BA-T: An Iterative Transformer for Two-View Bundle Adjustment

Ganlin Zhang[1,2] **Weirong Chen**[1,2] **Daniel Cremers**[1,2] **Xi Wang**[1,2,3,4]

[1] TU Munich [2] MCML [3] Ulm University [4] ETH Zurich

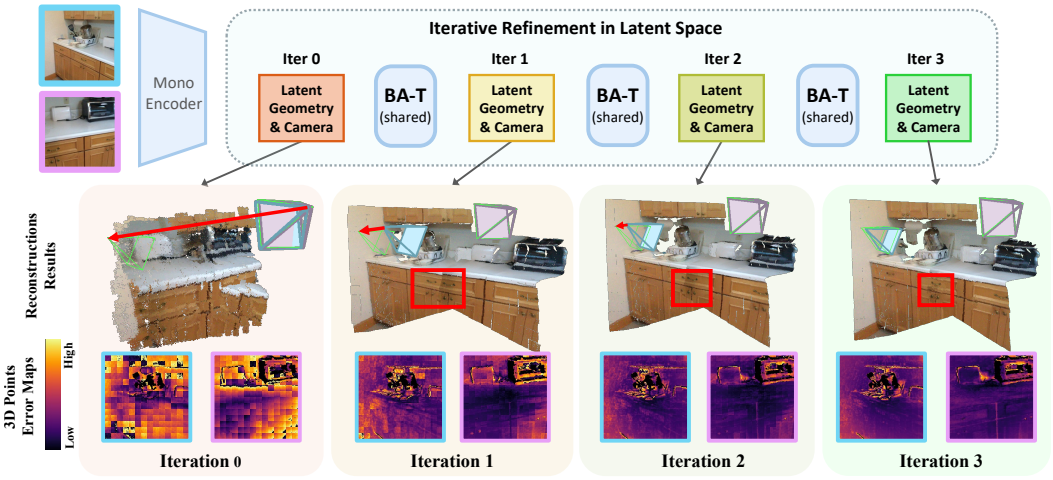

Figure 1: **Overview of BA-T.** Given input images, BA-T performs iterative updates on camera and local geometry tokens using a compact, reusable BA-T layer in latent space. ⟵ indicates error between GT poses and estimated poses (blue and pink), which gradually decreases, and red boxes highlight regions progressively refined across iterations. The 3D point error maps visualize the per-view 3D point errors.

Abstract

Feed-forward models for 3D reconstruction have achieved strong performance using deep cross-view attention to exchange information across images. However, these approaches often depend on heavy decoder stacks and lack a structured mechanism for geometry refinement, resulting in poor multi-view consistency. We address this by drawing inspiration from classical bundle adjustment (BA), which can be viewed as an *iterative information propagation process* between poses and local geometry. Inspired by BA, we propose BA-T, an iterative Transformer that implements BA-style structured updates as a repeatable layer in implicit token space. Instead of relying on deep attention stacks, BA-T refines predictions based on latent residual by a single lightweight layer. Experiments demonstrate that BA-T progressively improves pose and reconstruction accuracy across iterations, achieves stronger cross-view consistency than conventional decoders, and matches or surpasses substantially larger models while using only 16% of their decoder parameters. BA-T provides a compact, efficient, and structural alternative to depth-heavy attention, enabling accurate 3D reconstruction within a lightweight architecture. Github repository: `https://github.com/zhangganlin/BA-T`.

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

1 Introduction

Estimating dense 3D structure and camera poses from multi-view images is fundamental in 3D computer vision, with applications in robotics, AR/VR, and autonomous driving. To obtain dense reconstructions, recent feed-forward methods [42, 18, 39, 17, 43, 19] directly regress dense 3D pointmaps (and camera poses) in a single forward pass. To enforce multi-view consistency, they use deep cross-view attention blocks as decoders to exchange information across images. While effective, these models rely on heavy decoder stacks to approximate geometric constraints and lack a mechanism to self-correct misalignments, limiting their robustness.

In contrast, classical bundle adjustment (BA) [37, 1, 2] refines camera poses and scene structure through iterative optimization, explicitly enforcing geometric consistency. However, its explicit geometric formulation makes it hard to embed in learning-based methods using implicit token representations. Recently, there are learning-based 3D reconstruction systems [33, 35, 36, 40] trying to incorporate differentiable BA modules into their pipelines to improve reconstruction accuracy by adjusting predicted feature correspondences or optical flow. Nevertheless, these BA components still operate as external optimization stages: predictions are first converted into *explicit geometric representations* (*e.g.*, point clouds [33, 40] or feature tracks [35, 36]), after which explicit second-order solvers are applied. Although differentiable, this design still performs iterative updates on explicit geometry, limiting its ability to leverage the expressive power of latent representations. Related iterative refinement strategies operating directly on latent representations have been explored for 3DGS [44], but only for novel view synthesis rather than geometric reconstruction.

This reveals a key gap: feed-forward models leverage powerful latent token spaces but lack structured geometric refinement, while classical optimization is principled and iterative yet relies on explicit geometry, limiting integration with expressive latent representations. To bridge this gap, we draw inspiration from the *structured* iterative refinement of BA, where pose and geometry structure are progressively updated through repeated information propagation. Motivated by this, we propose **BA-T** (fig. 1), a BA-inspired iterative Transformer that mimics BA-style information flow through repeatable learnable updates directly in implicit token space.

BA-T replaces deep cross-view attention stacks with a single structured iterative refinement layer, enabling more compact and efficient updates of camera poses and 3D geometry than vanilla cross-attention blocks. By combining latent 3D representations, feed-forward efficiency, and BA-inspired self-consistency, it bridges end-to-end regression and iterative optimization. Experiments show that BA-T progressively improves pose and reconstruction accuracy, achieves stronger multi-view consistency than conventional decoder stacks, and matches or surpasses two-view models with up to $6\times$ more decoder parameters [42] while remaining competitive with much larger multi-view models at 5–6% of their decoder budget, showing that structured iterative refinement can replace depth-heavy attention while remaining lightweight and accurate.

In summary, our main **contributions** are as follows:

- We propose BA-T, a compact learnable, *bundle-adjustment-inspired* iterative Transformer layer for structured pose and geometry refinement.
- We refine relative camera poses and local geometry at the token level in an *implicit latent space*, leveraging the strong expressiveness of latent tokens.
- We achieve superior accuracy and multi-view consistency with a single *reusable* layer, matching or surpassing two-view models with up to $6\times$ more decoder parameters and remaining competitive with much larger multi-view models, while being more lightweight and computationally efficient.

2 Related Work

3D Reconstruction with Classical Bundle Adjustment. Classical 3D reconstruction systems [6] use bundle adjustment (BA) to jointly optimize camera poses and 3D structure through iterative refinement. Sparse feature-based methods [9, 29, 22, 24, 20] estimate camera trajectories and 3D points from keypoints, refining them by minimizing reprojection error via BA. In contrast, direct methods [13, 12, 14, 46, 45] optimize pixel intensities instead of sparse correspondences, formulating reconstruction as photometric error minimization. Dense extensions, such as DROID-SLAM [35] and others [15, 52, 50, 28, 51] integrate learned priors with differentiable bundle

adjustment, achieving strong geometric consistency through iterative optimization that explicitly couples pose and structure variables. However, these approaches still rely on solver-based explicit updates and require constructing and inverting large second-order Hessian matrices during inference. Such computational overhead makes seamless integration with modern large-scale feed-forward architectures challenging. Moreover, their dependence on optimization dynamics can limit robustness in texture-less or low co-visibility scenarios, where feed-forward models may instead benefit from large-scale data-driven priors.

3D Reconstruction with Feed-Forward Models. Recent learning-based methods perform geometry and pose estimation in a single forward pass. DUSt3R [42] perform two-view reconstruction by regressing pointmaps in a shared global frame, while sequential extensions [38, 5, 41] propagate reconstructions across multiple frames. Two-view follow-ups [49, 26] predict local, view-dependent geometry for greater downstream flexibility. More recent models, including VGGT [39] and others [17, 43, 19], are trained on multi-view data to generalize across more views, and SLAM-Former [48] supports mixed inputs of reconstructed and new frames to enhance consistency. A common characteristic of these feed-forward methods is the heavy use of cross-view attention blocks as geometric decoders for cross-view information exchange. Geometric consistency is enforced implicitly through deep stacks of attention layers, requiring increasingly large and computationally intensive models to approximate the constraints that classical optimization enforces explicitly. Despite their depth and capacity, these models lack an iterative correction mechanism at inference time: once a forward pass is completed, prediction errors cannot be systematically refined.

In contrast, our proposed BA-T replaces heavy cross-view attention stacks with structured, repeatable refinement steps inspired by bundle adjustment. By embedding BA-inspired information propagation into a compact architecture, our method combines efficient feed-forward prediction with iterative geometric refinement without explicit solver-based optimization.

