, M. Liu, D. Liu, J. Yang, Z. Fu, J. Chen, C. Shen, J. Pang, K. Zhang, and T. He. OmniWorld: A multi-domain and multi-modal dataset for 4D world modeling. In *ICLR*, 2026.

[53] Y. Zuo and J. Deng. OGNI-DC: Robust depth completion with optimization-guided neural iterations. In *ECCV*, 2024.

[54] Y. Zuo, W. Yang, Z. Ma, and J. Deng. OMNI-DC: Highly robust depth completion with multiresolution depth integration. In *ICCV*, 2025.

A Additional Ablations

Table A1 reports development runs that motivated the final design. Each block is a controlled comparison that isolates one design decision under comparable training conditions.

Table A1: **Model-development ablations.** We report development runs that isolate key choices in Scaffold3D. Each dataset block reports AUC at 1, 5, and 10 cm. The controlled blocks motivate adding point tokens to image tokens, aggregating point tokens so the model can use all available SfM points, and pretraining with GT-overlap points before real SfM. These ablations use earlier training configurations than the final model (e.g., smaller training and data scale, different source of SfM scaffold), but are comparable within each block. Cyan rows indicate the selected design choices.

Variant	ScanNet++$_4$			ETH3D$_4$			T&T$_4$		
	AUC@1	AUC@5	AUC@10	AUC@1	AUC@5	AUC@10	AUC@1	AUC@5	AUC@10
Token fusion (first sweep)									
Append point tokens as extra tokens	20.3	49.5	64.6	5.6	21.3	34.0	4.8	20.3	33.7
Mask image tokens at point locations	15.5	43.7	59.4	6.5	21.9	35.3	4.1	17.5	30.3
Replace image tokens with point tokens	17.6	49.3	65.5	7.4	26.5	40.6	5.9	24.1	38.9
Add point tokens to image tokens	**25.7**	**59.4**	**72.8**	**11.6**	**37.1**	**53.6**	**11.7**	**35.2**	**50.6**
Token fusion (second sweep)									
Replace image tokens with point tokens	27.1	59.4	71.9	12.5	37.0	52.6	14.6	38.9	54.2
Replace image tokens + point-token dropout	25.0	58.8	72.3	12.6	36.8	52.7	13.4	38.3	53.7
Gate image/point tokens	20.8	56.4	71.4	18.1	47.1	62.3	16.4	43.2	58.6
Add point tokens to image tokens	**29.8**	**61.3**	**74.0**	**20.3**	**50.1**	**64.6**	**18.5**	**45.9**	**60.5**
Point-token capacity									
Unaggregated point tokens (768 points)	5.2	31.3	52.3	4.2	30.3	52.4	9.8	46.9	66.8
Patch-aggregated point tokens (all points)	**5.6**	32.0	52.9	4.0	30.9	52.8	10.7	47.0	66.7
Patch aggregation + point self-attention	5.3	**32.6**	**53.6**	**4.9**	**32.8**	**54.1**	**11.5**	**48.8**	**67.9**
Sparse-input curriculum									
Real SfM from start	17.4	48.1	64.2	8.3	28.2	41.8	6.1	23.2	36.6
GT-overlap pretrain → real SfM	**20.3**	**51.9**	**67.2**	**11.5**	**33.7**	**48.3**	**11.7**	**34.0**	**48.9**

Table A2 studies Pi3X fine-tuning under our setup, comparing training duration and recipe choices before and after conditioning on the same SfM scaffold.

Table A2: **Pi3X fine-tuning ablations.** We compare released Pi3X [43] with variants fine-tuned under our SfM-conditioned training setup. The first numeric block reports mean AUC@10 over the five datasets. Each dataset block reports all-pixel AUC@10 for the conditioning-free feed-forward output (FF), conditioning on the same dataset-specific SfM scaffold with predicted cameras (Pred.), and conditioning with the scaffold cameras fixed (Fixed). Thresholds are in centimeters except MV-dVRK, which uses millimeters. Unless stated otherwise, variants use 67K steps, learning rate 10^{-5}, bfloat16, and a frozen image encoder.

Variant	Overall			ScanNet++$_4$			ETH3D$_4$			T&T$_4$			4D-DRESS			MV-dVRK		
	FF	Pred.	Fixed	FF	Pred.	Fixed	FF	Pred.	Fixed	FF	Pred.	Fixed	FF	Pred.	Fixed	FF	Pred.	Fixed
Training length																		
Released Pi3X (no fine-tuning)	44.1	67.4	72.1	66.5	67.1	68.5	**47.3**	**54.5**	57.8	**44.8**	**51.2**	**52.7**	26.5	90.0	91.9	35.3	74.1	89.5
34K steps (1 day)	47.4	68.1	**72.4**	69.2	69.7	71.6	38.8	53.0	**58.6**	40.2	46.2	49.6	35.9	88.8	91.9	**53.0**	83.0	90.2
67K steps (2 days)	51.6	68.1	72.3	70.5	70.5	72.0	39.4	52.5	57.9	39.8	45.2	49.3	55.5	89.6	91.8	52.8	82.5	**90.6**
123K steps (4 days)	**52.8**	**68.7**	72.3	**71.4**	**72.0**	**72.7**	39.2	52.3	57.4	41.9	45.4	48.6	**59.3**	**90.3**	**92.3**	52.3	**83.5**	90.5
Training choices (67K steps)																		
Base recipe	51.6	68.1	72.3	70.5	70.5	72.0	39.4	52.5	**57.9**	39.8	45.2	49.3	55.5	89.6	91.8	52.8	82.5	**90.6**
Learning rate 10^{-4}	45.5	63.5	70.3	61.8	63.5	67.9	32.1	46.1	55.4	33.4	42.3	46.7	47.0	82.1	91.6	52.9	83.7	89.7
Scaffold-quality gating	53.9	68.6	**72.4**	70.3	69.3	71.6	**40.7**	**53.9**	57.7	40.7	**46.2**	**50.3**	59.6	89.1	92.0	58.1	84.3	90.3
GT-scaffold probability 0.1	49.6	67.9	71.9	**70.9**	**71.2**	72.1	**40.7**	50.9	56.8	40.4	45.3	48.7	44.7	89.3	91.7	51.2	82.9	90.2
No confidence loss	52.8	67.8	72.2	**70.9**	70.4	72.1	40.1	52.7	57.7	**40.8**	45.4	49.1	56.1	89.5	92.1	56.1	81.2	90.3
Unfrozen image encoder	**56.1**	68.7	72.2	67.3	70.9	71.7	38.1	52.9	**57.9**	40.1	45.2	48.9	**69.2**	88.9	92.0	**66.0**	85.6	90.4
Conditioning probability 2.5×	53.7	**69.3**	72.2	68.9	70.7	**72.4**	39.5	53.7	57.4	40.5	44.9	48.7	61.8	**90.5**	**92.3**	57.7	**86.8**	90.3

Table A3 compares scaffold-conditioning interfaces using the same SfM inputs and evaluation protocol. The Pow3R-pretrained rows show that our SfM-anchored DGA improves mean AUC@10 over plain DGA. The MASt3R-initialized controls share scaffold construction and SfM-anchored final alignment. Their training recipes differ across model families in encoder freezing and losses; the final Scaffold3D checkpoint additionally uses the staged curriculum.

Table A3: **Comparison of scaffold-conditioning interfaces.** We compare scaffold-conditioning interfaces under the same SfM inputs and evaluation protocol. The first numeric column reports mean AUC@10 over the five dataset blocks. Each dataset block reports AUC at 1, 5, and 10 in the dataset unit: centimeters except MV-dVRK, which uses millimeters. Conditioning input icons denote : scaffold as patchified depth map, : scaffold as 3D point tokens or patchified pointmaps, : scaffold cameras or relative pose, and : scaffold intrinsics/rays. Cyan marks our final Scaffold3D checkpoint trained with the staged curriculum. The pretrained Pow3R using our SfM-anchored alignment improves over plain DGA by 3.9 AUC@10 on ETH3D and 3.3 on Tanks&Temples. Variants trained from the same MASt3R checkpoint suggest three takeaways: (1) in the simplified depth/pointmap-only interface, decoder fusion improves over encoder fusion by 0.5 mean AUC@10 for depth inputs and 1.7 for pointmap inputs, (2) pointmap decoder fusion improves over depth decoder fusion by 0.9 mean AUC@10, and (3) different conditioning mechanisms, including Pow3R-style conditioning and pointmap decoder fusion, achieve comparable performance. These results support SfM-conditioned training and alignment without establishing a uniquely superior conditioning interface. These conclusions are tied to the dataset mixture, training length, hyperparameters, and SfM construction used here.

Variant	Input	Overall	ScanNet++$_4$			ETH3D$_4$			T&T$_4$			4D-DRESS			MV-dVRK		
		AUC@10	AUC@1	AUC@5	AUC@10	AUC@1	AUC@5	AUC@10	AUC@1	AUC@5	AUC@10	AUC@1	AUC@5	AUC@10	AUC@1	AUC@5	AUC@10
Pow3R-pretrained references																	
Pow3R [14] (DGA)		71.2	20.0	49.1	62.2	18.7	45.3	58.4	18.9	43.0	56.0	59.2	84.6	90.9	67.6	83.4	88.3
Pow3R [14]		73.0	23.4	49.4	61.7	25.7	50.9	62.3	26.1	48.1	59.3	**76.9**	89.2	93.2	65.2	83.5	88.6
Pow3R-pretrained + SfM fine-tune		75.6	25.9	56.2	69.1	25.6	52.0	64.5	**27.0**	48.9	59.9	75.3	89.2	**93.4**	69.9	**86.1**	**91.0**
Variants trained from the MASt3R checkpoint																	
Pointmap encoder fusion		74.1	24.7	53.9	67.5	23.0	49.7	62.3	25.3	48.1	59.4	69.8	86.8	91.8	65.4	83.9	89.5
Depth encoder fusion		74.4	24.5	54.2	67.3	25.3	51.1	63.3	26.4	48.8	60.3	71.0	87.3	92.1	62.7	83.1	89.1
Depth decoder fusion		74.9	26.5	57.0	69.9	**25.9**	52.8	65.2	26.9	**49.3**	**60.9**	61.3	84.0	90.3	60.1	82.1	88.4
Pow3R with pointmap decoder fusion		75.4	26.3	55.6	68.9	25.3	52.7	65.5	24.1	47.4	59.2	76.0	88.9	93.1	69.8	85.3	90.3
Pow3R (native: depthmap encoder fusion)		75.7	**27.3**	56.7	69.4	24.4	52.7	66.0	25.2	48.0	59.5	76.3	**89.4**	**93.4**	68.8	85.2	90.3
Pow3R with depthmap decoder fusion		75.8	25.8	55.1	68.9	25.1	**53.7**	**66.6**	26.3	48.3	59.7	75.6	89.2	93.3	**71.6**	85.9	90.6
Pointmap decoder fusion		75.8	25.9	**57.3**	**70.1**	24.7	53.2	65.9	26.6	49.0	60.6	72.2	87.9	92.5	64.9	84.2	89.9
Final checkpoint trained with the staged Scaffold3D curriculum																	
Scaffold3D: pointmap decoder fusion		**76.0**	25.7	56.5	69.6	25.6	53.6	66.4	26.4	49.2	**60.9**	74.4	88.6	92.9	68.4	85.4	90.4

Table A4 evaluates how many SfM points are needed for conditioning. The results saturate with a modest number of points injected into the predictor on most datasets, while the full filtered scaffold remains available for final alignment. The zero-point row supplies no scaffold points to the predictor, while retaining SfM-anchored alignment, isolating the contribution of point conditioning.

Table A4: **Conditioning point-budget sensitivity.** The subsampled-budget rows vary the number of scaffold points injected into Scaffold3D, while final SfM-anchored alignment retains the full filtered scaffold. Each dataset block reports AUC@1, AUC@5, AUC@10, and AUC@M. M denotes AUC@10 on monocular pixels. Performance saturates after a moderate point budget in this subsampling experiment, suggesting diminishing returns from injecting every available scaffold point into the predictor. The separate BA-tracks row (4316 tracks on average) uses only bundle-adjusted tracks and omits the DLT-triangulated scaffold observations.

#Points	ScanNet++$_4$				ETH3D$_4$				T&T$_4$				4D-DRESS				MV-dVRK			
	AUC@1	AUC@5	AUC@10	AUC@M	AUC@1	AUC@5	AUC@10	AUC@M	AUC@1	AUC@5	AUC@10	AUC@M	AUC@1	AUC@5	AUC@10	AUC@M	AUC@1	AUC@5	AUC@10	AUC@M
Subsampled SfM scaffold observations																				
0	24.7	54.8	68.0	56.1	24.2	49.4	62.1	32.1	26.1	48.6	60.0	23.8	56.6	79.4	86.6	34.3	57.1	79.0	85.7	34.5
512	25.7	56.6	69.6	**58.7**	23.2	51.2	64.8	40.3	25.6	48.6	60.2	22.7	74.0	88.4	92.8	51.6	64.6	83.6	89.2	40.9
1024	26.1	56.6	69.5	58.2	23.0	51.3	65.0	40.7	26.4	48.8	60.1	23.5	74.2	88.5	92.9	52.8	65.3	83.7	89.3	41.7
2048	**26.3**	56.7	69.6	57.7	23.3	51.6	65.1	39.8	24.1	48.2	59.9	24.7	**75.1**	**88.7**	**93.0**	53.7	67.1	84.4	89.7	42.2
4096	25.8	56.8	69.7	57.8	22.7	52.0	65.7	40.2	25.5	48.7	60.2	24.8	72.4	88.1	92.7	53.5	69.0	84.9	90.0	42.8
8192	26.1	**56.9**	69.8	57.9	24.5	52.3	65.7	40.6	25.9	49.3	60.9	25.2	73.6	88.4	92.8	**53.9**	68.0	84.8	90.0	43.1
16384	25.5	56.6	69.7	57.1	24.2	52.3	65.9	**40.8**	25.2	48.9	60.4	25.2	73.8	88.5	92.8	52.6	63.9	84.1	89.7	42.8
32768	26.0	**56.9**	**70.0**	57.1	24.1	52.0	65.6	39.6	25.9	49.2	60.6	25.7	74.8	**88.7**	**93.0**	50.8	**69.4**	85.3	90.2	43.6
65536	25.3	56.2	69.3	56.4	23.8	52.6	65.9	40.5	**26.9**	**49.9**	**61.3**	**26.6**	73.8	88.5	92.9	51.1	68.3	85.2	90.2	**44.0**
131072	26.1	56.5	69.6	56.1	**24.6**	**53.3**	**66.3**	40.4	26.6	49.3	60.7	25.9	74.2	88.6	92.9	51.1	65.8	84.7	90.0	**44.0**
all	25.9	56.8	69.9	56.2	24.0	53.1	66.1	40.0	25.8	49.2	60.9	26.1	73.3	88.4	92.8	51.2	**69.4**	**85.6**	**90.5**	43.9
BA tracks only, no DLT triangulation																				
4316	26.0	56.3	69.3	57.0	24.0	52.6	65.5	**40.8**	26.6	48.9	60.5	24.6	66.7	85.1	90.2	37.1	64.8	83.0	88.5	35.8

Table A5 reports the percentage of image patches that receive a point token under the default evaluation protocol. Across evaluated scenes, this ranges from 8.9% to 93.1%.

Table A5: **Point-token coverage.** Percentage of the 768 image tokens per view (16×16 patches on the 512×384 canvas) that receive a point token, i.e., contain at least one scaffold observation. Statistics are over scenes; each scene value is the mean over its views.

Dataset	Scenes	Min	25%	50%	75%	Max	Mean
ScanNet++$_4$	29	8.9	54.0	60.6	64.5	78.6	58.1
ETH3D$_4$	26	37.5	56.8	69.9	76.9	88.9	67.1
Tanks&Temples$_4$	32	35.5	70.0	74.4	79.9	87.8	72.1
4D-DRESS	24	20.6	22.6	25.1	27.3	29.8	25.0
MV-dVRK	23	63.2	81.2	84.5	89.8	93.1	84.2

Table A6 checks whether using full-resolution 4:3 center crops for RoMaV2 matching changes the scaffold-conditioned comparison under the same evaluation protocol. The main effect is a stronger ETH3D scaffold, while the relative trends remain similar across methods.

Table A7 replaces scaffold points, camera poses, and intrinsics with ground truth at the original scaffold-supported pixels where ground truth is valid. This provides internally consistent geometry without adding scaffold support. Scaffold3D gains 10.0, 11.0, and 19.9 AUC@10 points on ScanNet++, ETH3D, and Tanks&Temples, respectively, indicating room for improvement through better scaffold construction. OMNI-DC also benefits substantially, with scaffold-depth overwrite raising its Tanks&Temples AUC@10 from 77.5 to 82.5.

Table A8 fuses multiple feed-forward backbones with the same scaffold after prediction. Scaffold3D exceeds every post-hoc variant of its own backbone on all five datasets in AUC@10, with the smallest margin of 0.5 over GGPT on ScanNet++. Variant C on Pi3 is the strongest post-hoc baseline overall and comes within 0.2 to 1.8 AUC@10 of Scaffold3D, but Scaffold3D leads it outside scaffold support on Tanks&Temples, 4D-DRESS, and MV-dVRK. Variant B on Pi3 and VGGT slightly exceeds Scaffold3D on 4D-DRESS (93.5 and 93.2 vs. 92.9), a domain where both backbones are much stronger than ours (71.4 and 73.3 vs. 26.5 AUC@10 before fusion). Native conditioning in Pi3X adds less to overall accuracy than per-view scale alignment and scaffold completion on the three standard benchmarks, although it is more accurate outside scaffold support on ETH3D and Tanks&Temples.

Table A6: **Full-resolution RoMaV2 scaffold matching.** We rerun scaffold-conditioned methods using full-resolution 4:3 center crops as RoMaV2 matcher inputs. Matches are expressed on the 512×384 grid used for BA/DLT, conditioning, and evaluation. After DLT, visible track observations are rasterized to this grid by rounding their 2D coordinates and averaging collisions. Each cell reports AUC@10 with the change relative to the corresponding 512×384 RoMaV2 scaffold run. Depth-completion rows include SfM overwrite. Full-resolution matching strengthens the scaffold most clearly on ETH3D, but the relative trends remain similar, suggesting that the main conclusions are not an artifact of low-resolution matching.