3 BA-T

3.1 Preliminary: Information Propagation of Bundle Adjustment

Bundle Adjustment (BA) jointly optimizes scene geometry and camera parameters by minimizing residuals through iterative refinement. In this process, information circulates between camera estimates and geometry estimates through repeated updates. Many BA variants exist, depending on the choice of residuals and the representation of geometry. We begin by presenting a general formulation of BA for the simplest two-view case. Given two images $\boldsymbol{I}_a$ and $\boldsymbol{I}_b$, BA aims to optimize their local geometry $\boldsymbol{g}_a$, $\boldsymbol{g}_b$ and their relative camera pose c. Consider a pair of local regions (e.g., a pixel or a patch) $p \in \boldsymbol{I}_a$ and $q \in \boldsymbol{I}_b$. The local geometry associated with p, denoted by $g_a^p \in \boldsymbol{g}_a$, is transformed into the local space of $\boldsymbol{I}_b$ via the space transform function $\pi_c^{a\to b}(\cdot)$. A property f of the transformed local geometry is then compared with the corresponding measurement using a *residual function* $r(\cdot,\cdot)$, which forms the core objective of bundle adjustment. The resulting optimization problem can be written as

$$\hat{c}, \hat{\boldsymbol{g}}_a, \hat{\boldsymbol{g}}_b = \underset{c, \boldsymbol{g}_a, \boldsymbol{g}_b}{\arg\min} \left(\sum_{p \in \boldsymbol{I}_a, q \in \boldsymbol{I}_b} r\left(f(g_b^q), f\left(\pi_c^{a\to b}(g_a^p)\right)\right) \right), \tag{1}$$

To solve the optimization problem, the classical way uses a second-order Gauss-Newton solver [23], taking use of the Jacobian matrix $\boldsymbol{J_c}$ and $\boldsymbol{J_g}$:

$$\begin{bmatrix} \boldsymbol{J_c}^T \boldsymbol{J_c} & \boldsymbol{J_c}^T \boldsymbol{J_g} \\ \boldsymbol{J_g}^T \boldsymbol{J_c} & \boldsymbol{J_g}^T \boldsymbol{J_g} \end{bmatrix} \begin{bmatrix} \Delta \boldsymbol{c} \\ \Delta \boldsymbol{g} \end{bmatrix} = - \begin{bmatrix} \boldsymbol{J_c}^T \\ \boldsymbol{J_g}^T \end{bmatrix} \boldsymbol{r} \xrightarrow{\text{rewrite as}} \begin{bmatrix} \boldsymbol{B} & \boldsymbol{E} \\ \boldsymbol{E}^T & \boldsymbol{C} \end{bmatrix} \begin{bmatrix} \Delta \boldsymbol{c} \\ \Delta \boldsymbol{g} \end{bmatrix} = \begin{bmatrix} \boldsymbol{v} \\ \boldsymbol{w} \end{bmatrix}, \tag{2}$$

and the update $\Delta \boldsymbol{c}$ and $\Delta \boldsymbol{g}$ can be solved by Schur Complement [16]:

$$\Delta \boldsymbol{c} = \left[\boldsymbol{B} - \boldsymbol{E}\boldsymbol{C}^{-1}\boldsymbol{E}^T\right]^{-1} (\boldsymbol{v} - \boldsymbol{E}\boldsymbol{C}^{-1}\boldsymbol{w}), \quad \Delta \boldsymbol{g} = \boldsymbol{C}^{-1}(\boldsymbol{w} - \boldsymbol{E}^T \Delta \boldsymbol{c}). \tag{3}$$

These equations summarize the information flow in BA: residuals are first computed, the camera update aggregates the coupled residual information through the Schur complement, and the resulting camera change is then used to refine the geometry.

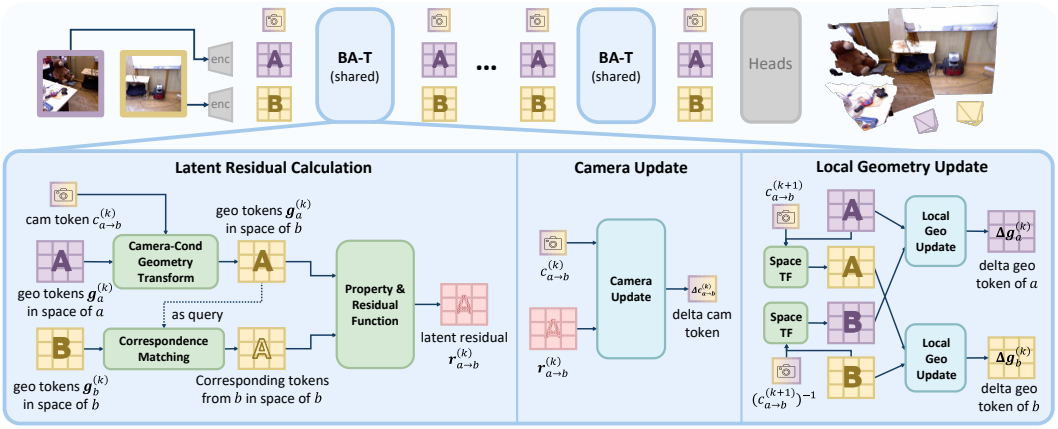

Figure 2: **Overview of the BA-T pipeline.** BA-T takes camera tokens (from learnable initialization) and local geometry tokens (from the image encoder) as input and refines them iteratively. At each step, it performs a BA-inspired implicit refinement step by transforming geometry tokens across camera spaces, matching correspondences, and computing latent residuals. Camera tokens and per-view geometry tokens are refined via the Camera Update and Local Geometry Update respectively.

3.2 Method Overview: BA-inspired Transformer for Implicit Refinement

Notation. We use the following typographic convention: for any token-valued quantity x, we write x for a single token, $\boldsymbol{x}$ for a set of tokens, and $x^{(k)}$ for its value at refinement iteration k.

BA-T. Classical BA explicitly constructs and solves the normal equations (eq. (2)) via iterative second-order optimization, providing strong geometric guarantees but relying on explicit geometry, hand-crafted residuals, Jacobians, and large matrix inversions, making it difficult to integrate into modern end-to-end learning frameworks or scale to latent implicit representations.

The update equations eq. (3) already reveal the key pattern we want to preserve: information flows back and forth between camera and geometry through repeated refinement. Rather than explicitly forming and solving linear systems, we take this *iterative information propagation* of BA as a design cue for a learnable refinement layer between camera parameters and local geometry. This dependency structure can be seen as a global information flow, where updates to one set of variables propagate and influence the other, motivating a learnable mechanism that emulates this interaction pattern.

We model this pattern of information propagation with a learnable layer, **BA-T** (fig. 2), treating camera and geometry parameters as latent tokens, and use attention layers to learn the interaction patterns suggested by classical BA. BA-T is organized around three functional roles that implement this information flow: (1) cross-view correspondence estimation and latent residual computation, (2) camera update, and (3) geometry update. This BA-inspired architecture realizes the update flow through structured token interactions that directly predict parameter updates. Thanks to the structured updates, BA-T also outperforms naive iterative cross-attention stacks, as shown in section 4.2.

Formally, let $c_{a\to b} \in \mathbb{R}^{1\times D}$ denote the camera token for an image pair $(\boldsymbol{I}_a, \boldsymbol{I}_b)$, where D is the token dimension. The camera token encodes relative pose from view a to view b. Let $\boldsymbol{g}_a, \boldsymbol{g}_b \in \mathbb{R}^{N\times D}$ denote the local geometry tokens for images $\boldsymbol{I}_a$ and $\boldsymbol{I}_b$, with N tokens per image. The initial camera token $c_{a\to b}^{(0)}$ is a learnable parameter shared across all pairs, while the initial geometry tokens $\boldsymbol{g}_a^{(0)}$ and $\boldsymbol{g}_b^{(0)}$ are obtained from image features extracted by an image encoder.

BA-T, denoted as $\mathcal{F}_\theta$, takes the set of camera and geometry tokens as input, computes latent residuals in the implicit token space, and updates camera and geometry tokens accordingly. Specifically, given the token set $\mathcal{T}^{(k)}$ from iteration k, $\mathcal{F}_\theta(\cdot)$ produces updated tokens,

$$\mathcal{T}^{(k)} = \left\{c_{a\to b}^{(k)}, \boldsymbol{g}_a^{(k)}, \boldsymbol{g}_b^{(k)}\right\}, \qquad \Delta\mathcal{T}^{(k)} = \mathcal{F}_\theta\left(\mathcal{T}^{(k)}\right), \qquad \mathcal{T}^{(k+1)} = \mathcal{T}^{(k)} + \Delta\mathcal{T}^{(k)}. \quad (4)$$

This update process can be applied iteratively for a variable number of steps $k = 0, \ldots, K$, where K is not fixed and may differ between training and inference. Through repeated refinement, BA-



Figure 3: **Token-level correspondence response.** Given a local geometry token from one view, its attention scores highlight responses in the other view. Correct responses are observed for both ambiguous (left, middle) and distinctive (right) regions.

T progressively enforces global geometric consistency across views, following the same broad refinement spirit as bundle adjustment while remaining a learned latent model.

3.3 Implementation of General Functions in BA-T

Inspired by the general form of BA, we design the following components to support iterative refinement and information exchange between camera and geometry tokens: a camera-conditioned geometry alignment function $\pi_c^{a\to b}$, a property function f, and a residual function r. Additionally, given a geometry token g_a^p in view a, we must identify its corresponding geometry token g_b^q in view b.

Camera-Conditioned Geometry Transform. Since the geometry tokens of the two views are defined in their respective local frames, they are not directly comparable. We first focus on how to inject the camera token $c_{a\to b}$ into the geometry token g_a^p so that they can be compared consistently in a shared space. Since the local coordinates are linked via the relative transformation (which is encoded in the camera token), it is natural to aggregate information from the camera token $c_{a\to b}$ into the geometry token g_a^p of view a. Inspired by the conditioning mechanism in DiT [25], we implement this transformation using a condition block with an adaLN layer, which conditions the local geometry token g_a^p on the camera token $c_{a\to b}$: $\pi_c^{a\to b}(g_a^p) := \text{Condition}(g_a^p, c_{a\to b})$.

Token-level Correspondence Matching. Once the transformed geometry token is obtained, we match it with the corresponding geometry token g_b^q in view b to compute the residual. Since each token represents a local image patch, the projected region in the other view may not align exactly with a single token as in pixel-level matching, but can instead fall between tokens or span multiple tokens. Therefore, a flexible correspondence matching mechanism is required. We adopt an attention-inspired correspondence method. As shown in fig. 3, for a given transformed token $\pi_c^{a\to b}(g_a^p)$, a cross-attention layer queries the corresponding tokens in view b. We can then obtain the attention score $\boldsymbol{\alpha}_p$, measuring the relevance of the query token $\pi_c^{a\to b}(g_a^p)$ and every token in view b, and the corresponding token g_b^q is obtained as the attention-weighted sum of all geometry tokens in view b,

$$\boldsymbol{\alpha}_p = \text{AttnScore}\left(\pi_c^{a\to b}(g_a^p),\, \boldsymbol{g_b}\right), \qquad g_b^q = \sum_{i<N} \alpha_p^i \cdot g_b^i. \tag{5}$$

This soft correspondence mechanism enables BA-T to model sub-token matches and occlusions that are difficult to capture with explicit classical BA.

Latent Property and Residual Function. For the property function $f(\cdot)$, we apply a linear projection to extract a suitable representation from each geometry token:

$$f(\pi_c^{a\to b}(g_a^p)) = \text{proj}\left(\pi_c^{a\to b}(g_a^p)\right), \qquad f(g_b^q) = \text{proj}(g_b^q). \tag{6}$$

The function $r(\cdot,\cdot)$ that calculates residual in latent space is implemented as an MLP applied to the difference between the properties of the two tokens:

$$r_{a\to b}\Big(f(g_b^q), f(\pi_c^{a\to b}(g_a^p))\Big) = \text{MLP}\left(f(g_b^q) - f(\pi_c^{a\to b}(g_a^p))\right). \tag{7}$$

The resulting latent residual token encodes the discrepancy (fig. 4) between the geometry in view a (expressed in b's space) and its corresponding region in view b, serving as a learned analogue of the classical residual (*e.g.*, reprojection error).

Relative Camera Information Update Inspired by the information propagation pattern of classical BA in eq. (3), we update the camera token $c_{a\to b}$ first at each iteration in BA-T. In practice, the camera

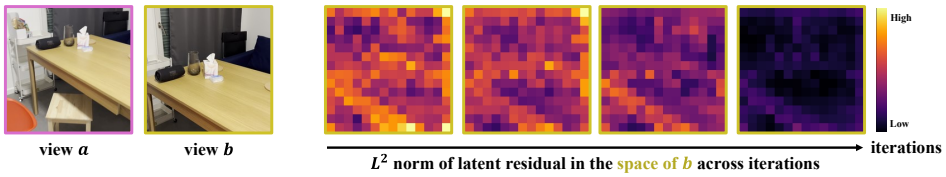

Figure 4: **Visualization of latent residuals.** Residuals are computed in the latent space of view b. Their magnitude decreases across refinement iterations, indicating increasingly accurate estimates.

update $\Delta c_{a\to b}$ is predicted from the current camera estimate $c_{a\to b}$ and the residual tokens $\boldsymbol{r}_{a\to b}$, which already carry the geometry information needed for the next refinement step. Each of the N geometry tokens of view a yields one residual token, so there are N residual tokens $\boldsymbol{r}_{a\to b}$ per image pair. Thus, we update the camera token by attending from the single camera token to all N residual tokens. Specifically, we employ a one-to-many cross-attention mechanism, where the camera token serves as the query and the residual tokens act as the context to be aggregated,

$$\Delta c_{a\to b}^{(k)} = \mathrm{CrossAttn}\left(c_{a\to b}^{(k)},\ \boldsymbol{r}_{a\to b}^{(k)}\right), \tag{8}$$

where $\mathrm{CrossAttn}(\cdot,\cdot)$ denotes a cross-attention operator that aggregates information from multiple residual tokens conditioned on the current camera token. This one-to-many attention-based update implicitly captures the information flow of the camera update in Schur complement in classical BA.

Local Geometry Update After obtaining the camera update $\Delta c_{a\to b}$, we proceed to update the local geometries $\boldsymbol{g}_a$ and $\boldsymbol{g}_b$. As the residual information $\boldsymbol{r}_{a\to b}$ has already been aggregated into the camera update $\Delta c_{a\to b}$, the geometry update depends on the current estimates of local geometry $\boldsymbol{g}_a$ and $\boldsymbol{g}_b$, and the updated camera parameters $c_{a\to b}^{(k+1)} = c_{a\to b}^{(k)} + \Delta c_{a\to b}^{(k)}$.

To effectively exploit cross-view geometric consistency, we update the local geometry using cross-attention between the two views. Specifically, when updating the geometry of one view, we condition on both its own current geometry and the camera-conditioned geometry of the other view, so that both geometry are aligned in the same space,

$$\Delta \boldsymbol{g}_a^{(k)} = \mathrm{CrossAttn}\left(\boldsymbol{g}_a^{(k)}, \boldsymbol{\pi}_{c^{(k+1)}}^{b\to a}\left(\boldsymbol{g}_b^{(k)}\right)\right),\ \Delta \boldsymbol{g}_b^{(k)} = \mathrm{CrossAttn}\left(\boldsymbol{g}_b^{(k)}, \boldsymbol{\pi}_{c^{(k+1)}}^{a\to b}\left(\boldsymbol{g}_a^{(k)}\right)\right). \tag{9}$$

This symmetric cross-view update encourages consistent geometric refinement across views and follows the same information propagation pattern as the geometry update step in classical BA, where local structure is refined conditioned on the updated camera parameters.

Together, these camera and geometry update steps form a complete iterative refinement procedure in BA-T, in which local geometry and relative camera parameters are progressively refined through repeated application of a single refinement layer.

3.4 Implementation and Training

We instantiate BA-T in a two-view reconstruction model that regresses local point maps and relative camera poses for an image pair, using a ViT [11] encoder to capture per-image features and decoding the implicit token from interative BA-T into per-view point maps and relative camera pose using regression heads (more details are provided in Appendix section A).

Loss functions. The supervision signals include point-level geometry, relative pose, and cross-view consistency constraints, following standard choices in prior works [42, 10, 49]. The overall training objective is defined as

$$L = \lambda_{\text{pmap}} L_{\text{pmap}} + \lambda_{\text{pose}} L_{\text{pose}} + \lambda_{\text{gc}} L_{\text{gc}}, \tag{10}$$

where L_{pmap} supervises the predicted local point maps, L_{pose} penalizes errors in relative rotation and translation with an additional cycle-consistency constraint, and L_{gc} enforces geometric consistency between point maps across views under the predicted relative transformation. The weighting factors λ_{pmap}, λ_{pose}, and λ_{gc} balance the three terms.

Table 1: **Relative pose and geometry evaluation on 7Scenes [30] and BundleFusion [8].** BA-T ($K = 4$) achieves best pose and competitive geometry performance among two-view methods, while maintaining a significantly smaller decoder size.

	7Scenes							BundleFusion							Decoder Param Size #↓
	Pose AUC			Geometry				Pose AUC			Geometry				
Method	@5° ↑	@10° ↑	@20° ↑	Chamf.↓	Acc.↓	Comp.↓	$\delta_{1.25}$ ↑	@5° ↑	@10° ↑	@20° ↑	Chamf.↓	Acc.↓	Comp.↓	$\delta_{1.25}$ ↑	
Trained with Multi-view Data															
VGGT [39]	0.19	0.39	0.60	0.10	0.10	0.07	0.97	0.18	0.36	0.56	**0.06**	**0.07**	**0.04**	**0.99**	605M
MA [17]	0.09	0.23	0.44	0.10	0.12	0.07	**0.98**	0.03	0.11	0.27	0.15	0.19	0.09	0.98	171M
DA3 [19]	**0.22**	**0.43**	**0.64**	**0.08**	**0.09**	**0.06**	**0.98**	**0.20**	**0.37**	**0.57**	**0.06**	**0.07**	**0.04**	**0.99**	765M
Trained with Two-view Data															
DUSt3R [42]	0.09	0.22	0.42	0.13	0.13	0.12	0.81	**0.16**	0.33	0.52	0.25	0.26	0.22	0.58	227M
MASt3R [18]	0.15	0.32	0.52	0.16	0.16	0.11	0.79	**0.16**	0.32	0.50	0.24	0.26	0.22	0.58	227M
ViSTA [49]	0.13	0.33	0.56	**0.10**	**0.10**	**0.10**	**0.98**	0.15	0.31	0.51	**0.06**	**0.06**	**0.05**	**0.99**	113M
ViSTA† [49]	0.01	0.06	0.17	0.15	0.12	0.15	0.94	0.02	0.07	0.17	0.09	0.09	0.09	0.98	**38M**
BA-T	**0.16**	**0.37**	**0.59**	0.11	**0.10**	0.11	**0.98**	**0.16**	**0.34**	**0.54**	0.07	0.07	0.06	**0.99**	**38M**

Iterative Supervision. Since BA-T refines camera and geometry tokens over multiple iterations using the same Transformer layer, we apply supervision at the output of each iteration. Let $L^{(k)}$ denote the loss at iteration k, and let K be the total number of iterations. To encourage progressive refinement, later iterations are given higher weights, while earlier iterations contribute less, This design ensures the network learns to gradually refine predictions rather than only focusing on the final output.

$$L_{\text{total}} = \sum_{k=1}^{K} \alpha_{\lambda}^{K-k} L^{(k)}, \quad (0 < \alpha_{\lambda} < 1) \tag{11}$$

4 Experiments

We present the evaluation of BA-T in three parts. First, we compare it with recent feed-forward 3D reconstruction models with deep stacked decoders to assess pose estimation and geometry reconstruction results. Next, we analyze the performance of iterative refinements. Finally, we ablate the components of BA-T and discuss its potentials and limitations. Best results are highlighted as **first** and second.

4.1 Evaluation of Pose Estimation and 3D Reconstruction

In table 1, BA-T (with $K = 4$ iterations for inference) is compared with recent feedforward 3D reconstruction models trained with multi-view data [39, 17, 19] and with two-view data [42, 18, 49]. Since these models employ much larger architectures than BA-T, we train an additional variant of ViSTA [49] with a decoder size comparable to BA-T, denoted as ViSTA†, to enable a fair comparison. ViSTA† shares the encoder, training data, training steps, loss and decoder budget with BA-T and differs only in the decoder design; the full specification is given in Appendix section B.

Metrics and Datasets. We evaluate both pose estimation and geometry reconstruction. For relative pose, we report AUC@5°/10°/20°, computed as the area under the pose accuracy curve with thresholds $5°/10°/20°$, where the pose error is the maximum of rotation and translation angular errors. For geometry, we report Chamfer distance, accuracy, completeness, and depth threshold accuracy $\delta_{1.25}$, defined as the percentage of pixels satisfying $\max(y_{\text{pred}}/y_{\text{gt}}, y_{\text{gt}}/y_{\text{pred}}) < 1.25$, capturing local geometric quality. Evaluation uses 3,600 and 1,109 co-visible image pairs from real world datasets 7Scenes [30] and BundleFusion [8], which provide ground-truth camera poses and 3D geometry.