Method	ScanNet++$_4$	ETH3D$_4$	Tanks&Temples$_4$	4D-DRESS	MV-dVRK
Monocular depth completion					
OMNI-DC [54]	55.6 (-0.5)	63.6 (+6.4)	59.1 (-0.4)	93.6 (+1.0)	87.3 (-0.7)
Marigold-DC [35]	56.8 (+0.5)	66.0 (+6.4)	58.1 (+0.0)	**93.8** (+0.7)	88.8 (-0.9)
Point-token propagation					
GGPT (VGGT) [5]	63.5 (+0.3)	58.7 (+2.1)	53.8 (-1.2)	92.7 (+0.8)	83.6 (+0.6)
GGPT (Pi3) [5]	67.1 (+1.0)	50.5 (+0.5)	51.2 (-0.7)	93.1 (+0.4)	88.3 (+0.4)
GGPT (Pi3X) [5]	65.7 (+0.3)	63.0 (+3.0)	54.8 (-1.3)	89.3 (+1.7)	77.9 (-3.1)
GGPT (MASt3R†) [5]	69.1 (+0.0)	60.9 (+4.7)	57.9 (-0.5)	82.7 (-0.7)	88.7 (-0.4)
Geometry-conditioned predictors					
MVSAnywhere [13]§	40.9 (+0.8)	41.6 (-0.2)	42.6 (-0.2)	77.2 (-0.3)	79.1 (-0.6)
MapAnything [17]	51.2 (+0.8)	42.7 (+3.2)	44.4 (-0.5)	82.5 (-0.4)	78.9 (-0.2)
Pow3R [14]	61.9 (+0.2)	68.0 (+5.7)	58.5 (-0.8)	**93.8** (+0.6)	87.9 (-0.7)
Scaffold3D (ours)	**69.5** (-0.1)	**71.0** (+4.5)	**60.0** (-0.9)	93.5 (+0.6)	**90.5** (+0.1)

Table A7: **Ground-truth scaffold geometry.** We replace scaffold points, camera poses, and intrinsics with ground truth, retaining the original scaffold-support pixels where ground truth is valid. Each cell reports AUC@10 and its change from the corresponding estimated-scaffold evaluation. All methods use the same dataset-specific scaffold support, including GGPT. OMNI-DC is reported with and without scaffold-depth overwrite.

Method	ScanNet++$_4$	ETH3D$_4$	Tanks&Temples$_4$	4D-DRESS	MV-dVRK
Monocular depth completion					
OMNI-DC [54]	64.3 (+10.1)	72.0 (+23.4)	77.5 (+28.1)	94.4 (+6.7)	91.1 (+7.5)
+ SfM overwrite	67.5 (+11.4)	74.7 (+17.6)	**82.5** (+23.0)	**98.7** (+6.1)	97.0 (+9.0)
Point-token propagation					
GGPT (VGGT) [5]	68.8 (+5.2)	56.3 (+1.9)	62.7 (+8.8)	95.4 (+4.8)	89.4 (+5.0)
GGPT (Pi3) [5]	71.4 (+3.9)	46.5 (+0.1)	57.7 (+8.4)	95.6 (+3.1)	93.2 (+4.8)
GGPT (Pi3X) [5]	74.6 (+6.2)	58.1 (+3.3)	64.4 (+8.9)	90.3 (+3.1)	86.1 (+2.8)
GGPT (MASt3R†) [5]	76.6 (+7.5)	66.0 (+9.8)	73.6 (+15.1)	86.3 (+2.8)	94.2 (+5.0)
Geometry-conditioned predictors					
MVSAnywhere [13]§	42.1 (+2.1)	42.5 (+0.7)	42.2 (-0.7)	78.1 (+0.6)	80.7 (+1.0)
MapAnything [17]	56.2 (+5.7)	26.6 (-12.9)	42.1 (-2.7)	85.2 (+2.3)	82.0 (+2.8)
Pi3X [43]	78.4 (+9.9)	64.1 (+6.2)	58.9 (+6.2)	96.4 (+4.4)	95.2 (+5.7)
Pow3R [14]	74.6 (+12.9)	76.3 (+14.0)	81.2 (+21.8)	98.6 (+5.4)	96.7 (+8.1)
Scaffold3D (ours)	**79.6** (+10.0)	**77.4** (+11.0)	80.7 (+19.9)	98.0 (+5.1)	**97.3** (+7.0)

All reported VGGT-Omega results use the original `vggt_omega_1b_512.pt` checkpoint. The authors recommend a replacement for benchmarking following a potential, unconfirmed training-data contamination concern.[1]

Table A8: **Post-hoc alignment of feed-forward predictions to the scaffold.** Each backbone is fused with the same dataset-specific SfM scaffold as Scaffold3D by training-free variants: (A) rigidly align the scaffold to the prediction with Umeyama and paste the scaffold over the prediction; (B) complete the scaffold depth maps with predicted depth aligned by scale and shift in inverse depth, after rigidly aligning the prediction to the scaffold; (C) as B, but with a single per-view median scale in depth, without a shift or rigid pre-alignment; (GGPT) point-token propagation [5] from the backbone's prediction. "Conditioned" feeds the scaffold through the model's own conditioning interface (Pi3X and MapAnything rows of Table 1). Each dataset block reports AUC@10 on all pixels and outside SfM support (AUC@NS). For unconditioned rows, NS is evaluated with the same scaffold mask. Here GGPT uses the shared scaffold, whereas in Table 1 each GGPT variant constructs its scaffold with the named initializer.

Backbone	Variant	ScanNet++$_4$		ETH3D$_4$		T&T$_4$		4D-DRESS		MV-dVRK	
		AUC@10	NS	AUC@10	NS	AUC@10	NS	AUC@10	NS	AUC@10	NS
SfM scaffold	–	25.2	0.0	42.0	0.0	52.0	0.0	87.6	0.2	83.1	0.0
Pi3 [43]	–	68.0	**65.8**	44.1	37.1	46.6	31.0	71.4	57.2	67.0	47.4
	A	68.3	65.1	57.8	33.1	58.4	25.6	92.0	58.4	88.0	47.2
	B	65.9	60.8	63.1	41.8	59.6	31.1	**93.5**	**77.7**	88.1	48.9
	C	69.4	65.7	65.7	48.9	60.0	31.7	92.6	66.6	88.6	53.0
	GGPT	67.6	65.2	46.4	35.4	49.2	30.7	92.4	71.6	88.4	58.1
Pi3X [43]	–	66.5	64.5	47.3	39.8	44.8	29.1	26.5	17.2	35.3	17.8
	A	67.3	63.5	59.4	36.7	58.9	28.7	88.7	12.0	84.5	15.7
	B	66.4	61.7	62.1	40.2	59.3	30.3	92.5	67.5	86.5	34.3
	C	69.4	65.7	64.7	46.7	60.0	31.6	91.1	49.1	86.7	38.2
	GGPT	68.3	65.0	54.8	35.7	55.5	32.9	87.2	42.7	83.3	41.5
	conditioned	68.5	65.0	57.8	47.5	52.7	33.8	91.9	76.6	89.5	65.1
VGGT [39]	–	55.6	52.7	41.4	33.9	43.3	28.4	73.3	52.4	49.1	26.8
	A	59.0	51.9	57.0	29.9	56.7	20.3	91.9	53.9	85.7	26.4
	B	62.2	55.4	61.7	39.9	59.8	32.3	93.2	71.8	86.4	33.9
	C	59.2	51.2	61.6	39.8	59.1	30.7	90.9	41.1	86.2	33.5
	GGPT	63.6	58.5	54.4	36.2	53.8	30.7	90.6	67.3	84.4	43.3
VGGT-Omega [39]	–	39.3	38.1	52.0	47.2	45.7	30.0	88.2	79.5	67.8	53.4
	A	46.8	35.2	62.0	41.0	57.6	22.9	93.5	79.1	88.5	52.0
	B	57.5	48.8	66.2	48.0	60.3	32.8	93.6	81.1	88.3	50.8
	C	62.9	56.3	70.6	57.4	60.4	33.7	93.7	80.1	89.0	57.3
	GGPT	57.3	49.3	62.0	45.6	59.2	31.8	93.7	82.1	89.6	64.1
MapAnything [17]	–	61.0	58.4	21.2	18.3	30.2	16.1	40.2	25.0	30.8	15.8
	A	63.4	57.7	49.4	14.2	55.2	14.9	89.7	27.0	84.4	14.5
	B	24.7	4.6	46.1	7.3	52.8	2.7	90.2	31.6	83.6	6.4
	C	65.3	59.5	56.2	28.1	56.8	20.3	90.5	37.3	85.0	21.0
	GGPT	63.5	58.1	51.8	23.1	56.6	22.8	89.2	43.3	83.4	32.5
	conditioned	50.4	42.0	39.5	25.8	44.8	22.0	82.9	51.3	79.2	28.0
MASt3R†	–	65.6	62.3	40.6	32.5	44.4	27.0	26.5	12.6	70.1	43.2
	A	64.3	61.3	53.6	23.6	55.5	13.7	88.6	13.1	87.3	43.2
	B	64.6	58.8	58.0	32.1	58.6	26.2	89.8	27.5	88.6	50.8
	C	68.4	64.1	61.4	39.7	58.5	27.2	90.4	34.0	88.1	45.9
	GGPT	69.1	64.9	56.3	33.2	58.5	29.4	83.5	23.2	89.1	56.5
Scaffold3D (ours)	conditioned	**69.6**	**65.8**	**66.4**	**49.1**	**60.9**	**35.5**	92.9	72.7	**90.4**	**66.4**

Table A9 tests whether keeping the SfM cameras fixed during alignment limits reconstruction accuracy. Initializing the SfM-anchored alignment from them and refining intrinsics and extrinsics changes AUC@10 by at most 0.5.

[1] `https://github.com/facebookresearch/vggt-omega/blob/48b23c8ce72c9a8fcf987f0d7f43d8e4870759f6/reproduction.md`

Table A9: **Fixed vs. refined scaffold cameras.** AUC@10 of Scaffold3D when the SfM-anchored global alignment keeps the SfM cameras fixed (default) or is initialized from them and refines intrinsics and extrinsics. The better value per dataset is **bold**.