Results. As shown in table 1, BA-T achieves the best overall pose estimation among two-view models. It even delivers competitive results compared to models trained on multi-view data, despite having a significantly lighter structure (5% of their size). For geometry, BA-T performs on par with substantially larger models. Compared to a model with similar decoder size, *i.e.*, ViSTA† [49], BA-T consistently outperforms in both pose and geometry metrics, demonstrating the effectiveness of structured iterative refinement. Notably, it achieves this with a compact reusable layer (38M parameters), much smaller than prior methods (113M–765M), balancing accuracy and efficiency (running speed and memory footprint are reported in the Appendix table 5).

Table 2: (left) **Performance comparison on 7Scenes [30] across iterative refinement steps.** Increasing the number of refinement iterations consistently improves pose and geometry accuracy. ViSTA† *w/ iterative training* performs iterative update by conventional stacked decoder with comparable size. BA-T outperforms it at all iterations, showing the effectiveness of the proposed structural design.

Method	Iter	Rot. AUC (deg)			Trans. AUC (m)			Geometry				Decoder Size #
		@5°↑	@10°↑	@20°↑	@0.05↑	@0.10↑	@0.20↑	Corr.↓	Rel.↓	$\delta_{1.05}$ ↑	$\delta_{1.25}$ ↑	
ViSTA†	/	0.426	0.662	0.823	0.104	0.272	0.510	0.052	0.079	0.380	0.940	**38M**
ViSTA† *w/* iterative training	1	0.442	0.676	0.830	0.101	0.270	0.505	0.062	0.061	0.497	0.967	**38M**
	2	0.472	0.699	0.844	0.133	0.322	0.555	0.048	0.060	0.513	0.968	
	3	0.485	0.708	0.848	0.146	0.341	0.575	0.045	0.060	0.515	0.968	
	4	0.491	0.712	0.851	0.151	0.346	0.580	0.045	0.060	0.516	0.968	
BA-T	1	0.650	0.819	0.908	0.322	0.555	0.749	0.046	0.051	0.569	0.972	**38M**
	2	0.671	0.830	0.914	0.401	0.624	0.794	0.032	0.050	0.585	0.975	
	3	0.679	0.834	0.916	0.421	0.642	0.805	0.028	0.049	0.594	0.976	
	4	**0.683**	**0.836**	**0.917**	**0.422**	**0.643**	**0.805**	**0.026**	**0.048**	**0.601**	**0.976**	

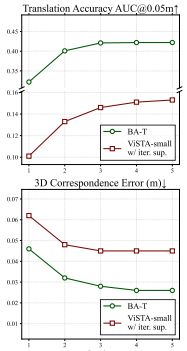

Figure 5: (right) **Iterative refinement behavior.** BA-T (green curve) consistently outperforms ViSTA† *w/* iterative training (red curve) and converges within 3 $\sim$ 4 refinement steps.

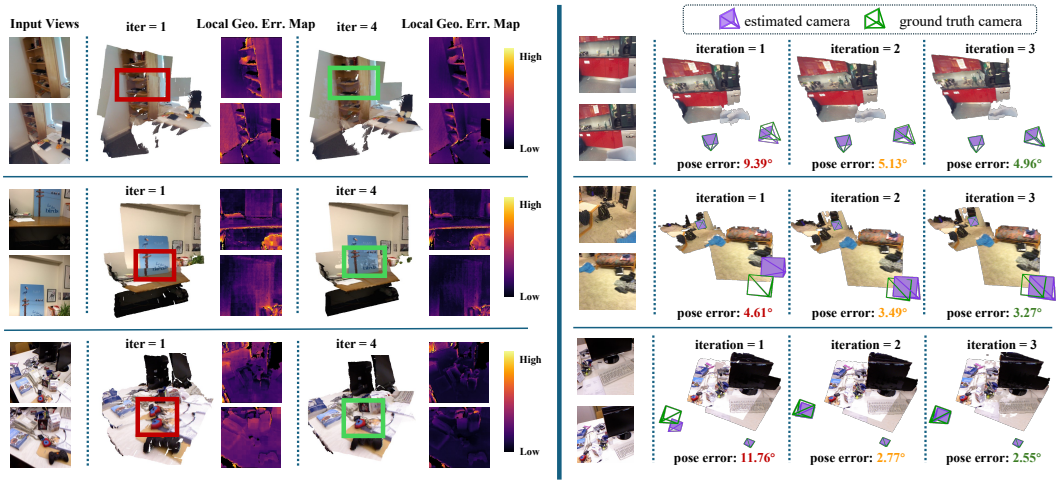

Figure 6: (left) **Qualitative reconstruction results.** We visualize reconstructed geometry and 3D points (in local frame) error maps at iteration 1 and iteration 4. Red boxes highlight noticeable misalignments in the 1_{st} iteration, which are corrected in the 4_{th} iteration, as indicated by green boxes. This demonstrates the effectiveness of the iterative refinement in BA-T for improving local geometric consistency. (right) **Pose estimation visualization.** Estimated relative poses across different refinement iterations are shown. BA-T progressively refines the pose estimates over iterations and remains robust even in low-visibility cases (second row).

4.2 Performance under Iterative Refinements

Setup and Metrics. To evaluate the effectiveness of iterative refinement in BA-T, we analyze performance across multiple update steps. For comparison, we also train a variant of ViSTA† [49] using the same training settings as BA-T. This variant, denoted as ViSTA† *w/ iterative training*, employs iterative training in the same manner as BA-T. The only difference is that this baseline performs iteration with a conventional cross-view attention stacked decoder design, whereas BA-T adopts a BA-inspired structured update mechanism. We report rotation (in degrees) and translation (in meters) AUCs, along with geometry metrics: relative depth error (Rel.) and depth threshold accuracies $\delta_{1.05}$, $\delta_{1.25}$, to evaluate local geometry quality; 3D correspondence error (Corr.), which is the Euclidean distance between corresponding 3D points from two images, and is used to assess the consistency of the local 3D reconstructions across the two views, with ground-truth correspondences obtained from GT depths and camera poses.

Table 3: **Ablation study of BA-T on 7Scenes [30].** *Every* row uses a **38M** decoder and shares the same encoder, training data, training steps and loss, so all differences are attributable to the decoder design alone. Rows (a)–(b) use a conventional stacked cross/self-attention decoder with *no* latent residual; rows (h) and (g) build up the BA-inspired structured update on top of (b); rows (c)–(e) remove one component from the full model. The dominant factor is the latent residual itself: step (b)→(h) alone recovers most of the gain over an identically-trained conventional decoder.

Setting	Rot. AUC (deg)			Trans. AUC (m)			Geometry Reconstruction			
	@5° ↑	@10° ↑	@20° ↑	@0.05↑	@0.10↑	@0.20↑	Corr.↓	Rel.↓	$\delta_{1.05}$ ↑	$\delta_{1.25}$ ↑
Conventional stacked decoder (no latent residual)										
(a) ViSTA† *(no iteration)*	0.426	0.662	0.823	0.104	0.272	0.510	0.052	0.079	0.380	0.940
(b) ViSTA† *w/ iterative training*	0.491	0.712	0.851	0.151	0.346	0.580	0.045	0.060	0.516	0.968
Building up the BA-inspired structured update on top of (b)										
(h) + latent residual *(minimal BA-T)*	0.637	0.810	0.904	0.296	0.534	0.735	0.039	0.051	0.565	0.971
(g) ++ geo. trans. & corr. matching **(full BA-T)**	**0.683**	**0.836**	**0.917**	**0.422**	**0.643**	**0.805**	**0.026**	**0.048**	0.601	**0.976**
Removing one component from the full model										
(c) BA-T *w/o* geometry transform	0.665	0.827	0.912	0.385	0.613	0.787	0.027	0.050	0.599	0.974
(d) BA-T *w/o* iterative supervision	0.667	0.827	0.912	0.404	0.625	0.793	0.036	0.049	**0.620**	0.974
(e) BA-T *w/o* correspondence matching	0.677	0.833	0.916	0.412	0.635	0.802	**0.026**	**0.048**	0.591	**0.976**

Quantitative Results. As shown in table 2, iterative refinement in BA-T steadily improves both pose and geometry. Both pose and geometry metrics get better across iterations, reflecting consistent gains in reconstruction. Most improvements occur within the first 3–4 iterations (fig. 5), after which performance converges. For comparison, ViSTA† [49] reduces 3D correspondence error by only 27%, whereas BA-T achieves 44%, showing that our BA-inspired design more effectively enforces reconstruction consistency. Results demonstrate that structured iterative updates in BA-T efficiently refine camera poses and 3D geometry, enabling high-accuracy reconstruction with a compact layer.

Qualitative Results. In fig. 6 (left), we show reconstruction results after 1 and 4 iterations, along with 3D points error maps, for scenes from 7Scenes [30], BundleFusion [8], and TUM-RGBD [32]. After the first iteration, local geometry roughly captures the scene, but cross-view alignment is still inaccurate, producing misaligned planes and edges. Subsequent iterations progressively correct these discrepancies, yielding better-aligned structures, sharper details, demonstrating improved consistency through structured iterative updates. In fig. 6 (right), it shows geometry and relative camera poses from iteration 1 to 3. Estimated poses become increasingly accurate with each iteration. Notably, in cases with very limited visual overlap (*e.g.*, only a small co-visible region), BA-T still gradually refines camera poses, demonstrating its robustness under challenging low-overlap conditions.

4.3 Ablation: Effectiveness of Components in BA-T

In table 3, we ablate BA-T as a single ladder at a fixed 38M decoder budget. Crucially, the two ViSTA† baselines are included as ablation rows: they share the encoder, training data, training steps and loss with BA-T and differ only in the decoder design, so they separate the contribution of *iteration* from that of the *BA-inspired structure*. Detailed descriptions of each setting are given in Appendix section B.

Iteration alone is not the contribution. Adding iterative training to a conventional stacked decoder, (a)→(b), recovers only 25% of the total rotation gain, 15% of the translation gain and 27% of the cross-view consistency gain. The remaining 75% / 85% / 73% comes from replacing the stacked decoder with the BA-inspired structured update, (b)→(g).

The latent residual is the dominant factor. To identify *what* in the structured update produces that gap, row (h) adds only the simplest possible residual mechanism to (b), with every other BA-T component disabled, so that the two views communicate through an explicit latent residual instead of through direct cross-attention. This single change already recovers 76% of the rotation gain, 54% of the translation gain and 75% of the relative depth gain of (b)→(g). Forming an explicit residual measurement inside the loop, even the crudest one, is therefore what drives the iterative refinement.

The remaining components improve the quality of the residual. The geometry transform and correspondence matching, (h)→(g), make the residual more accurate and informative: they account for the majority of the cross-view consistency gain (68% of the Corr. improvement) and for a further 46% of the translation gain. This is consistent with the leave-one-out rows. Removing the geometry

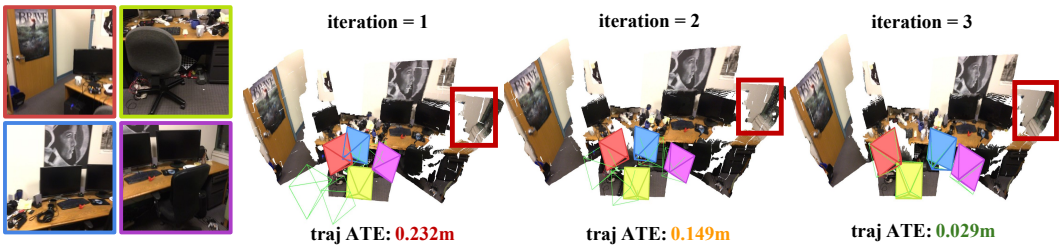

Figure 7: **Multiview reconstruction results.** Estimated poses and reconstructed scenes are visualized across iterations for a 4-view input setup, demonstrating BA-T's ability to handle multi-view settings. The green frustums indicate GT camera poses. The red boxes and decreasing trajectory ATE (0.232m $\rightarrow$ 0.029m) highlights the effectiveness of iterative refinements.

transform (c) forces the network to operate directly in the two different local coordinate frames of the input views, degrading both pose and geometry and confirming that cross-view alignment is essential for effective token updates. Removing correspondence matching (e) mainly costs relative pose while leaving per-view geometry largely unchanged, suggesting these tokens facilitate cross-view pose alignment more than per-view geometry.

Iterative supervision. In row (d) the decoder is still applied K times, but only the output of the *final* iteration is supervised during training, instead of supervising the decoded result of every refinement iteration as in eq. (11). This lowers rotation and translation accuracy and increases 3D correspondence error, indicating that supervising every iteration is what makes the refinement progressive. Overall, the full BA-T achieves the strongest results, highlighting the importance of structured token interaction for accurate pose estimation and geometry.

4.4 Extension: BA-T for Multiview Reconstruction

Thanks to the BA-inspired design, all cross-view interactions in BA-T are performed in view's local space, making it possible to directly extend to multi-view inputs ***without*** architectural changes. The most straightforward way for multiview extension is to allow each query view to attend to all other views, producing per-view pointmaps (details in Appendix). Visualization of a multiview example is shown in fig. 7. More importantly, the model refines all related views together, allowing information from one view to immediately help improve the others and strengthen overall consistency. This makes BA-T a natural fit for multi-view reconstruction and allows it to scale beyond the two-view setting.

While our design is not inherently limited to two views, the current implementation and training mainly focus on two-view reconstruction, broader multi-view configurations remain unexplored. Future work includes integrating BA-T into multi-view reconstruction frameworks [39, 19] to replace cross-view attention decoders and leverage large-scale data, potentially improving global consistency and reconstruction accuracy.

5 Conclusion

We propose BA-T, a Bundle-Adjustment-inspired iterative Transformer that jointly refines camera poses and 3D geometry via an implicit, structured, learnable BA-inspired layer. Unlike conventional feed-forward decoders that predict in a single pass, BA-T enables coordinated, progressive refinement within a compact architecture. Experiments show it consistently improves pose and geometry estimates across iterations, achieves stronger consistency than standard stacked-attention decoders, and delivers high-accuracy pose estimation and 3D reconstruction efficiently.

Acknowledgment This research was funded by the German Federal Ministry of Education and Research through the ExperTeam4KI funding program for UDance (Grant No. 16IS24064).

References

[1] Agarwal, S., Snavely, N., Seitz, S.M., Szeliski, R.: Bundle adjustment in the large. In: ECCV. pp. 29–42. Springer (2010)

[2] Alismail, H., Browning, B., Lucey, S.: Photometric bundle adjustment for vision-based slam. In: ACCV. pp. 324–341. Springer (2016)

[3] Avetisyan, A., Xie, C., Howard-Jenkins, H., Yang, T.Y., Aroudj, S., Patra, S., Zhang, F., Frost, D., Holland, L., Orme, C., et al.: Scenescript: Reconstructing scenes with an autoregressive structured language model. In: ECCV. pp. 247–263. Springer (2024)

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

[5] Cabon, Y., Stoffl, L., Antsfeld, L., Csurka, G., Chidlovskii, B., Revaud, J., Leroy, V.: MUSt3R: Multi-view network for stereo 3D reconstruction. In: CVPR. pp. 1050–1060 (2025)

[6] Carlone, L., Kim, A., Barfoot, T., Cremers, D., Dellaert, F. (eds.): SLAM Handbook. From Localization and Mapping to Spatial Intelligence. Cambridge University Press (2026)

[7] Dai, A., Chang, A.X., Savva, M., Halber, M., Funkhouser, T., Nießner, M.: ScanNet: Richly-annotated 3D reconstructions of indoor scenes. In: CVPR (2017)

[8] Dai, A., Nießner, M., Zollhöfer, M., Izadi, S., Theobalt, C.: BundleFusion: Real-time globally consistent 3D reconstruction using on-the-fly surface reintegration. ACM TOG (2017)

[9] Davison, A.J., Reid, I.D., Molton, N.D., Stasse, O.: MonoSLAM: Real-time single camera SLAM. IEEE TPAMI **29**(6), 1052–1067 (2007)

[10] Dong, S., Wang, S., Liu, S., Cai, L., Fan, Q., Kannala, J., Yang, Y.: Reloc3r: Large-scale training of relative camera pose regression for generalizable, fast, and accurate visual localization. In: CVPR. pp. 16739–16752 (2025)

[11] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. In: ICLR (2021)

[12] Engel, J., Koltun, V., Cremers, D.: Direct sparse odometry. IEEE TPAMI **40**(3), 611–625 (2017)

[13] Engel, J., Schöps, T., Cremers, D.: LSD-SLAM: Large-scale direct monocular SLAM. In: ECCV. pp. 834–849. Springer (2014)

[14] Gao, X., Wang, R., Demmel, N., Cremers, D.: LDSO: Direct sparse odometry with loop closure. In: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 2198–2204. IEEE (2018)

[15] Hagemann, A., Knorr, M., Stiller, C.: Deep geometry-aware camera self-calibration from video. In: ICCV. pp. 3438–3448 (October 2023)

[16] Hartley, R., Zisserman, A.: Multiple view geometry in computer vision. Cambridge university press (2003)

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

[18] Leroy, V., Cabon, Y., Revaud, J.: Grounding image matching in 3D with MASt3R. In: ECCV. pp. 71–91. Springer (2024)

[19] Lin, H., Chen, S., Liew, J.H., Chen, D.Y., Li, Z., Shi, G., Feng, J., Kang, B.: Depth Anything 3: Recovering the visual space from any views. arXiv preprint arXiv:2511.10647 (2025)

[20] Liu, S., Gao, Y., Zhang, T., Pautrat, R., Schönberger, J.L., Larsson, V., Pollefeys, M.: Robust incremental structure-from-motion with hybrid features. In: ECCV. pp. 249–269. Springer (2024)

[21] Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101 (2017)

[22] Mur-Artal, R., Montiel, J.M.M., Tardos, J.D.: ORB-SLAM: A versatile and accurate monocular SLAM system. IEEE transactions on robotics **31**(5), 1147–1163 (2015)

[23] Nocedal, J., Wright, S.J.: Numerical optimization. Springer (2006)

[24] Pan, L., Baráth, D., Pollefeys, M., Schönberger, J.L.: Global structure-from-motion revisited. In: ECCV. pp. 58–77. Springer (2024)

[25] Peebles, W., Xie, S.: Scalable diffusion models with transformers. In: ICCV. pp. 4195–4205 (2023)

[26] Qian, S., Zhang, G., Wu, S., Cremers, D.: Flow4R: Unifying 4d reconstruction and tracking with scene flow. arXiv preprint arXiv:2602.14021 (2026)

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

[28] Sandström, E., Zhang, G., Tateno, K., Oechsle, M., Niemeyer, M., Zhang, Y., Patel, M., Van Gool, L., Oswald, M., Tombari, F.: Splat-SLAM: Globally optimized rgb-only SLAM with 3D gaussians. In: CVPRW (2025)

[29] Schonberger, J.L., Frahm, J.M.: Structure-from-motion revisited. In: CVPR. pp. 4104–4113 (2016)

[30] Shotton, J., Glocker, B., Zach, C., Izadi, S., Criminisi, A., Fitzgibbon, A.: Scene coordinate regression forests for camera relocalization in RGB-D images. In: CVPR. pp. 2930–2937 (2013)

[31] Straub, J., Whelan, T., Ma, L., Chen, Y., Wijmans, E., Green, S., Engel, J.J., Mur-Artal, R., Ren, C., Verma, S., Clarkson, A., Yan, M., Budge, B., Yan, Y., Pan, X., Yon, J., Zou, Y., Leon, K., Carter, N., Briales, J., Gillingham, T., Mueggler, E., Pesqueira, L., Savva, M., Batra, D., Strasdat, H.M., Nardi, R.D., Goesele, M., Lovegrove, S., Newcombe, R.: The Replica dataset: A digital replica of indoor spaces. arXiv preprint arXiv:1906.05797 (2019)