Cameras	ScanNet++$_4$	ETH3D$_4$	T&T$_4$	MV-dVRK
Fixed SfM cameras	**69.6**	**66.4**	**60.9**	90.4
Refined	69.4	65.9	**60.9**	**90.6**

Table B1: **Dataset support statistics.** Views denotes the number of input views. GT is the fraction of pixels covered by the evaluation ground truth. The view-count columns split GT-covered pixels by the estimated number of views they are visible in. S and NS denote pixels with and without SfM-scaffold support, respectively. 4D-DRESS and MV-dVRK have few monocular pixels and high SfM-scaffold coverage, so they leave less room for gains from geometry propagation beyond SfM support. DexYCB-4A/B/C use camera subsets 1–4, 3–6, and 5–8, respectively; DexYCB-8 uses all eight views. DexYCB and EgoBody have much lower scaffold support due to difficult matching and triangulation. **Black** marks the easiest extremes in overlap and SfM-scaffold coverage, whereas **red** marks the most difficult extremes.

Dataset	Views	GT	Visible in					Coverage	
			1	>1	2	3	4	≥ 4	S	NS
ScanNet++$_4$	4	96.6	24.0	76.0	39.7	27.1	9.2	9.2	30.8	69.2
ScanNet++$_8$	8	96.5	9.1	90.9	19.8	22.6	21.7	48.5	78.8	21.2
ScanNet++$_{16}$	16	96.3	2.9	97.1	6.0	8.4	10.4	82.7	90.1	9.9
ETH3D$_4$	4	22.6	**26.5**	73.5	30.7	32.5	10.4	10.4	52.0	48.0
ETH3D$_8$	8	21.5	12.9	87.1	18.7	22.0	17.9	46.4	83.3	16.7
ETH3D$_{16}$	16	21.4	3.9	96.1	9.6	13.8	13.8	72.7	92.4	7.6
Tanks&Temples$_4$	4	34.6	10.2	89.8	22.6	28.0	39.2	39.2	71.0	29.0
Tanks&Temples$_8$	8	36.4	4.7	95.3	7.1	10.6	12.6	77.6	93.5	6.5
Tanks&Temples$_{16}$	16	34.6	2.7	97.3	4.6	5.3	5.3	87.4	**95.7**	4.3
4D-DRESS	4–12	17.4	0.9	99.1	4.4	9.9	16.8	84.8	92.6	7.4
MV-dVRK	4–12	83.5	0.4	**99.6**	13.2	17.5	17.5	68.8	88.7	11.3
DexYCB-4A	4	56.2	19.6	80.4	32.4	32.3	15.7	15.7	5.6	94.4
DexYCB-4B	4	69.2	21.6	78.4	31.9	38.3	8.2	8.2	5.5	**94.5**
DexYCB-4C	4	75.5	13.2	86.8	40.2	31.5	15.1	15.1	13.4	86.6
DexYCB-8	8	66.6	7.2	92.8	11.6	13.7	18.9	67.5	7.7	92.3
EgoBody-Exo	3/5	11.7	11.4	88.6	19.5	30.0	16.6	39.1	31.0	69.0

B Dataset Details

Table B1 reports how much of each dataset is covered by ground truth, how often pixels are visible in one or multiple input views, and how much of the evaluation region is supported by the SfM scaffold. These statistics explain several trends in the main paper. ScanNet++, ETH3D, and Tanks&Temples contain substantial monocular or no-scaffold regions in the four-view setting, leaving room for learned geometry propagation. In contrast, 4D-DRESS and MV-dVRK have high scaffold coverage and few monocular pixels, while DexYCB and EgoBody expose failure modes of our scaffold construction.

C Implementation Details

Image resolution and cropping. The training preprocessing uses landscape sizes of 512×384, 512×336, 512×288, 512×240, and 512×160 pixels (width × height). For evaluation, we resize and center-crop input images to a shared 512×384 canvas, applying the same transformation to depth maps and updating camera intrinsics accordingly. This can remove peripheral content from images with other aspect ratios. All reported reconstruction comparisons use this common evaluation protocol. The full-resolution matching experiment in Table A6 changes the images used to construct the scaffold, not the pointmap predictor's input resolution. Because development experiments showed reduced

Table B2: **Training data mixture for the final fine-tuning stage.** Dataset selection and approximate ratios are mainly inspired by MASt3R [19]. The table reports the number of preprocessed training samples per source dataset after filtering and the input-view range sampled during preprocessing.

Dataset	Samples	Share (%)	Views
BlendedMVS [46]	27,324	15.2	3–6
MegaDepth [20]	27,267	15.2	3–6
ARKitScenes [2]	21,656	12.1	3–6
OmniWorld-Game [52]	20,480	11.4	3–6
ScanNet++ v2 [47]	18,122	10.1	3–6
Waymo [33]	17,719	9.9	3–6
Habitat [34]	16,959	9.4	3–5
Co3D-v2 [30]	11,445	6.4	3–6
WildRGBD [45]	9,091	5.1	3–6
StaticThings3D [26]	9,089	5.1	3–6
Virtual KITTI 2 [3]	464	0.3	3–6
Total	**179,616**	**100.0**	3–6

accuracy at other input resolutions, we recommend evaluating our released checkpoint at 512×384 pixels, with broader resolution and aspect-ratio training, as used in Pi3 and MapAnything [17, 43], remaining a possible route to better generalization. Here, C denotes the shared 512×384 input, while F retains the source-image pixel scale after a 4:3 center crop. For reference, retained sequences from ScanNet++, ETH3D, Tanks&Temples, 4D-DRESS, and MV-dVRK yield F crops of 1453×1090, 1695×1271, 1233×925, 1280×960, and 1920×1440 pixels, respectively. These are examples, not fixed dataset-wide resolutions, and individual matchers may resize their inputs internally.

Scaffold filtering. The default filtering thresholds are as follows. For bundle adjustment, the matching-score threshold is 0.6 and the cycle-error threshold is 2 pixels. For DLT, these thresholds are 0.1 and 4 pixels. Tracks must be observed in at least two views. The epipolar-error and mean reprojection-error thresholds are both 4 pixels, and at least one observing-camera pair must have a triangulation angle above 3°. We then retain points whose reprojection score $1/(1+e^{\mathrm{reproj}})$ exceeds 0.6, cycle-consistency score $1/(1+e^{\mathrm{cycle}})$ exceeds 0.7, and mean RoMaV2 matching score exceeds 0.7, where the errors are averaged over visible track observations. On 4D-DRESS and MV-dVRK, we omit the additional score-based filtering because development experiments showed that conditioning on these scaffolds improved reconstruction accuracy. We initialize the scaffold from MASt3R†, except on 4D-DRESS, where we use Pi3 because camera estimation by our conditioning-free predictor largely fails on this dataset (pose AUC@5° of 0.3, compared with 36.8 for Pi3 in Table F2). Bundle adjustment and DLT otherwise follow the GGPT pipeline [5].

Input-depth filtering. After scale alignment, we discard scaffold points with nonfinite coordinates or depths at most 10^{-8} in the reference camera's frame. We also discard depths above 20 times the 99th percentile of finite positive unconditional depths pooled across the pair. For each view separately, we then discard depths above 50 times the median of its remaining scaffold depths. If any points are removed, we recompute the median depth ratio from the retained points and update the scale if the result is finite and positive. We then apply the same filters again.

Global alignment. We set $\lambda_{\mathrm{anc}} = 3$ and optimize for 300 iterations, using Adam with a cosine learning-rate schedule from 10^{-2} to 10^{-6}.

Architecture and training. The SfM-conditioned pointmap predictor follows the MASt3R-Large image model configuration [19]: patch size 16, a 24-layer encoder with width 1024 and 16 heads, two 12-layer decoders with width 768 and 12 heads, and DPT-style dense pointmap prediction heads. In point MLP ϕ_{pt}, we use a four-layer ReLU MLP and a three-layer ReLU MLP with width 512 to encode 3D coordinates and within-patch 2D offset separately, then a four-layer ReLU MLP to project both features into a single 768-dimensional feature. The patch point feature sequence is processed by four PointBlock blocks with 12-head self-attention. We freeze the MASt3R image encoder throughout training. We train the remaining parameters using the four-stage curriculum in Table C1, with MP-SfM scaffolds in the earlier mixed stages and GGPT-pipeline scaffolds in the final stage. For computational reasons, we precompute the SfM scaffolds for all training samples before

training starts. During the final fine-tuning stage, we apply point-token dropout, randomly dropping 5–72% of the point tokens of each view. The final training stage uses batch size 2 per GPU on 8 GPUs. We train and evaluate on NVIDIA GH200 nodes with four 96GB GPUs per node; the training runs that produced the final Scaffold3D checkpoint used less than 1K GPU-hours, while the full model-development and evaluation effort, including scaffold precomputation, baseline evaluations, ablations, and preliminary or failed runs, used on the order of tens of thousands of GPU-hours. We optimize trainable parameters with AdamW ($\beta_1 = 0.9$, $\beta_2 = 0.95$, weight decay 0.05, no weight decay for biases and normalization layers). The learning rate is linearly warmed up to $2 \cdot 10^{-5}$ over 1.2 epochs and then follows a half-cycle cosine decay to $5 \cdot 10^{-7}$.

Table C1: **Training curriculum.** GT denotes ground-truth scaffolds sampled from multi-view-overlap regions. Percentages give the probability of using GT rather than the listed SfM source. Sample and update counts for stages 1–3 are approximate and derived from the training configurations. Only stage 4 uses the exact training mixture in Table B2.