[32] Sturm, J., Engelhard, N., Endres, F., Burgard, W., Cremers, D.: A benchmark for the evaluation of RGB-D SLAM systems. In: IROS (Oct 2012)

[33] Tang, C., Tan, P.: BA-Net: Dense bundle adjustment networks. In: ICLR (2019)

[34] Teed, Z., Deng, J.: RAFT: Recurrent all-pairs field transforms for optical flow. In: ECCV. pp. 402–419. Springer (2020)

[35] Teed, Z., Deng, J.: DROID-SLAM: Deep visual SLAM for monocular, stereo, and RGB-D cameras. NeurIPS **34**, 16558–16569 (2021)

[36] Teed, Z., Lipson, L., Deng, J.: Deep patch visual odometry. NeurIPS pp. 39033–39051 (2023)

[37] Triggs, B., McLauchlan, P.F., Hartley, R.I., Fitzgibbon, A.W.: Bundle adjustment—a modern synthesis. In: International workshop on vision algorithms. pp. 298–372. Springer (1999)

[38] Wang, H., Agapito, L.: 3D reconstruction with spatial memory. In: 3DV (2025)

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

[40] Wang, J., Karaev, N., Rupprecht, C., Novotny, D.: VGGSfM: Visual geometry grounded deep structure from motion. In: CVPR. pp. 21686–21697 (2024)

[41] Wang, Q., Zhang, Y., Holynski, A., Efros, A.A., Kanazawa, A.: Continuous 3D perception model with persistent state. In: CVPR. pp. 10510–10522 (2025)

[42] Wang, S., Leroy, V., Cabon, Y., Chidlovskii, B., Revaud, J.: DUSt3R: Geometric 3D vision made easy. In: CVPR. pp. 20697–20709 (2024)

[43] Wang, Y., Zhou, J., Zhu, H., Chang, W., Zhou, Y., Li, Z., Chen, J., Pang, J., Shen, C., He, T.: π^3: Permutation-equivariant visual geometry learning (2025)

[44] Xu, H., Barath, D., Geiger, A., Pollefeys, M.: Resplat: Learning recurrent gaussian splatting. arXiv preprint arXiv:2510.08575 (2025)

[45] Yang, N., Stumberg, L.v., Wang, R., Cremers, D.: D3VO: Deep depth, deep pose and deep uncertainty for monocular visual odometry. In: CVPR. pp. 1281–1292 (2020)

[46] Yang, N., Wang, R., Stuckler, J., Cremers, D.: Deep virtual stereo odometry: Leveraging deep depth prediction for monocular direct sparse odometry. In: ECCV. pp. 817–833 (2018)

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

[48] Yuan, Y., Chen, Z., Li, K., Wang, W., Zhao, H.: SLAM-Former: Putting SLAM into one transformer. arXiv preprint arXiv:2509.16909 (2025)

[49] Zhang, G., Qian, S., Wang, X., Cremers, D.: ViSTA-SLAM: Visual SLAM with symmetric two-view association. arXiv preprint arXiv:2509.01584 (2025)

[50] Zhang, G., Sandström, E., Zhang, Y., Patel, M., Van Gool, L., Oswald, M.R.: GlORIE-SLAM: Globally optimized RGB-only implicit encoding point cloud SLAM. arXiv preprint arXiv:2403.19549 (2024)

[51] Zhang, W., Cheng, Q., Skuddis, D., Zeller, N., Cremers, D., Haala, N.: HI-SLAM2: Geometry-aware gaussian SLAM for fast monocular scene reconstruction. arXiv preprint arXiv:2411.17982 (2024)

[52] Zhang, W., Sun, T., Wang, S., Cheng, Q., Haala, N.: HI-SLAM: Monocular real-time dense mapping with hybrid implicit fields. IEEE Robotics and Automation Letters **9**(2), 1548–1555 (2024)

BA-T: An Iterative Transformer for Two-View Bundle Adjustment

Appendix

A Architecture and Training Details

Architecture Details. In BA-T, tokens have dimension $D = 768$. In the Space Transform module, geometry tokens are transformed using an AdaLN-style conditional block [25], where each token is normalized and modulated with shift and scale vectors predicted from the camera token (also 768-D). Token-level Correspondence Matching uses 8 attention heads, enabling tokens from different views to interact and establish geometric correspondences. The Latent Property and Residual Function applies linear projections to matched tokens, followed by a 2-layer MLP with expansion ratio 4 to produce normalized residuals that inform the camera update. The camera token is updated through cross-attention (12 heads) and a 2-layer feed-forward MLP with expansion ratio 4. Local geometry tokens are refined via cross-view attention (12 heads, 2-layer MLP with expansion ratio 4) and three intra-view self-attention layers (each with 12 heads and a 2-layer MLP with expansion ratio 4) to improve per-view feature consistency. Training uses $K = 5$ iterations for iterative supervision.

Training Details. We initialize the encoder with a pretrained ViT [49], while BA-T and the regression heads are randomly initialized. All components are trained jointly. Following the setting of prior work [49], BA-T is trained on ScanNet [7], ScanNet++ [47], ARKitScenes [4], CO3D [27], ASE [3], and Replica [31], for 2 days on 8 NVIDIA H100 GPUs. Training uses AdamW [21] with a learning rate of 1.5×10^{-4}, weight decay 0.01, and $\alpha_\lambda = 0.8$, $K = 5$.

B Baseline and Ablation Details

This section specifies the ViSTA† baselines used in table 1, table 2 and table 3, and describes each ablation setting of table 3 in detail. Every model discussed here has a **38M** decoder, so that all reported differences are attributable to the decoder design alone.

ViSTA† (row a). Identical in kind to the original ViSTA [49] decoder: a stack of identical decoder layers, each with (i) self-attention applied to each view separately, followed by (ii) cross-attention in which each view's tokens act as queries and the other view's tokens as keys and values. The original ViSTA decoder has **12** such layers (113M); we use **3** so that the decoder parameter count (38M) matches BA-T exactly.

ViSTA† *w/* iterative training (row b). Identical to ViSTA†, but trained with the iterative strategy of BA-T: the 3-layer decoder is applied K times, the K intermediate token sets are decoded by the same regression heads, and supervision is the weighted sum of eq. (11).

Controlled factors. Both baselines use the same pretrained ViSTA encoder as BA-T, with decoder and regression heads randomly initialised; the same training datasets; the same number of training steps; and the same loss with the same loss weights. Since each BA-T training step backpropagates L_{total}, a weighted sum over refinement outputs, we additionally multiply the non-iterative ViSTA† loss by $\sum_{k=1}^{K} \alpha_\lambda^{K-k}$ so that the loss magnitude per backward pass is comparable across all three models. Without this control the comparison would be confounded by the effective learning rate.

Ablation settings. The remaining rows of table 3 are defined as follows.

- **(h) Minimal BA-T**: the simplest possible BA-inspired decoder, obtained by adding only a latent residual to row (b). The geometry transform and the correspondence matching are both disabled, so the residual reduces to its crudest form: for each of the N token positions we subtract the two views' tokens at the same 2D coordinate and pass the difference through an MLP, $r^i = \mathrm{MLP}(g_a^i - g_b^i)$, in place of the correspondence and residual computation of section 3.3. The difference between (b) and (h) is therefore *only* the medium of cross-view communication: (b) lets the two views' tokens cross-attend to each other directly, whereas (h) forms an explicit latent residual and conditions the update on it. Everything else (encoder, data, steps, loss, iterative training and supervision) is identical.

- **(c) *w/o* geometry transform**: all camera-conditioned geometry transform operations $\pi_c^{a\to b}(\cdot)$ are replaced by the identity.
- **(d) *w/o* iterative supervision**: the decoder is still applied K times, but only the output of the final iteration is supervised, instead of the decoded result of every refinement iteration.
- **(e) *w/o* correspondence matching**: when computing the residual, we directly use the view b token at the same 2D coordinate as the view a token, instead of the attention-fused corresponding token.

Note that rows (c)–(e) are leave-one-out ablations measured from the full model, so they all still contain the BA-inspired residual mechanism and cannot on their own isolate what separates the iteratively-trained conventional decoder from BA-T. Row (h) is included for exactly that purpose.

C Bidirectional Camera Representation

In BA-T, each two-view pair is represented by a single camera token that encodes the relative transformation. However, the local geometry update requires both directions of the relative transformation, and the camera-conditioned operator $\pi^{b\to a}_{c^{(k+1)}}(\cdot)$ still depends on the camera parameters $c^{(k+1)}_{b\to a}$. To obtain the opposite direction of the transformation, we recover $c_{b\to a}$ by applying a lightweight MLP to $c_{a\to b}$. This MLP learns the inverse transformation in latent space, which avoids redundant camera tokens while preserving bidirectional geometric consistency.

D Solver-based BA Comparison

We further compare BA-T with Droid-SLAM, which relies on classical bundle adjustment (BA) using reprojection error as the residual and Gauss–Newton as the optimizer. For a fair comparison, both methods are initialized with the same local geometry and camera poses (taken from iteration 0 of BA-T). Since Droid-SLAM requires camera calibration, we additionally provide the ground-truth intrinsic matrix. Quantitative and qualitative results are reported in table 4 and fig. 8, respectively. The results show that classical BA underperforms compared to BA-T, particularly in geometry reconstruction. This is largely due to the highly non-convex optimization objective, which is difficult to solve reliably under limited co-visibility constraints (e.g., few-view settings). In contrast, BA-T benefits from expressive latent representations and learned priors from training data, leading to improved robustness and accuracy. Moreover, BA-T is substantially more efficient, achieving approximately $5\times$ faster inference while reducing memory consumption by about 45%.

Table 4: **Quantitative comparison of two-view inputs on 7Scenes [30].**

Method	Rot AUC @5°	Trans AUC @0.05m	Rel.	$\delta < 1.25$	Time (ms)	Peak Mem (GB)
Droid-SLAM [35]	0.53	0.08	1.19	0.36	160.93	2.43
BA-T	**0.68**	**0.42**	**0.20**	**0.98**	**30.92**	**1.35**

E Running Time and Peak Memory

We evaluate BA-T against several multiview [39, 19, 17] and two-view [35, 18, 42, 49] methods under the two-view setting, measuring decoder size, peak GPU memory, and inference time on the same input images with the same resolution. As shown in table 5, BA-T achieves the most compact architecture with only 38M parameters, substantially smaller than all prior methods. It also requires the lowest peak memory (1.32 GB) and delivers faster inference, demonstrating a significant improvement in computational efficiency while maintaining competitive accuracy. These results highlight the effectiveness of BA-T's compact and iterative design.