Stage	Scaffold sampling	Samples per epoch	Global batch	Epochs	Optimizer updates
1	95% GT / MP-SfM	≈200K	16	3	≈37.5K
2	GT only	≈200K	32	10	≈62.5K
3	50% GT / MP-SfM	≈200K	16	3	≈37.5K
4	40% GT / GGPT-SfM	179,616	16	6	≈67K

D Runtime

Table D1 breaks the inference time into scaffold construction, the two network forwards, and SfM-anchored global alignment, from 2 to 50 views and for both pair graphs. Scaffold construction is unchanged between the two graphs, retaining the same initialization and all-pairs matching. The network passes are relatively small compared with the final alignment at higher view counts. With the complete graph, the pairwise stages grow quadratically with the number of views; the sparse graph keeps 17, 41, 89, and 143 of 28, 120, 496, and 1225 pairs at 8, 16, 32, and 50 views and cuts total runtime by 33% to 81%. Table D2 reports AUC@1/5/10 for both graphs. Beyond 16 views, the complete graph's many low-overlap pairs destabilize the global alignment (on 7 of 16 scenes at 32 views), while the sparse graph stays stable and outperforms the feed-forward baselines at every view count from 4 to 50 views and every metric.

Table D1: **Runtime breakdown.** Wall-clock time per stage for Scaffold3D on one ETH3D scene per view count, in seconds. Total is the sum of the four stages. The sparse graph connects each view to its neighbor in input order plus two hub views to all views. At 2 and 4 views it keeps every pair. These timings use Pi3 to initialize the scaffold, so no initial pairwise DGA is required. The scaffold stage includes this initialization and matches all pairs in both graphs. Parentheses give the change with respect to the complete graph. SfM-anchored DGA accounts for the largest share of runtime at higher view counts.

Views	Graph	Pairs	SfM scaffold	Initial forward	Conditioned forward	SfM-anchored DGA	Total
2	complete = sparse	1	4.3	0.1	0.1	4.3	8.8
4	complete = sparse	6	16.3	0.7	0.9	18.0	35.9
8	complete	28	26.1	3.2	4.4	52.3	86.0
8	sparse	17 (−39%)	26.1	1.9	2.2	27.4	57.6 (−33%)
16	complete	120	61.2	10.5	22.2	219.0	312.9
16	sparse	41 (−66%)	61.2	3.7	7.8	79.5	152.2 (−51%)
32	complete	496	71.3	43.6	114.1	557.0	786.0
32	sparse	89 (−82%)	71.3	7.6	21.5	120.1	220.5 (−72%)
50	complete	1225	128.8	109.5	277.2	1361.0	1876.5
50	sparse	143 (−88%)	128.8	12.8	33.2	186.8	361.6 (−81%)

Table D2: **Accuracy with more views.** AUC@1/5/10 on the full ETH3D set at each view count. At 4 views the sparse graph keeps every pair, so both graphs are identical.

Method	4 views			8 views			16 views			32 views			50 views		
	@1	@5	@10	@1	@5	@10	@1	@5	@10	@1	@5	@10	@1	@5	@10
Scaffold3D (sparse graph)	**25.6**	**53.6**	**66.4**	**29.2**	55.4	66.9	**29.3**	**58.1**	**69.9**	**28.8**	**57.1**	**67.8**	**26.6**	**50.8**	**60.3**
Scaffold3D (complete graph)	**25.6**	**53.6**	**66.4**	29.0	**55.7**	**67.4**	28.2	55.7	67.0	21.0	43.0	51.3	9.6	19.8	24.0
Pi3X [43]	9.5	33.3	47.3	5.6	27.1	42.3	3.7	23.5	38.9	3.4	19.5	32.7	3.3	19.8	32.9
Pi3 [43]	6.2	29.2	44.1	3.9	21.5	35.7	2.4	19.7	34.9	2.1	14.8	26.5	2.3	14.4	25.3
VGGT [39]	6.4	26.3	41.4	4.7	23.5	37.1	3.8	18.6	31.2	3.4	15.6	26.0	3.5	16.2	26.1
MapAnything [17]	1.7	11.2	21.2	1.2	8.7	17.0	0.8	6.5	13.5	0.8	5.6	11.6	0.7	5.5	11.2

E Additional Evaluation Results

Table E1 evaluates Scaffold3D on DexYCB and EgoBody, which contain hand-object interactions and human-centric scenes. Despite limited and partly inaccurate scaffold support, conditioning improves AUC@10 over our conditioning-free backbone by 3.3–8.3 points on all five splits, although Pi3 remains more accurate. Pi3X benefits from conditioning on three splits and loses accuracy on two. Table E2 reports camera accuracy, and Fig. E1 shows representative reconstructions.

Examining the retained DexYCB matches revealed an egregious failure case involving textureless surfaces and extreme viewpoint changes: incorrect correspondences remained close to the epipolar lines and therefore survived epipolar filtering. Our diagnostics found median displacements of 2 to 6 px from the correct correspondences, predominantly along the epipolar line, leading to incorrect triangulated depths.

To separate camera errors from errors in the triangulated points, we substitute ground-truth cameras and points, separately and together (Table E3). On DexYCB, holding the cameras at their ground-truth values and replacing the triangulated points with ground-truth points improves AUC@10 by up to 7.5 points, confirming that triangulation errors limit reconstruction accuracy. On EgoBody-Exo, correcting the cameras alone or the points alone is enough for Scaffold3D to outperform both Pi3 and Pi3X. On DexYCB, however, correcting both still leaves a gap to Pi3. This supports the importance of a strong backbone prior when scaffold coverage remains sparse, even if the available geometry is accurate.

Table E1: **Additional out-of-domain datasets.** We report the same AUC@1/5/10/M format as Table 1 on hand-object and human-scene datasets. These datasets test reconstruction with limited and partly inaccurate SfM support. Conditioning improves Scaffold3D over its conditioning-free pass (MASt3R†) in AUC@10 on every split, although Pi3 remains more accurate. Pi3X benefits from conditioning on three of the five splits. DexYCB-4A/B/C use camera subsets 1–4, 3–6, and 5–8, respectively; DexYCB-8 uses all eight views. EgoBody-Exo uses 3–5 exocentric cameras.

Method	DexYCB-4A				DexYCB-4B				DexYCB-4C				DexYCB-8				EgoBody-Exo			
	AUC@1/5/10/M				AUC@1/5/10/M				AUC@1/5/10/M				AUC@1/5/10/M				AUC@1/5/10/M			
	M pixels: 19.6%				*M pixels: 21.6%*				*M pixels: 13.2%*				*M pixels: 7.2%*				*M pixels: 11.4%*			
Feed-forward predictors																				
MASt3R-SfM [8]	6.6	16.9	22.7	10.1	13.9	29.9	36.9	36.2	33.9	54.0	61.2	35.7	23.4	38.3	45.1	20.5	2.5	18.2	33.5	16.5
Pow3R [14]	12.0	24.7	30.0	10.4	11.1	34.8	46.8	22.6	6.8	33.5	47.7	13.4	7.4	32.2	43.9	6.1	2.5	18.4	34.3	20.3
DUSt3R [41]	14.1	28.4	35.8	14.6	12.8	37.2	47.5	25.8	9.5	39.8	52.0	17.0	9.6	38.0	50.7	9.6	**3.2**	19.5	34.8	17.8
MapAnything [17]	12.4	43.4	57.9	27.9	17.3	57.1	70.2	41.8	12.6	58.7	72.0	33.0	12.1	54.2	69.0	19.2	2.1	12.4	23.6	21.2
MASt3R†	13.3	46.3	59.3	28.7	20.0	53.5	66.5	43.0	25.6	57.1	68.0	31.3	29.6	61.8	73.0	24.1	2.3	19.0	35.1	15.7
MASt3R [19]	17.4	50.2	64.2	33.3	21.9	53.8	66.3	45.3	31.8	64.8	75.2	37.1	30.8	63.6	75.0	32.6	2.7	19.1	34.9	18.0
VGGT [39]	11.1	48.1	65.4	34.9	28.6	68.0	79.8	58.0	17.8	57.8	73.3	37.2	15.6	59.3	75.0	31.1	2.8	22.2	40.3	29.1
VGGT-Omega‡ [39]	25.7	66.7	78.7	52.1	46.7	77.4	84.9	66.2	32.5	71.9	81.7	46.6	38.5	76.2	84.9	45.6	2.8	18.1	34.9	22.1
Pi3X [43]	27.5	**68.5**	78.8	44.1	15.6	63.9	76.9	58.8	34.5	73.1	82.9	48.1	35.4	75.5	84.8	41.9	3.0	23.4	42.9	**29.3**
Pi3 [43]	**27.6**	68.2	**78.9**	**46.3**	**34.7**	**73.1**	**82.4**	**63.8**	**42.3**	75.9	84.5	51.6	40.9	75.8	85.0	43.0	2.8	23.1	41.8	23.5
Monocular depth completion																				
OMNI-DC [54]	22.8	50.6	62.8	27.2	23.8	52.8	64.8	25.5	30.0	66.3	77.6	38.3	33.7	68.0	78.2	23.3	2.3	16.4	31.4	6.8
Marigold-DC [35]	22.8	51.9	63.7	31.2	26.0	58.2	70.1	38.9	36.0	72.0	81.3	42.8	41.7	72.9	81.3	30.5	2.6	19.3	35.8	11.3
Point-token propagation																				
GGPT (MASt3R†) [5]	17.7	50.7	62.8	28.2	25.1	57.8	69.9	39.0	32.0	67.4	76.6	35.3	38.3	70.8	79.3	25.4	2.7	21.2	39.0	16.6
GGPT (VGGT) [5]	20.5	54.6	68.7	36.4	29.2	64.2	75.8	50.8	37.1	73.3	82.7	45.1	38.2	73.2	82.6	36.6	2.6	**23.5**	**43.3**	28.4
GGPT (Pi3X) [5]	24.3	59.9	71.3	37.9	27.3	62.9	74.9	50.7	41.2	75.9	84.2	50.4	40.7	74.8	83.3	38.3	2.5	22.9	41.8	25.3
GGPT (Pi3) [5]	26.3	61.5	73.1	41.9	32.5	66.7	77.0	53.4	**42.3**	**76.3**	**84.7**	**52.7**	**42.2**	75.8	84.3	42.8	2.0	21.2	40.6	21.8
Geometry-conditioned predictors																				
MVSAnywhere [13]	6.5	20.0	29.6	9.1	14.0	37.6	50.6	30.2	18.9	50.6	64.9	32.1	16.7	43.9	58.2	16.9	2.5	16.0	30.3	11.2
Pow3R [14]	14.6	39.4	51.6	20.8	19.4	44.5	56.5	34.0	29.1	64.1	75.3	40.0	27.0	57.4	68.8	25.7	2.9	20.5	36.6	14.2
MapAnything [17]	17.1	43.5	53.7	20.3	20.7	52.7	64.1	37.2	30.4	59.3	67.8	30.3	36.7	63.7	71.1	19.1	2.2	18.2	34.5	14.9
Pi3X [43]	23.2	61.2	74.3	43.3	29.1	69.3	80.2	62.4	40.1	75.5	83.9	50.4	39.8	**77.2**	**86.1**	**49.3**	2.8	21.2	38.8	21.6
Scaffold3D (ours)	22.1	53.3	65.7	30.7	30.2	61.8	72.5	48.6	34.8	66.9	76.3	39.5	40.2	70.2	79.3	31.7	3.1	21.2	38.4	21.8

Table E2: **Supplemental camera diagnostics.** We report camera-pose AUC at 1, 5, and 10 degrees and F°, the mean field-of-view error in degrees, for the datasets in Table E1. These diagnostics separate camera accuracy from dense reconstruction quality on the hand-object and human-centric stress tests. Lower is better for F°.

Method	DexYCB-4A				DexYCB-4B				DexYCB-4C				DexYCB-8				EgoBody-Exo			
	AUC@1/5/10/F°				AUC@1/5/10/F°				AUC@1/5/10/F°				AUC@1/5/10/F°				AUC@1/5/10/F°			
Feed-forward predictors																				
DUSt3R [41]	0.0	11.0	20.6	1.9	0.0	12.5	34.2	2.6	0.8	21.5	44.4	2.6	0.0	17.3	42.5	2.1	27.5	68.0	82.7	1.1
MASt3R [19]	3.3	54.7	77.2	1.3	1.7	41.5	62.8	1.8	7.5	56.7	76.5	1.5	13.4	64.3	80.5	1.3	38.9	83.4	91.7	0.9
MASt3R-SfM [8]	16.7	22.3	27.5	**0.8**	**23.4**	31.3	32.4	1.1	16.7	82.6	91.4	**0.5**	48.6	58.4	60.6	**0.5**	36.4	83.2	91.6	**0.6**
Pow3R [14]	0.0	5.3	10.3	3.6	0.0	7.0	27.8	2.8	0.0	10.2	36.7	2.2	0.0	10.9	27.9	2.5	26.0	74.9	87.2	1.7
MapAnything [17]	5.8	52.5	74.5	1.3	1.7	41.8	69.3	1.4	4.2	58.7	78.5	1.6	2.7	54.3	76.2	1.5	19.6	66.8	83.1	4.2
VGGT [39]	8.3	59.5	78.9	1.1	9.2	66.7	82.9	**1.0**	2.5	38.0	61.9	2.0	13.4	65.5	82.2	0.7	49.7	83.1	91.5	0.9
Pi3 [43]	**26.7**	**77.8**	**88.9**	1.4	4.2	58.0	78.5	**1.0**	18.3	67.8	83.8	0.7	15.2	72.0	85.8	0.9	48.9	83.8	91.9	0.8
Pi3X [43]	17.5	73.0	86.5	1.0	6.7	57.8	75.8	1.7	15.8	74.3	87.2	1.8	16.4	76.8	88.4	0.7	42.9	85.1	92.5	**0.6**
MASt3R†	5.0	45.3	69.0	1.3	4.2	36.7	63.5	1.1	5.0	48.5	72.1	1.3	6.6	57.7	77.2	1.4	37.0	79.5	89.8	1.5
Monocular depth completion																				
OMNI-DC [54]	12.5	62.7	80.5	0.9	14.2	73.8	86.9	2.1	46.7	83.2	91.6	1.5	**66.8**	**92.6**	**96.3**	1.0	55.7	86.6	93.3	0.8
Marigold-DC [35]	12.5	62.7	80.5	0.9	14.2	73.8	86.9	2.1	46.7	83.2	91.6	1.5	**66.8**	**92.6**	**96.3**	1.0	55.7	86.6	93.3	0.8
Point-token propagation																				
GGPT (VGGT) [5]	10.0	63.3	81.2	1.1	14.2	70.7	85.3	2.2	**52.5**	85.3	92.7	1.5	48.2	88.5	94.2	1.0	54.2	86.7	93.3	0.9
GGPT (Pi3) [5]	14.2	60.7	79.0	1.0	14.2	75.0	87.5	2.1	42.5	83.3	91.7	1.7	52.0	89.5	94.8	1.1	56.0	86.8	93.4	0.8
GGPT (Pi3X) [5]	15.0	66.3	82.1	0.9	15.8	**75.5**	**87.8**	2.1	51.7	**86.0**	**93.0**	1.4	51.2	89.2	94.6	1.0	53.8	86.8	93.4	0.8
GGPT (MASt3R†) [5]	12.5	62.7	80.5	0.9	14.2	73.8	86.9	2.1	46.7	83.2	91.6	1.5	**66.8**	**92.6**	**96.3**	1.0	55.7	86.6	93.3	0.8
Geometry-conditioned predictors																				
MVSAnywhere [13]	12.5	62.7	80.5	0.9	14.2	73.8	86.9	2.1	46.7	83.2	91.6	1.5	**66.8**	**92.6**	**96.3**	1.0	55.7	86.6	93.3	0.8
MapAnything [17]	15.0	65.0	81.6	1.1	10.0	71.8	85.9	2.3	45.0	83.7	91.8	1.6	58.0	90.8	95.4	1.4	**58.3**	**87.1**	**93.6**	0.9
Pi3X [43]	12.5	62.7	80.5	0.9	14.2	73.8	86.9	2.1	46.7	83.2	91.6	1.5	**66.8**	**92.6**	**96.3**	1.0	55.7	86.6	93.3	0.8
Pow3R [14]	12.5	62.7	80.5	0.9	14.2	73.8	86.9	2.1	46.7	83.2	91.6	1.5	**66.8**	**92.6**	**96.3**	1.0	55.7	86.6	93.3	0.8
Scaffold3D (ours)	12.5	62.7	80.5	0.9	14.2	73.8	86.9	2.1	46.7	83.2	91.6	1.5	**66.8**	**92.6**	**96.3**	1.0	55.7	86.6	93.3	0.8

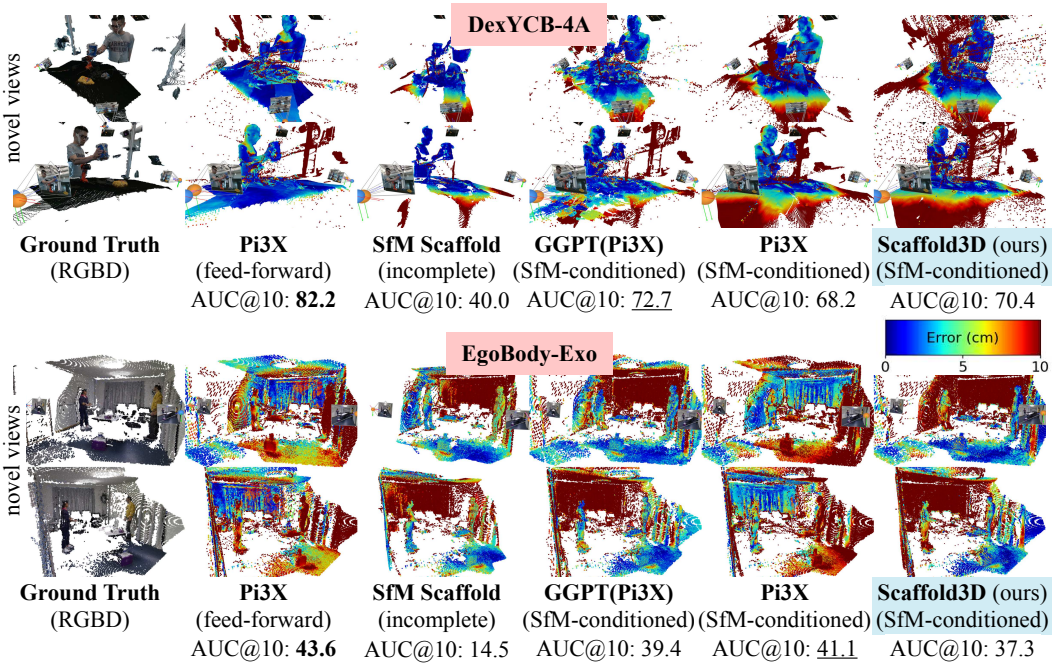

Figure E1: **Failure mode under inaccurate SfM scaffolds.** We visualize DexYCB-4A and EgoBody-Exo examples from two viewpoints each. Columns show ground truth RGB-D geometry, feed-forward Pi3X, the incomplete SfM scaffold, GGPT(Pi3X), Pi3X conditioned on the scaffold, and Scaffold3D. Predicted points are colored by metric 3D error, clipped at 10 cm, where dark blue indicates low error and red indicates high error. These examples illustrate limited scaffold coverage and geometric errors that can affect conditioned predictions. Aggregate reconstruction results, camera diagnostics, and ground-truth-scaffold tests are reported in Tables E1 to E3.