F Discussion on Iteration and Refinement Behavior

F.1 Comparison with Unrolling and Recurrent Methods

Although BA-T is inspired by the iterative refinement spirit of bundle adjustment, it differs from prior unrolling and recurrent methods in both the representation it updates and the role played by each

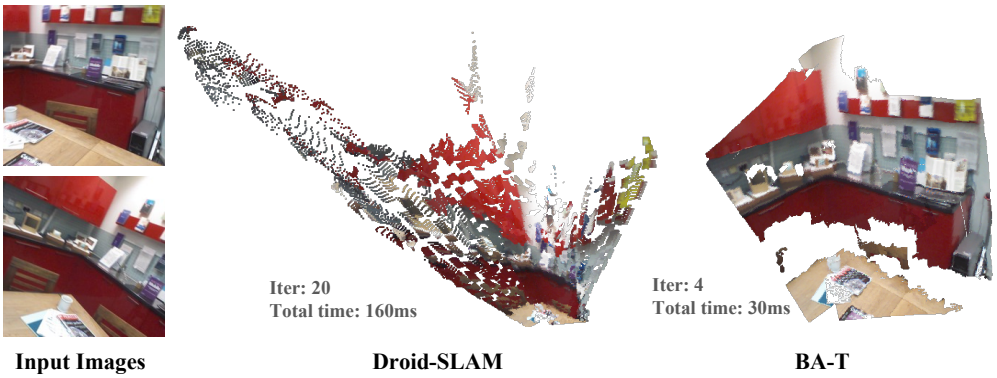

Figure 8: **Reconstruction comparison.** Both methods are initialized from BA-T (iter = 0). Droid-SLAM uses ground-truth intrinsics, whereas BA-T does not. BA-T achieves stronger performance with fewer iterations and faster runtime, benefiting from its more expressive latent space.

Table 5: **Model size and efficiency comparison under two-view inputs.** Decoder size is reported in millions of parameters (M), peak GPU memory in GB, and running time in milliseconds (ms). BA-T achieves the most compact architecture and the highest computational efficiency among all compared methods.

	Droid [35]	VGGT [39]	DA3 [19]	MapAny [17]	MASt3R [18]	DUSt3R [42]	ViSTA [49]	**BA-T** (k=1/2/3/4)
Running time ↓ (ms)	160.93	128.71	78.14	41.13	93.25	66.47	37.96	21.40 / 24.65 / 28.32 / **30.92**
Decoder size ↓ (M)	/	605	765	171	227	227	113	**38**
Peak GPU mem ↓ (GB)	2.43	4.93	6.52	3.23	2.72	2.02	1.76	**1.32**

iteration. BA-Net [33], RAFT [34], ReSplat [44], and BA-T all share an iterative refinement flavor, but the differences are substantial. BA-Net unrolls BA-style updates on explicit geometric variables, RAFT iteratively refines dense correspondence fields for optical flow, and ReSplat recurrently updates latent representations for Gaussian-splat novel-view synthesis; in all three cases, the update target and problem setting differ a lot from ours. The ablation with ViSTA† using iterative training supports the same point: iterative training helps, but a large gap remains compared to BA-T, indicating that iteration alone is not enough. By contrast, BA-T performs the refinement entirely in latent space and is designed around the BA structural information propagation path, where camera and geometry tokens exchange information through a compact, reusable refinement layer. In other words, our method is not just iterative in the generic sense: it is specifically structured to propagate information between camera and geometry in a reconstruction setting, rather than unrolling an explicit solver or recurrently refining flow or rendering variables.

F.2 Pose and Geometry Behavior in Experiments

In our experiments, pose estimation improves more consistently than geometry reconstruction, while the geometry gains are more limited. This behavior follows from the update design: the residual tokens directly influence the camera update, so pose benefits from a more immediate refinement signal. Geometry, in contrast, is updated after the camera has been refined, so its improvement is more indirect.

A second factor is an **initialisation asymmetry**, which applies equally to BA-T and to all baselines in our comparison. The camera token is initialised from a single shared learnable token, which we verified decodes to approximately the identity transform; it therefore starts far from the solution and the decoder has to produce essentially the whole pose, leaving considerable room to improve. The geometry tokens, by contrast, are initialised from the pretrained encoder, which already carries a strong monocular geometry prior and already yields a good per-view prediction. Most of the final geometry accuracy is thus in place before the decoder runs, so the decoder performs alignment and minor refinement rather than construction.

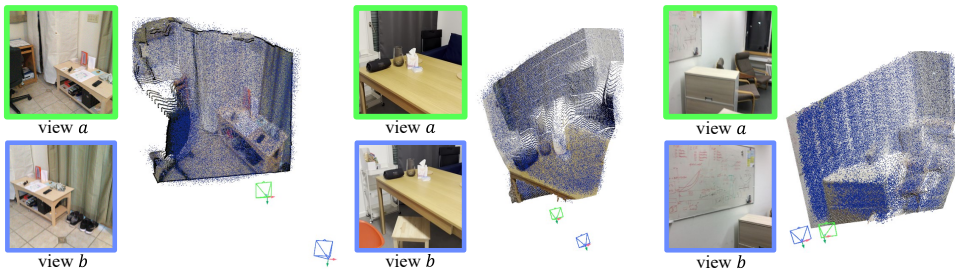

Figure 9: **Visualization of camera-conditioned geometry transformation.** Given two input views, a and b, we visualize the point map regressed from the geometry tokens $\boldsymbol{g}a$ of view a as colored point clouds in the coordinate frame of view a. We also visualize the point map regressed from the transformed tokens $\boldsymbol{\pi}_c^{a\to b}(\boldsymbol{g}_a)$, which is obtained by conditioning $\boldsymbol{g}_a$ on the camera token $c_{a\to b}$, as blue point clouds in the coordinate frame of view b. The two point clouds align closely under the pose transformation encoded in the camera token, demonstrating that the Camera-Conditioned Geometry Transform component of BA-T operates as intended by transforming latent local geometry into the coordinate frames of other views.

A third factor is **dimensionality**. Pose is a single 6-DoF quantity that is heavily overdetermined by many correspondences, so aggregating N residuals into one update filters noise effectively and benefits pose disproportionately. Geometry is high-dimensional and per-token, so the same aggregated signal is structurally weaker for it.

Another factor is that we reconstruct geometry on a per-view basis. The final scene geometry is therefore formed by combining the local geometry predictions with the estimated camera poses, rather than by optimizing a single global geometry representation at once. As a result, the local geometry outputs may not look dramatically different even when the pose estimates improve substantially.

When geometry refinement helps and when it does not. Geometry refinement succeeds where the error is an *alignment* error between two locally plausible reconstructions, which is precisely what the latent residual can express. It contributes least in two regimes: where the encoder's monocular prior is already accurate, leaving little to correct, and where co-visibility is almost absent, so that no meaningful residual can be formed in the first place.

This is why the quantitative gains are often clearer for pose than for geometry. The refinement process mainly strengthens the camera-geometry interaction, and the benefit to reconstruction is accumulated through the updated poses and the per-view local reconstructions.

F.3 Component-Level Behavior

In fig. 9, we also visualize the camera-conditioned geometry transform component by comparing the geometry regressed directly from the local geometry tokens with the geometry regressed from the camera-conditioned transformed tokens. The two geometry predictions roughly align under the provided camera transform, which shows that the transform behaves as designed.

Together with the correspondence and residual visualizations already shown in the main text, it provides component-level evidence that the learned modules follow the intended BA-inspired behavior. Although we do not explicitly train each component to exactly match its BA counterpart, the end-to-end training results show that the learned modules still roughly play the intended functional roles. This further supports the view that the BA-inspired structural design is valuable.

G Multiview Extension Details

To extend BA-T to the multi-view setting, we generalize the two-view formulation (algorithm 1) to operate on multiview inputs. Additionally, it requires a view graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ as input, where each node $i \in \mathcal{V}$ represents a view with geometry tokens $\boldsymbol{g}_i$, and each edge $(i, j) \in \mathcal{E}$ is associated with a relative camera token $c_{i\to j}$. The overall architecture of BA-T remains unchanged; the same refinement layer is applied without any modification to the network structure. The only difference lies in how tokens interact according to the view graph connectivity, shown in algorithm 2.

For each edge (i, j), residual tokens $\boldsymbol{r}_{i\to j}$ are computed using the same procedure as in the two-view case: geometry tokens from view i are first transformed into view j using the camera token $c_{i\to j}$, latent correspondences are then obtained, and residuals are computed in latent space. Camera tokens are subsequently updated independently for each edge using cross-attention between the camera token and its associated residual tokens.

The only difference from the two-view formulation arises in the local geometry update. Instead of performing cross-attention between a pair of views, each view attends to geometry tokens from all neighboring views in the graph, transformed into the same coordinate frame using the updated camera tokens. This effectively changes the interaction from a one-view-to-one-view cross-attention to a one-view-to-many-views cross-attention, allowing each view to aggregate multiview geometric constraints.

Importantly, this extension does not modify the network architecture: the same refinement layer and parameters are reused without introducing additional modules or increasing the model depth or width. The multiview capability therefore emerges purely from the changed token connectivity defined by the input view-graph.