Table E3: **What limits the scaffold on DexYCB and EgoBody.** AUC@10 of Scaffold3D when ground-truth points, ground-truth cameras, or both are substituted into the estimated scaffold. Components not replaced by ground truth remain estimated. Coverage (in parentheses) is the scaffold's pixel coverage. Substituting points does not change the support, so coverage is shown once per camera setting. Correct cameras tighten the epipolar and score filters and reduce coverage, most sharply on EgoBody-Exo. Feed-forward results from Table E1 are given for reference.

	Feed-forward			Scaffold3D			
Dataset	Pi3	Pi3X	MASt3R†	Estimated (cov.)	GT points	GT cameras (cov.)	GT points and cameras
DexYCB-4A	**78.9**	78.8	59.3	65.7 (5.6%)	71.8	67.7 (4.1%)	75.0
DexYCB-4B	**82.4**	76.9	66.5	72.5 (5.5%)	81.0	74.7 (4.0%)	82.2
DexYCB-4C	**84.5**	82.9	68.0	76.3 (13.4%)	77.7	78.7 (13.3%)	79.2
DexYCB-8	**85.0**	84.8	73.0	79.3 (7.7%)	82.1	78.8 (5.7%)	81.9
EgoBody-Exo	41.8	42.9	35.1	38.4 (31.0%)	44.1	44.0 (5.8%)	**45.3**

F Scaffold Quality and Reliability

Scaffold sources. Table F1 gives the per-dataset results of the scaffold-source study in Table 4 together with the quality of each scaffold. For every matcher, the full scaffold construction (track building, bundle adjustment, DLT triangulation, and the same filtering) is rerun from that matcher's correspondences. In this study, accuracy tracks scaffold camera and point accuracy more closely than coverage: the sparse LoMa scaffolds beat the much denser UFM scaffolds. Sparse but accurate scaffolds still help: the LoMa scaffolds cover at most 3.5% of pixels but reach 67.0 mean AUC@10 at full resolution. The mean hides some per-dataset losses: on ScanNet++, the SuperPoint+LightGlue, UFM, RoMa, and MASt3R scaffolds fall below the conditioning-free pass (65.6), and on MV-dVRK, the SuperPoint+LightGlue and UFM (canvas) scaffolds fall below it (70.1).

Table F1: **Scaffold-source study: per-dataset results and scaffold quality.** Extends Table 4 (left). For each dataset, @10 is the AUC@10 of Scaffold3D conditioned on that scaffold. The remaining columns describe the scaffold itself: camera pose AUC@5° (pose), mean field-of-view error in degrees (F°), pixel coverage in % of ground-truth-valid pixels (cov), and Chamfer accuracy and completeness at 5 cm (5 mm for MV-dVRK) after Sim(3) alignment (acc, comp). C and F denote matching on 512×384 inputs and full-resolution 4:3 center crops; SP+LG denotes SuperPoint+LightGlue. The "None" rows use no scaffold, and their pose and F° columns describe the feed-forward cameras. Lower is better for F°. "Best FF" is the per-dataset best feed-forward baseline ranked in Table 1: Pi3 on ScanNet++ and MASt3R-SfM on the other datasets.

Scaffold source	Res.	ScanNet++$_4$						ETH3D$_4$						Tanks&Temples$_4$					
		@10↑	pose↑	F° ↓	cov↑	acc↑	comp↑	@10↑	pose↑	F° ↓	cov↑	acc↑	comp↑	@10↑	pose↑	F° ↓	cov↑	acc↑	comp↑
None (MASt3R†)	–	65.6	74.5	1.5	–	–	–	40.6	43.1	3.0	–	–	–	44.4	61.0	1.5	–	–	–
None (best FF)	–	68.0	84.0	0.7	–	–	–	49.3	76.3	0.5	–	–	–	52.9	90.0	0.7	–	–	–
SP+LG [6, 24]	–	64.6	85.7	0.5	0.3	79.4	10.7	42.7	76.4	1.5	0.4	57.8	7.1	49.4	92.6	**0.3**	0.7	53.1	14.7
UFM [51]	C	58.9	81.1	0.2	19.6	81.6	44.8	49.7	78.6	0.9	36.8	63.4	43.5	49.5	93.6	0.8	48.0	59.1	54.5
UFM [51]	F	59.9	79.9	0.5	20.7	82.6	44.3	52.1	83.6	0.5	40.9	67.6	47.4	49.8	92.9	0.7	47.2	59.0	55.3
LoMa [27]	C	65.8	86.3	0.4	0.2	79.0	13.2	47.9	84.7	0.5	0.2	51.6	6.7	50.8	94.6	0.4	0.5	49.4	13.2
RoMa [9]	F	58.1	69.2	1.2	4.9	89.5	13.5	55.7	85.1	0.5	21.3	77.3	30.3	55.9	91.3	0.5	37.7	**69.7**	49.7
MASt3R [19]	F	64.5	87.4	0.6	13.0	86.5	52.2	50.5	83.3	0.5	22.1	64.2	53.8	52.4	93.1	0.9	34.2	58.9	66.9
RoMa [9]	C	60.8	74.8	0.6	6.8	85.4	17.1	54.7	83.7	0.3	18.5	74.0	28.0	56.9	91.0	0.5	44.2	68.2	54.5
MASt3R [19]	C	64.9	86.1	0.5	13.1	87.2	**52.6**	50.8	83.8	0.4	21.7	65.6	53.4	52.8	92.9	0.8	34.4	59.1	66.9
LoMa [27]	F	66.1	89.8	0.2	0.3	84.5	12.6	53.1	96.0	**0.1**	0.3	62.5	8.3	51.2	94.8	0.4	0.5	55.3	13.9
RoMa v2 [10]	C	**69.3**	92.8	**0.1**	30.8	90.3	50.0	66.1	96.3	0.2	52.0	77.5	56.9	**60.5**	**97.0**	0.5	71.0	65.7	**70.3**
RoMa v2 [10]	F	69.2	**93.6**	**0.1**	32.4	**90.8**	50.8	**71.0**	**98.3**	**0.1**	61.8	**81.6**	**65.1**	60.0	94.3	0.7	71.6	65.2	69.0

Scaffold source	Res.	4D-DRESS						MV-dVRK					
		@10↑	pose↑	F° ↓	cov↑	acc↑	comp↑	@10↑	pose↑	F° ↓	cov↑	acc↑	comp↑
None (MASt3R†)	–	26.5	0.3	29.2	–	–	–	70.1	29.0	2.5	–	–	–
None (best FF)	–	76.9	67.5	1.0	–	–	–	75.8	78.9	1.9	–	–	–
SP+LG [6, 24]	–	68.2	79.7	3.4	2.6	89.0	76.9	66.5	59.8	1.4	0.4	92.3	56.9
UFM [51]	C	77.9	51.5	4.7	83.5	77.4	94.2	70.0	56.9	1.3	72.7	92.1	89.4
UFM [51]	F	77.5	58.0	3.8	83.1	75.2	93.0	71.4	57.9	1.4	72.1	93.6	89.8
LoMa [27]	C	81.1	89.1	0.4	2.7	97.8	93.9	75.8	82.7	1.6	0.6	91.2	83.7
RoMa [9]	F	83.7	84.1	0.5	70.2	97.2	90.0	73.4	62.6	1.8	51.9	96.4	83.2
MASt3R [19]	F	83.7	75.2	2.0	36.7	94.5	94.3	81.4	82.6	1.8	31.2	96.7	93.0
RoMa [9]	C	86.2	85.7	0.6	72.1	98.3	91.5	74.0	67.5	1.0	58.9	95.1	86.5
MASt3R [19]	C	82.0	70.7	1.2	36.9	83.5	93.5	83.4	84.6	0.9	31.3	97.9	94.4
LoMa [27]	F	86.2	95.6	**0.3**	3.5	**99.0**	94.3	78.3	92.8	1.0	0.6	94.4	79.1
RoMa v2 [10]	C	93.2	**96.9**	0.5	92.6	**99.0**	**97.5**	**90.7**	**96.3**	**0.2**	88.7	**98.5**	**97.0**
RoMa v2 [10]	F	**93.5**	**96.9**	**0.3**	92.4	97.9	**97.5**	90.4	96.2	0.3	88.3	98.2	96.8

The main reconstruction tables compare methods under a shared scaffold whenever geometry conditioning is used (except GGPT variants, which construct scaffolds with their named initializers). Table F2 separates camera quality from dense reconstruction quality: many conditioned rows inherit the same cameras, so differences in point accuracy are primarily due to how methods use the scaffold rather than due to different camera estimates.

Evaluation variance. We report a single run per configuration and measured how much the numbers vary. Rerunning the same code and configuration in fresh caches changes AUC@10 on ETH3D$_4$ by 0.003, due to non-deterministic attention kernels. With the network predictions cached, five different random seeds for the latent initialization of the global alignment give identical AUC@10 on ETH3D$_4$ and MV-dVRK and a standard deviation of 0.05 on Tanks&Temples$_4$. An update of the evaluation code that fixes the random seeds per scene for point-token sampling and for the alignment initialization moves a few scenes to a different solution. With the current code, Scaffold3D reads 23.5/52.9/66.1/40.2 (AUC@1/5/10/M) on ETH3D$_4$ and 71.8/86.1/90.7/44.1 on MV-dVRK, against

Table F2: **Camera diagnostics.** We report camera-pose AUC at 1, 5, and 10 degrees and F°, the mean field-of-view error in degrees. Most geometry-conditioned rows inherit the same SfM cameras, so the reconstruction gaps in Table 1 cannot be explained by camera accuracy alone. MapAnything receives the same camera conditioning as Scaffold3D, but nevertheless predicts camera updates rather than keeping the input cameras fixed. Lower is better for F°.

Method	ScanNet++$_4$				ETH3D$_4$				Tanks&Temples$_4$				4D-DRESS				MV-dVRK			
	AUC@1	AUC@5	AUC@10	F°	AUC@1	AUC@5	AUC@10	F°	AUC@1	AUC@5	AUC@10	F°	AUC@1	AUC@5	AUC@10	F°	AUC@1	AUC@5	AUC@10	F°
Feed-forward predictors																				
DUSt3R [41]	5.8	48.6	67.6	2.8	7.7	39.4	58.3	3.4	15.6	51.8	69.3	2.6	0.0	1.1	8.1	27.2	0.7	16.7	36.9	4.1
MASt3R [19]	19.5	53.3	69.1	3.4	12.8	47.3	61.3	3.2	18.8	59.0	74.4	2.7	0.1	10.3	24.0	4.8	0.3	3.9	19.7	19.9
MASt3R-SfM [8]	40.2	77.6	86.1	1.2	52.6	76.3	81.3	0.5	68.8	90.0	93.4	0.7	39.1	67.5	73.2	1.0	58.8	78.9	85.2	1.9
Pow3R [14]	10.9	55.4	69.4	2.1	9.0	40.8	60.5	3.4	18.8	56.8	71.4	1.8	0.0	2.5	9.7	17.8	0.2	4.3	19.6	18.5
MapAnything [17]	46.5	78.3	86.1	0.6	5.8	33.8	51.5	6.0	22.4	69.5	82.3	3.2	0.0	3.9	16.9	30.9	0.0	2.1	8.5	23.7
VGGT [39]	6.9	40.3	60.5	1.4	26.9	65.0	77.4	2.0	65.1	84.0	88.1	0.6	0.0	9.3	28.9	28.4	0.0	9.2	25.3	14.5
Pi3 [43]	60.9	84.0	89.7	0.7	39.1	67.6	78.4	2.3	55.2	81.8	87.6	0.8	1.7	36.8	66.3	5.2	2.3	30.3	51.5	3.4
Pi3X [43]	55.7	81.4	88.2	0.7	47.4	78.8	87.1	0.9	66.7	86.4	90.5	**0.4**	0.1	10.6	33.2	23.9	1.2	3.9	18.5	19.2
MASt3R†	37.9	74.5	83.6	1.5	8.3	43.1	62.2	3.0	29.2	61.0	74.8	1.5	0.0	0.3	1.3	29.2	2.2	29.0	52.5	2.5
Monocular depth completion																				
OMNI-DC [54]	**85.6**	**92.8**	**94.1**	**0.1**	**91.7**	**96.3**	97.8	**0.2**	**92.7**	**97.0**	**97.7**	**0.4**	**93.5**	**96.9**	**97.3**	**0.5**	**91.5**	**96.3**	97.1	**0.2**
Marigold-DC [35]	**85.6**	**92.8**	**94.1**	**0.1**	**91.7**	**96.3**	97.8	**0.2**	**92.7**	**97.0**	**97.7**	**0.4**	**93.5**	**96.9**	**97.3**	**0.5**	**91.5**	**96.3**	97.1	**0.2**
Point-token propagation																				
GGPT (VGGT) [5]	79.9	91.4	93.4	0.3	87.8	94.5	97.0	**0.2**	89.1	91.8	92.8	**0.4**	92.2	93.3	94.4	1.0	76.8	92.0	94.9	0.6
GGPT (Pi3) [5]	79.9	91.4	93.4	0.3	87.8	94.9	97.4	0.3	89.1	93.1	94.2	**0.4**	**93.5**	**96.9**	**97.3**	**0.5**	79.8	94.6	**97.3**	0.4
GGPT (Pi3X) [5]	79.3	91.2	93.5	0.4	88.5	95.9	**97.9**	0.3	91.1	93.4	94.4	0.5	91.2	96.5	97.1	**0.5**	77.8	92.6	95.5	0.4
GGPT (MASt3R†) [5]	**85.6**	**92.8**	**94.1**	**0.1**	**91.7**	**96.3**	97.8	**0.2**	**92.7**	**97.0**	**97.7**	**0.4**	55.2	56.9	57.2	2.8	**91.5**	**96.3**	97.1	**0.2**
Geometry-conditioned predictors																				
MVSAnywhere [13]	**85.6**	**92.8**	**94.1**	**0.1**	**91.7**	**96.3**	97.8	**0.2**	**92.7**	**97.0**	**97.7**	**0.4**	**93.5**	**96.9**	**97.3**	**0.5**	**91.5**	**96.3**	97.1	**0.2**
MapAnything [17]	79.3	88.6	91.4	0.3	73.7	89.0	93.4	0.5	90.1	95.5	96.5	0.6	83.4	94.6	96.2	0.9	62.5	87.6	92.6	0.9
Pi3X [43]	**85.6**	**92.8**	**94.1**	**0.1**	**91.7**	**96.3**	97.8	**0.2**	**92.7**	**97.0**	**97.7**	**0.4**	**93.5**	**96.9**	**97.3**	**0.5**	**91.5**	**96.3**	97.1	**0.2**
Pi3X† [43]	**85.6**	**92.8**	**94.1**	**0.1**	**91.7**	**96.3**	97.8	**0.2**	**92.7**	**97.0**	**97.7**	**0.4**	**93.5**	**96.9**	**97.3**	**0.5**	**91.5**	**96.3**	97.1	**0.2**
Pow3R [14]	**85.6**	**92.8**	**94.1**	**0.1**	**91.7**	**96.3**	97.8	**0.2**	**92.7**	**97.0**	**97.7**	**0.4**	**93.5**	**96.9**	**97.3**	**0.5**	**91.5**	**96.3**	97.1	**0.2**
Scaffold3D (ours)	**85.6**	**92.8**	**94.1**	**0.1**	**91.7**	**96.3**	97.8	**0.2**	**92.7**	**97.0**	**97.7**	**0.4**	**93.5**	**96.9**	**97.3**	**0.5**	**91.5**	**96.3**	97.1	**0.2**

25.6/53.6/66.4/40.3 and 68.4/85.4/90.4/44.0 in Table 1, while all baselines we re-ran reproduce to within 0.02. In these checks, AUC@10 and AUC@M vary by at most about 0.3, but AUC@1 can move by up to 3.4 points, and margins below 0.5 AUC@10 or below a few AUC@1 points should not be read as reliable advantages. The scaffold-source study and the sparse-graph, 32-view, and 50-view results in Table 4 were evaluated with the current code.

G Broader Impacts

More accurate multi-view 3D reconstruction can benefit robotics, AR/VR, digital content creation, and surgical and embodied perception. Like other reconstruction methods, it could also be used to reconstruct people or private spaces without their consent, and improved reconstruction accuracy could make such privacy-invasive uses more effective.

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 method, assumptions, and empirical scope, and the claims are supported by the main quantitative results in Tables 1 to 4.

   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: Section 4.3 and the discussion and limitations section describe the dependence on the SfM scaffold (including scaffold errors that survive filtering), when a feed-forward model is preferable, and runtime, with supporting evidence in Tables 4, D1, E1 and E3.

   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: [N/A].

Justification: The paper does not present theoretical results or formal proofs; the equations define the 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 method, training setup, datasets, metrics, baselines, and evaluation protocol. Training and evaluation code, checkpoints, and configurations are released through the project page linked in the abstract, and Section C lists the scaffold settings.

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: [Yes].

   Justification: Training and evaluation code, pre-trained checkpoints, and dataset preprocessing scripts are released through the project page linked in the abstract. The evaluation datasets are public.

   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 experiments section specifies datasets, metrics, baselines, architecture, optimizer, learning-rate schedule, training mixture, and evaluation protocol, with additional dataset/support and runtime tables in the appendix.

   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: We report one run per configuration over fixed benchmark splits and do not repeat training, because the experiments are computationally expensive. Instead, Section F quantifies evaluation variance across reruns, random seeds, and code versions and states which margins should not be read as significant.

   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 experiments section reports the GPU type and memory, training GPU count, optimizer-update count, approximate total node-hours/GPU-hours including preliminary and failed runs, and the appendix reports inference runtime breakdowns from 2 to 50 views. All experiments ran on NVIDIA GH200 nodes with four 96GB GPUs per node.

   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 public or previously released datasets and models for multi-view reconstruction research, and we are not aware of any deviation from the NeurIPS Code of Ethics.

   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: Section G discusses potential benefits of improved reconstruction and potential privacy risks from reconstructing people or private spaces without consent.

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 does not introduce high-risk generative models, language models, or scraped datasets; release artifacts are reconstruction code, checkpoints, preprocessing scripts, and evaluation outputs.

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: We use standard public or previously released datasets, methods, and model assets, cite their original sources, and use them for research purposes under their respective terms. Any released code or checkpoints will respect the licenses and terms of the third-party assets they depend on.

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: [Yes].

Justification: The released code, configurations, and checkpoints are documented on the project page and in the code repository.

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 research with human subjects.

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 paper does not conduct crowdsourcing experiments or new research with human subjects.

    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 an important, original, or non-standard component of the core method or experiments; any LLM use was limited to writing, editing, or code-assistance workflows.

    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.