Algorithm 1: Two-view BA-T

Input: $\boldsymbol{g}_a^{(k)}, \boldsymbol{g}_b^{(k)}, c_{a\to b}^{(k)}$

Output: $\boldsymbol{g}_a^{(k+1)}, \boldsymbol{g}_b^{(k+1)}, c_{a\to b}^{(k+1)}$

// Residual computation

$\tilde{\boldsymbol{g}}_a^{(k)} \leftarrow \pi_{c^{(k)}}^{a\to b}(\boldsymbol{g}_a^{(k)})$

$\boldsymbol{g}_b^{q,(k)} \leftarrow \mathrm{LatentMatch}(\tilde{\boldsymbol{g}}_a^{(k)}, \boldsymbol{g}_b^{(k)})$

$\boldsymbol{r}_{a\to b}^{(k)} \leftarrow \mathrm{MLP}\Big(f(\boldsymbol{g}_b^{q,(k)}) - f(\tilde{\boldsymbol{g}}_a^{(k)})\Big)$

// Camera update

$\Delta c_{a\to b}^{(k)} \leftarrow \mathrm{CrossAttn}\Big(c_{a\to b}^{(k)},\ \boldsymbol{r}_{a\to b}^{(k)}\Big)$

$c_{a\to b}^{(k+1)} \leftarrow c_{a\to b}^{(k)} + \Delta c_{a\to b}^{(k)}$

// Geometry update

$\Delta \boldsymbol{g}_a^{(k)} \leftarrow \mathrm{CrossAttn}\Big(\boldsymbol{g}_a^{(k)},\ \boldsymbol{\pi}_{c^{(k+1)}}^{b\to a}(\boldsymbol{g}_b^{(k)})\Big)$

$\Delta \boldsymbol{g}_b^{(k)} \leftarrow \mathrm{CrossAttn}\Big(\boldsymbol{g}_b^{(k)},\ \boldsymbol{\pi}_{c^{(k+1)}}^{a\to b}(\boldsymbol{g}_a^{(k)})\Big)$

$\boldsymbol{g}_a^{(k+1)} \leftarrow \boldsymbol{g}_a^{(k)} + \Delta \boldsymbol{g}_a^{(k)}$

$\boldsymbol{g}_b^{(k+1)} \leftarrow \boldsymbol{g}_b^{(k)} + \Delta \boldsymbol{g}_b^{(k)}$

Algorithm 2: Multiview BA-T

Input: $\{\boldsymbol{g}_i^{(k)}\}_{i\in\mathcal{V}}, \{c_{i\to j}^{(k)}\}_{(i,j)\in\mathcal{E}}, \mathcal{G}(\mathcal{V}, \mathcal{E})$

Output: $\{\boldsymbol{g}_i^{(k+1)}\}_{i\in\mathcal{V}}, \{c_{i\to j}^{(k+1)}\}_{(i,j)\in\mathcal{E}}$

// Residual computation for each edge

foreach $(i, j) \in \mathcal{E}$ **do**

$\tilde{\boldsymbol{g}}_i^{(k)} \leftarrow \pi_{c_{i\to j}^{(k)}}^{i\to j}(\boldsymbol{g}_i^{(k)})$

$\boldsymbol{g}_j^{q,(k)} \leftarrow \mathrm{LatentMatch}(\tilde{\boldsymbol{g}}_i^{(k)}, \boldsymbol{g}_j^{(k)})$

$\boldsymbol{r}_{i\to j}^{(k)} \leftarrow \mathrm{MLP}\Big(f(\boldsymbol{g}_j^{q,(k)}) - f(\tilde{\boldsymbol{g}}_i^{(k)})\Big)$

// Camera update for each edge

foreach $(i, j) \in \mathcal{E}$ **do**

$\Delta c_{i\to j}^{(k)} \leftarrow \mathrm{CrossAttn}\Big(c_{i\to j}^{(k)},\ \boldsymbol{r}_{i\to j}^{(k)}\Big)$

$c_{i\to j}^{(k+1)} \leftarrow c_{i\to j}^{(k)} + \Delta c_{i\to j}^{(k)}$

// Geometry update for each view

foreach $i \in \mathcal{V}$ **do**

$\Delta \boldsymbol{g}_i^{(k)} \leftarrow \mathrm{CrossAttn}\Big(\boldsymbol{g}_i^{(k)},\ \{\boldsymbol{\pi}_{c_{j\to i}^{(k+1)}}^{j\to i}(\boldsymbol{g}_j^{(k)})\}_{j:(j,i)\in\mathcal{E}}\Big)$

$\boldsymbol{g}_i^{(k+1)} \leftarrow \boldsymbol{g}_i^{(k)} + \Delta \boldsymbol{g}_i^{(k)}$

Table 6: **Quantitative comparison of 4 view inputs on BundleFusion [8].** BA-T achieves comparable accuracy to prior methods while requiring significantly lower memory and substantially faster inference.

Method	Trans. (m)	Rot. (°)	Rel.	$\delta < 1.25$	Time (ms)	Peak Mem (GB)
MapAny [17]	0.14	29.65	0.36	0.30	<u>62.95</u>	<u>3.42</u>
VGGT [39]	**0.06**	**10.72**	**0.15**	**0.79**	230.13	5.11
BA-T	<u>0.09</u>	<u>16.39</u>	<u>0.20</u>	<u>0.70</u>	**50.62**	**1.35**

For multi-view setting, we train an additional 12 hours on 4-view inputs. We evaluate BA-T on 4-view inputs from BundleFusion [8] (547 4-view samples) and compare it with recent multi-view methods [39, 17]. In this experiment, the four views form a fully connected view graph, where a camera token is defined for every pair of views. The evaluation metrics include absolute trajectory error (Trans.), absolute rotation error (Rot.), median relative depth error (Rel.), $\delta < 1.25$ accuracy, inference time, and peak memory. As shown in table 6, BA-T achieves accuracy comparable to prior methods while being substantially more efficient, using only 39% of the peak memory of MapAnything. These results demonstrate that BA-T scales naturally to the multi-view setting while maintaining strong accuracy and significantly improving computational efficiency. Additional multi-view results are presented in fig. 10.

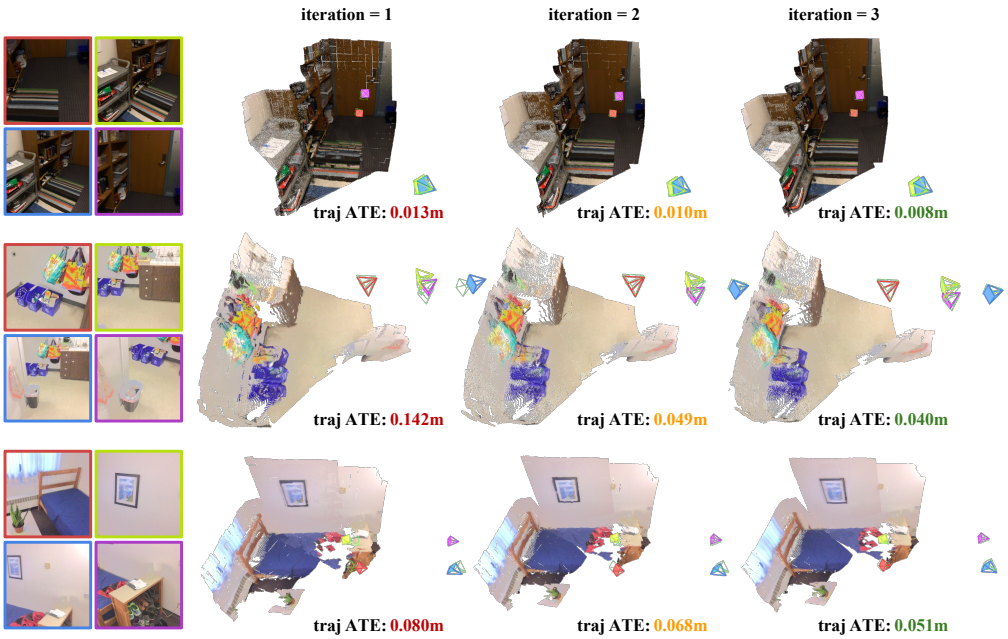

Figure 10: **More qualitative 4-view results.**

H Limitations and Failure Modes

The failure modes of the current model follow directly from its training distribution. Ground-truth geometry is far easier to obtain for indoor, static scenes, so our training mixture is indoor and static and uses pinhole or near-pinhole cameras throughout. Consequently we *cannot* claim generalisation to (i) **dynamic scenes**, (ii) **outdoor scenes**, in particular large-scale outdoor scenes, or (iii) **heavily distorted inputs** such as strong fisheye or wide-angle lenses. This is a limitation of the current *model* rather than of the architecture: the BA-inspired refinement layer makes no assumption that the scene is static, indoor or undistorted, and we expect these regimes to be addressable by broadening the training data.

Dataset	License / Terms of Use
ScanNet [7]	Non-commercial
ScanNet++ [47]	Non-commercial
ARKitScenes [4]	Non-commercial
CO3D [27]	CC BY-NC 4.0
ASE [3]	Non-commercial
Replica [31]	Non-commercial

Table 7: Summary of the licenses or terms of use for the datasets used in training.

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 accurately describe the main contribution as a BA-inspired iterative Transformer in latent token space, and the experimental sections support the claimed gains in pose, geometry consistency, and efficiency.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: The paper discusses key limitations, including the current two-view implementation, the limited exploration of broader multiview settings.

   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 proofs.

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 main paper and supplement disclose the experimental settings needed to reproduce the reported results.

   Guidelines:

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

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

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

Answer: [No]

Justification: The data used is publicly available, and the code will be released upon publication with instructions for reproducing the main results.

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: Training datasets, settings and hardware are all described in the main paper and supplement.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [No]

   Justification: The paper reports main results without error bars or significance tests, following the standard practice in the field.

   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 paper specifies training setting and resources and reports inference memory and runtime statistics for the main experiments in supplement.

   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 research is a standard reconstruction method evaluated on public datasets, and we are not aware of any ethical issues that would violate 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: [No]

    Justification: The paper does not discuss broader impacts explicitly. The method has clear positive applications in reconstruction, but potential misuse or downstream harms are not addressed in the text.

    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 release a high-risk model or dataset that requires special safeguards.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: The paper cites the datasets and baseline methods used, the license and terms of use for the datasets are mentioned in the appdendix.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [N/A]

    Justification: The paper does not introduce a newly released dataset or other standalone asset package.

    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 involve crowdsourcing or human-subject experiments.

    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 involve 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 part of the core method; they are not used as an important, original, or non-standard component of the research.

    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.